IDENTIFICATION OF HEART SOUNDS FOR THE ANALYSIS OF CARDIAC PATHOLOGY USING MACHINE LEARNING

David Susič

Doctoral Dissertation
Jožef Stefan International Postgraduate School
Ljubljana, Slovenia

Supervisor: Asst. Prof. Anton Gradišek, Jožef Stefan Institute, Ljubljana, Slovenia
Co-Supervisor: Prof. Matjaž Gams, Jožef Stefan Institute, Ljubljana, Slovenia

Evaluation Board:
Prof. Mitja Luštrek, Chair, Jožef Stefan Institute, Ljubljana, Slovenia
Prof. Juan Carlos Augusto, Member, Middlesex University, London, United Kingdom
Asst. Prof. Aleksandra Rashkovska Koceva, Member, Jožef Stefan Institute, Ljubljana, Slovenia

MEDNARODNA PODIPLOMSKA ŠOLA JOŽEFA STEFANA
JOŽEF STEFAN INTERNATIONAL POSTGRADUATE SCHOOL

David Susič

IDENTIFICATION OF HEART SOUNDS FOR THE ANALYSIS OF CARDIAC PATHOLOGY USING MACHINE LEARNING

Doctoral Dissertation

IDENTIFIKACIJA SRČNIH ZVOKOV ZA ANALIZO BOLEZNI SRCA S POMOČJO STROJNEGA UČENJA

Doktorska disertacija

Supervisor: Asst. Prof. Anton Gradišek

Co-Supervisor: Prof. Matjaž Gams

Ljubljana, Slovenia, September 2024

"If you don't believe, you shouldn't be here."

Acknowledgments

I am grateful to my supervisor, Asst. Prof. Anton Gradišek, for his consistent support throughout my studies. His encouragement and guidance have enhanced both the quality and enjoyment of my academic journey. I also extend my sincere thanks to my co-supervisor, Prof. Matjaž Gams, for his insightful feedback and for always inspiring me to achieve excellence.

My appreciation goes to Asst. Prof. Gregor Poglajen, M.D., and the team at University Medical Centre Ljubljana for their dedicated efforts in collecting the data essential to this study.

I thank the members of the evaluation board, Prof. Mitja Luštrek, Prof. Juan Carlos Augusto, and Asst. Prof. Aleksandra Rashkovska Koceva, for their time, constructive feedback, and thoughtful suggestions, which have significantly improved the quality and clarity of this thesis.

Lastly, I am grateful to those closest to me for their unwavering support and for the invaluable time we share together.

This thesis was supported by the Slovenian Research Agency (ARIS) through the Young Researcher Grant PR-10495.

Abstract

Cardiovascular diseases (CVDs) are a leading cause of mortality globally, significantly affecting patient quality of life and imposing considerable demands on healthcare systems. Chronic heart failure (CHF), a common outcome of CVDs, represents a growing health burden with an increasing incidence. Accurate and early identification of CVDs is critical for enabling timely diagnosis and effective treatment.

Artificial intelligence (AI) algorithms have shown promise in detecting CVDs, particularly through the application of deep-learning methods using phonocardiogram (PCG) data. However, these models require large datasets, which are often difficult to obtain. Data augmentation has emerged as a viable solution to this challenge. This thesis investigates advanced methods for improving the detection and classification of heart conditions, focusing on CHF, and the development of a heart-sound-specific data-augmentation method to enhance PCG data analysis.

The first investigation analyses heart sounds from CHF patients in their decompensated and recompensated phases. The decompensated phase occurs when a CHF patient requires hospitalization due to worsening symptoms, whereas the recompensated phase indicates a patient's subsequent stabilization. Using a PCG dataset from 37 CHF patients and a combination of machine learning approaches, we achieved up to 72% classification accuracy between the two phases, surpassing the cardiologists' average accuracy of 50%. Key predictive features were derived from diastole and included both time and frequency domains.

The second investigation introduces PCGmix, a novel data-augmentation technique specifically designed for PCG data. PCGmix generates new instances from original recordings through meticulous interpolation, preserving key diagnostic features essential for CVD detection. In this way, it also differs from existing data-augmentation methods, which are general and not specialised for heart sounds. Empirical assessments using a publicly available database of 851 normal and abnormal heart-sound recordings demonstrated that PCGmix outperforms state-of-the-art augmentation techniques when data availability is limited. Specifically, at 10, 30, 56, 112, and 168 PCG recordings, our method achieves the same performance as the no-augmentation approach when train on the amount of data that is 1.35–1.46, 1.34–1.69, 1.46–1.69, 1.08–2.25, and 1.01–1.65 times bigger, respectively.

This thesis contributes to the knowledge at the intersection of AI and medicine, offering novel insights and methodologies to advance cardiovascular health monitoring and diagnosis. It demonstrates the performance achievable in detecting CHF decompensation using PCGs alone, and introduces and thoroughly validates PCGmix, a novel heart-sound-specific data-augmentation method.

Povzetek

Bolezni srca in žilja ali kardiovaskularne bolezni (KVB) so vodilni vzrok smrti v svetu. Močno vplivajo na kakovost življenja bolnikov in predstavljajo precejšno obremenitev za zdravstvene sisteme. Povečevanje pojavnosti kroničnega srčnega popuščanja (KSP), ki je pogosta posledica KVB, predstavlja naraščajočo obremenitev za zdravje ljudi. Natančna in zgodnja identifikacija KVB je ključna za omogočanje pravočasne diagnoze in učinkovitega zdravljenja.

Algoritmi umetne inteligence (UI) so pokazali obetavne rezultate pri odkrivanju KVB. Na veljavi so pridobile predvsem metode globokega učenja z uporabo zvočnih posnetkov srca ali fonokardiogramov (FKD). Modeli globokega učenja potrebujejo veliko podatkov, ki so pogosto težko dostopni. Eden izmed pristopov, ki se je izkazal za učinkovito rešitev tega izziva, je bogatenje podatkov. Ta disertacija preiskuje napredne metode za izboljšanje odkrivanja in klasifikacije bolezni srca, pri čemer se osredotoča na KSP in razvoj metode bogatenja podatkov, specializirane za zvočne posnetke srca.

Prva preiskava analizira zvočne posnetke srca bolnikov s KSP v njihovih dekompenziranih in rekompenziranih fazah. Dekompenzirana faza se pojavi, ko bolnik s KSO potrebuje hospitalizacijo zaradi poslabšanja simptomov, medtem ko faza rekompenzacije predstavlja bolnikovo sledečo stabilizacijo. Z uporabo FKD podatkov 37 bolnikov s KSP in kombinacije metod strojnega učenja smo dosegli do 72 % točnost klasifikacije med obema fazama, kar presega povprečno točnost kardiologov, ki je znašala 50 %. Ključne napovedne značilke so bile pridobljene iz diastole in vključujejo tako časovno kot frekvenčno domeno.

Druga preiskava predstavi PCGmix, novo tehniko bogatenja podatkov, posebej zasnovano na FKD podatkih. PCGmix generira nove primere iz izvirnih posnetkov preko skrbne interpolacije, ki ohranja ključne diagnostične značilke za odkrivanje KVB. S tem se tudi razlikuje od obstoječih metod bogatenja podatkov, ki so splošne in niso specializirane za domeno srčnih posnetkov. Empirične ocene z uporabo javno dostopne baze podatkov z 851 posnetki normalnih in abnormalnih srčnih zvokov so pokazale, da PCGmix presega trenutne tehnike bogatenja podatkov v primerih, ko imamo omejeno količino podatkov. Natačjene, pri 10, 30, 56, 112 in 168 FKD posnetkih naša metoda doseže enako uspešnost kot pristop brez bogatenja, ko se uči na količini podatkov, ki je 1.35–1.46, 1.34–1.69, 1.46–1.69, 1.08–2.25 oziroma 1.01–1.65-krat večja.

Ta disertacija prispeva k znanju na presečišču UI in medicine ter ponuja nove vpoglede in metodologije za izboljšanje spremljanja in diagnoze kardiovaskularnega zdravja. Prikazuje učinkovitost, ki jo je mogoče doseči pri detekciji dekompenzacije srčnega popuščanja izključno na podlagi FKD in predstavi ter temeljito validira PCGmix, novo metodo bogatenja podatkov, specifično za zvočne posnetke srca.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

Heart pathologies pose significant medical challenges. This dissertation explores these challenges through the perspective of computer science, with a focus on the identification of heart sounds as a critical tool for detecting cardiac abnormalities. Leveraging the capabilities of machine learning (ML), we tackle the complexities of training neural networks (NN) with limited datasets to enhance the accuracy and reliability of automated cardiac pathology diagnosis. This research contributes to the knowledge at the intersection of medicine and artificial intelligence (AI), offering novel insights and methodologies aimed at advancing cardiovascular health monitoring and diagnosis.

1.1 Description of the Problem

Cardiovascular diseases (CVDs) are the leading cause of death globally. They include a variety of heart and blood vessel disorders that result in cardiovascular dysfunctions, such as narrowing of blood vessels in the heart and throughout the body, heart and blood vessel complications during childbirth, malfunctioning heart valves, and irregular heart rhythms [1]. CVDs are responsible for approximately 20 million deaths annually, accounting for 32% of all global deaths [2], [3]. By 2030, annual CVD deaths are projected to rise to 23.3 million [4]. Early-stage cost-effective treatments highlight the importance of advanced identification of individuals at high risk of CVD to prevent premature deaths and manage the global health impact.

CVDs can also contribute to the development of chronic heart failure (CHF). CHF is a complex chronic condition, characterized by the inability of the heart muscle to provide sufficient perfusion to meet the metabolic demands of the body. Globally, CHF has reached epidemic proportions, affecting roughly 2% of the world's overall population, with the incidence increasing at 2% annually. The prevalence of CHF reaches around 10% in the overall population aged over 65 years and additionally carries a significant burden in terms of healthcare costs and staff workload [5]. Available literature suggests that identifying the CHF decompensation episodes in their pre-symptomatic phase may enable clinicians to make appropriate and timely changes to patient's medical therapy thus preventing CHF decompensation to occur or to occur in much milder forms that do not require hospitalization. This may significantly improve patients' outcomes [5]. There is thus a significant unmet need for automated, effective, cost-efficient, and robust protocols for detection of heart abnormalities.

Microelectromechanical systems technology, which involves the integration of mechanical elements, sensors, and electronics at a microscale, has the potential to revolutionize various medical applications. However, due to its invasive nature, limited availability, and high costs, it has not yet made a significant impact on the management of heart fail-

ure. Importantly, studies using clinical parameters (body weight, blood pressure, heart rate, etc.) displayed only limited success in accurately predicting CHF decompensation episodes. While electrocardiogram (ECG) data has shown promise in distinguishing between healthy individuals and CHF patients, the signal changes induced by CHF are not always unique to the condition, limiting its diagnostic specificity [6]. In contrast, heart sounds (phonocardiograms or PCGs) provide distinct advantages by directly capturing physiological changes related to valve function and blood flow abnormalities. PCGs can be collected noninvasively using cost-effective electronic stethoscopes, offering valuable diagnostic information efficiently and with minimal procedural demands, thus enhancing their applicability in detecting CVDs. With the use of computers and automated methods for detecting heart abnormalities, patients could monitor the condition of their hearts themselves without having to visit the hospital regularly to have their hearts checked by the medical staff, reducing medical costs and staff workload.

The most comprehensive heart-sound database available to date is the PhysioNet/CinC Challenge 2016 database that consists of 2488 recordings of normal and 665 recordings of abnormal heart sounds [7] with a total time of 20.2 h. The database is curated from nine different sources that vary in recording equipment, signal quality, recording environments, recording body position, patient types, and methods used to determine the diagnosis. While the PhysioNet database is a valuable resource for the development and evaluation of algorithms, practical applications require standardized, specific and high-quality data. In the clinical setting, customized models for specific CVDs are essential for accurate diagnosis and treatment. Relying solely on publicly available datasets can overlook nuances that occur in individual patient populations or specific cardiac conditions. In addition, access to proprietary data allows for customization and refinement of algorithms to meet the unique characteristics of local patient demographics and healthcare practices. The process of gathering proprietary heart-sound data is challenging due to several factors. Firstly, the procedure requires strict adherence to ethical guidelines and patient consent, which can be time-consuming. Secondly, ensuring data accuracy and reliability requires meticulous attention to detail during the recording and annotation stages. In addition, the need for specialized equipment and trained personnel adds to the financial burden of data acquisition. Finally, the complexity of cardiac pathologies requires large sample sizes in different demographic and clinical settings, so larger amounts of data need to be collected to create robust and powerful detection models. Given these challenges, it is essential to develop methods that maximize model performance while taking into account limited data availability.

1.2 Purpose

The purpose of this thesis is to develop a technique for the early detection of decompensation episodes in CHF using heart sounds. With such a technique, CHF patients could monitor their heart at home with affordable electronic stethoscopes, and the recordings would then be sent to the cloud or to a home PC and analysed for possible early signs of a decompensation episode. This would improve the quality of life of CHF patients and reduce medical costs and the burden on medical staff.

With this purpose in mind, we aim to propose, implement and robustly validate two tools. One to classify between decompensated (a state in which the patient requires medical attention) and recompensated (a state in which the patient is healthy and discharged from the hospital) CHF phases, and the second one to maximize model performance with limited data availability using data augmentation. Accurate classification of the two CHF phases would help to detect early signs of decompensation, which in turn would allow physicians

to modify the patient's medical therapy appropriately and in a timely manner. This would prevent decompensation or ensure that it occurs in much milder forms that do not require hospitalization. Maximizing model performance with limited heart-sound data availability is essential for creating robust and reliable models, particularly when dealing with a smaller number of patients. Such tools are invaluable not only because data collection is time-consuming and expensive but also because it can be impractical or impossible due to the rarity of certain diseases or challenges in data collection.

1.3 Hypothesis and Goals

In this thesis, we investigate the following two hypotheses:
Hypothesis 1: Deep neural networks employing data-augmentation techniques on smaller heart-sound datasets perform comparably to those trained on larger datasets.
Hypothesis 2: Machine-learning methods perform comparably to cardiology experts in detecting decompensation episodes in patients with chronic heart failure based solely on heart-sound analysis.

In additon to investigating the hypotheses, the goals of the dissertation are as follows. For the first part of our study, which deals with the classification of CHF phases, our goals are:
Goal 1: Obtain heart-sound data from CHF patients in the decompensated and recompensated states. We will achieve this with the help of cardiologists at the University Medical Centre Ljubljana, who will perform the data acquisition and labeling.
Goal 2: Robustly preprocess this data using the established heart-sound pre-processing techniques.
Goal 3: Extract the most informative features about CHF and other CVDs according to the literature.
Goal 4: Train ML models using these features, validate their performance, and compare them to a baseline of cardiology experts.

For the second part of our study, which deals with maximizing model performance with limited data availability, our goals are:
Goal 5: Obtain and robustly preprocess the PhysioNet/CinC Challenge 2016 heart-sound database using the latest heart-sound pre-processing techniques. This database is the largest publicly available collection of heart sounds and is widely used for developing heart-sound analysis algorithms.
Goal 6: Develop and implement a novel heart-sound-specific data-augmentation method to create new heart sounds.
Goal 7: Train deep-learning models on a variety of smaller fractions of the data (from 1% to 100% of the original database size) with the standard data-augmentation methods, with the developed heart-sound-specific data-augmentation method, and without augmentation.
Goal 8: Robustly validate the performance of all augmentation methods against the baseline without data augmentation on the selected fractions of the data.
Goal 9: Validate the performance of the developed heart-sound data-augmentation method on the heart-sound data obtained from the CHF patients.

1.4 Scientific Relevance and Contributions

The prevalence of CHF has severe implications for patient well-being and mortality, imposing substantial economic and resource burdens on the healthcare system. Early detection of decompensation episodes in CHF has the potential to enable timely adjustment of patient therapy, thereby preventing or mitigating the severity of decompensation events. Such

interventions can significantly improve patients' quality of life. Our research will set limits of how well the decompensation episodes can be detected solely based on heart sounds. In addition, analysing the decision-making processes of ML models will provide new insights into the features of heart-sound data, improving our understanding of pathology of CHF decompensation and diagnostic methods. This research has profound implications for optimising clinical management strategies and improving patient outcomes in the treatment of CHF.

In the context of telemedicine, the integration of affordable electronic stethoscopes enables patients to record their heart sounds independently. These recordings are then automatically analysed in the cloud, eliminating the need for frequent hospital visits for heart examinations. This not only reduces the workload of medical staff, but also improves patient convenience and access to healthcare services. By using ML algorithms for real-time analysis of heart sounds in telemedicine, healthcare providers can remotely monitor patients' cardiac status and intervene immediately if necessary. This research drives innovation in telemedicine and provides scalable solutions to improve access to cardiac care and the management of patients with CHF.

Heart-sound datasets for analysing CVDs often suffer from small size, primarly due to the lack of subjects with specific CVDs. Data augmentation is recognised as one of the most important strategies to address such problems. Augmentation techniques are used to improve model performance even with limited data by diversifying the data set through the generation of additional samples. While various augmentation methods are known, novel techniques tailored to the specific domain of heart sounds represent valuable contributions. The advancement of such augmentation methods is therefore of significant scientific relevance as it improves the effectiveness of ML models in the diagnosis and treatment of CVDs.

The main scientific contributions of this work can be summarised as follows:
Contribution 1: A novel method for augmenting heart-sound data, particularly useful when trained on small datasets.
Contribution 2: Applying and thoroughly validating the data-augmentation method on a variety of subsets of the PhysioNet/CinC Challenge 2016 heart-sound database for heart-sound classification using NNs.
Contribution 3: Demonstrating the performance that can be achieved in detecting decompensation in CHF patients using heart sounds alone.

Additional contributions include:
Contribution 4: Extending the existing labelled dataset of heart sounds from CHF patients in recompensated and decompensated phases.
Contribution 5: A novel metric specialized for assessment of an augmentation method's performance in the context of experiments with datasets of different sizes.

1.5 Thesis Overview

This thesis is organised as follows.

Chapter 2 begins with an exploration of the medical background of CVDs and CHF. Following this, it provides an overview of computer-aided diagnosis systems for heart diseases and describes the characteristics of heart-sound signals. The chapter then reviews the publicly available heart-sound datasets. Next, the chapter details the application of ML in heart-sound classification, covering pre-processing, feature engineering, and classification techniques. This is followed by a discussion of related studies on CHF detection. Finally, the chapter introduces the concept of data augmentation as a regularization strategy for deep learning and reviews a selection of studies that utilize data augmentation for

heart-sound classification.

Chapter 3 outlines the materials and methods employed in our study. It begins with an overview of the datasets, followed by a description of the data pre-processing techniques utilized. Subsequently, the chapter details the methods for detecting decompensation episodes of CHF, including feature engineering, and the experimental setup. The chapter then describes the heart-sound data-augmentation methods, starting with the formulation of our proposed method and followed by an overview of the baseline methods and deep-learning models used. This is followed by a description of the experimental setup. Finally, the chapter defines the performance metrics used for the evaluation of the ML models.

Chapter 4 presents the experimental results of the study. It begins with the detection of decompensation episodes of CHF, including classification by experts as a baseline, evaluation of the model's performance, and model fine-tuning, followed by a parameter sensitivity analysis. The chapter then presents results on heart-sound data augmentation in both time-series and spectrogram domains. This includes analyses of parameter sensitivity, the impact of constrained mapping on instance reliability, feature space visualizations, out-of-manifold intrusion, saliency map visualizations, and the use of saliency information in augmentation. The chapter concludes with an analysis of computational complexity and the results of data augmentation for CHF decompensation detection.

Chapter 5 concludes the thesis by discussing the findings, contributions, and relevance of the research. It provides an analysis of the hypothesis, outlines the limitations of the study, and discusses directions for future work.

Chapter 2

Background and Related Work

2.1 Cardiovascular Diseases and Heart Failure

The most common and serious CVDs include coronary artery disease (CAD), stroke, aortic diseases, valvular heart disease, and heart failure. Despite the broad spectrum of CVDs, in this work we focus specifically on heart-related diseases, especially heart failure. Anatomy of the heart and its cross-section view is given in Figure 2.1.

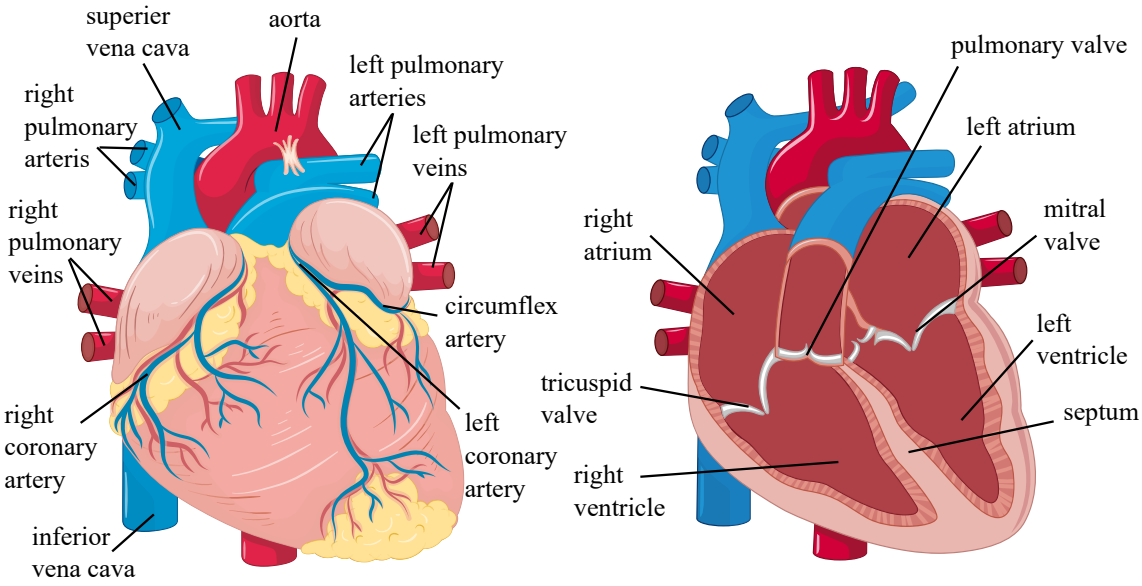


Figure 2.1: Anatomy of the heart. Adapted from [8].

CAD occurs when the coronary arteries that supply the heart with the oxygen-rich blood become narrowed due to the buildup of fats, cholesterols and other substances. This buildup can also burst, which leads to a blood cloth. CAD leads to symptoms such as chest pain (angina), heart attack, irregular heart rhythms and heart failure [9]. Valvular heart disease affects the heart valves that control the blood flow in the heart chambers. A pathological condition in which a valve does not open properly due to narrowing is called stenosis, whereas a condition in which a valve leaks because it does not close properly is called regurgitation. **Aortic stenosis (AS)** is the narrowing of the aortic valve due to the accumulation of calcium, which can also lead to calcification of the valve leaflets [10]. Similarly, **aortic regurgitation (AR)** is characterized by the backflow of blood from the aorta into the left ventricle due to improper closure of the valve [11]. AR can be caused

by **aortic diseases (AD)**, such as aortic dissection (a tear in the inner layer of the aortic wall), aortic rupture, or aortic occlusion [12]. **Mitral stenosis (MS)** is narrowing of the mitral valve, mainly due to the thickening of the leaflets, which leads to the increased blood pressure in the left atrium [13]. **Mitral regurgitation (MR)** is characterized by the backflow of blood from the left ventricle into the left atrium. Over time, MR causes decline in left ventricular ejection fraction [14]. One of the common causes for MR is **mitral valve prolapse (MVP)**, a condition where valve leaflets prolapse back into the left atrium during ventricular contraction [15]. **Pulmonary stenosis (PS)** is characterized by the narrowing in the pulmonary valve. If the narrowing is great, the lower right heart chamber must pump blood with much higher pressure, which causes the right ventricle to become thick [16]. **Tricuspid stenosis (TS)** is the narrowing of the tricuspid valve, obstructing the blood flow from the right atrium to the right ventricle. It most often co-exists with mitral valve pathology and is almost always due to rheumatic fever [16]. **Tricuspid regurgitation (TR)** occurs when the tricuspid valve does not close properly, allowing blood to flow backward from the right ventricle into the right atrium. This dysfunction is often due to the growth of abnormal tissue underneath the valve's leaflets, causing it to degenerate and become incompetent [16]. **Septal defects (SD)** are defects that are present from birth where there are "ruptures" or "discontinuities" in the walls (septum) that separate the heart chambers. Atrial septal defects are located in the wall between the atria, whereas ventricular septal defects are located in the wall between the ventricles. These openings allow blood to flow freely between the ventricles and mix oxygen-rich blood with oxygen-poor blood [16]. **Hypertrophic obstructive cardiomyopathy (HOC)** is an inherited disease in which the heart muscle becomes abnormally thick (hypertrophic). This thickening makes it difficult for the heart to pump blood effectively. In **patent ductus arteriosus (PDA)**, the ductus arteriosus blood vessel, which normally closes shortly after birth, remains open. This opening creates a pathway for oxygen-rich blood from the aorta to flow back into the pulmonary artery and return to the lungs [16]. Alongside the described conditions, additional factors like hypertension (high blood pressure), abnormal heart rhythms, and congenital heart defects can contribute to the development of **heart failure**. Heart failure is a complex condition, characterized by the inability of the heart muscle to provide sufficient perfusion to meet the metabolic demands of the body; alternatively the failing heart stabilizes the circulation by operating at the higher filling pressures, which generate the majority of the symptoms and signs, characteristic of heart failure [5]. Based on left ventricular ejection fraction (LVEF), heart failure has three types, specifically, heart failure with reduced ejection fraction (HFrEF) with LVEF $\leq 40\%$, mildly reduced ejection fraction (HFmrEF) with LVEF between 41 and 49%, and preserved ejection fraction with LVEF $\geq 50\%$ [17], [18].

An analysis of various proposed research and clinical criteria for the diagnosis of heart failure shows that approximately 64 million people worldwide are living with some form of heart failure [19]. In developed countries, the prevalence of heart failure in the general population is around 2%, although some estimates, which also take undetected cases into account, assume a figure of up to 4.2%. The prevelance in people aged 65 years and above is above 10% [5], [20]. Although the incidence is stable, the prevalence is increasing due to the ageing of the population and improvements in treatment. This will lead to a further increase in hospitalisation rates and consequently in healthcare costs in the coming years [5]. The cost of treating this condition is very high, reaching up to 2% of total health care costs in developed countries [21]. In contrast to acute heart failure, which occurs very suddenly and requires immediate medical intervention, CHF is a long-term condition that gradually worsens over time (years). Although CHF cannot be cured, its symptoms can be controlled for many years if detected in early stages. Importantly, the prognosis of CHF

patients is concerning, with a 50% mortality at 5 years, which is largely related to heart failure decompensation episodes that require in-hospital management [5], [17].

The prevalence and incidence of heart failure is influenced by a number of risk factors, including age, sedentary habit, smoking, obesity, excesive alcohol intake, cardiotoxic drugs, chest radiation, diabetes, dyslipidaemia, microbes, and influenza. In addition to a healthy lifestyle, early detection of heart failure symptoms and assessment of severity play a crucial role in halting the progression of the disease, thereby improving the quality of life of those affected and reducing medical costs [5], [22]. The most successful and promising methods for diagnosing CVDs today come from the field of computer science. Computers are able to analyse and learn from enormous amounts of data in a relatively short time, and the so-called computer-aided diagnosis systems have developed rapidly in recent times.

2.2 Computer-Aided Diagnosis Systems for Heart Diseases

The first automated detections of CHF and CVDs in general relied primarily on data modalities such as photoplethysmogram (PPG) data, heart-rate-variability (HRV) data derived from ECG, and clinical parameters [23]. PPG employs light sensors to estimate the blood flow by measuring the changes between the transmitted and reflected light, which gives insight into various heart conditions [24], [25]. HRV data provides information about variation between consecutive heartbeats and is commonly used for CHF-detection tasks [26], [27]. Clinical data includes information about weight, pulse rate, age, systolic blood pressure, respiratory rate and parameters selected from blood tests [28]. However, despite their use, these modalities have shown limited success in accurately detecting CVDs.

Nowadays, diagnosis of CVDs relies on several non-invasive techniques, many of which are imaging-based. Myocardial perfusion imaging (MPI) provides information on the distribution of blood flow in the heart muscle (myocardium) by recording the distribution of an intravenously injected radiotracer during exercise (e.g., exercise) and at rest [29]. Echocardiogram, an ultra sound scan, creates a moving image of the heart and provides information about its size, shape, structure, and function [30]. Cardiac magnetic resonance imaging (CMRI) uses radio waves and magnets to create both still and moving images of the heart and the major bloos vessels [31]. In computed tomography (CT), the heart is scanned with X-rays to create detailed images of the heart and its vessels. Advanced image processing techniques can be used to create 3D and 4D (i.e. spatial-temporal) heart model reconstructions from CMRI and CT images, allowing for better understanding and visualization than 2D images [32], [33]. The biggest disadvantage of imaging techniques is that they require trained personnel and often expensive equipment that only a fraction of the world's healthcare facilities can afford. Another modern diagnostic tools for CVDs screening involves recording the carotid pulse using piezoelectric sensor. Carotid pulse provides information about indication of variations in blood pressure signals in the arteries and the volume of each heartbeat [34]. However, this approach also requires the expertise of trained medical personnel.

One of the most popular methods for recording the heart cycle signal today is electrocardiography. Electrocardiography is the recording of the electrical currents caused by the contraction of the muscle fibers of various parts of the heart. The electrical activity of the heart is recorded in the electrocardiogram through electrodes placed on the surface of the body. The ECG trace consists of a series of waves of different morphology, time, and rate, with peaks and valleys designated by the letters P, Q, R, S, and T (see Figure 2.2). The P-peak and QRS complex represent depolarisation of the atria and ventricles, respectively, whereas the T-peak represents repolarisation of the ventricles.

While widely used, ECG has limitations in detecting structural abnormalities in heart

valves and conditions associated with heart murmurs [35]. Additionally, the ECG signal alterations associated with CHF lack specificity, as these changes are not exclusive to the condition [6].

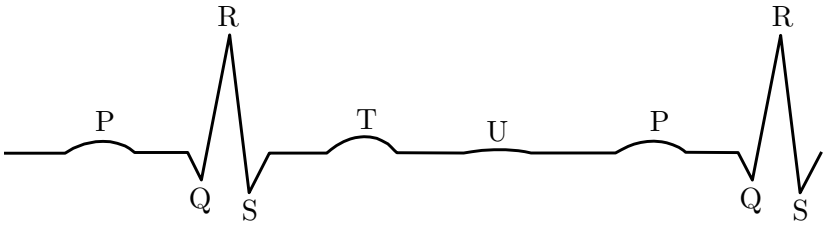


Figure 2.2: Schematic representation of an ECG signal.

Recent research findings emphasise the central role of heart sounds in the diagnosis of CVDs. Their ability to detect nuanced variations in heart function makes them a promising tool for improving the accuracy and reliability of CVD-detection methods. The affordability of cardiac auscultation, combined with the compact and portable nature of electronic stethoscopes, underscores their value for the adoption of computer-aided auscultation in primary care and telemedicine. In addition, the number of publicly available datasets containing recordings of heart sounds has increased dramatically over the past decade, facilitating the development of robust methods for analysing heart sounds and detecting CVDs.

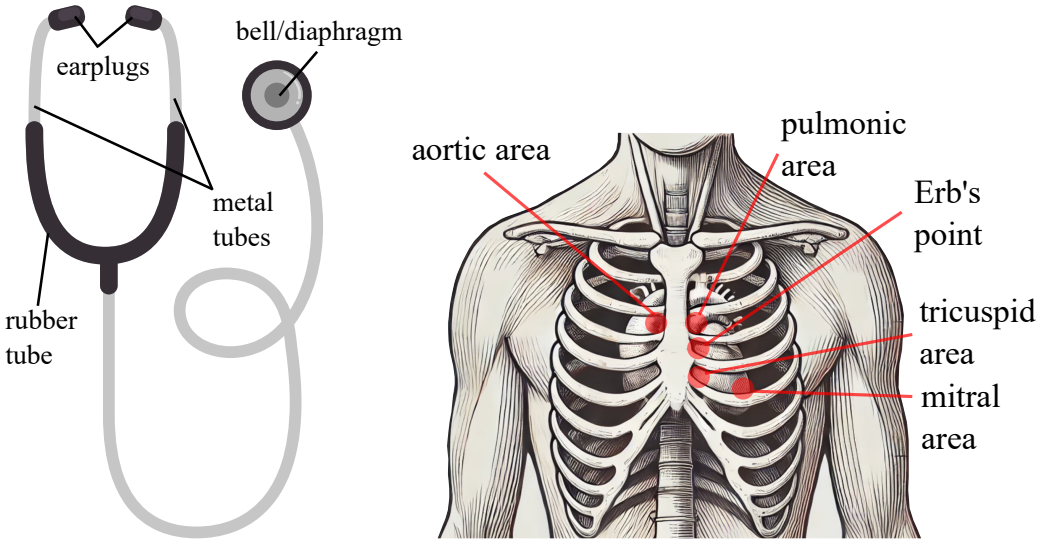
2.3 Heart-Sound Signals

Heart sounds are produced by the mechanical activity of the heart muscle and by the sudden starting or stopping of blood flow within its chambers. The opening and closure of heart valves regulate blood flow between the heart chambers and throughout the circulatory system. These physiological processes induce audible vibrations that provide valuable insights into the health of the heart. The time-series audio recording capturing these sounds is called a phonocardiogram (PCG) [36].

2.3.1 Acquisition

Heart sounds are mainly listened to and recorded with a stethoscope, a device equipped with a diaphragm made of flexible material that produces sound through vibration. This sound is conveyed through hollow tubing, acting as a filter, and transmitted to earpieces for auditory perception. The traditional stethoscope comprises three primary components: a chest piece, predominantly crafted from stainless steel, a tubing section, and earpieces, as depicted in Figure 2.3a [37]. Electronic stethoscopes, which offer various advantages over conventional models such as enhanced accuracy, ease of use, sound quality, sensitivity, and real-time waveform display [38], have become increasingly prevalent. Additionally, electronic stethoscopes convert sound signals into electrical signals, facilitating the possibility of saving the data onto electronic devices such as smartphones or computers. Medical personnel listen to and record the heart sounds by placing the stethoscope diaphragm on the body in the region of the heart. This process is known as heart **auscultation**. The standard positions for heart auscultation are shown in Figure 2.3b. They are the aortic area (in the middle of the 2-nd intercostal space), the pulmonic area (2-nd space along the left sternal border), the Erb's point (3-rd intercostal space along the left sternal border),

the tricuspid area (4-th intercostal space along the left sternal edge) and the mitral area (5-th intercostal space on the midclavicular line) [39].



(a) Stethoscope schematic (b) Standard heart auscultation positions.

Figure 2.3: (a) Schematic illustration of a stethoscope, detailing its main components, and (b) a diagram of the chest, highlighting the standard heart auscultation positions. Adapted from [8].

2.3.2 Characteristics

Example of a PCG signal is shown in Figure 2.4. The heart cycle, also known as the heartbeat, is composed of two intervals called the systole and the diastole. In a normal heartbeat, the duration of systole is about 0.35 s and the duration of diastole is about 0.4 s. The sounds produced within a heart include the first sound (S1), the second sound (S2), the third sound (S3), the forth sound (S4), gallop, murmurs, opening snaps, rubs, and clicks. The first and the second heart sounds compose a normal heart signal and are also called the fundamental heart sounds [36]. The other heart sounds are caused by abnormalities of the heart and are associated with CVDs, with the exception of the S3 sound, which may also be present in normal hearts of young children and athletes [40].

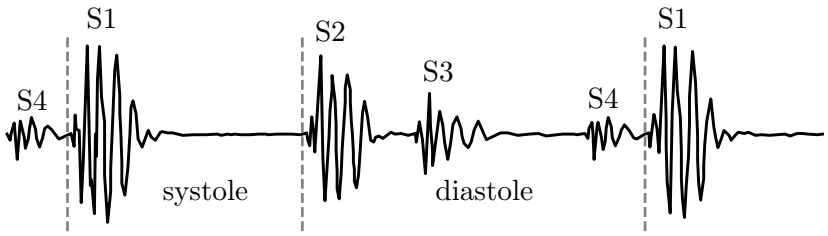


Figure 2.4: Schematic representation of a PCG signal, indicating the fundamental heart sounds S1 and S2, as well as the S3 and S4 heart sounds.

The **first sound** occurs in the beginning of the systole and is caused by the closure of the mitral and tricuspid valves. It is manifested by the vibrations caused by the tension

on the valves, deceleration of the blood and contraction of the ventricular muscles. It is the loudest and the longest heart sound, with a duration of 0.12 – 0.15 s. The **second sound** occurs in the beginning of the diastole and is caused by the closure of the aortic and pulmonic valves. The sound is manifested by the vibrations in the arteries due to deceleration of blood and by the vibrations of ventricles and atria. Its duration is 0.08 – 0.12 s. The **third sound** occurs just after the second sound due to the twitching of the ventricular wall during the rapid ventricular filling. Because the walls are relaxed, the frequency of the sound is low. The **forth sound** is a low frequency sound that occurs at the end of the diastole, just before the S1 of the new heart cycle. It occurs when the atria contract during the late diastolic filling phase. **Gallop** sounds are low frequency sounds that occur during the diastole period. They are caused due to rapid deceleration of the blood flow into the ventricle and due to decreased ventricular distensibility[1]. **Murmurs** are noises that can be present both in systole and diastole and reach frequencies of up to 1000 Hz. They are caused by the loose structures within the heart, the backward regurgitation through a valve that is leaking, high blood flow through valves, and flow through a narrowed valve. **Opening snaps** occur right after the second sound and are a result of a sudden pathological arrest of the opening of the mitral and tricuspid valves. **Rubs** can be heard both in the systole and diastole and are cause due to the friction between layers and abrasion of pericardial surfaces. **Clicks** are heart in the early part of systole and occur due to the opening of a rigid and calcified aortic or pulmonary valve [40], [42]. Information about the heart-sound frequency ranges, qualitative characteristics, durations, locations, and causes are listed in Table 2.1.

AS produces a harsh systolic crescendo-decrescendo murmur that is best heard at the right upper sternal border radiating into the carotid arteries. AR produces a decrescendo-bubbling diastolic murmur that is best heard at the lower left sternal border. MS produces a diastolic murmur that is best heard at the apex, while MR produces a systolic murmur at the apex that radiates into the left axilla. MVP is characterized by an early systolic click at the apex, often followed by a late systolic murmur. PS is manifested by a crescendo-decrescendo systolic ejection murmur that is loudest at the upper left sternal border. TS produces a diastolic murmur that is best heard at the lower left sternal border, while TR is associated with a systolic murmur best heart at the same location. SD produce distinct sounds: atrial SD show a loud, broad S1 with a fixed split S2 at the upper left sternal border, while ventricular SD produce a holosystolic murmur at the apex, the intensity of which varies depending on the size of the defect. In HOC, a systolic ejection murmur is heard between the apex and the left sternal border, which is enhanced by Valsalva or abrupt standing maneuvers. The PDA is characterized by a continuous "machine-like" murmur that is loudest at the upper left sternal border [43].

PCGs for abnormal heart sounds of the most common heart abnormalities are given in Figure 2.5. We can deduce that heart sounds are very complex and informative data modality that is sensitive to a variety of heart abnormalities. Based on the locations, durations, and frequency of the heart sounds and murmurs, PCGs can serve as a primary diagnostic method for heart valve diseases. There are, however, some drawbacks of PCGs as graphical representations of the heart sounds. Distinguishing between the individual frequencies of the different heart sounds and providing detailed information about the energy variations in these sounds is a challenge for PCGs. In addition, artifacts and noise in PCGs can obscure weak or subtle heart sounds, making them difficult to detect visually. Furthermore, problems arise in identifying specific boundaries of heart sounds within PCGs [44]. Overall, the complexity of PCGs presents an opportunity for robust computer-aided diagnostic systems. By utilizing advanced signal-processing and ML techniques, these

[1] A metric of the stiffness of blood vessels [41]

systems can improve interpretation, leading to more reliable cardiac diagnoses in clinical practice.

Table 2.1: Pathophysiology of heart sounds [40], [42].

Heart sound	Frequency range	Qualitative characteristics	Duration/ location	Causes
S1	10 – 200 Hz	Dull and prolonged	0.12 – 0.15 s	Closure of the tricuspid and mitral valves
S2	20 – 250 Hz	Sharp and short	0.08 – 0.12 s	Closure of the pulmonic and aortic valves
S3	25 – 70 Hz	Soft	≈0.04 s, early diastole	Rapid ventricular filling
S4	15 – 70 Hz	Weak and rumbling	Slightly before S1	Atria contraction
Gallop (ventricular)	15 – 50 Hz	Galloping rhythm	≈0.15 s, early diastole	Rapid deceleration of blood flow into the ventricle
Gallop (atrial)	15 – 50 Hz	Galloping rhythm	0.08 – 0.2 s, just before S1	Decreased ventricular distensibility
Gallop (summing)	15 – 50 Hz	Quadruple rhythm	During diastole	S3 and S4 are superimposed
Murmurs (innocent)	120 – 250 Hz	Whooshing, roaring	Early systole	Turbulent blood flow
Murmurs (systolic)	Up to 600 Hz	Rasping, blowing	Systole	Systolic and ventricular ejection
Murmurs (diastolic)	Up to 600 Hz	Puffing, rumbling	Diastole	Ventricular relaxation and filling
Opening snaps	100 – 800 Hz	Snapping sound	Diastole	Inspissating of valve leaflets
Rubs	100 – 800 Hz	Scratching sound	Systole, diastole	Friction between layers
Clicks	100 – 800 Hz	Short and loud	Early systole	Opening of a rigid aortic or pulmonary valve

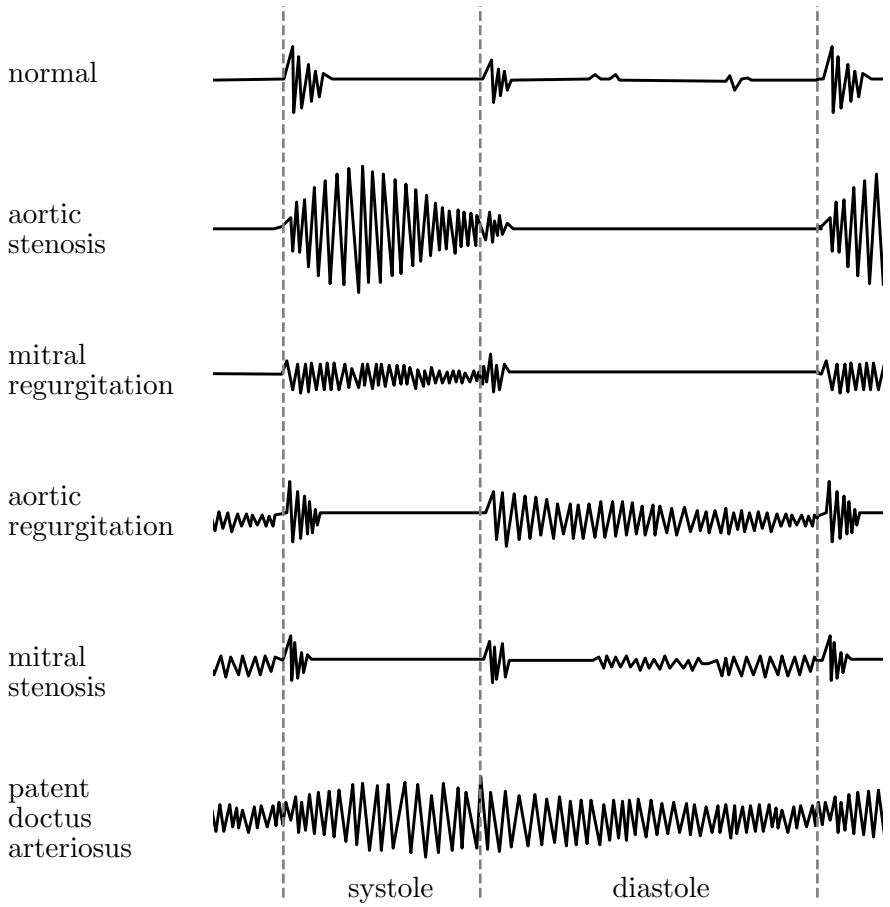


Figure 2.5: Schematic representations of PCGs for some of the most common heart abnormalities. Adapted from [45].

2.4 Available Heart-Sound Datasets

Large-scale and high-quality datasets are essential for the development of heart-sound analysis algorithms. There are several publicly available heart-sound datasets that have been released in the past decade that have been extensively used for the validation of such algorithms. Among them, the PhysioNet/Computing in Cardiology Challenge 2016 (PhysioNet/CinC Challenge 2016) database [7], [46] stands out as one of the best known resources in the field. The available heart-sound databases are listed in Table 2.2. For a database to be included in the table, it has to be accessible online, it has to include relevant information about the study subjects (e.g., gender, age), and the acquisition process has to be described in detail (recording device, auscultation locations). In this thesis, we employ the PhysioNet database alongside a CHF dataset collected for this study. Comprehensive descriptions of both datasets are provided in Section 3.1. For information on the other datasets listed, refer to the respective references.

In addition to the above, there also other heart-sound datasets. Texas Heart Institute database (2010) [55] includes openly available 44 heart sounds and murmurs, but the information about the subjects and the acquisition process is lacking. Biosciences Database of Heart Sound Signals (2017) includes 25 recordings of normal and abnormal heart sounds with the total length of approximately 1 min. No information about the subjects and the recording conditions is available [56]. The Cardiac Auscultatory Recording Database

Table 2.2: Available heart-sound datasets.

Dataset	#Subjects	#Samples	Duration	Types
PhysioNet/CinC Challenge 2016 [7]	764	3240	20.2 h	Normal, abnormal
Michigan [47]	Unknown	23	0.4 h	Normal, abnormal
PASCAL [48]	Unknown	176 656	0.4 h 1.2 h	Normal, abnormal
Heart Sounds Shenzhen [49]	170	845	7.0 h	Normal, abnormal
DigiScope [50]	29	29	≈4 min	Normal children
CirCor DigiScope [43]	942	3163	20.1 h	Normal, abnormal children
HSCT-11 [51]	206	412	5.2 h	Unknown
Fetal Heart Sound [52]	26	26	-	Fetal and pregnant
EPHNOGRAM [53], [54]	24	69	30.6 h	At rest, activities

(CARD) (2001) [57] consists of normal and pathological recordings from 800 different patients with each recording being 20 s in length. Heart sounds were captures simultaneously with ECG data. Information about the patient types and demographics is unknown. The Yaseen (2018) dataset was collected by [58] and consists a total of 1000 2 s recordings, 200 from each of the subject types: healthy, AS, MR, MS, and MVP. Details on subject demographics and recording conditions are unknown.

In addition to the publicly available datasets, numerous researchers have also collected their own datasets with the purpose of developing developing heart-sound analysis algorithms. Comprehensive overview of various private heart-sound datasets and their applications can be found in [42], [59].

2.5 Machine Learning for Heart-Sound Classification

In this thesis, we focus on the identification of heart sounds for the analysis of cardiac pathology using ML classification techniques. A typical heart-sound analysis pipeline, based on traditional sound analysis methods, consists of several key steps. The initial step is pre-processing, which includes signal denoising and segmentation. This is followed by feature engineering, where a small number of representative features are extracted from the signals. The final step is classification, where these features are fed into an appropriate classification model. Our focus is on ML classification techniques, encompassing both classical ML methods and NNs. Convolutional neural networks (CNNs), a type of NN, utilize convolutional and pooling layers to directly extract features from the signal, thereby integrating feature engineering and classification into a single process. When pre-processing is bypassed and raw signals are input directly into CNNs, this approach is termed end-to-

end learning, as the model learns all steps from initial input to output. Alternatively, the signal can be transformed using Fast Fourier transform (FFT) to produce a spectrogram, a time-frequency representation of the signal, which is then input into the CNN. The heart-sound analysis pipelines using both classical ML methods and NNs are illustrated in Figure 2.6. In the following, all of the heart-sound analysis steps are described in detail.

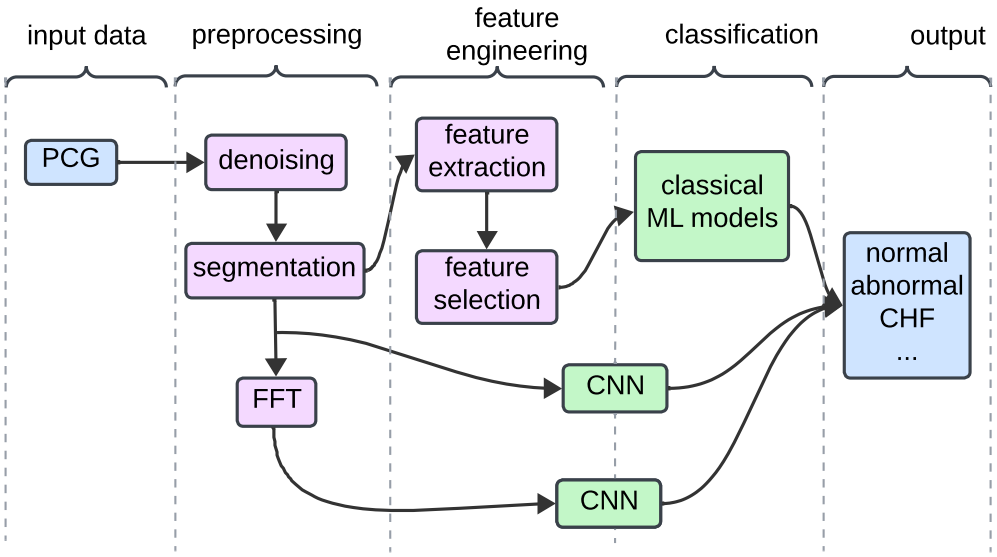


Figure 2.6: A diagram illustrating standard ML heart-sound analysis pipelines from input data to output [42].

2.5.1 Pre-processing

After data acquisition and labelling, pre-processing steps, denoising and segmentation, are essential to ensure the quality and relevance of the data for subsequent analysis. The goal of denoising is to eliminate the noise without altering the original signals, whereas the goal of segmentation is to segment the data into meaningful and manageable segments. The PCG signals are typically segmented into heartbeats and the goal of the segmentation is to also determine the location of the four heart states in each heartbeat. In PCG analysis, a heart state is defined as per Definition 2.1.

Definition 2.1 (Heart state). A *heart state* is one of the four non-overlapping phases within each heartbeat: S1, systole, S2, and diastole.

It should be emphasised that in the context of heartbeat segmentation, S1 and S2 are treated as separate states, although they occur at the beginning of systole and diastole respectively. This means that the states do not overlap; the systole state begins immediately after the end of the S1 state, and the diastole state begins immediately after the end of the S2 state. This precise segmentation ensures a clear delineation of the individual heart states within the heartbeat. The example of a raw, denoised and segmented PCG is given in Figure 2.7.

2.5.1.1 Denoising

Apart from being very faint, heart-sound recordings typically contain a high amount of noise coming from various sources, e.g., ambient noise, breathing noise, lung noise, noise

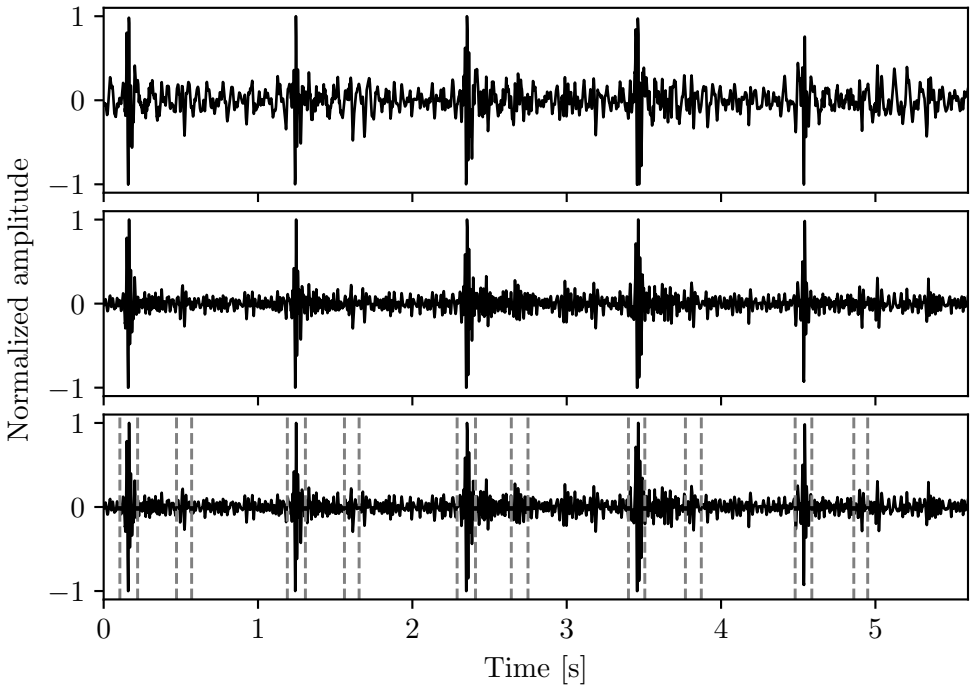


Figure 2.7: PCG denoising and segmentation: (top) raw signal, (middle) denoised signal and (bottom) segmented signal. The dashed vertical lines indicate the beginnings of the heartbeat states, starting with S1.

due to contact between the recording device and the human body, and humans' speech. Additionally, since heart sounds are converted into electrical signals, they are susceptible to electromagnetic interference and power frequency noise. The goal of denoising is to eliminate the noise without altering the original PCG signals and is a crucial pre-processing step for accurate automated identification of heart-sound signals. The most commonly used denoising techniques for PCG signal analysis are linear filters, e.g., low-pass, band-pass and high-pass filters, wavelets, e.g., continuous wavelet transform and discrete wavelet transform, and Kalman filters [42], [60]–[64].

The process of **linear filtering** in signal processing can be understood as a methodical manipulation of signal components within the frequency domain [65]. In this domain, signal components are represented by their respective magnitudes. Controlled scaling, an integral aspect of linear filtering, involves the selective adjustment of these magnitudes according to the specific characteristics of the applied filters. Low-pass filters preserve low-frequency signal components while attenuating higher frequencies. Conversely, high-pass filters preserve high-frequency signal components while attenuating lower frequencies. Band-pass filters preserve a specified frequency range, known as the passband, while attenuating frequencies outside this range. Although both high- and low-pass filters are used [66], [67], band-pass filters are more often used to denoise heart sounds [68]–[70]. Butterworth band-pass filters with different orders and passbands have been successfully employed [71]–[78]. A 4-th-order Butterworth filter with a passband of 25 – 400 Hz was employed in [79]. A 5-th-order Butterworth filter was employed with a passband of 50 – 500 Hz in [80] and 25 – 250 Hz in [81]. A 6-th-order Butterworth filter was employed with a passband of 50 – 950 Hz in [82], [83] and 30 – 900 Hz in [84]. Other linear filters, such as a Savitzky–Golay filter in [85], [86], a Chebyshev low-pass filter in [87], [88], and Notch filter in [89] have also been successfully employed for PCG denoising.

Another very commonly used method for denoising heart-sound signals is based on the **wavelet transform (WT)**, a powerful signal analysis tool with the ability to represent a signal simultaneously in time and in frequency. Wavelet transform decomposes a function (a signal) into a set of wavelets, a wavelike oscillation localized in time. The idea behind the wavelet transform is to calculate how much of a wavelet is contained in a signal for a given wavelet scale and position — also called wavelet coefficients. Different wavelets are shifted over the entire signal, checking the similarities for each time step. In the continuous wavelet transform (CWT), an infinite number of wavelets are shifted across the signal, covering a continuous range of scales and positions to provide a comprehensive analysis. In the discrete wavelet transform (DWT), on the other hand, only a finite set of wavelets is selected. After decomposing a signal, thresholding is performed, where only coefficients above the selected threshold are selected and others are ignored. The selection of a threshold is a delicate work, since a poor thresholding leads to a poor denoising performance. The final step is to reconstruct the signal using the selected coefficients [90]. The WT approches for PCG denoising employed a variety of mother wavelets[2] [91]–[99]. For example, Morlet is used in [100], Coif-4 in [101], Coif-5 in [102], and Daubechies-10 in [103].

Kalman filters are mathematical tools used to remove noise from signals by combining predictions of the signal behaviour with actual measurements. They predict what the signal should look like based on its past behaviour, take into account how the signal is expected to change over time, and then compare that prediction to new measurements of the signal. If the measurement differs from the prediction (e.g. due to noise), the filter adjusts its estimate to be closer to the measurement. Through a mathematical optimization process, the filter determines the best estimate of the true signal, taking into account both the prediction and the measurement [104]. Kalman filters have been employed for PCG denoising by several researchers [105]–[107].

In addition, researchers have proposed numerous other algorithms for PCG signal denoising. Those include empirical mode decomposition [108], adaptive noise canceller [109], tunable-Q wavelet transform [87], variational mode decomposition [110], non-negative matrix factorization [111], blind source separation [112], wavelet packet transform [113], short–time Fourier transform [114], [115], adaptive filtering [116], spike removal [73], [79], deep learning (DL) models for denoising [117], [118], and singular value decomposition [119].

Empirical mode decomposition, unlike other methods such as wavelets that require a predefined basis function, is a fully data-driven method that does not require an a priori defined basis system. In this method, a signal is decomposed into a series of simple oscillatory functions by an iterative procedure, which allows a perfect reconstruction of a denoised signal [108]. Salman et al. [120], [121] compared the denoising methods wavelet transform, empirical mode decomposition, and total variation on heart sounds obtained from the Michigan Heart Sound and Murmur database. Based on extensive simulations, the empirical mode decomposition method consistently showed the best performance in terms of signal-to-noise ratio, root-mean-square error, and percent root-mean-square difference metrics.

2.5.1.2 Segmentation

In traditional ML pipelines for signal analysis, segmentation is a crucial step where the signal is divided into equally-sized segments to facilitate subsequent analysis. This segmentation is often achieved using a sliding window technique, with or without overlap. In the heart-sound analysis, however, the purpose of segmentation is to segment the signal

[2] a specific basic waveform that serves as the building block for generating other wavelets [90]

into individual heartbeats and to identify the locations of the fours states within each heartbeat. Segmentation then allows detection of the presence of abnormal sounds and allows subsequent feature extraction to extract the features of the abnormal sounds. The segmentation can be performed on both the raw signal and the denoised signal.

Accurate segmentation is a major challenge in heart-sound analysis, especially when the signals are contaminated by real-world artifacts. While existing algorithms focus primarily on locating fundamental heart sounds, there is a need to locate abnormal sounds and low amplitude irregularities. In addition, current segmentation methods often rely on absolute measures such as time or frequency distributions, which vary widely between subjects and result in insufficient segmentation accuracy. The segmentation methods proposed by the researchers can be broadly categorized into envelope-based methods, ECG reference-based methods, probabilistic models, feature-based methods, time-frequency and wavelet analysis-based methods and learning-based methods [42], [60]–[64], [122].

Envelope-based segmentation methods aim to identify the boundaries of individual heartbeats by analysing the envelope of the signal. The aim is to identify peaks and valleys in the envelope that correspond to the different states. These methods often use techniques such as peak detection algorithms or amplitude thresholds to locate the prominent features in the envelope. By capturing the temporal variations in signal amplitude, envelope-based methods provide a robust approach to heart-sound segmentation. The primary approaches for extracting the signal envelope include normalized Shannon energy [83], [123]–[126], homomorphic filtering, and Hilbert transform [127]–[129]. There are some limitations to the envelope-based PCG segmentation. Many approaches assume that the systole is shorter than the diastole, which is not always the case for small children and subjects with CVDs [130]. Next, the peaks of the artefacts that overlap with the fundamental heart sounds often cause incorrect segmentation [131]. In addition, medium amplitude peaks (e.g., murmurs) are often attenuated in envelope analysis, while large and low peaks can merge into a single envelope, resulting in very low amplitude peaks not being detected [132].

ECG reference-based methods use an additional ECG signal as a reference to identify the heart states [131], [133]–[136]. The main drawback of such methods is that both PCG and ECG signals are required, which adds complexity to both sensing and synchronization. Moreover, the methods are affected by the slight mismatch between the electrical and mechanical activities of the heart, which also depends on the condition of the patient [137]. In addition, methods that rely on the identification of R and T peaks are computationally expensive. Furthermore, accuracy can be compromised by low amplitude signals and sudden changes in QRS morphologies, making R and T peak identification inherently difficult [42].

In the last decade, the **probabilistic models** have been the main approch to PCG segmentation. They include dynamic clustering [138], Hilbert–Huang transform [139], hidden Markov models (HHM), and hidden semi-Markov models (HSHM). HHMs [140], [141] and HSHMs [142]–[146] predict the most likely sequence of heart-sound states by considering the expected duration of each of the states. Currently, they represent the most successful approaches to PCG segmentation and are often employed in combination with NN segmentation approaches [50], [147]. The most widely used segmentation approach today is the Springer's modification [39] of Schmidt's method [148]. The algorithm uses a HSMM model and Viterbi decoding and provides a state-of-the-art method for segmenting heart sounds. The main disadvantage of HHM and HSHM models is that require many parameters and are therefore computationally intensive and slow.

The **feature-based** methods involve extracting specific features from the signal, such as energy fraction, sample entropy, total variation filtering, Shannon entropy, instantaneous phase boundary, boundary location identification, likelihood computation, and others, to

identify peaks present in the cardiac cycles [132], [149]–[157]. In more recent approaches, researchers have segmented the cardiac signals directly into cardiac cycles and bypassed the steps of identifying the individual positions of the S1 and S2 peaks [82], [83], but this requires prior knowledge of the cardiac cycles. Once features are extracted, they are compared against predefined thresholds or patterns, often derived from expert knowledge or prior studies, to determine the location of heart states. The effectiveness of feature-based segmentation depends on selecting the right features that accurately capture the nuances of heart sounds, which can be challenging due to the variability in heart sounds across different patients and conditions. Additionally, such methods may require extensive computational resources, particularly when dealing with large datasets or complex feature sets.

Methods based on **time-frequency analysis** decompose the signal into time-frequency representations that allow simultaneous capture of temporal and spectral features [56], [123], [158]–[164]. Techniques such as the short-time Fourier transform [165], [166] or the wavelet transform [167], [168] show how the frequency content of the signal changes over time, which facilitates the detection of transient events and frequency variations. Wavelet analysis uses scaled and translated wavelet functions to decompose the signal into different frequency bands, resulting in a multi-resolution representation that captures high-frequency components at fine scales and low-frequency components at coarse scales. These methods effectively capture localized temporal features and frequency content suitable for segmentation of complex heart-sound signals. However, careful selection of parameters and wavelet features is required, and interpretation may require additional complexity compared to other segmentation approaches.

As opposed to the probabilistic models, which incorporate prior knowledge about the PCG signal structure, **learning-based** methods utilize ML algorithms to learn the mapping between the signal and the segmentation boundaries, either directly from the signal or from the extracted features [169]. The learning-based approaches can be unsupervised, meaning that they do not require the labels of the data, and supervised, which means that the segmentation boundary labels are required. In [170], the authors employed an unsupervised learning approach based on spectral clustering to locate the S1 and S2 sounds. Due to the recent advances of DL in image segmentation tasks [171], supervised methods [172] for heart-sound segmentation nowadays mostly rely on DL approaches [147], [173]–[176]

2.5.2 Feature Engineering

In ML and pattern recognition, a **feature** refers to a measurable property extracted from data. Features are numerical representations that capture relevant information about the data, enabling algorithms to learn patterns, make predictions, or perform tasks. They serve as input variables for ML methods and are typically derived through feature-engineering techniques, which involve extracting, selecting features, and normalizing features. Additionally, the **target** variable, representing the outcome of interest, is often considered alongside features during model training and evaluation. The goal of feature engineering is to identify a small number of representative features to replace the high-dimensional raw signals. This reduces data dimensionality, removes redundant or irrelevant information, and transforms the data into a more suitable form for subsequent classification [177].

2.5.2.1 Feature Extraction

Feature extraction in sound analysis can be done with the raw or denoised signal, segmented or not, but is most effective when done with the denoised and segmented signal. There are many established and standardized features for audio analysis and can be split

into several categories: time-domain features, statistical features, frequency-domain features, wavelet features, and deep features. While some literature treats time-domain and statistical features as a single group, most related work treats them as separate categories; therefore, we do so as well.

Time-domain features describe characteristics of the signal in the time domain and provide insight into the temporal dynamics. They include measures such as length, mean, median, minimum, maximum, root-mean-square, zero-crossing rate, energy, power, entropy, periodicity, and autocorrelation.

Statistical features aim to encapsulate the overall distribution and statistical characteristics of the signal, offering insights into its shape and variability. They include measures such as standard deviation, skewness, and kurtosis.

In order to extract the **frequency-domain features**, the signal has to first be transformed into the frequency domain. For this purpose, Fourier analysis is widely used. It states that every periodic signal can be represented as the sum of sine waves of different frequencies. A FFT provides representation of the distribution of the frequency content of sounds, i.e., of sound spectrum. Example of signal decomposition is shown in Figure 2.8.

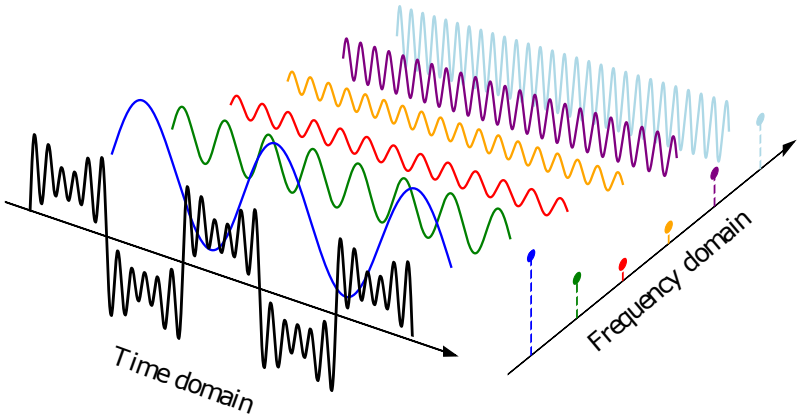


Figure 2.8: Fourier transform: decomposing a signal into its individual time-domain components and transforming it into the frequency domain.

The black signal depicted on the left represents the time-domain view, while the coloured signals on the right represent its individual time-domain components. When represented in the frequency domain, the signal showcases the distribution of its constituent frequency components. Each coloured dot along the frequency axis corresponds to a specific frequency component in the spectrum. Low-frequency components are positioned at the beginning of the axis, while high-frequency components are located toward the end. While the FFT produces complex values that contain both the amplitude and phase information of the signal's frequency components (known as FFT coefficients), we are usually only interested in the magnitude (absolute value) of these coefficients. These magnitudes represent the strength of each frequency component in the signal. The spectrum is as a plot of the magnitudes of the FFT coefficients against their corresponding frequencies.

Frequency-domain features include power spectral density, spectral centroid, spectral bandwidth, spectral constrast, spectral flatness, spectral entropy, spectral roll-off, Mel-frequency cepstrum coefficients (MFCCs), and chroma vector.

In a **wavelet decomposition**, the signal is decomposed into various frequency components, providing decomposition coefficients at different levels. At each level, filters determined by the selected wavelet function are applied to separate the signal into low-pass and

high-pass components. The coefficients of these components can be extracted and used as features. The details to wavelet decomposition can be found in Section 3.3.1.1.4.

Deep features refer to the representations of data that are learned by DL models during the training process. These features are abstract, high-level representations that capture complex patterns and structures in the data. They are automatically learned from the input data by multiple layers of the network, with each layer transforming the data into increasingly complex and informative representations. As a rule, deep features are extracted from the last hidden layer of the NN.

The listed features can also be extracted from the first and the second derivative of a signal, which greatly increases the amount of possible extracted features.

2.5.2.2 Feature Selection

Feature selection methods aim to reduce the input features down to those deemed most useful for predicting the target variable in a model. This serves several purposes. It addresses the "curse of dimensionality" by reducing overfitting and computational complexity. By focusing on relevant features, it improves the performance of the model while minimizing the risk of learning noise in the data. In addition, feature selection simplifies the model, making it more interpretable, and improves computational efficiency by working with a smaller feature set. Feature selection techniques can be split into two categories: unsupervised and supervised [178].

If the target variable is not required during the process of selecting the most informative features, the technique is **unsupervised**. Correlation methods utilize statistical measures to evaluate the interrelationships among features, enabling the selection of the most relevant features through filtering. Unsupervised methods are typically applied after the supervised methods to remove the features that exhibit high correlation. Features with high correlation are considered redundant as they convey similar information, and retaining only one of such similar features suffices to preserve the relevant information.

As opposed to the unsupervised methods, the **supervised** feature-extraction methods require the target variable during the selection process. Supervised methods can be split into three categories: filter, wrapper, and intrinsic methods. The diagram showing hierarchy of the feature selection methods is shown in Figure 2.9.

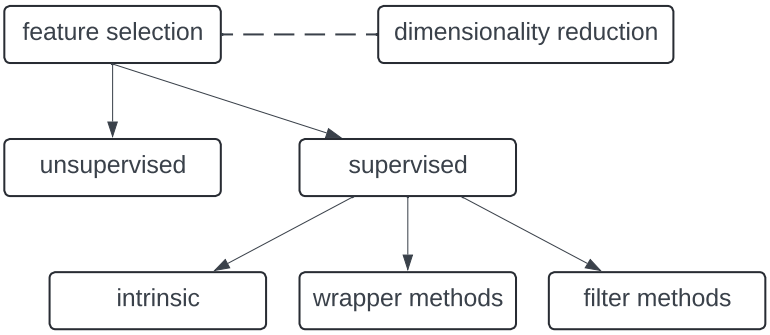


Figure 2.9: Hierarchy of feature selection techniques [179].

Filter methods select the features independently of the model and rely on the data characteristics to evaluate the importance of the features. They are generally more computationally efficient than wrapper methods. However, since they lack a specific learning algorithm to guide feature selection, the selected features may not be optimal for the target model. A filter method usually involves two steps. Firstly, the importance of the

features is evaluated using a predefined scoring metric that compares them with the target, seeking correlation. This evaluation can be univariate, where each feature is evaluated individually, or multivariate, where several features are evaluated simultaneously. Secondly, features with low scores are discarded during filtering [178]. A filter metric depends on the data types of the input features and the output target. Data type can be numeric or categorical. Numeric types include integer (e.g., 1, 52, 100) and float (e.g., 0.1, 3.2, 8.33), whereas categorical types include nominal (e.g., red, green, blue), ordinal (e.g., 1-st, 2-nd, 3-rd), and boolean (e.g., true, false). When the output target is numerical, the task involves regression, whereas for a categorical target, it involves classification.

The popular scoring metrics include Pearson's correlation coefficient, Spearman's rank coefficient, Kendall's rank coefficient, ANOVA correlation coefficient, χ^2 test, and mutual information. All correlation coefficients vary between -1 (perfect negative relationship) and 1 (perfect positive relationship), with 0 meaning no correlation. Pearson's correlation coefficient measures the linear relationship between two sets that can be either discrete or continuous in nature [180]. The Spearman's rank correlation coefficient is a non-parametric measure of correlation that evaluates the extent to which an arbitrary monotonic function can describe the relationship between two variables, irrespective of assumptions about their frequency distribution [181]. Kendall's rank correlation coefficient measures ordinal association between variables, ideal for non-parametric data. It evaluates data point ordering similarity, robust to outliers and distribution assumptions. Because it is calculated from concordant and discordant observation pairs, it avoids the need for normal distribution assumptions [182]. ANOVA correlation coefficient compares means among two or more groups. When examining correlation coefficients, ANOVA becomes valuable for assessing the interrelationships among multiple variables concurrently [183]. The χ^2 test evaluates the independence between categorical variables. It is designed to analyse the discrepancy between observed counts in categorical data and the counts expected under the assumption that the categories are independent [184]. Mutual information quantifies how much knowing one variable reduces uncertainty about another, detecting both linear and nonlinear relationships. It is applicable across various types and distributions of data [185].

Wrapper methods systematically evaluate various models by iteratively modifying features to identify the combination that optimizes model performance. These methods employ a greedy search approach, considering all possible feature combinations against a specified evaluation criterion, which depends on the problem type. For instance, in regression, p-values may serve as an evaluation criterion, while accuracy could be used for classification. The search algorithm stops when the predetermined number of features is selected, or when the predetermined evaluation criterion threshold is met. Ultimately, wrapper methods select the feature combination that yields the best performance for the designated ML algorithm. However, their effectiveness is directly tied to the selection process, and wrapper methods are often computationally intensive compared to filter methods, with a risk of over-fitting. The wrapper selection approaches include forward selection, backward elimination and stepwise selection [178].

In forward selection, the process is initiated with a null model, and each individual feature is sequentially added to the model. Their performance is evaluated based on predefined evaluation criterion. The feature providing the most significant improvement is initially selected. Subsequently, the feature set is expanded by exploring combinations of the previously selected features with the remaining ones until the stopping criteria are met [178].

In backward selection, all features are initially included in the model and then iteratively removed based on the evaluation criterion to identify those that do not significantly contribute to the model. This iterative process continues until the final set of significant

features is obtained [178].

Stepwise selection combines aspects of both forward selection and backward elimination. Similar to forward selection, this method involves adding new features iteratively. However, unlike forward selection, it also evaluates the significance of previously added features. If any of these features are found to be insignificant, they are removed via backward elimination [178].

Intrinsic or embedded methods combine the advantages of both wrapper and filter methods. They incorporate feature interactions while ensuring computational efficiency. These methods iterate through the model-training process, extracting features that contribute the most to each iteration's training. Some of the approaches include L1 regularization, L2 regularization, and decision tree-based methods.

L1 regularization involves imposing a penalty on the model's parameters to constrain its flexibility and prevent overfitting. In linear model regularization, this penalty is imposed on the coefficients associated with each predictor. Among various regularization methods, L1 regularization can effectively reduce some coefficients to zero, thereby eliminating certain features from the model [186].

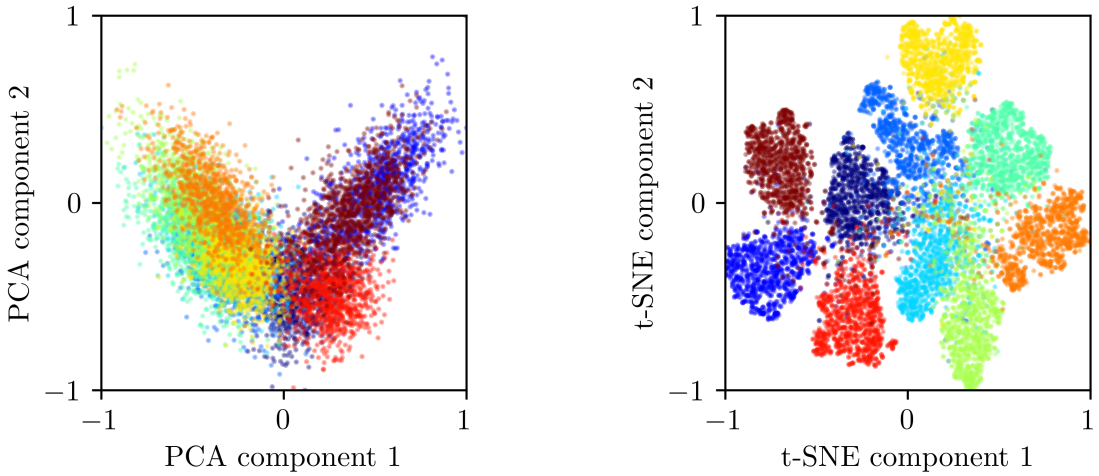
L2 regularization is a technique used to prevent overfitting in ML models by adding a penalty term to the model's loss function. This penalty term is proportional to the square of the magnitude of the coefficients associated with each predictor, thus encouraging smaller coefficient values. As a result, L2 regularization tends to shrink the coefficients towards zero, but unlike L1 regularization, does not usually bring them exactly to zero. This helps to reduce the impact of irrelevant or redundant features in the model while retaining all features in the final model [187].

Decision tree-based methods involve using decision trees to automatically select relevant features during the training process. These methods work by recursively partitioning the feature space based on the predictive power of each feature. Features that contribute more to reducing impurity or increasing information gain at each split are favoured and retained in the final model, while less important features are pruned away. Such methods include decision tree, random forest, XGBoost and LighGBM [179]. The models are thoroughly described in Section 2.5.3.1.

In addition to feature selection methods, **dimensionality reduction** techniques offer alternative approaches for reducing the data's dimensionality. The outcome of these techniques aligns with that of feature selection as they both reduce the number of input features. However, the main difference is that the output features of dimensionality reduction techniques are not interpretable as they are computed as combinations of the original features and are thus mainly used as high-dimensional data visualization techniques. Two widely employed dimensionality reduction techniques are the Principal Component Analysis (PCA) and the t-Distributed Stochastic Neighbor Embedding (t-SNE). Comparison of PCA and t-SNE employed on the same high-dimensional publicly available dataset consisting of 10 classes is given in Figure 2.10.

PCA is a deterministic linear transformation method that transforms the data into a new coordinate system. In this new system, the principal components, representing the transformed variables, are linear combinations of the original variables. These components are uncorrelated, and the highest variance is aligned with the first coordinate, followed by subsequent coordinates capturing decreasing variances [189].

On the other hand, t-SNE is a randomized non-linear method. It works by modeling the relationships between data points in the high-dimensional space and then representing these relationships in the lower-dimensional space. In essence, t-SNE aims to preserve the local structure of the data, ensuring that similar data points are placed close together in the lower-dimensional space while dissimilar points are placed farther apart [190]. Compared



(a) PCA (b) t-SNE

Figure 2.10: Comparison of dimensionality reduction techniques on the CIFAR10 dataset consisting of 10 classes [188]. PCA and t-SNE were used to reduce hidden features from a pretrained NN model to two components, with the class clusters still visible.

to PCA, t-SNE is much more computationally expensive.

2.5.3 Classification

ML algorithms are computer techniques developed to help machines or computer systems learn patterns, make predictions or perform specific tasks without the need for explicit programming. They enable machines to learn on their own and improve their performance based on data or experience [191]. From the ML point of view, detection of heart-sound abnormalities is a classification problem. A classifier maps the input data to a target category from a predefined set of possibilities.

2.5.3.1 Classical Machine Learning

Classical ML methods are termed "classical" because they have been foundational in the development of ML and have a long history of use before the advent of DL and NNs. They are simpler and less computationally intensive compared to NNs. These methods require significant feature engineering, where domain knowledge is utilized to create relevant features from raw data. Classical ML methods are often utilized due to their interpretability and lower computational requirements compared to NNs.

In the following, we describe the classical ML methods used in this thesis. Only probabilistic classifiers were chosen because they provide classification labels along with probability estimates of class membership, which are crucial for understanding prediction confidence and uncertainty in the medical domain.

2.5.3.1.1 Decision Tree Classifier A decision tree (DT) is a supervised machine learning algorithm that makes predictions by recursively partitioning the data into subsets based on feature values. This process is visualized as a tree structure, where each internal node represents a decision point on a specific feature, each branch represents an outcome of that decision, and each leaf node represents a final class prediction. The algorithm aims

to achieve maximum "purity" at each leaf node, ideally grouping instances of a single class. At pure leaf nodes, prediction probabilities reach 1.0, indicating complete confidence. For partially grown trees, probabilities are calculated as the proportion of instances in the leaf node that belong to the predicted class relative to the total number of instances at that node [191].

DTs are considered "white-box" models due to their interpretability, allowing straightforward tracing of decision paths. However, they are susceptible to over-fitting and variance, especially with small datasets. Pruning techniques and restrictions on tree depth can mitigate these issues by removing or limiting branches that add complexity without improving predictive accuracy [191].

2.5.3.1.2 Random Forest In addition to pruning and depth limitations, ensemble methods are effective strategies for reducing the variance of DTs. Ensemble methods combine predictions from multiple models (base learners) to improve overall accuracy and robustness. One well-known DT-based ensemble model is the random forest (RF). In RF, multiple DTs are trained on different bootstrapped samples - randomly selected subsets with replacement from the original training data. Further, at each split within each tree, only a randomly selected subset of features is available for consideration, which intentionally limits the features that each tree can use at each decision point. This approach of bootstrapping and randomized feature selection reduces correlation between individual trees, thereby increasing model diversity and reducing the risk of over-fitting. The final prediction in an RF is typically determined by a majority vote across all trees, which generally enhances predictive accuracy and stability over that of a single DT [191].

2.5.3.1.3 Gradient Boosting Classifier Another example of a DT-based ensemble model is the gradient boosting (GB) classifier. GB constructs an ensemble by sequentially adding DTs, where each tree is trained to minimize the residual errors from the previous tree. Specifically, the first tree is fit to the training data, and then the second tree is trained on the residuals of the first model's predictions, calculated as $\hat{y} - y,$, where $\hat{y}$ represents the true labels and y the initial predictions. This process is repeated, with each subsequent tree addressing the errors left by its predecessor. The final model's predictions are made by summing the outputs of all the trees, resulting in a refined prediction that corrects cumulative errors across iterations [191].

GB has evolved to include optimized implementations that enhance its performance and computational efficiency. Extreme gradient boosting (XGB) integrates advanced features such as parallelization, tree-pruning, and regularization, and is built on a distributed computing framework to accelerate training and reduce memory requirements [192].

Another optimized variant, light gradient boosting machine (LGBM), emphasizes speed and scalability, employing a histogram-based approach for tree construction instead of the traditional depth-first method. This approach makes LGBM particularly effective for handling large datasets with many features and supports parallel and distributed computing for even greater efficiency [193].

2.5.3.1.4 Support Vector Machine A support vector machine (SVM) is a supervised learning model that aims to find an optimal hyperplane in an N-dimensional feature space, effectively separating data points from different classes with maximum margin. The margin is defined as the distance between the closest data points (support vectors) of each class to the hyperplane, and maximizing it contributes to better generalization.

For cases where classes are not linearly separable, SVM incorporates two key strategies: the soft margin and the kernel trick. The soft margin approach allows SVM to accommo-

date some misclassifications to balance the trade-off between maximizing the margin and minimizing classification errors. The kernel trick, on the other hand, enables SVM to handle non-linear separations by transforming data into a higher-dimensional space where the classes become linearly separable. This is achieved by applying a kernel function, such as a polynomial or radial basis function, to project the original features into a more complex feature space. Once separated linearly, the data can be mapped back to the original space for interpretability. An illustration of the kernel trick's effect on a non-linear separation is provided in Figure 2.11.

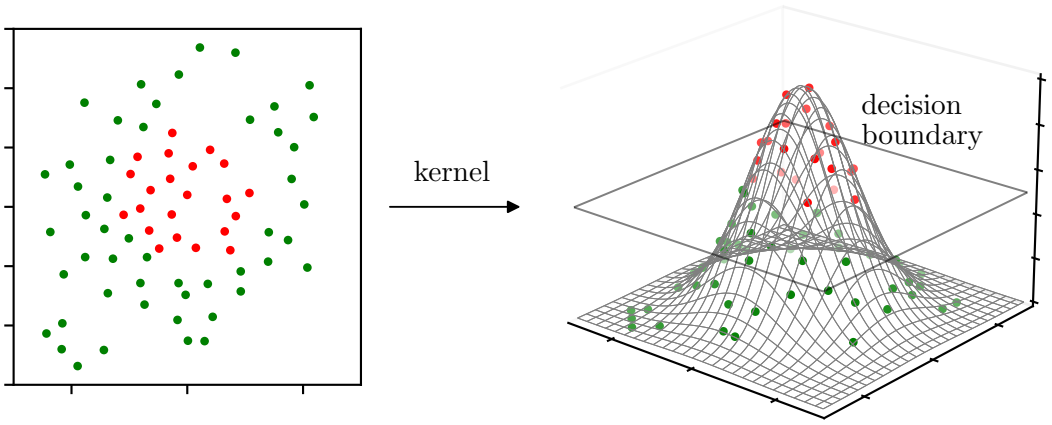


Figure 2.11: Kernel trick in SVM.

Although SVM is binary (not probabilistic) classifier, calibration methods, e.g., Platt scaling, can be used to convert the outputs to class probabilities [193].

2.5.3.1.5 K-Nearest Neighbor The k-nearest neighbor (kNN) classifier is a non-parametric, instance-based learning algorithm that operates without an explicit training phase. Instead of training a model, kNN stores the entire training dataset in memory. When a new instance is introduced, the algorithm identifies the k training samples that are closest to the new instance, based on a specified distance metric. Common distance measures employed in kNN include Euclidean distance, Manhattan distance, and Minkowski distance, among others. Once the k nearest neighbors are identified, the classifier assigns a class label to the new instance through a majority voting mechanism, where the class most frequently represented among the k neighbors is chosen as the prediction.

Although kNN is not a probabilistic classifier by design, it is possible to derive a probabilistic interpretation of its outputs. For instance, the classifier can calculate the probability of an instance belonging to a particular class as the ratio of the number of neighbors belonging to that class to the total number of neighbors considered. This characteristic allows kNN to provide not only class predictions but also insights into the confidence of those predictions [194].

2.5.3.1.6 Gaussian Naive Bayes The Gaussian naive Bayes (GNB) classifier is a variant of the naive Bayes algorithm, which is based on Bayes' theorem. It assumes that the features are independent and have a Gaussian (normal) distribution. In GNB, each feature is assumed to be normally distributed within each class, and the mean and standard deviation of the distribution are computed from the training data. During the classification process, GNB employs Bayes' theorem to compute the posterior probability of an instance belonging to each class based on its feature values. The classifier calculates this probability

by multiplying the prior probability of the class—derived from the training data—with the likelihood of observing the feature values under the assumption of a Gaussian distribution for that class. The instance is subsequently assigned to the class that yields the highest posterior probability, thereby allowing for effective classification while leveraging the assumptions of feature independence and normality [194].

2.5.3.1.7 Logistic Regression Logistic regression (LR) is a classification algorithm that estimates the probability of an instance belonging to a specific class. It functions similarly to linear regression, wherein it computes a weighted sum of the input features. However, instead of predicting a continuous output, LR utilizes the logistic function—commonly referred to as the sigmoid function—to transform this weighted sum. The logistic function is mathematically expressed as $\sigma(t) = \frac{1}{1+\exp(-t)}$, where t represents the weighted sum of input features. This transformation constrains the output to a range between 0 and 1, effectively representing the probability that an instance is classified as belonging to the positive class. In practice, the decision boundary for classification is typically established at a threshold of 0.5. Instances with a predicted probability exceeding this threshold are classified as belonging to the positive class, while those with a predicted probability below this threshold are assigned to the negative class. This probabilistic framework enables LR to provide not only class predictions but also an associated measure of confidence in those predictions, making it particularly valuable in various applications requiring interpretability and probabilistic outputs [193].

2.5.3.1.8 Stohastic Gradient Descent Stochastic gradient descent (SGD) classifiers are linear classifiers, such as SVM and LR, optimized using the stochastic gradient descent algorithm. SGD computes the gradient of the loss function with respect to model parameters iteratively, updating them based on these gradients. The learning rate, which determines the size of each update, typically follows a decreasing schedule to enhance convergence. In contrast to traditional gradient descent, which calculates gradients over the entire training dataset, SGD selects random individual training instances at each iteration. This stochastic approach allows for more frequent updates and improves computational efficiency, making it suitable for large-scale datasets. However, the inherent randomness of SGD may cause fluctuations in the loss function during training as the algorithm converges toward the minimum. This variability is a characteristic feature of the stochastic nature of the method, as illustrated in Figure 2.12.

Among the most commonly employed classical ML methods for heart-sound classification using extracted features from PCGs are SVM, RF, kNN, and GB models [195]–[203]. In [196] and [197], the authors extracted the deep features from the latent space of pre-trained NN classifiers to be used as input to the SVM.

2.5.3.2 Deep Learning

NNs are inspired by the structure of the human brain, featuring neurons and synaptic connections. While this analogy is mostly conceptual, it is the basis for their name. They consist of layers of interconnected nodes (neurons), including an input layer, one or more hidden layers, and an output layer. Each connection has an associated weight that is adjusted during training. The term "deep learning" refers to NNs with many layers.

NNs automatically learn features from raw or preprocessed data through training, especially in DL, where multiple layers learn hierarchical data representations. Training involves backpropagation and gradient descent to minimize a loss function, requiring large datasets and significant computational resources. They are often called "black box" models

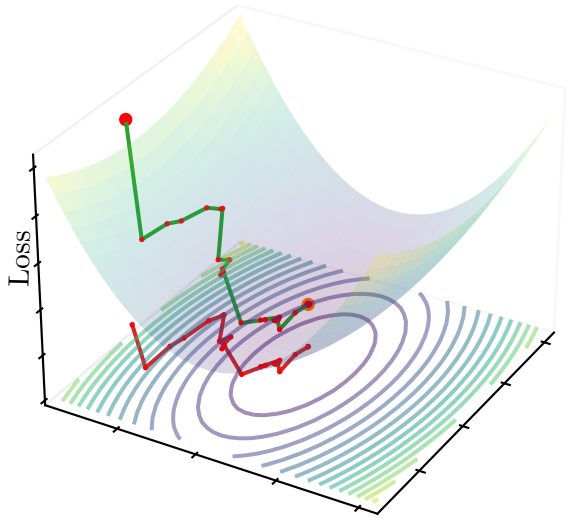


Figure 2.12: Visualization of the stochastic gradient descent method.

due to their complexity and the difficulty of interpreting them. However, various methods exist that can help interpret these models to some degree. NNs are well-suited for tasks like image and audio recognition, natural language processing, and playing games, as they can capture intricate patterns and relationships within the data that classical models might miss. Their ability to generalize from large amounts of data makes them particularly powerful for these applications.

Although NNs are sometimes used for classification with extracted features as input, they are most commonly employed with either a time-series signal or its spectrogram as the input. The spectrogram is a time-frequency representation that can be computed by applying the FFT to expand the dimensionality of a time-series signal. Initially, the time-series signal is divided into short, overlapping segments using a sliding-window technique. The length of each segment is determined by the window size, while the overlap between consecutive segments is specified by the hop length. Subsequently, the FFT is computed for each segment to convert the time-domain data into the frequency domain. The power spectrum is then computed by taking the square of the magnitude of the FFT outputs. The resulting power spectra of all segments are then combined over time to construct the spectrogram. The x-axis represents time, the y-axis represents frequency, and the value indicates the magnitude of the spectrum at each time-frequency point. The dimension of the spectrogram is equal to the number of windows times the number of frequency bins. An example spectrogram of a PCG is shown in Figure 2.13.

When features extracted from PCGs are used as input to NNs, the models are typically fully connected networks, fully convolutional networks, or long short-term memory (LSTM) networks [143], [168], [204], [205]. When spectrograms are used as input, 2-dimensional CNNs are most often used [206]–[213]. In [208] and [209], the frequency information was converted to the Mel-scale to approximate the human ear's nonlinear perception of distinct frequencies. Similarly, for time-series data input, 1-dimensional CNNs are most often employed [214]–[223]. The authors of [224] used deep features extracted from variational autoencoder (VAE) as input to 1-dimensional CNN.

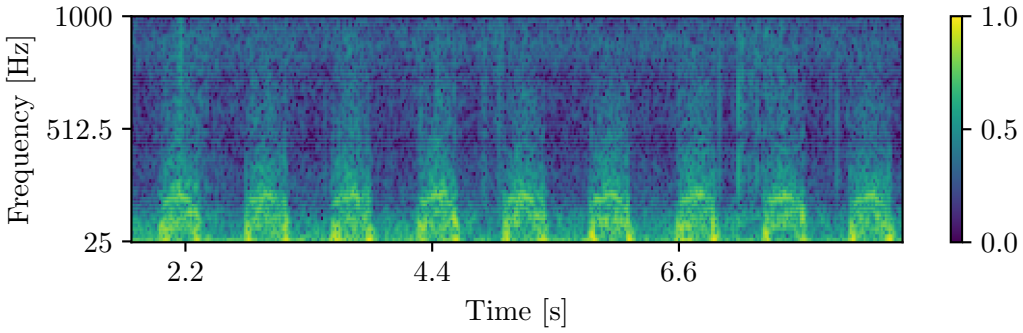


Figure 2.13: Mel-scaled spectrogram of an (unsegmented) PCG.

2.5.4 Chronic Heart Failure Detection Studies

Extensive lists of works on ML for detecting CVDs from heart sounds can be found in various review papers [23], [42], [60]–[64], [122], [225]. This thesis focuses on detecting decompensation episodes in CHF patients. We list works that address CHF classification specifically. Unlike other studies that generally classify heart sounds as normal or abnormal, this list is confined to those where the abnormal sounds are associated with CHF, its subtypes, or its phases. These works are also summarised in Table 2.3. In [23], the authors describe works that address detection of CHF, classification of subtypes, severity estimation, and prediction of adverse events such as decompensation, rehospitalization, and mortality using ML models, but they use predictive features other than heart sounds. In [226], the authors list literature on CHF detection from ECG, HRV measures, clinical and laboratory data, and demographics data. In addition, CHF-detection studies from heart sounds that do not use ML methods include [227].

Gjoreski et al. [228] collected PCG recordings from 23 subjects diagnosed with CHF and 99 healthy subjects. The recordings were filtered using a low-pass Butterworth filter with a 1000 Hz threshold. For segmentation, they employed a sliding window technique with a window length of 1 s and a 50% overlap. From each segment, they extracted 1582 general audio features from both the time and frequency domains using the OpenSmile tool. These features served as inputs for training seven different classifiers: J48, Naive Bayes, kNN, SVM, RF, Bagging, and Boosting. To aggregate the predictions from all segments for each classifier, they computed the minimal, maximal, and average values of the predicted probabilities for the CHF class, resulting in 21 new features (3 features per classifier across 7 models) for each PCG recording. These aggregated features were then used as inputs to a recording-based ML classifier to predict the final outcome for each recording. The evaluation of their approach was conducted using the leave-one-subject-out (LOSO) cross-validation (CV) method, where each experiment was repeated as many times as there were subjects, with one subject always used as the test set and the others as the training set. This method let to the accuracy of 96%. This was a preliminary study that was improved in the following study.

In their improved study, Gjoreski et al. [206] expanded their data collection to include PCG recordings from 22 CHF patients in both decompensated (upon hospitalization) and recompensated (upon discharge) phases. The recordings were filtered with a low-pass Butterworth filter at a 1000 Hz threshold and segmented using a sliding window technique. Their methodology integrated both classical ML and end-to-end DL approaches. For the classical ML approach, features were extracted using the OpenSmile tool from both segments and entire recordings, ensuring that segment features included corresponding record-

ing features. They employed mutual information to select the top-ranked features and used a RF classifier for classification. In the end-to-end DL approach, they implemented a spectro-temporal ResNet model with two branches: one processing the raw PCG signal in the time domain and the other working with the spectrogram. The final classification for each PCG recording was determined by a recording-based classifier that averaged input features for the classical ML model, the output of the classical ML model, the output of the end-to-end approach, and a hidden layer representation from the end-to-end model. Their method was validated using subsets of the PhysioNet database. Using their own dataset, they achieved an accuracy of 86.3% for classifying normal versus CHF recordings. Additionally, they developed a DT to differentiate between the two CHF phases using LOSO CV, achieving an accuracy of 93.3%.

In the work by Gao et al. [229], the authors collected data from 42 HFrEF patients, 66 HFpEF patients, and 1286 random recordings from healthy subjects sourced from the PhysioNet database. All PCG recordings were down-sampled to 600 Hz. They utilized Springer's segmentation algorithm to pinpoint the onset of the S1 heart sound and segmented the recordings into 1.6 s segments, each starting with a consecutive S1 sound. The amplitudes of these segments were normalized between 0 and 1. This pre-processing resulted in datasets comprising 7670 segments labeled as HFrEF, 7710 as HFpEF, and 7740 as normal. The authors compared the performance of various models, including fully connected network (FCN), SVM, LSTM, and gated recurrent unit (GRU), using a 10-fold CV approach. The GRU model achieved the highest accuracy at 98.92%, followed by the LSTM at 96.29%, the FCN at 94.65%, and the SVM at 87.62%.

In the paper by Liu et al. [205], the authors collected 401 PCG recordings from healthy subjects and 441 PCG recordings from CHF patients with HFpEF. They down-sampled the recordings to 2205 Hz and applied wavelet denoising techniques to remove noise. Segmentation was performed using Springer's segmentation algorithm. The authors extracted 12 features describing the multifractal characteristics of the time series, which were used as inputs to two classification models: a single hidden layer feedforward NN called extreme learning machine (ELM) and SVM. The ELM model achieved an accuracy of 96.32%, which was 2.31% higher than the SVM model. Additionally, ELM proved to be 1.38 times faster than SVM.

In their study, Zheng et al. [195] collected PCG recordings from 88 healthy subjects and 64 CHF patients, including 36 with HFrEF and 28 with HFpEF. The recordings were amplitude-normalized between -1 and 1 and denoised using wavelet denoising. Features were extracted from both the time and frequency domains, and cardiac reserve (CR) indices were also utilized. These indices provide information about the difference between the heart's current pumping rate and its maximum capacity. To classify the recordings into CHF and normal groups, the authors implemented three models: SVM, HMM, and FCN. The SVM demonstrated the best performance with an accuracy of 95.39%, significantly outperforming HMM (81.59% accuracy) and FCN (84.21% accuracy).

In their paper, Zheng et al. [230] collected PCG recordings from 275 subjects, including 51 healthy individuals and 224 CHF patients. The CHF patients were categorized into four stages of heart failure by experienced cardiologists: stage A (at risk), stage B (pre-heart failure), stage C (symptomatic heart failure), and stage D (advanced heart failure). The recordings were down-sampled to 1000 Hz and denoised using a high-pass Butterworth filter with a 10 Hz threshold frequency, followed by noise cancellation via singular value decomposition. The recordings were normalized and segmented using Springer's segmentation algorithm. They extracted time and frequency features from three data types: PCGs, heart-sound sub-sequences, and heart-sound sub-band signals. The sub-sequences and sub-band signals were generated from the PCGs. L1 regularization was employed to select the

most informative features. The authors compared three ML methods, SVM, RF, and a deep belief network (DBN), to classify the heart failure stages and healthy subjects. A DBN is a kind of deep NN, which is composed of stacked layers of restricted Boltzmann machines. In a 5-fold CV, the SVM achieved the highest accuracy at 82.0%, followed by RF at 79.5%, and DBN at 74.3%.

Table 2.3: Literature on CHF detection from heart sounds using ML methods. Unless stated otherwise, the datasets were collected by the authors.

Study	Target	Features	Classifier	Dataset	Classification accuracy
Zheng (2022) [230]	4 CHF stages, normal	time, frequency	SVM, RF, DBN	224 CHF, 51 healthy subjects	82.0%, 79.5% , 74.3%
Gjoreski (2020) [206]	CHF, normal; decomp., recomp. CHF phases	time, frequency	RF, ResNet; DT	52 decomp., 22 recomp., 159 healthy recordings; 76 min	86.3%; 93.3%
Gao (2020) [229]	HFrEF, HFpEF, normal	-	FCN, SVM, LSTM, GRU	42 HFrEF, 66 HFpEF subjects; 1286 healthy recs. (PhysioNet)	98.92%, 96.29%, 94.65%, 87.62%
Liu (2019) [205]	HFpEF, normal	time	ELM, SVM	441 HFpEF, 401 healthy subjects	96.32%, 94.01%
Gjoreski (2017) [228]	CHF, normal	time, frequency	J48, Naive Bayes, kNN, SVM, RF, Bagging, Boosting	99 healthy, 23 CHF subjects	96%
Zheng (2015) [195]	CHF, normal	time, frequency, CR indices	SVM, HMM, FCN	64 CHF, 88 healthy subjects	95.39%, 81.59% , 84.21%

2.6 Data Augmentation

One of the best ways to tackle limited data availability is by using data augmentation, which involves applying transformations to the original data to generate new data. Data augmentation is defined in Definition 2.2.

Definition 2.2 (Data augmentation). *Data augmentation* is the process of creating new data by transforming the original data for the purpose of model training [231].

In this thesis, we focus on developing a heart-sound-specific data-augmentation technique to be used during DL model training for improved performance and better gen-

eralizability. Generally, techniques that modify the learning process of an algorithm are considered regularization techniques, as per Definition 2.3.

Definition 2.3 (Regularization technique). A *regularization technique* is any modification made to a learning algorithm that is intended to reduce its generalization error but not its training error [231].

The generalization error, also called the test error is defined as the expected value of the error on a new input. In the context of DL, the regularizations most often employed include early stopping, ensemble methods, dropout, adversarial training, and data augmentation [231].

When augmenting data, it is crucial to apply transformations that do not alter the correct class. For instance, in optical character recognition tasks, transformations like horizontal flips or 180° rotations are inappropriate because they can turn a "b" into a "d" or a "6" into a "9". Additionally, transformations should not be too intensive, as this can create unrealistic instances that do not reflect real-world scenarios. Properly applied data augmentation helps enhance the model's robustness and generalization without requiring additional real-world data [231].

Although there are many approaches to generating synthetic data, which is generating new instances by utilizing the underlying distribution of the original dataset, such as generative models [231], autoencoders [231], generative adversarial networks (GANs) [232], and variational autoencoders (VAEs) [233], this thesis focuses on data augmentation, which involves applying transformations to the original data to create new variations. For heart sounds, data-augmentation techniques can be applied to both time-series data and spectrograms, drawing from general audio- and image-augmentation domains. These techniques can be categorized into two main types: those that generate new instances from a single instance and those that create new instances by combining multiple (two or more) instances. The hierarchy of strategies, including some of the most common techniques, is shown in Figure 2.14

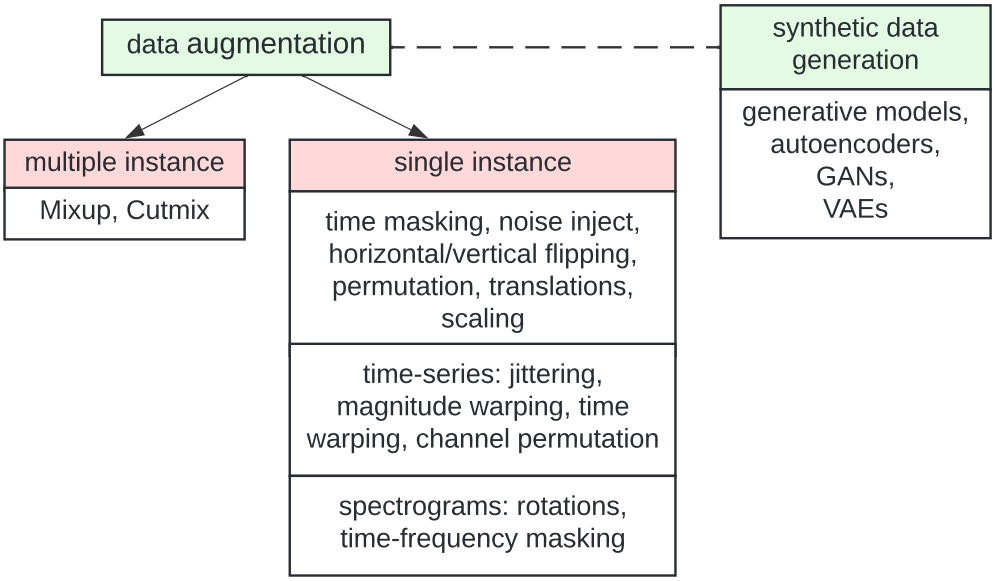


Figure 2.14: Hierarchy of strategies for generating new data. The listed single instance techniques are those most commonly found in literature for heart-sound classification, whereas the listed multiple instance and synthetic data generation techniques are more general [207], [209], [213], [234]–[238].

2.6.1 Studies on Heart-Sound Classification with Data Augmentation

In the context of heart-sound classification using ML, numerous existing studies employ data-augmentation strategies. In the following, we list and describe a selection of those works. The list is limited to those that focus on heart-sound augmentation specifically and those that were published in recent years. The works are also given in Table 2.4.

In their paper, Mishra et al. [235] used codec data augmentation to improve the classification of a CNN model on the Yaseen heart-sound dataset. Codec augmentation involves passing audio recordings through various audio codecs to introduce distortions and artifacts similar to those encountered in real-world audio transmission and storage. The authors simulated the Opus (OGG) codec at bitrates of 4.5k, 5.5k, and 7.7k, which increased the data to 4000 PCGs from the original 1000. Their experiments showed that codec data augmentation is effective in getting around the data limitation.

In the study by T. Koike et al. [213], the authors investigated data augmentation on the PhysioNet database for heart-sound classification. They applied magnitude scaling, random trimming, and respiratory scaling to the time-series signals before converting them into spectrograms. Subsequently, they employed time masking, frequency masking, and Cutout (an image-augmentation technique where a patch of pixels is masked out [239]) on the resulting spectrograms. The implemented data-augmentation techniques substantially outperformed the no-augmentation baseline.

In the research conducted by G. Zhou et al. [209], different combinations of augmentations were analysed on the PhysioNet database to improve heart-sound classification. The authors applied pitch shifting, time warping, and noise injection to the time-series signals. They also transformed the signals to Mel-spectrograms and employed frequency masking, time masking, horizontal flipping, vertical flipping, and colour transformations. Using a CNN model, they found that horizontal flipping of the spectrogram alone improved performance the most compared to other augmentation techniques and their combinations.

In the study by Y. Jeong et al. [236], the authors used the PASCAL and PhysioNet databases and collected their own data to perform heart-sound classification using a CNN. They applied six different augmentations to abnormal sounds to create six additional instances. In the time-series domain, they used noise injection, vertical flip, and time shift. In the spectrogram domain, they applied time masking, frequency masking, and salt & pepper noise injection (the masking of random points in an image). They found that training the model with both original and augmented data diminished performance compared to using only the original data. However, when data augmentation was combined with data normalisation and segmentation, the performance improved.

M. Habijan et al. [207] used the PASCAL Dataset and LSTM to classify heart sounds. They found that implementing noise injection, time stretching, and time shifting as augmentations for the time-series signals significantly improved the classification accuracy. They extracted MFCC features from the augmented signals, which were then used as the input for classification.

In their work, N. Baghel et al. [222] utilized the Yaseen heart-sound dataset and CNN to classify heart sounds. They argued that common audio-augmentation techniques, such as pitch changing, speed changing, and time shifting, are not suitable for PCGs. Instead, they found that noise injection is a more appropriate augmentation method for PCGs. Consequently, they used a background deformation technique to add noise to the signals before converting them into spectrograms for classification. The augmentation and no-augmentation results were very similar, although the augmentation results were slightly better in some metrics.

In their work, M. H. Asmare et al. [240] utilized the Yaseen heart-sound dataset and CNN for classification task. They experimented with time shifting and time stretching

augmentations applied to the spectrograms. However, they found that these augmentation techniques resulted in slightly worse performance compared to using unaugmented training data.

Al-Issa et al. [237] utilized the Yaseen dataset and PhysioNet database to evaluate their diagnostic system, which combines CNN and LSTM components, for automated diagnosis of PCGs. They compared the performance of their system when trained on original data versus augmented data, employing techniques such as time warping, time shift, noise injection, and magnitude scaling. Their findings indicated that data augmentation improved PCG classification performance.

In the work of Lu et al. [238], the authors utilized the CirCor DigiScope dataset and a CNN. They employed noise injection in time domain and time-frequency masking in spectrogram domain. They ranked 1-st and 10-th in the "murmur detection" and "clinical outcome" categories of the George B. Moody PhysioNet Challenge 2022, respectively.

Table 2.4: A selection of works on heart-sound classification with data augmentation.

Study	Dataset	Time-series transforms	Spectrogram transforms	Classifier input modality
Mishra (2023) [235]	Yaseen	Codec	-	Time series
Habijan (2023) [207]	PASCAL	Noise injection, time stretching, time shifting	-	Spectrograms
Al-Issa (2022) [237]	Yaseen, PhysioNet	Time warping time shift noise injection, magnitude scaling	-	Time series
Lu (2022) [238]	CirCor DigiScope	Noise injection	Time-frequency masking	Spectrograms
Zhou (2022) [209]	PhysioNet	Pitch shifting, time warping, noise injection,	Time masking, frequency masking, horizontal flipping, vertical flipping, colour transformations	Spectrograms
Koike (2021) [213]	PhysioNet	Magnitude scaling, random trimming, respiratory scaling	Time masking, frequency masking, Cutout	Spectrograms
Jeong (2021) [236]	PhysioNet, PASCAL, collected by authors	Noise injection, time shift, vertical flipping	Time masking, frequency masking, noise injection	Spectrograms
Baghel (2020) [222]	Yaseen	Background deformation	-	Spectrograms

2.7 Explainable AI

Before moving on to the next chapter, it is important to discuss the concept of explainable AI (XAI), which plays a crucial role in AI-driven medicine. In the medical domain, concepts such as explainability, transparency, robustness and generalization are essential to ensure trust and effectiveness. **Explainability** specifically refers to the ability to understand how a model makes its decisions, which is crucial for medical professionals who need to understand the rationale behind AI-driven diagnoses or treatment recommendations. Without this clarity, medical professionals may be hesitant to rely on AI tools, especially in situations where patient care is at stake.

Transparency means making the internal workings of an AI model accessible and understandable. This is crucial in medicine to verify that AI decisions are based on sound medical reasoning. Transparent models such as DTs are easier to interpret, while more complex models such as NNs often work as a "black box", making it more difficult to understand and trust their decisions.

Robustness in AI means that the models perform reliably even with different data, conditions, patient populations and data collection methods. A robust AI model should be effective across different hospitals and patients, which is crucial in the medical field due to the variability of patient demographics, ensuring consistent performance in different settings.

Generalization is the ability of an AI model to perform well on new, unseen data. In healthcare, this is crucial as medical models need to perform accurately across different patient types and conditions, not just the specific dataset they have been trained on. Good generalization ensures that AI tools can be used broadly in real-world clinical settings.

Together, these concepts ensure that AI systems in medicine are reliable, ethical, and effective, ultimately enhancing trust in AI technologies and improving patient outcomes. In Section 4.2.8, we deal with the explainability of NNs for PCG classification, in particular the extraction of saliency maps.

Chapter 3

Materials and Methods

In this chapter, we present the materials and methods for the two-part study that comprises this thesis: the detection of decompensation episodes of CHF and the development of a heart-sound data-augmentation technique for heart-sound classification with limited data. The methodology pipeline of our study is given in Figure 3.1.

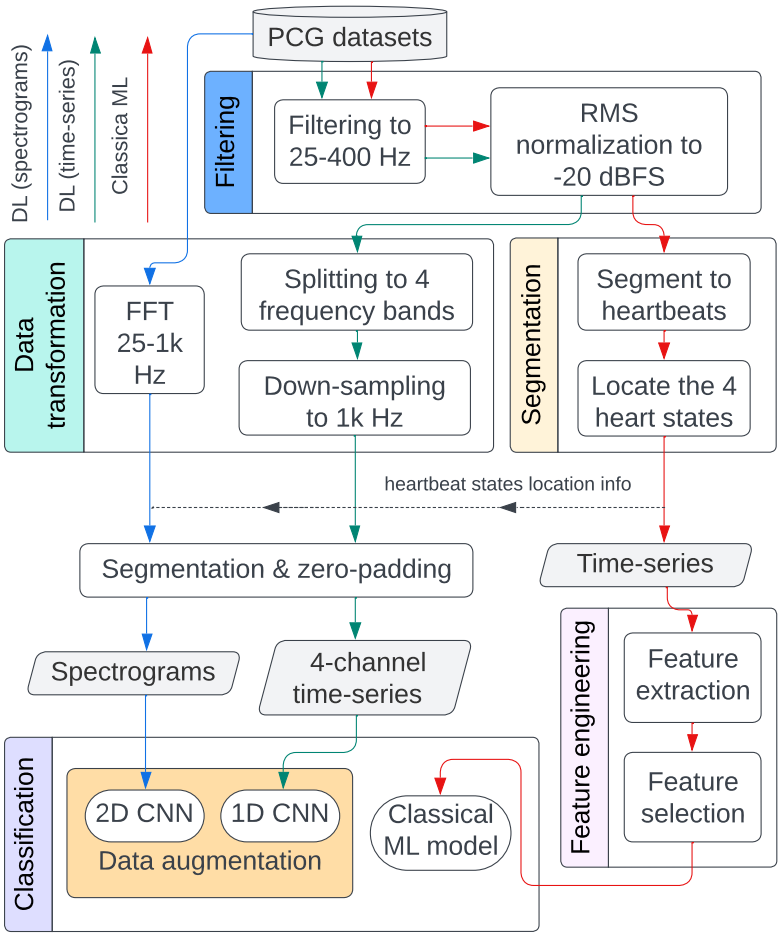


Figure 3.1: Diagram illustrating the methodological pipelines employed in our experiments: the blue arrows indicate the DL pipeline using spectrograms and 2D CNN, the green arrows represent the DL pipeline using multi-channel time-series data and 1D CNN, and the red arrows denote the classical ML approach using manually extracted features.

Section 3.1 describes the datasets used in our study. For the detection of decompensation episodes of CHF, the dataset was collected by cardiology experts at the University Medical Centre Ljubljana (UMCL). For the development of the heart-sound augmentation technique as DL regularization strategy, a subset of the PhysioNet/CinC Challenge 2016 Database was used. The pre-processing steps employed in this study, which involve the filtering and normalisation, data transformation, and segmentation of heart sounds, are detailed in Section 3.2. Section 3.3 describes the methods used for developing the pipeline for detecting decompensation episodes of CHF, whereas Section 3.4 explains the methods applied to develop the heart-sound data-augmentation technique. Finally, Section 3.5 lists the metrics used to evaluate the performance of our classification tasks.

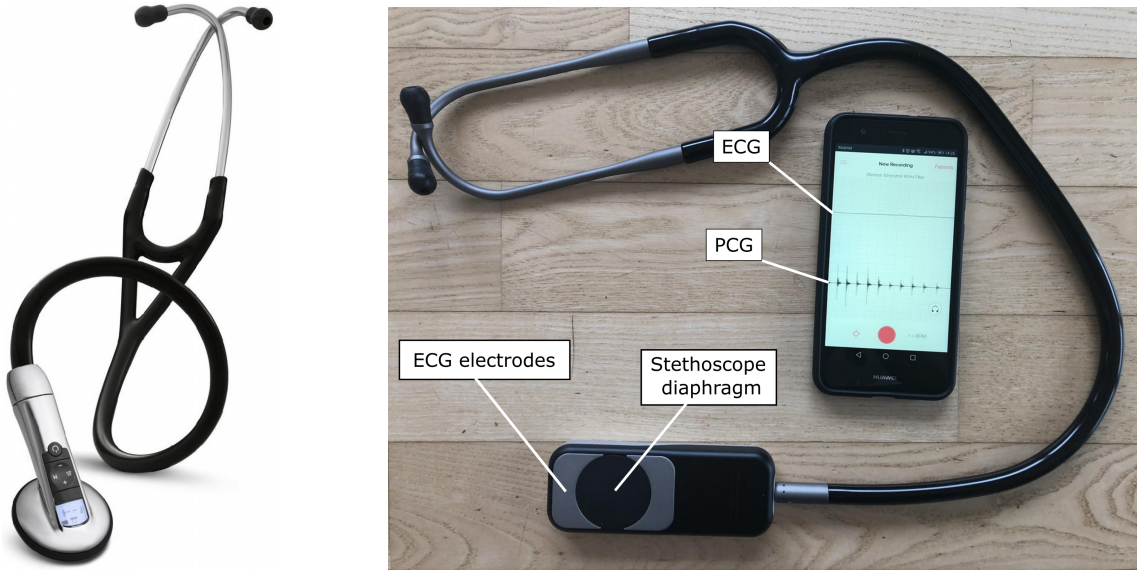
3.1 Datasets

This section covers each of the heart-sound datasets used in our study in enough detail for the reader to understand their properties and to interpret the results. It describes recording equipment, recording positions, recording environment, sampling rate, number of patients, number of recordings, duration of the recordings, and patient demographics. The first dataset, the UMCL, was collected by ourselves and our partners, whereas the second dataset, the PhysioNet, is a public-domain dataset [7], [46].

3.1.1 UMCL

The UMCL dataset consists of PCGs of 37 CHF patients (average age of 51.3 $\pm$ 13.3 yr). The dataset was obtained by two different setups. The first part (21 subjects) was obtained with a 3M^{TM} Littmann Electronic Stethoscope Model 3200 [241] digital stethoscope and consists of PCGs 30 s in length. The second part (16 subjects) was obtained with the Eko DUO ECG + Digital Stethoscope [242] and consists of PCGs 15 s in length. Both devices are shown in Figure 3.2. They use built-in filters to reduce ambient noise and record single channel audio signals at a sampling rate of 4000 Hz. According to the PCA analysis that we conducted, the difference between the recordings from the two devices after prep-rocessing were small, thus it was reasonable to consider both as a device-independent dataset. The subjects were recorded in both the decompensated and the recompensated phase. The decompensated episode was recorded when the patient was admitted to the hospital for worsening heart failure episode, while the recompensated one was recorded upon discharge from the hospital when the patient was optimally recompensated and was deemed optivolemic, meaning their fluid levels and blood volume have returned to a healthy, balanced state. The PCGs were collected by medical professionals at UMCL from left parasternal 3-rd intercostal space body position. Overall, our dataset consists of 75 PCGs, 37 and 38 for compensated and decompensated phases, respectively, and adds up to 29 min 15 s in length. A detailed profile of the UMCL dataset is presented in Table 3.1. The study protocol was reviewed and approved by the Republic of Slovenia National Medical Ethics Committee (decision number 0120-276/2016-5).

The patient selection process involved a prospective nonrandomized cohort study that included 37 consecutive patients hospitalized for worsening heart failure at the Advanced Heart Failure Center, Dept. of Cardiology, UMC Ljubljana. Inclusion criteria were as follows: CHF of ischemic or non-ischemic etiology, hospitalization for worsening heart failure $<$ 24 h ago, age $>$ 18 yr. We did not consider patients with severe valvular disease, artificial valves, patients with acute myocardial infarction and/or de-novo acute heart failure, patients in cardiogenic shock, on vasoactive and/or inotropic support, on mechanical ventilation or on short- or long-term mechanical circulatory support for this analysis, or



(a) (b)

Figure 3.2: Recording devices used to collect the UMCL dataset: (a) Littmann Electronic Stethoscope Model 3200 (image adapted from [243]) and (b) Eko DUO ECG + Digital Stethoscope connected to a smartphone.

Table 3.1: Detailed profile of the UMCL heart-sound dataset.

Data source	Recording class	#PCGs	#Beats	Recording position	Sample rate	Acquisition device
Ours	Recomp.	37	942	3rd intercostal space	4000 Hz	3M Littmann 3200/ Eko DUO
	Dekcomp.	38	927			
	Total	75	1869			

patients that were hospitalized for worsening heart failure more than 24 h ago. Clinical, biochemical, and medical therapy data were collected for all the patients at the time of the initial heart-sound sampling. All patients included in this analysis were recompensated using levosimendan, followed by the intravenous diuretic therapy. In all study participants, heart sounds were recorded before the infusion of levosimendan (decompensated phase) and upon reaching the optivolemic phase (recompensated phase). Demographic and clinical characteristics of the patients are given in Table 3.2.

3.1.2 PhysioNet

We used the publicly available PhysioNet/CinC Challenge 2016 database [7], [46] in both parts of our study. In the first part, it was used to test robustness of our classical ML pipeline for detection of decompensation episodes. In the second part, it was used to develop and evaluate the performance of a heart-sound augmentation technique for heart-sound classification with limited data. This database is a collection of six different datasets of heart sounds obtained by independent research groups around the world. It is the largest and the most diverse collection of heart sounds and was curated by the challenge organisers to provide researchers with a heart-sound database to train and evaluate automated CVD

Table 3.2: UMCL dataset patient demographic and clinical characteristics (N=37).

Parameter	Value
Age [yr]	56.6 ± 12
Gender (male) [%]	88
Heart failure etiology (ischemic) [%]	27
Cause of decompensation	
Volume overload [%]	78
Infection [%]	15
Arrhythmia [%]	7
LVEF [%]	26.7 ± 7.7
NT-proBNP [pg/mL]	4593 (953, 5102)
Medical therapy	
ARNI/ACEI/ARB [%]	96
Beta blockers [%]	100
MRA [%]	96
SGLT2i [%]	70
Diuretic [%]	96
Ca-antagonist [%]	9
Digoxin [%]	15

Abbreviations: LVEF, left ventricular ejection fraction; ARNI, angiotensin receptor antagonist neprylisin inhibitor; ACEI, angiotensin convertase enzyme inhibitor; ARB, angiotensin receptor blocker; MRA, mineralocorticoid receptor blocker; SGLT2i, sodium glucose transporter 2 inhibitor; Ca, calcium.

diagnostics algorithms [244]. The six datasets included in the database differ in a variety of aspects: recording equipment, signal quality, recording environments, recording body position, patient types, and methods used to determine the diagnosis. All the heart-sound recordings in the database are divided into two types based on expert labeling from the original data contributors: normal and abnormal (pathological). The normal recordings were from healthy subjects and the abnormal ones were from patients with heart abnormalities, including MVP, MR, innocent or benign murmurs, AD, AS, and CAD. The data include recordings from the clean to the very noisy (flagged by the data curators as being of poor signal quality and having unreliable labels), reflecting real-world conditions.

The database contains 3153 recordings from 1297 patients, with each patient contributing between one and six recordings. Of these, 297 recordings are designated as the "validation" set, and 2874 as the "training" set, according to the Challenge organizers. Although patient identities are not linked to specific recordings, the organizers ensured that each patient's recordings are exclusively assigned to either the training or validation set, with no overlap.

Since the Challenge's original test set is not publicly available, we utilize the "validation" set as our test set to enhance reproducibility. The 2874 recordings in the "training" set pose some potential biases that must be addressed. Class imbalances within these recordings can skew model predictions towards the majority class, and the differing sizes of the six constituent datasets may introduce biases related to recording device, auscultation site, diagnostic protocol, environmental noise, or patient demographics. Additionally, training on all 2874 recordings would pose a considerable computational load. To mitigate these issues, we created a balanced and computationally feasible subset of the training data. We removed recordings flagged as poor quality and balanced each of the six datasets by randomly down-sampling the majority class to match the size of the minority class, thus achieving class balance and minimizing dataset-size discrepancies.

Table 3.3: Detailed profile of the heart-sound data used for the development and evaluation of the heart-sound augmentation techniques. Based on [7], [46].

Data source	Class	#PCGs		#Beats		Rec. position	Sample rate	Acq. device
		Train	Test	Train	Test			
MIT	Normal	76	40	2784	1517	Nine different positions	44 100 Hz	Welch Allyn Meditron
	Abnormal	76	40	2738	1431			
AAD	Normal	37	49	304	403	Tricuspid area	4000 Hz	3M Littmann E4000
	Abnormal	37	49	292	398			
AUTH	Normal	4	3	249	107	Apex	4000 Hz	Welch Allyn Meditron
	Abnormal	4	4	363	270			
UHA	Normal	21	5	256	52	Unknown	8000 Hz	Prototype (Infral Corp.)
	Abnormal	21	5	418	63			
DLUT	Normal	108	49	3072	1585	Multiple chest positions	800 Hz/ 22 050 Hz/ 8000 Hz	MLT201/ piezo/ 3M Littmann
	Abnormal	108	53	2043	1365			
SUA	Normal	31	0	1096	0	Apex	8000 Hz	JABES
	Abnormal	31	0	1140	0			
Total		554	297	14755	7191			

Abbreviations: MIT, Massachusetts Institute of Technology; AAD, Aalborg University; AUTH, Aristotle University of Thessaloniki; UHA, University of Haute Alsace; DLUT, Dalian University of Technology; SU, Shiraz University.

We were left with 554 recordings as our full training data. To ensure the selected 554 recordings were representative, we tested 10 additional random subsets of 554 recordings chosen using the same removal process. The results showed no significant differences in model performance, confirming our subset's representativeness. Furthermore, we found that the models' performance on the subset of 554 recordings without augmentation was comparable to the models trained on the full set of 2874 recordings. This was likely because the smaller subset was balanced (as the test set is), free of noisy recordings, and retained sufficient diversity. The list of the exact 554 recordings that were used can be found in Appendix A.1. Note that, as no hyper-parameter fine-tuning was performed, no data was withheld from the training set for validation purposes.

A detailed profile of the data used is presented in Table 3.3. For additional information about the database, including demographics data, recordings, associated synchronously recorded data, and acquisition devices, see [7].

3.2 Data Pre-processing

After data acquisition and labelling, pre-processing steps such as denoising, data transformation, and segmentation are crucial to ensure its quality and relevance for subsequent analysis. Noise is removed during the denoising process without altering the original signals. Following denoising, data transformation is performed to convert the data into formats suitable for input to DL models. The data is then divided into meaningful and manageable parts through segmentation. In PCG analysis, signals are typically segmented into heartbeats, with the goal of identifying the location of all four heart states within each heartbeat.

3.2.1 Denoising and normalisation

In addition to being very faint, heart-sound recordings usually contain significant noise from various sources, such as ambient sounds, breathing, lung sounds, contact between the recording device and the body, and speech. As an electric signal, heart sound is also susceptible to electromagnetic and power frequency interference. Although heart sounds have frequencies of up to 800 Hz (see Table 2.1), the most dominant frequencies are in the frequency range of 25 – 400 Hz [245]. To reduce the effects of different recording settings and to reduce noise, the PCGs were filtered with a band-pass Butterworth filter of 4-th order and a frequency range from 25 – 400 Hz. The filtering was performed using Matlab R2021a [246]. Although Section 2.5.1.1 presents various denoising methods, we chose linear filtering for its widespread use and ease of implementation. In addition, Butterworth filter was selected for its smooth frequency response, which minimizes signal distortion.

As the PCGs in the datasets were recorded at different amplitudes, the next step was heart-sound signal normalisation. We used the root-mean-square (RMS) normalisation with the target amplitude of −20 dBFS. As opposed to the peak normalisation, which normalizes the signal based on the highest peak, the RMS normalisation normalizes the signal based on the average power level by calculating the average value of all peaks. The normalisation was performed using Python 3.7 and the library Pydub 0.25.1 [247].

3.2.2 Data Transformation

For the purpose of the DL analysis, we created additional time-series and spectrogram datasets to be used as inputs to DL models. For the time-series data, the recordings were first split into four non-overlapping frequency bands. Specifically, each normalized PCG recording was divided into frequency bands of 25 – 45 Hz, 45 – 80 Hz, 80 – 200 Hz, and 200 – 400 Hz. This process generated four separate channels for each recording. The rationale behind selecting these specific bands was to align each band with different heart sounds as closely as possible, as these sounds generally appear in distinct frequency ranges (see Table 2.1). A Butterworth filter of 4-th order was used for the band-pass filtering. Next, we down-sampled all channels to 1000 Hz. The Nyquist frequency of 500 Hz is sufficient to accurately capture the heart sounds, as it is more than the highest frequency of interest (400 Hz), thus preventing information loss and aliasing in accordance with the Nyquist theorem [248]. Additionally, down-sampling from 4000 Hz to 1000 Hz reduced the size of the dataset by a factor of four, which saved memory and sped up the model-training process. After down-sampling, each channel was standardized using the mean and standard deviation values calculated separately for each dataset, following the equation

$$\hat{\boldsymbol{x_i}} = \frac{\boldsymbol{x_i} - \mu_i}{\sigma_i}. \tag{3.1}$$

Here, $\boldsymbol{x_i}$, $\hat{\boldsymbol{x_i}}$, μ_i, and σ_i are the signal, its standardized version, mean, and standard deviation of the i-th channel, respectively.

Additionally, we computed Mel-scale spectrograms from the raw PCGs using the lowest and highest cutoff frequencies of 25 Hz and 1000 Hz, respectively. The spectrograms were then standardization as per Eq. (3.1). In this case, there was only one channel, $i = 1$, and both $\boldsymbol{x_i}$ and $\hat{\boldsymbol{x_i}}$ were 2-dimensional. The spectrograms were computed using Python 3.7 and the library Librosa 0.9.1 [249].

3.2.3 Signal Segmentation and Removal of Bad Segments

The Springer's modification [39] of Schmidt's method [148] was used to split the heart sound into separate heartbeats and to find the four main states of each segment: S1, systole, S2, and diastole. This algorithm uses a HSMM and Viterbi decoding, which includes the duration densities of the heart states, and provides a state-of-the-art method for segmenting heart sounds. The segmentations were determined from the non-transformed normalized PCGs. The segmentation was performed using Matlab R2021a [246].

After segmentation, some of the segments were removed. In manually reviewing the segmented PCGs of the UMCL dataset, we found that seven (9%) of the recordings either consisted of a significant number of segments that were not correctly determined, or the recording itself was so unclear that it was impossible to tell whether the segments were correct or not. The two most common reasons for the segmentation error were that one of the main sounds (S1 or S2) was not detected by the PCG acquisition device, resulting in a segment that was too long (longer than one heartbeat), or that the high-amplitude noise was detected as one of the two main sounds, resulting in a segment that was too short (shorter than one heartbeat). As some of the features in our classical ML pipeline were calculated based on the characteristics of the S1 and S2 sounds, the segments where either of the sounds was not present or the signal-to-noise ratio was too high were excluded in the analysis. To find such segments, we first computed the so-called analytic signal using Hilbert transform. For a real-valued signal $s(t)$, its Hilbert transform $\hat{s}(t)$ is defined as a composition

$$s_A(t) = s(t) + j\hat{s}(t), \tag{3.2}$$

where $s_A(t)$ is the analytic signal and j is the imaginary unit. Because the analytic signal is a complex function, we can express it in exponential notation as

$$s_A(t) = A(t)e^{j\psi(t)}. \tag{3.3}$$

Here, $A(t)$ is called the instantaneous amplitude or the envelope and $\psi(t)$ is the instantaneous phase [250]. In Figure 3.3, $s(t)$ is depicted in black and the corresponding $A(t)$ is depicted in red.

After that, we calculated the mean (μ_A) and standard deviation (σ_A) of the area under the envelope across all segments of each PCG. Segments with areas under the envelope falling outside the interval

$$\mu_A \pm f_A \cdot \sigma_A \tag{3.4}$$

were removed. The z-score[1], f_A, was determined by visually inspecting a subset of the segments to find an optimal balance between removing as many problematic segments as possible while preserving as many good segments as possible. We set $f_A = 2.2$, which resulted in 3.3 (2.6%) of the segments per PCG recording being removed on average.

[1] a statistical measure that quantifies how far a specific value is from the mean of a set of values, expressed in terms of the number of standard deviations away from the mean.

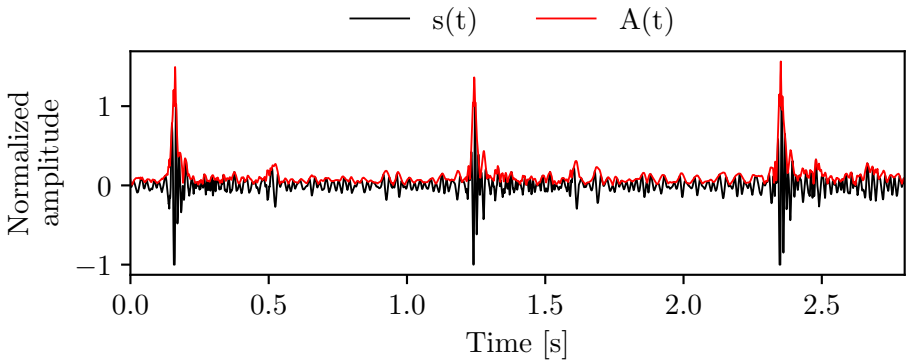


Figure 3.3: Example of a signal and its corresponding envelope.

Figure 3.4 shows an example of a clear segment, an example of a segment that was removed because S2 is missing, and an example of a segment that was removed because it is too noisy. The envelopes were computed using Python 3.7 and the libraries Scipy 1.5.2 [251] and Numpy 1.18.5 [252].

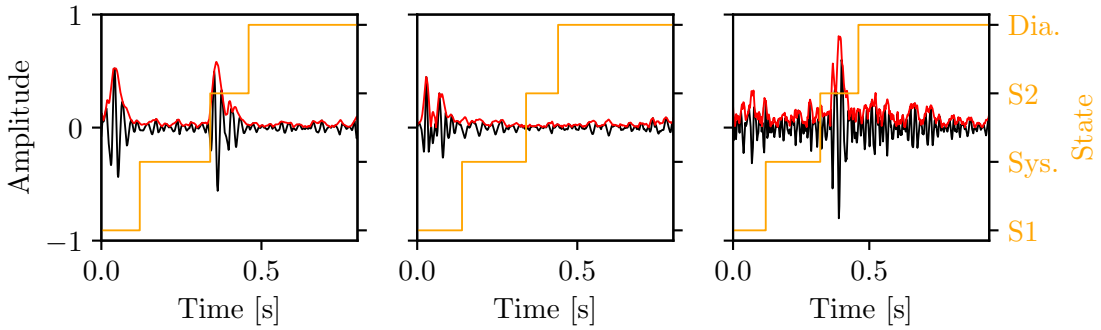
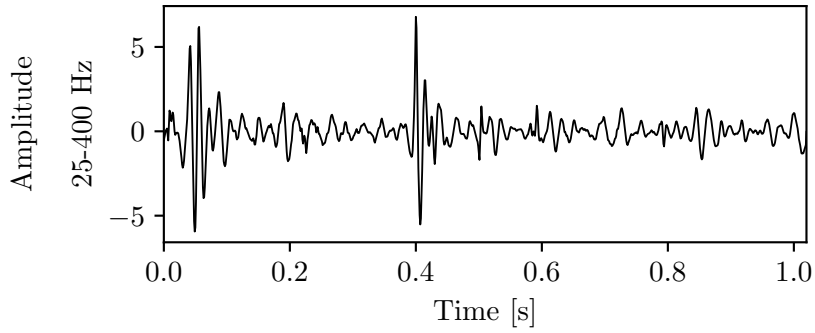


Figure 3.4: Example of a (left) clear PCG segment, (middle) PCG segment with missing S2 sound, and (right) a noisy PCG segment. Envelopes are shown in red.

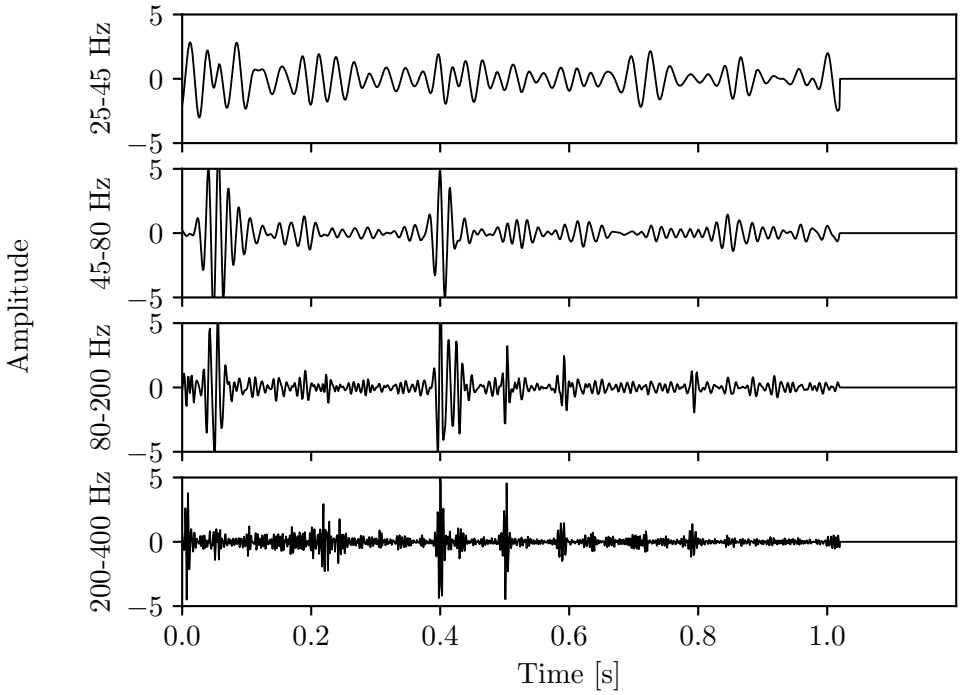
For the PhysioNet dataset, segmentation was performed by the challenge organisers using the same Springer's segmentation algorithm. The segments were manually reviewed and corrected by medical professionals, and these annotations were included in the database. Some segments were marked by the medical professionals as too noisy to accurately determine the correct timestamps of the heart states. We excluded these segments from our analysis. As a result, for this dataset, we did not need to calculate the envelopes to remove bad segments.

3.2.4 Data Formatting

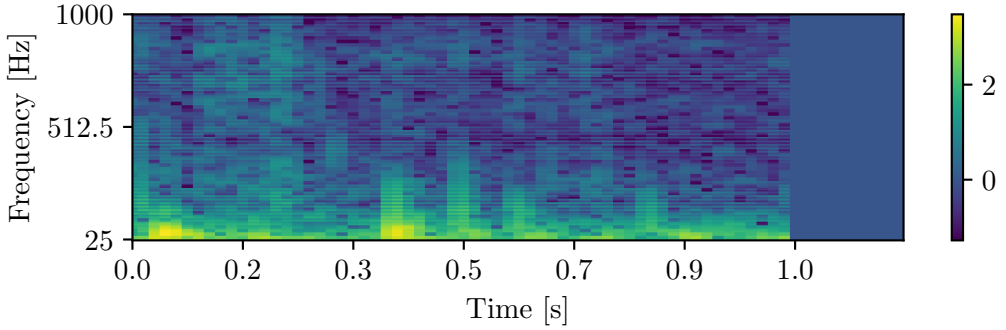
After segmentation, the instances in the PhysioNet dataset, which was intended for DL, were zero-padded to ensure a uniform input shape for all instances, as required by the DL models. The segment length was determined by the longest segment in each dataset, which was 2.5 s. Consequently, the shape of the time-series instances 4×2500 for the PhysioNet dataset, whereas the shape of the spectrograms was 128×128 for both datasets. To illustrate the comparison between the three data formats used in our study, we display a random heartbeat segment in all three formats in Figure 3.5.



(a)



(b)



(c)

Figure 3.5: The three data formats after the pre-processing and segmentation: (a) single-channel time series, (b) multi-channel time series, and (c) spectrogram. For clarity, the zero-padded instances used for DL, (b) and (c), are cropped to the length of 1.2 s.

3.3 Detection of Decompensation Episodes of CHF

To develop a method for detecting decompensation episodes of CHF, we used the UMCL dataset described in Section 3.1.1. The outcome of interest was a binary variable indicating whether a PCG represented a recompensated or a decompensated CHF phase. After pre-processing the data, feature engineering was performed, followed by the training and evaluation of classical ML models.

This section is structured as follows. Our approach to feature engineering is described in Section 3.3.1. The classical ML models employed in the study are described in Section 2.5.3.1. Lastly, the experimental setup is detailed in Section 3.3.2.

3.3.1 Feature Engineering

The goal of feature engineering is to identify a small set of representative features that can effectively replace the high-dimensional signals. Feature engineering included extraction and selection of numerical features to be used as input to classification models.

3.3.1.1 Feature Extraction

We extracted a total of 177 features from each normalized, 25 – 400 Hz filtered heartbeat segment (Figure 3.5, left). These included features in time domain, statistical features, features in frequency domain, and features generated by a 4-level wavelet decomposition.

3.3.1.1.1 Time-domain Features Time-domain features are derived directly from the signal in time domain. They focus on capturing the signal's characteristics over time, providing insights into its temporal dynamics. We extracted measures such as heartbeat duration, beats per minute, mean envelope, root-mean-square, zero-crossing rate, energy, and entropy. Let $x_i(t)$, where $t = 1, \ldots, n_i$ represent a time-series sequence of n_i samples. In our case, each instance, $x_i(t)$, represents a heartbeat of length n_i with the (original) sampling rate of $f_s = 4000$ Hz. The heartbeats per minute can be computed as

$$\mathrm{BPM}_i = \frac{60 \cdot f_s}{n_i}. \tag{3.5}$$

The envelope of a signal can be derived using Eqs. (3.2) and (3.2). The root-mean-square of a signal can be computed using

$$\mathrm{RMS}_i = \sqrt{\frac{1}{n_i} \sum_{t=1}^{n_i} x_i(t)^2}. \tag{3.6}$$

Zero-crossing rate measures how many times the signal changes value, from positive to negative and vice versa, divided by the number of samples in the signal as [253]

$$\mathrm{ZCR}_i = \frac{1}{2n_i} \sum_{t=1}^{n_i} |\operatorname{sgn}(x_i(t)) - \operatorname{sgn}(x_i(t-1))| \tag{3.7}$$

where $\operatorname{sgn}(x_i(t))$ equals to 1 if $x_i(t) \geq 0$ and -1 otherwise. The energy of the signal is computed as

$$\mathrm{Energy}_i = \sum_{t=1}^{n_i} |x_i(t)^2|. \tag{3.8}$$

An alternative metric for energy is $\frac{\sigma_i^2}{\mu_i}$, where μ_i and $\sigma_i = \sqrt{\frac{\sum_{t=1}^{n_i}(x_i(t)-\mu_i)^2}{n_i}}$ are the mean and the standard deviation of the signal, $x_i(t)$, respectively [253]. Entropy is a measure of

abrupt changes in the energy of a signal. To compute it, we first divide the signal into L frames of equal length, compute the energy of each frame, $\text{Energy}_{i,l}$, where $l = 1, \ldots, L$, and divide it by the total energy as in $\text{e}_{i,l} = \frac{\text{Energy}_{i,l}}{\sum_{l=1}^{L} \text{Energy}_{i,l}}$. The entropy is then computed according to [253]

$$\text{Entropy}_i = -\sum_{l=1}^{L} \text{e}_{i,l} \cdot \log_2(\text{e}_{i,l}). \tag{3.9}$$

3.3.1.1.2 Statistical Features Statistical features focus on summarising the signal's overall distribution and statistical properties, providing insights into its shape and variability. We extracted statistical measures skewness and kurtosis. Skewness is a measure of the asymmetry of a distribution of values. A positive skewness indicates that the distribution is skewed to the left, with a longer tail on the right side of the distribution, whereas a negative skewness indicates that the distribution is skewed to the right, with a longer tail on the left side of the distribution. It is calculated as [253]

$$\text{Skewness}_i = \frac{\sum_{t=1}^{n_i}(x_i(t) - \mu_i)^3}{(n_i - 1) \cdot \sigma_i^3}. \tag{3.10}$$

Similarly, kurtosis measures the distribution's tail or peak relative to a normal distribution. Positive kurtosis indicates heavier tails and a sharper peak, while negative kurtosis indicates lighter tails and a flatter peak. It is calculated as [253]

$$\text{Kurtosis}_i = \frac{\frac{1}{n_i}\sum_{t=1}^{n_i}(x_i(t) - \mu_i)^4}{\sigma_i^4}. \tag{3.11}$$

3.3.1.1.3 Frequency-domain Features In the frequency domain, we computed power spectral density, MFCCs, spectral centroid, spectral bandwidth, spectral contrast, spectral flatness, spectral roll-off, and coefficients of polynomial fit to spectrum. Let $X_i(k)$ be the FFT coefficients of the signal $x_i(t)$, where $k = 1, \ldots, m_i$, and let f_k be the frequencies corresponding to the k-th FFT bin. Power spectral density represents the proportion of the total signal power contributed by each frequency component. For a frequency component f_k, the power spectral density can be computed as

$$\text{PSD}_i(f_k) = \frac{|X_i(k)|^2}{f_s}. \tag{3.12}$$

To compute the PSD for a specific frequency interval $[f_{\min}, f_{\max}]$, the PSD values for all frequency components f_k within this interval are summed, $f_k \in [f_{\min}, f_{\max}]$. When calculating PSDs for frequency intervals, the results are normalized by dividing each interval's PSD by the total sum of PDSs across all frequency components [254]. One of the most popular features in audio analysis, especial in speech processing, are the MFCCs [255]. MFCCs employ a Mel-scale distribution of frequency bands to capture the nonlinear perception of frequency in the human auditory system. This scale, known for its perceptually motivated design, features frequency intervals that are judged to be equally spaced by human listeners. To calculate MFCCs, the spectrum is first mapped to a Mel-scale that consists of L filters. If $P_{i,l}$, where $l = 1, \ldots, L$, represent the power at the output of the l-th filter, then the MFCCs are calculated as

$$\text{MFCC}_{i,m} = \sum_{l=1}^{L} (\log P_{i,l}) \cos\left[m\left(l - \frac{1}{2}\right)\frac{\pi}{L}\right], \quad m = 1, \ldots, L. \tag{3.13}$$

In other words, MFCCs are the discrete cosine transform coefficients of the Mel-scaled log-power spectrum [253]. The spectral centroid and spectral bandwidth are two basic metrics used to assess spectral position and shape. Spectral centroid represents the center of gravity of the spectrum and is defined as [253]

$$\text{Centroid}_i = \frac{\sum_{k=1}^{m_i} f_k |X_i(k)|}{\sum_{k=1}^{m_i} |X_i(k)|}. \tag{3.14}$$

Spectral bandwidth, on the other hand, is defined as [253]

$$\text{Bandwidth}_i = \left(\frac{\sum_{k=1}^{m_i} (f_k - \text{Centroid}_i)^p |X_i(k)|}{\sum_{k=1}^{m_i} |X_i(k)|} \right)^{\frac{1}{p}}, \tag{3.15}$$

where p denotes the order. Spectral contrast measures the difference in amplitude between peaks and valleys in a sound spectrum. A spectrum is first divided into B sub-bands (e.g., octaves). For each sub-band, $b = 1, \ldots, B$, the contrast is estimated by comparing the mean energy of the highest quantile (peak energy) to the mean energy of the lowest quantile (valley energy). Spectral contrast for a band b in decibels (dB) is computed as [256]

$$\text{Contrast}_{i,b} = 10 \log \left(\frac{\max\limits_{\forall k \in \text{band}_b} |X_i(k)|}{\min\limits_{\forall k \in \text{band}_b} |X_i(k)|} \right). \tag{3.16}$$

Spectral flatness quantifies the degree to which a sound resembles noise rather than a distinct tone. This metric is determined by the ratio of the geometric mean to the arithmetic mean of the magnitudes of the FFT coefficients. It is computed as [257]

$$\text{Flatness}_i = \frac{\left(\prod_{k=1}^{m_i} |X_i(k)| \right)^{\frac{1}{m}}}{\frac{1}{m} \sum_{k=1}^{m_i} |X_i(k)|}. \tag{3.17}$$

The spectral rolloff denotes the frequency below which a certain percentage, c, of the total spectral energy lies. It is defined as [253]

$$\text{Rolloff}_{i,c} = \min\{ f_k : \sum_{k=1}^{k_0} |X_i(k)|^2 \geq c \sum_{k=1}^{m_i} |X_i(k)|^2 \}. \tag{3.18}$$

The threshold c is often set to 85% or 95%. The coefficients of an n-th order polynomial fit are computed by fitting a polynomial to the log-power spectrum. The fitting can be expressed as

$$\log P_i(k) \approx c_{i,0} + c_{i,1} f_k + c_{i,2} f_k^2 + \cdots + c_{i,n} f_k^n, \tag{3.19}$$

where $P_i(k)$ is the power of a k-th bit with the frequency f_k, $P_i(k) = |X_i(k)|^2$, and $c_{i,j}$ are the coefficients of the fit.

3.3.1.1.4 Wavelet Decomposition Features In a multi-level wavelet decomposition, the signal is recursively filtered with a pair of low-pass and high-pass filters, g and h, and down-sampled. This process decomposes the signal into different frequency components and provides both approximation and detail coefficients at different decomposition levels.

The two filters are defined by a selected wavelet function (e.g., Daubechies). At each decomposition level, the filters are applied with a step size of two and the outputs are down-sampled by a factor of two. The wavelet decomposition at the first level can be represented as

$$cA_1(t) = \sum_k x(k) g(2t - k) \tag{3.20}$$

and

$$cD_1(t) = \sum_k x(k)h(2t-k), \tag{3.21}$$

where the cA_1 and cD_1 are the approximation and detail coefficients, respectively. At each recursive decomposition, the low-pass filtered and down-sampled signal is further decomposed at subsequent levels using the same pair of filters. For subsequent levels j, decomposition can be represented as

$$cA_{j+1}(t) = \sum_k cA_j(k)g(2t-k) \tag{3.22}$$

and

$$cD_{j+1}(t) = \sum_k cA_j(k)h(2t-k), \tag{3.23}$$

where cA_j and cD_j are the approximation and detail coefficients at level j, respectively [258].

The complete list of features extracted from the PCG segments can be found in Table 3.4. The features were extracted both from the full segments and from each of the four heart "*_states*". In some cases, the features were calculated as a ratio of the features of the states. The "*_ratio*" features include S1/Full, Sys/Full, S2/Full, Dia/Full, S1/S2, Sys/Dia, Sys/S1, and Dia/S2. To extract the frequency domain features, the segments were transformed from the time domain to the frequency domain using a FFT with a Hanning window of 256 ms in length and a hop length of 64 ms. For the power spectral density frequency features, we selected frequency "*_bands*" of 25 – 40 , 40 – 60 , 60 – 80 , 80 – 100 , 100 – 120 , 120 – 140 , 140 – 160 , 160 – 180 , 180 – 200 , 200 – 250 , 250 – 300 , and 300 – 400 Hz. The selected frequency bands are similar to those selected by the authors of Potes et al. [216]. We extracted the first 13 MFCC coefficients from each of the four states. The "2–5" bands of the spectral contrast features include: 25 – 50 , 50 – 100 , 100 – 200 , and 200 – 400 Hz. For spectral roll-off, a treshold of 85% was used. The 1-st order polynomial (linear function) was used for the fitting to the log-power spectrum. Daubechies-4 wavelet was used as a basis for the DWT features.

To smooth out the outliers, we generated another set of 2×177 features representing the mean and standard deviation of the features taken as a sliding window with window size $w = 6$ across the segments of each PCG. To ensure that each segment was equally represented, windowing was performed cyclically. The resulting 354 features were then used for the models' evaluation. The features were computed using Python 3.7 and libraries Librosa 0.9.1 [249], Scipy 1.5.2 [251], PyWavelets 1.3.0 [259] and Numpy 1.18.5 [252].

3.3.1.2 Feature Selection

During training, feature selection was performed by computing the mutual information [185] between the features and the target variable. We experimentally determined that selecting the top $k_b = 40$ features provided the best performance. This selection was based on observing the point at which increasing the number of features no longer resulted in performance improvements. The goal of this process was to minimize the number of predictor features while still achieving the highest possible model performance. Although we discussed more complex feature selection methods, such as wrapper and intrinsic approaches, in Section 2.5.2.2, we opted for this simpler mutual information-based filtering method due to its widespread use and ease of implementation. The top 40 features selected from the entire dataset are shown in Table 3.5.

Table 3.4: List of features extracted from PCG heartbeat segments.

Feature type (N)	Per segment state	Domain	Description
BPM (1)	Full	Time	Inverse of segment duration in beats per minute
Dur_*state* (4)	S1, Sys, S2, Dia	Time	Duration in milliseconds
Dur_Ratio_*ratio* (8)	-	Time	Duration ratios
MeanEnv_Ratio_*ratio* (8)	-	Time	Mean envelope ratios
RMS_*state* (4)	S1, Sys, S2, Dia	Time	Root-mean-square of a signal
RMS_Ratio_*ratio* (8)	-	Time	Root-mean-square ratios
ZCR_*state* (4)	S1, Sys, S2, Dia	Time	Zero-crossings rate
Entropy_*state* (4)	S1, Sys, S2, Dia	Time	Entropy
Skewness_*state* (4)	S1, Sys, S2, Dia	Statistical	Skewness
Kurtosis_*state* (4)	S1, Sys, S2, Dia	Statistical	Kurtosis
PSD_*region*_*band* (24)	Sys, Dia	Frequency	Power spectral density for different frequency bands
MFCC1-13_*state* (52)	S1, Sys, S2, Dia	Frequency	13 Mel-frequency cepstral coefficients
Centroid_*state* (4)	S1, Sys, S2, Dia	Frequency	Spectral centroid
Bandwidth_*state* (4)	S1, Sys, S2, Dia	Frequency	2nd order spectral bandwidth
Contrast2-5_*state* (16)	S1, Sys, S2, Dia	Frequency	Spectral contrast for different frequency bands
Flatness_*state* (4)	S1, Sys, S2, Dia	Frequency	Spectral flatness
Rolloff_*state* (4)	S1, Sys, S2, Dia	Frequency	Frequency below which 85% of the total spectral energy lies
Poly_*state* (4)	S1, Sys, S2, Dia	Frequency	Coefficients of degree-1 polynomial fit to the spectrogram
DWT1-4_*state* (16)	S1, Sys, S2, Dia	Frequency	Level 4 DWT mean detail coefficients

Table 3.5: Top 40 predictor features of the UMCL dataset according to their mutual information with the outcome. The prefixes "m_" and "sd_" correspond to the mean and the standard deviation of the features taken as a sliding window across the segments for each PCG (see Section 3.3.1.2 for details).

Rank	Feature (mutual info)	Rank	Feature (mutual info)
1	m_BPM (0.16)	21	m_Dur_Ratio_S1Full (0.10)
2	m_Dur_Ratio_DiaFull (0.15)	22	m_PSD_Dia_140_160Hz (0.10)
3	m_Dur_Dia (0.14)	23	sd_PSD_Sys_250_300Hz (0.10)
4	sd_PSD_Dia_200_250Hz (0.14)	24	sd_PSD_Sys_200_250Hz (0.10)
5	sd_BPM (0.14)	25	m_MFCC6_Dia (0.10)
6	m_Dur_Ratio_SysS1 (0.13)	26	m_ZCR_Dia (0.10)
7	m_MFCC4_Dia (0.13)	27	sd_PSD_Sys_140_160Hz (0.10)
8	m_Dur_Ratio_SysDia (0.13)	28	m_PSD_Dia_300_400Hz (0.10)
9	sd_PSD_Dia_250_300Hz (0.13)	29	m_PSD_Sys_300_400Hz (0.10)
10	m_MFCC2_Sys (0.12)	30	m_Contrast5_Dia (0.09)
11	sd_PSD_Dia_180_200Hz (0.12)	31	m_MFCC1_Sys (0.09)
12	sd_PSD_Sys_120_140Hz (0.12)	32	m_Centroid_Dia (0.09)
13	sd_PSD_Sys_120_140Hz (0.12)	33	m_Bandwidth_Dia (0.09)
14	m_MFCC2_Dia (0.12)	34	m_MFCC6_S1 (0.09)
15	sd_Dur_Ratio_SysS1 (0.11)	35	sd_PSD_Sys_300_400Hz (0.09)
16	m_MFCC1_Dia (0.11)	36	m_PSD_Dia_250_300Hz (0.09)
17	sd_PSD_Dia_140_160Hz (0.11)	37	m_MFCC4_Sys (0.09)
18	sd_PSD_Sys_180_200Hz (0.11)	38	m_Dur_Ratio_SysFull (0.09)
19	m_PSD_Dia_160_180Hz (0.11)	39	m_Dur_Sys (0.09)
20	m_MFCC6_Sys (0.11)	40	sd_PSD_Dia_100_120Hz (0.08)

3.3.2 Experimental Setup

We implemeted 10 classical ML methods for the classification between the decompensated and recompensated CHF phases, all of which are described in Section 2.5.3.1. They were implemented in Python 3.7 using Scikit 0.24.2 [260] and lightgbm 3.3.1 [261] libraries.

The methods were evaluated using a subject-wise 10-fold CV approach, in which all recordings from a given patient were assigned to the same fold, ensuring no patient had recordings in multiple folds. To maintain balanced representation, we applied stratification based on the two data acquisition setups, so each fold included a proportional number of patients from both acquisition setups. Additionally, since each subject provided recordings for both recompensated and decompensated CHF phases, this subject-wise approach also ensured balance across target classes in each fold.

During the evaluation, we discovered that models trained exclusively on correctly segmented PCGs performed significantly better. Because of this, for each training set, the subjects associated with any of the seven PCGs that we had manually identified as incorrectly segmented were removed. No removal was done for the test set. By excluding these faulty PCGs, we ensured that the models were trained on high-quality data, which contributed to their improved performance.

To keep the models explainable and as transparent as possible and to avoid overfitting, we performed feature selection. The features were selected independently of the models, meaning that all of the models used the same features. Features were selected by calculating the mutual information [185] between each feature and the outcome variable. Each training fold was divided into five stratified sub-folds, and 40 features that had the highest mutual

information with the outcome variable on average across the five sub-folds were used for training. Note that although Table 3.5 lists the top features across the entire dataset, feature selection was conducted separately within each training fold, which led to slight variations in selected features per fold, though most features remained consistent.

The models were implemented with the default parameter values as per libraries' documentations (see Appendix A.2) [260], [261]. Experiments were also conducted with model parameter fine-tuning using grid-search. For each model, we defined a set of values for each parameter as given in Appendix A.2. During each split of the 10-fold CV, the training data was further divided into five sub-folds. The grid search performed 5-fold CV multiple times, testing each parameter combination. The combination with the highest average accuracy across the five folds was selected for training and testing in that fold of the 10-fold CV. Algorithm 3.1 details our grid-search fine-tuning approach.

Algorithm 3.1: K-fold cross-validation with grid-search fine-tuning.

Input: M, model; $\mathcal{P}$, sets of values for all model parameters; k, number of folds of the CV; n, number of folds of the fine-tuning CV; $\mathcal{D}$, data
Output: $\overline{a}$, model performance

$\{\mathcal{D}_i\}_{i=1}^{k} \leftarrow$ split data into k folds
$a \leftarrow$ initialize an empty array of length k
for $i \leftarrow 1$ **to** k **do**
  $\{\mathcal{D}_{i,j}\}_{j=1}^{n} \leftarrow$ split fold data into n sub-folds
  $b \leftarrow$ initialize an empty array of length p
  for each combination $\mathcal{P}_{l=1}^{p}$ **in** $\mathcal{P}$ **do**
    $c \leftarrow$ initialize an empty array of length n
    for $j \leftarrow 1$ **to** n **do**
      $c_j \leftarrow$ evaluate $M(\mathcal{D}_{i,j}, \mathcal{P}_l)$
    end for
    $b_l = \overline{c} \leftarrow$ compute mean performance of the nested n-fold CV
  end for
  $\hat{b} = \max b \leftarrow$ find best mean performance across all parameter combinations
  $\hat{\mathcal{P}} \leftarrow$ parameter combination that achieved performance $\hat{b}$
  $a_i \leftarrow$ evaluate $M(\mathcal{D}_i, \hat{\mathcal{P}})$
end for
$\overline{a} \leftarrow$ compute mean performance of the k-fold CV

While Algorithm 3.1 is designed primarily for fine-tuning model parameters, it can also be extended to optimize other pipeline parameters, such as pre-processing settings and feature selection, using the same grid-search approach. Typically, pre-processing steps based on domain expertise—such as setting band-pass filtering thresholds to isolate specific frequency ranges—are determined prior to training and applied uniformly across the entire dataset. However, by incorporating these pre-processing parameters into the tuning process, it is possible to further refine the model's adaptability to the data characteristics.

3.4 Heart-Sound Data Augmentation

To develop a heart-sound data-augmentation technique, we used the PhysioNet dataset as described in Section 3.1.2. The outcome of interest was a binary variable indicating whether a PCG represented a normal or an abnormal heart sound. To evaluate the effect of augmentation on datasets of limited sizes, we performed experiments with varying amounts

of PCGs, from very few PCGs to using the whole dataset. The experiments were carried out with both time-series and spectrogram data.

This section is structured as follows. It starts with the mathematical preliminaries in Section 3.4.1. Then, in Section 3.4.2, our proposed method for heart-sound augmentation is presented. The baselines are described in Section 3.4.3. In Section 3.4.4, the architectures of the CNN models used in our experiments are explained. Section 3.4.5 gives description of how the data was split into smaller subsets. Sections 3.4.6 and 3.4.7 describe the experimental setup and implementation details, respectively.

3.4.1 Preliminaries

Let an instance x_i and its label y_i represent a single data sample from the dataset $\mathcal{X} = \{(x_i, y_i)\}_{i=1}^{s}$ with s samples. Each instance, x_i, is a time series of length n_i where every time slice is a vector of amplitudes, $x_i(t)$, where $t = 1, \ldots, n_i$. We filtered the recordings into $m = 4$ different frequency bands so that $x_i(t, p)$, where $1 \leq p \leq m$, denotes the amplitude of the p-th frequency band at timestamp t. Subsequently, we omit explicitly indicating the dependence of t and p in x_i, unless doing so enhances clarity. Each x_i includes a single heartbeat that is composed of four heart states, i.e., S1, systole, S2, and diastole, with lengths denoted as l^{S1}, l^{S}, l^{S2}, and l^{D}. Their sum,

$$l_i^{\mathrm{full}} = l_i^{\mathrm{S1}} + l_i^{\mathrm{S}} + l_i^{\mathrm{S2}} + l_i^{\mathrm{D}}, \tag{3.24}$$

represents the length of the full heartbeat of x_i. All instances are zero padded to a fixed length n so that $n_i = n$, $\forall i \in \{1 \ldots, s\}$, and $x_i(l_i^{\mathrm{full}} \leq t < n, p) = 0$. Let $M(t_{\min}, t_{\max})$ be a binary rectangular mask of size $n \times m$ defined as

$$M(t_{\min}, t_{\max}) = \{(i, j) | t_{\min} \leq i < t_{\max}\}. \tag{3.25}$$

Element-wise multiplication between the mask and the instance, $M(t_{\min}, t_{\max}) \odot x_i(t, p)$, preserves the elements of the instance for which it holds that $t_{\min} \leq t < t_{\max}$, whereas others are turned to zero. Let $S(l)$ be an $n \times n$ shift matrix where the (i, j)-th element is

$$S_{ij}(l) = \delta_{i+l,j} \tag{3.26}$$

with δ_{ij} denoting the Kronecker delta[2]. The matrix product of the shift matrix and the instance, $S(l) \times x_i(t, p)$, shifts the elements of x_i by $|l|$ positions, either toward $t = 0$ if $l > 0$ (upper-shift matrix), or toward $t = n$ if $l < 0$ (lower-shift matrix), with zeros appearing in the empty positions.

Note that although we defined x_i to have a time-series format, the same notation can be used for a data instance in spectrogram format. In that case, the $x_i(t, p)$ denotes the power of the p-th frequency bin in the audio frame at time t, whereas everything else stays the same.

3.4.2 Proposed Method

In the following, we describe our proposed method and its upgraded version.

3.4.2.1 PCGmix

We propose the phonocardiogram mix (PCGmix) data-augmentation method, which is inspired by the Mixup image-augmentation technique, where the pixels are linearly interpolated between two instances [262]. Instead of image pixels, PCGmix operates on

[2] δ_{ij} equals to 1 if $i = j$, otherwise it is 0

multichannel time-series heart sounds or the spectrogram data. The augmentation takes two heartbeats as the input, x_i and x_j, and uses a mixing coefficient, λ, to interpolate the sections of the heartbeats in such a way that the original durations of the heart states are preserved. We define PCGmix as

$$\begin{aligned}
&h_{\mathrm{P}}(x_i, x_j, \lambda, l_i^{\mathrm{HS}}, l_j^{\mathrm{HS}}) = \\
&(1 - M_{\mathrm{P},i,1} - M_{\mathrm{P},i,2} - M_{\mathrm{P},i,3} - M_{\mathrm{P},i,4}) \odot x_i \\
&+ M_{\mathrm{P},i,1} \odot x_i \cdot \lambda + (M_{\mathrm{P},j,1} \odot x_j) \cdot (1 - \lambda) \\
&+ M_{\mathrm{P},i,2} \odot x_i \cdot \lambda + S_{\mathrm{P},j,2} \times (M_{\mathrm{P},j,2} \odot x_j) \cdot (1 - \lambda) \\
&+ M_{\mathrm{P},i,3} \odot x_i \cdot \lambda + S_{\mathrm{P},j,3} \times (M_{\mathrm{P},j,3} \odot x_j) \cdot (1 - \lambda) \\
&+ M_{\mathrm{P},i,4} \odot x_i \cdot \lambda + S_{\mathrm{P},j,4} \times (M_{\mathrm{P},j,4} \odot x_j) \cdot (1 - \lambda).
\end{aligned} \tag{3.27}$$

Here,

$$\begin{aligned}
M_{\mathrm{P},i,1} =& M(0, \min\{l_i^{\mathrm{S1}}, l_j^{\mathrm{S1}}\}), \\
M_{\mathrm{P},i,2} =& M(l_i^{\mathrm{S1}}, l_i^{\mathrm{S1}} + \min\{l_i^{\mathrm{S}}, l_j^{\mathrm{S}}\}), \\
M_{\mathrm{P},i,3} =& M(l_i^{\mathrm{S1}} + l_i^{\mathrm{S}}, l_i^{\mathrm{S1}} + l_i^{\mathrm{S}} + \min\{l_i^{\mathrm{S2}}, l_j^{\mathrm{S2}}\}), \\
M_{\mathrm{P},i,4} =& M(l_i^{\mathrm{S1}} + l_i^{\mathrm{S}} + l_i^{\mathrm{S2}}, l_i^{\mathrm{S1}} + l_i^{\mathrm{S}} + l_i^{\mathrm{S2}} + \min\{l_i^{\mathrm{D}}, l_j^{\mathrm{D}}\}), \\
M_{\mathrm{P},j,1} =& M_{\mathrm{P},i,1}, \\
S_{\mathrm{P},j,2} =& S(l_j^{\mathrm{S1}} - l_i^{\mathrm{S1}}), \\
M_{\mathrm{P},j,2} =& M(l_j^{\mathrm{S1}}, l_j^{\mathrm{S1}} + \min\{l_i^{\mathrm{S}}, l_j^{\mathrm{S}}\}), \\
S_{\mathrm{P},j,3} =& S(l_j^{\mathrm{S1}} + l_j^{\mathrm{S}} - l_i^{\mathrm{S1}} - l_i^{\mathrm{S}}), \\
M_{\mathrm{P},j,3} =& M(l_j^{\mathrm{S1}} + l_j^{\mathrm{S}}, l_j^{\mathrm{S1}} + l_j^{\mathrm{S}} + \min\{l_i^{\mathrm{S2}}, l_j^{\mathrm{S2}}\}), \\
S_{\mathrm{P},j,4} =& S(l_j^{\mathrm{S1}} + l_j^{\mathrm{S}} + l_j^{\mathrm{S2}} - l_i^{\mathrm{S1}} - l_i^{\mathrm{S}} - l_i^{\mathrm{S2}}), \\
M_{\mathrm{P},j,4} =& M(l_j^{\mathrm{S1}} + l_j^{\mathrm{S}} + l_j^{\mathrm{S2}}, l_j^{\mathrm{S1}} + l_j^{\mathrm{S}} + l_j^{\mathrm{S2}} + \min\{l_i^{\mathrm{D}}, l_j^{\mathrm{D}}\}).
\end{aligned}$$

The PCGmix is visualized in Figure 3.6. The $(1 - M_{\mathrm{P},i,1} - M_{\mathrm{P},i,2} - M_{\mathrm{P},i,3} - M_{\mathrm{P},i,4}) \odot x_i$, $M_{\mathrm{P},j,1} \odot x_j$, $M_{\mathrm{P},j,2} \odot x_j$, $M_{\mathrm{P},j,3} \odot x_j$, and $M_{\mathrm{P},j,4} \odot x_j$ regions from Eq.(3.27) that mix to form a new instance are marked with yellow, green, blue, and purple, whereas the grey regions of the inputs are preserved. We can interpret the generation of a new instance described in Eq. (3.27) in such a way that the four heart states of x_i and x_j are interpolated individually and combined to form a new instance, whereby the lengths of the heart states of x_i are preserved. Specifically, each heart state of x_i is either fully or partially replaced by the interpolated ones. If the heart state of x_j is longer than that of x_i, it is trimmed to match the length of x_i's state. The two states are then interpolated and the interpolated state replaces the original state in x_i (S1 and systole in Figure 3.6). If, on the other hand, the heart state of x_j is shorter, the entire state is interpolated with the first part of the x_i's state, leaving the excess part of x_i's state preserved (diastole in Figure 3.6). The S2 states in Figure 3.6 are of the same length and thus no trimming/preserving takes place. The shifts in the 3-rd, 4-th, and 5-th terms of Eq. (3.27) ensure that the states of x_j are correctly aligned with those of x_i for the interpolation. For the interpolation, a mixing coefficient, λ, is randomly sampled between 0 and 1, $\lambda \in (0, 1)$. We employ the beta distribution for this sampling, as in $\lambda \sim \mathrm{Beta}(\alpha, \alpha)$. Notably, when $\alpha = 1.0$, λ is drawn from a uniform distribution, $\lambda \sim U(0, 1)$. The Algorithm 3.2 shows how the augmentation was carried out programmatically.

PCGmix leverages the theoretical framework provided by studies on Mixup, which indicate that enforcing a linear combination of inputs simplifies decision boundaries and

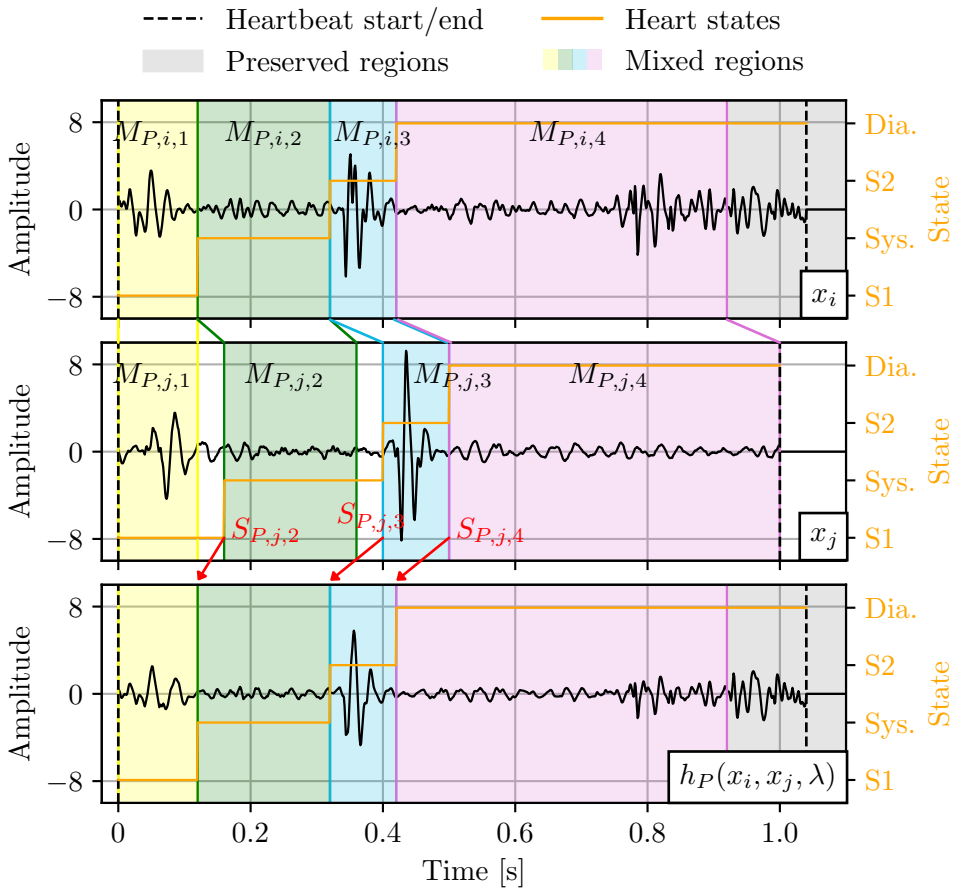


Figure 3.6: Visualization of the PCGmix process: (top) x_i, (middle) x_j, and (bottom) produced $h_P(x_i, x_j, \lambda)$. For clarity, the instances are cropped to the length of 1.1 s. In addition, only one channel per instance is shown.

can reduce overfitting by discouraging overly complex models. This technique enhances the learning of rare features by blending them with more common ones, a significant advantage in medical data where pathological features may be infrequent. By interpolating between multiple instances, PCGmix ensures that rare but important features are robustly represented in the training data, thereby improving model sensitivity and specificity [263].

A unique feature of PCGmix is its ability to preserve the duration ratios of heart states, a fundamental diagnostic characteristic crucial for accurate CVD detection. This preservation aligns directly with the clinical diagnostic process, where the timing of heart sounds provide valuable information about the heart's function and structure. PCGmix thus not only serves as a regularization tool but also retains essential diagnostic information. Standard data-augmentation methods often lack this domain-specific consideration, potentially leading to less effective models in medical applications where precise feature representation is critical.

3.4.2.2 PCGmix+

In addition to PCGmix, we introduce PCGmix+, which builds on PCGmix by additionally transforming the new instance with magnitude warping. Magnitude warping modifies the amplitudes across different parts of a PCG signal, simulating the effect of different record-

Algorithm 3.2: PCGmix augmentation.

Input: x_i and x_j, instances; λ, mixing coefficient; l_i^{HS} and l_j^{HS}; inputs' heart state durations

Output: x_{aug}, new instance

$x_{\mathrm{aug}} \leftarrow x_i$ // initialize new instance as a copy of x_i

$l_{\mathrm{min}}^{\mathrm{S1}} \leftarrow \min\{l_i^{\mathrm{S1}}, l_j^{\mathrm{S1}}\}$ // find the length of the shorter S1

$x_{\mathrm{aug}}[0 : l_{\mathrm{min}}^{\mathrm{S1}}] \leftarrow x_i[0 : l_{\mathrm{min}}^{\mathrm{S1}}] \cdot \lambda + x_j[0 : l_{\mathrm{min}}^{\mathrm{S1}}] \cdot (1 - \lambda)$ // replace (a part of) S1 with the interpolated S1 between x_i and x_j

$l_{\mathrm{min}}^{\mathrm{S}} \leftarrow \min\{l_i^{\mathrm{S}}, l_j^{\mathrm{S}}\}$ // find the length of the shorter systole

$x_{\mathrm{aug}}[l_i^{\mathrm{S1}} : l_i^{\mathrm{S1}} + l_{\mathrm{min}}^{\mathrm{S}}] \leftarrow x_i[l_i^{\mathrm{S1}} : l_i^{\mathrm{S1}} + l_{\mathrm{min}}^{\mathrm{S}}] \cdot \lambda + x_j[l_j^{\mathrm{S1}} : l_j^{\mathrm{S1}} + l_{\mathrm{min}}^{\mathrm{S}}] \cdot (1 - \lambda)$ // replace (a part of) systole with the interpolated systole between x_i and x_j

$l_{\mathrm{min}}^{\mathrm{S2}} \leftarrow \min\{l_i^{\mathrm{S2}}, l_j^{\mathrm{S2}}\}$ // find the length of the shorter S2

$x_{\mathrm{aug}}[l_i^{\mathrm{S1}} + l_i^{\mathrm{S}} : l_i^{\mathrm{S1}} + l_i^{\mathrm{S}} + l_{\mathrm{min}}^{\mathrm{S2}}] \leftarrow x_i[l_i^{\mathrm{S1}} + l_i^{\mathrm{S}} : l_i^{\mathrm{S1}} + l_i^{\mathrm{S}} + l_{\mathrm{min}}^{\mathrm{S2}}] \cdot \lambda + x_j[l_j^{\mathrm{S1}} + l_j^{\mathrm{S}} : l_j^{\mathrm{S1}} + l_j^{\mathrm{S}} + l_{\mathrm{min}}^{\mathrm{S2}}] \cdot (1 - \lambda)$ // replace (a part of) S2 with the interp. S2 between x_i and x_j

$l_{\mathrm{min}}^{\mathrm{D}} \leftarrow \min\{l_i^{\mathrm{D}}, l_j^{\mathrm{D}}\}$ // find the length of the shorter diastole

$x_{\mathrm{aug}}[l_i^{\mathrm{S1}} + l_i^{\mathrm{S}} + l_i^{\mathrm{S2}} : l_i^{\mathrm{S1}} + l_i^{\mathrm{S}} + l_i^{\mathrm{S2}} + l_{\mathrm{min}}^{\mathrm{D}}] \leftarrow x_i[l_i^{\mathrm{S1}} + l_i^{\mathrm{S}} + l_i^{\mathrm{S2}} : l_i^{\mathrm{S1}} + l_i^{\mathrm{S}} + l_i^{\mathrm{S2}} + l_{\mathrm{min}}^{\mathrm{D}}] \cdot \lambda + x_j[l_j^{\mathrm{S1}} + l_j^{\mathrm{S}} + l_j^{\mathrm{S2}} : l_j^{\mathrm{S1}} + l_j^{\mathrm{S}} + l_j^{\mathrm{S2}} + l_{\mathrm{min}}^{\mathrm{D}}] \cdot (1 - \lambda)$ // replace (a part of) diastole with the interpolated diastole between x_i and x_j

ing body positions. This is consistent with clinical observations that certain abnormalities are best detected from specific recording positions. PCGmix+ therefore mimics these amplitude variations, upgrading the PCGmix-generated instances to better reflect the clinical diversity observed in practice.

Magnitude warping is performed in the time-series domain, therefore PCGmix+ is exclusively used when the data is in time-series format. Magnitude warping changes the amplitude of each sample in a signal by multiplying the instance with a cubic spline with a set number of knots at random magnitudes [234], [264]. Let us define a set of knots $\mathcal{U} = \{(u_i, v_i)\}_{i=1}^{k_W}$, where k_W is the number of the knots. The knots are evenly distributed along the length of an instance and represent the steps at which the scaling is performed. The values of the knots, v_i, are generated using a Gaussian distribution with the mean μ_W and the standard deviation σ_W, $v_i \sim \mathcal{N}(\mu_W, \sigma_W^2)$, where the mean is fixed to $\mu_W = 1$, and the σ_W is determined empirically. For each time point in the instance, the scaling magnitudes are determined using a cubic spline interpolation of the knots, $S(t, k_W, \sigma_W)$. From now on we omit indicating the dependence of k_W and σ_W in S, unless doing so enhances clarity. It holds that for each subinterval between two consecutive knots, $[u_i, u_{i+1}]$, $S(t)$ is a 3-rd order polynomial, where $i = 1, \cdots, k_W - 1$. It also holds that $S(u_i) = v_i$ for all $i = 1, \cdots, k_W$. Let $S(t, p)$ denote the multichannel interpolation function with the same function $S(t)$ applied to all channels. Magnitude warping transformation can be computed as [264]

$$h_W(x_i, k_W, \sigma_W) = S(t, p, k_W, \sigma_W) \odot x_i(t, p). \tag{3.28}$$

We define PCGmix+ as applying magnitude warping to the output of the PCGmix from Eq. (3.27). The PCGmix+ transformation can be mathematically represented as

$$h_{\mathrm{P+}}(x_i, x_j, \lambda, l_i^{\mathrm{HS}}, l_j^{\mathrm{HS}}, k_W, \sigma_W) = h_{\mathrm{W}}(h_{\mathrm{P}}(x_i, x_j, \lambda, l_i^{\mathrm{HS}}, l_j^{\mathrm{HS}}), k_W, \sigma_W). \tag{3.29}$$

The PCGmix+ is visualized in Figure 3.7. From hereon, we refer to both PCGmix and PCGmix+ collectively as PCGmix(+).

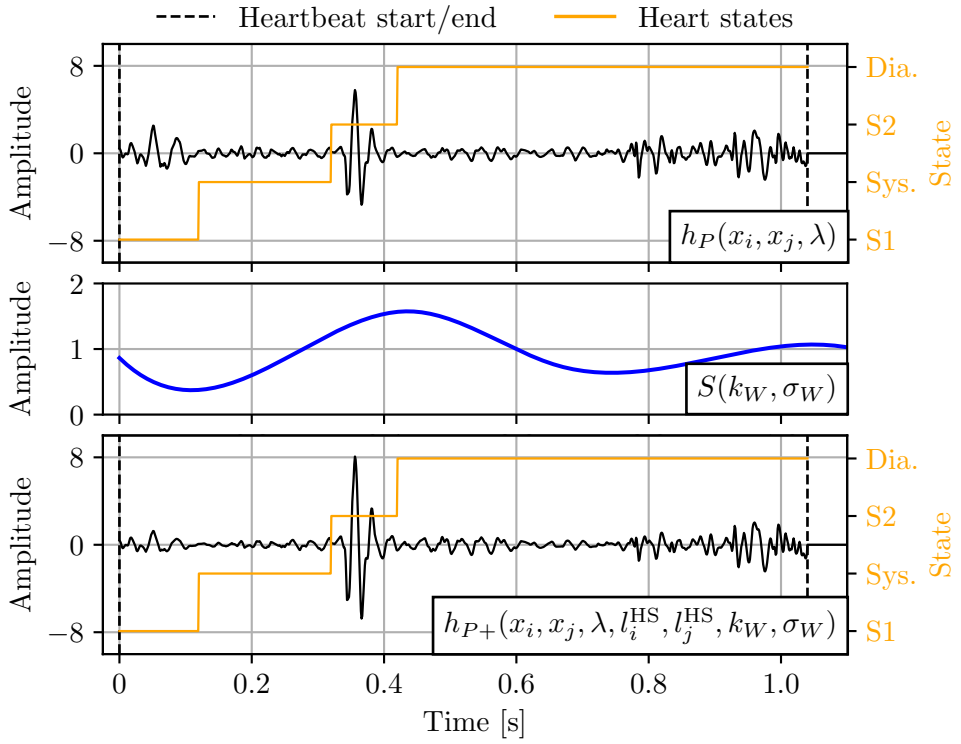


Figure 3.7: Visualization of the PCGmix+ process: (top) $h_P(x_i, x_j, \lambda)$, (middle) warper $S(k_W, \sigma_W)$, and (bottom) $h_{P+}(x_i, x_j, \lambda)$. For clarity, the instances are cropped to the length of 1.1 s and only one channel per instance is shown.

3.4.3 Baselines

We compared PCGmix(+) with the approach without augmentation, as well as with time-series and spectrogram data-augmentation methods used by other authors for heart-sound classification. Specifically, for time-series augmentations, we implemented noise injection, time masking, magnitude warping, and respiratory scaling. For spectrogram augmentations, we used frequency masking, time masking, and Cutout. We omitted time warping from our analysis as it was found to perform significantly worse than other methods. Additionally, we compared our proposed methods with the modality-agnostic state-of-the-art data-augmentation techniques Mixup [262] and Manifold Mixup [265], which are commonly utilized as benchmarks for data augmentation. To determine the baselines' parameters, we initially used the values reported in the original studies, then further refined them through empirical tuning to achieve optimal performance.

Our primary focus was on identifying the most effective augmentation method for heart-sound classification for various dataset sizes. While synthetic data generation methods such as generative models, autoencoders, GANs, and VAEs, as well as other regularization strategies like model ensembles, are valuable, they are beyond the scope of our study and are therefore not included. Implementing these complex models to achieve state-of-the-art performance would be highly time-consuming and therefore will be considered for future work. We did not wish to compare our method against an inadequately tuned or mediocre implementation just to claim superiority. Likewise, we ensured all baseline methods were optimized to perform as well as possible before comparing our approach, to provide a fair and rigorous evaluation.

3.4.3.1 Vanilla

Vanilla represents the baseline where models are trained solely on the original, unmodified data. This approach uses the same NN architectures as the augmentation methods, ensuring that any performance differences are due to the augmentation. It serves as a reference to quantify the impact of data-augmentation techniques, helping to determine if the added variability improves model performance.

3.4.3.2 Mixup

Mixup involves the linear interpolation of two instances at the input level to create a new instance [262]. It can be written as

$$h_{\mathrm{M}}(x_i, x_j, \lambda) = \lambda x_i + (1 - \lambda)x_j, \tag{3.30}$$

with the mixing coefficient $\lambda \sim \mathrm{Beta}(\alpha, \alpha)$ as proposed by [262]. Although Mixup was originally introduced as an image-augmentation technique, it is modality-agnostic.

3.4.3.3 Manifold Mixup

In Manifold Mixup, the concept of Mixup is extended by performing the mixing in the latent space of NNs, as proposed by V. Verma et al. [265]. This adaptation allows Manifold Mixup to be applied to diverse data representations, in contrast to Mixup, which is primarily designed for the image domain.

For a latent space g, the Manifold Mixup can be written as

$$h_{\mathrm{MM}}(x_i, x_j, \lambda) = \lambda g(x_i) + (1 - \lambda)g(x_j). \tag{3.31}$$

The implementation of the Manifold Mixup for time-series classification was previously reported in [266], where it showed promising performance on human-activity and sleep-stage datasets.

3.4.3.4 Standard Augmentations

Standard augmentations were applied to a single data instance, x_i, to generate a new one. Such a process can be written as

$$h_{\mathrm{S}}(x_i) = p(x_i), \tag{3.32}$$

where p represents either a single augmentation or a composition of augmentations.

3.4.3.4.1 Noise Injection We employed noise injection in the time-series domain. It involves adding random noise to a signal while maintaining s specified signal-to-noise ratio (SNR), where the amplitude of the noise signal follows a Gaussian distribution [267]. While the choice of a Gaussian distribution could be further explored, it is beyond the scope of this work. Let $r(t)$ represent noise of length n, where each value is sampled from a Gaussian distribution, $r(t) \sim \mathcal{N}(\mu_N, \sigma_N^2)$, with the mean of the distribution fixed to $\mu_N = 0$. SNR is measured as the the ratio between the signal power and the noise power. In decibels, SNR is defined as

$$\mathrm{SNR}_{\mathrm{dB}} = \left(\frac{P_{\mathrm{signal}}}{P_{\mathrm{noise}}}\right)_{\mathrm{dB}} = 10\log\left(\frac{P_{\mathrm{signal}}}{P_{\mathrm{noise}}}\right) = 10\log\left(\frac{\sum_{t=1}^{n}|x(t)|^2}{\sum_{t=1}^{n}|r(t)|^2}\right). \tag{3.33}$$

To inject noise with a specific $\mathrm{SNR}_{\mathrm{dB}}$, we must compute the standard deviation of the Gaussian distribution from which thenoise is sampled, σ_N. It can be derived from Eq. (3.33) by first computing the noise power as

$$P_{\mathrm{noise,dB}} = 10\log(P_{\mathrm{signal}}) - \mathrm{SNR}_{\mathrm{dB}}, \tag{3.34}$$

transforming it to Watt (W) units as

$$P_{\text{noise}} = 10^{\frac{P_{\text{noise,dB}}}{10}}, \tag{3.35}$$

and taking the sqare root to obtain

$$\sigma_N = \sqrt{P_{\text{noise}}}. \tag{3.36}$$

Putting together Eqs. (3.33, 3.34, 3.35, 3.36), σ_N can be computed as [268]

$$\sigma_N = \sqrt{10^{\frac{10\log\left(\frac{\sum_{t=1}^{n}|x(t)|^2}{n}\right) - \text{SNR}_{\text{dB}}}{10}}}. \tag{3.37}$$

Instead of using a fixed SNR_{dB} for all transformations, we generated noise by randomly sampling SNR_{dB} from an interval with predefined lower and upper limits, $\text{SNR}_{\text{dB,min}}$ and $\text{SNR}_{\text{dB,max}}$. The multichannel noise injection transform, where each channel is injected with the noise of the same SNR_{dB}, can thus be written as

$$h_{\text{NI}}(x_i, \text{SNR}_{\text{dB,min}}, \text{SNR}_{\text{dB,max}}) = r(t, p, x(t, p), \text{SNR}_{\text{dB,min}}, \text{SNR}_{\text{dB,max}}) + x_i(t, p). \tag{3.38}$$

In real-world scenarios, heart-sound recordings might be contaminated by various types of noise such as ambient sounds, breathing, lung sounds, contact between the recording device and the body, and speech. By adding noise during training, the model learns to differentiate actual heart sounds from noise.

3.4.3.4.2 Magnitude Warping We employed magnitude warping in the time-series domain. The transform has already been described in Section 3.4.2.2 in text and defined with Eq (3.28).

Magnitude warping modifies amplitudes across different segments of PCGs, mimicking different recording body positions. This is consistent with clinical observations that various abnormalities are best detected from specific recording positions, e.g., the aortic auscultation area in the second right intercostal space [7]. By mimicking amplitude variations using magnitude warping, the augmented heart sounds better reflect clinical diversity in practice.

3.4.3.4.3 Respiratory Scaling We employed respiratory scaling in the time-series domain. Its goal is to simulate breathing by generating sinusoidal weights that are multiplied to the signal at each sample point [213]. Let $s(t)$ represent sinusoidal weights defined as

$$s(t) = \sin(\omega_S t + \phi_S) \tag{3.39}$$

where ϕ_S and ω_S denote the phase and the respiration rate, respectively. The phase denotes the shift of the weights along the time axis, ranging from 0 to 2π, $\phi_S \in (0, 2\pi)$. The respiration rate determines the number of breathing cycles n_S per one minute as

$$\omega_S = \frac{n_S}{60\,\text{s}}. \tag{3.40}$$

Instead of using a fixed respiration rate for all transformations, we randomly sampled it from an interval with predefined lower and upper limits, $\omega_{S,\text{min}}$ and $\omega_{S,\text{max}}$. The multichannel respiratory scaling transform, where each channel is multiplied with the same sinusoidal weights, is computed as

$$h_{\text{S}}(x_i, \omega_{S,\text{min}}, \omega_{S,\text{max}}) = s(t, p, \omega_{S,\text{min}}, \omega_{S,\text{max}}) \odot x_i(t, p). \tag{3.41}$$

Breathing can be heard in cycles, with the sounds of inhalation and exhalation often overpowering heart sounds. Respiratory scaling simulates this effect by periodically diminishing the power of the heart-sound signal through multiplication with sinusoidal weights. This forces the model to recognise heart sounds even when they are obscured by breathing noise.

3.4.3.4.4 Time Masking We employed time masking in both time-series and spectrogram domains. Time masking creates a blank band along the time axis of the instance, simulating interruptions such as hitting the stethoscope, which generates a pulse-like sound that overpower all other sounds.

This augmentation can be computed by multiplying a masking matrix (introduced in Section 3.4.1) with an instance as

$$h_{\text{TM}}(x_i, t_{\min}, t_{\max}) = M(t_{\min}, t_{\max}) \odot x_i(t, p). \tag{3.42}$$

Here, the $t_{\min}$ and $t_{\max}$ are the lower and higher bounds of the interval within which the values in the instance are set to zero. Instead of specifying these bounds directly, we denote the percentage of the signal being masked out as

$$p_{\text{TM}} = \frac{t_{\max} - t_{\min}}{n}. \tag{3.43}$$

The percentage, p_{TM}, is randomly sampled from an interval between 0 and $p_{\text{TM, max}}$, $p_{\text{TM}} \in (0, p_{\text{TM, max}})$, where $p_{\text{TM, max}}$ is the maximum allowable percentage of the signal to be masked. The starting position of the masking is also selected randomly.

Time masking effectively creates a blank region in a PCG with no information, challenging the model to handle sudden loss of data, thereby improving its robustness and performance in real-world scenarios where such interruptions are common.

3.4.3.4.5 Frequency Masking Frequency masking was employed in the spectrogram domain. Similar to time masking, which creates a blank band along the time axis of an instance, frequency masking creates a blank band along the frequency axis [269].

Frequency masking can be computed by multiplying a frequency masking matrix with an instance as

$$h_{\text{FM}}(x_i, p_{\min}, p_{\max}) = M_{\text{freq}}(p_{\min}, p_{\max}) \odot x_i(t, p). \tag{3.44}$$

Here, $M_{\text{freq}}(p_{\min}, p_{\max})$ is a binary rectangular mask of size $n \times m$, defined as

$$M_{\text{freq}}(p_{\min}, p_{\max}) = \{(i, j) | p_{\min} \leq j < p_{\max}\}, \tag{3.45}$$

where $p_{\min}$ and $p_{\max}$ are the lower and upper bounds of the interval within which the values along the frequency axis in the instance are set to zero. Instead of specifying these bounds directly, we denote the percentage of the signal being masked out as

$$p_{\text{FM}} = \frac{p_{\max} - p_{\min}}{n}. \tag{3.46}$$

The percentage, p_{FM}, is randomly sampled from an interval between 0 and $p_{\text{FM, max}}$, $p_{\text{FM}} \in (0, p_{\text{FM, max}})$, where $p_{\text{FM, max}}$ is the maximum allowable percentage of the signal to be masked. The starting position of the masking along the frequency axis is also selected randomly.

Medical machines often generate sounds within specific frequency bands, which are visible on a spectrogram. Frequency masking involves deliberately blanking such regions, effectively simulating the real-world scenario where machine noise can obscure parts of a heart-sound recording.

3.4.3.4.6 Cutout We employed Cutout in the spectrogram domain. Cutout is an image-augmentation technique where a patch of pixels is masked out [270], which in our case, is a combination of both time and frequency masking. Instead of masking two separate bands, a rectangular area of a spectrogram is set to zero.

The Cutout transform can be computed as

$$h_{\mathrm{CO}}(x_i, t_{\min}, t_{\max}, p_{\min}, p_{\max}) = M_{\mathrm{cutout}}(t_{\min}, t_{\max}, p_{\min}, p_{\max}) \odot x_i(t, p), \tag{3.47}$$

where, $M_{\mathrm{cutout}}(t_{\min}, t_{\max}, p_{\min}, p_{\max})$ is a binary rectangular mask of size $n \times m$, defined as

$$M_{\mathrm{cutout}}(t_{\min}, t_{\max}, p_{\min}, p_{\max}) = \{(i, j) | t_{\min} \leq i < t_{\max}, \text{and } p_{\min} \leq j < p_{\max}\}. \tag{3.48}$$

According to definitions for time and frequency masking from Eqs. (3.43) and (3.46), we define $p_{\mathrm{CO,TM,\ max}}$ and $p_{\mathrm{CO,FM,\ max}}$ as the maximum allowable percentages of the signal to be masked out along the time and frequency axis, respectively.

Cutout simulates situations where specific background noise is observed, or parts of the spectrogram information are randomly dropped. This augmentation technique helps the model become robust to missing or corrupted data.

The visual representations of the baseline augmentations used in our study are given in Figure 3.8 and Figure 3.9.

3.4.4 Deep-Learning Models

This section describes the DL models used in the experiments. The formulation begins with the mathematical foundations of feedforward multilayer perceptrons and an explanation of the most common concepts in DL. The architectures of the implemented models are described in Sections 3.4.4.1, 3.4.4.2, and 3.4.4.3

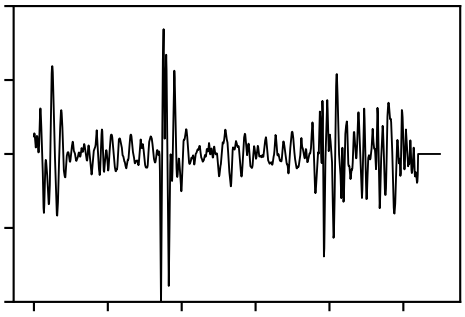
The fundamental architecture in DL is a feedforward multiLayer perceptron (MLP), depicted in Figure 3.10. In general, a classifier is designed to map an input vector $\boldsymbol{x}$ to a target y, which follows a target function $y = f^*(\boldsymbol{x})$. The MLP utilizes a mapping $\boldsymbol{y} = f(\boldsymbol{x}; \boldsymbol{\theta})$ to learn the optimal parameters $\boldsymbol{\theta}$ that approximate f^* most effectively. DL models, often referred to as networks, are composed of multiple functions or layers, denoted as $f(\boldsymbol{x}) = f^{(3)}(f^{(2)}(f^{(1)}(\boldsymbol{x})))$. The layers between the input and the output layers are also called the hidden layers. Each layer contains numerous units, known as neurons, which operate in parallel. These neurons represent vector-to-scalar functions, where the vector denotes the output of the preceding layer. If every neuron in a layer is connected to every neuron in the subsequent layer, the network is referred to as a *dense network*. A neuron's mathematical model, termed a perceptron, combines a linear model and a nonlinear activation function, expressed as

$$y = g(\boldsymbol{w}\boldsymbol{x} + b). \tag{3.49}$$

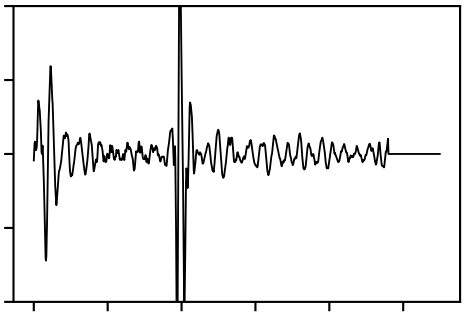
Here, $\boldsymbol{w}$ and b denote the weights and bias term of the linear model, respectively, whereas g represents the activation function. This activation function imparts nonlinearity to the perceptron and consequently to the entire network. The prevalent choice for activation functions in NNs today is the rectified linear unit (ReLU), defined as $g(z) = \max\{0, z\}$. The output vector of the l-th layer, as per Eq. (3.49), is computed as

$$\boldsymbol{y_l} = g(\boldsymbol{W_l}\boldsymbol{y_{l-1}} + \boldsymbol{b_l}). \tag{3.50}$$

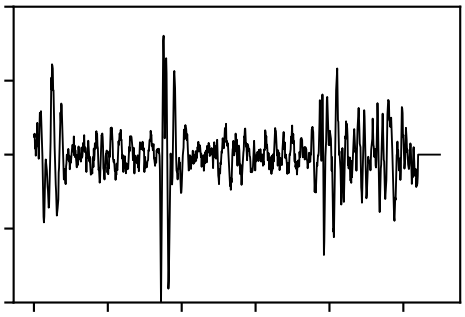
Here, $\boldsymbol{W_l}$ refers to the weight matrix of dimensions $m^{(l)} \times m^{(l-1)}$, where $m^{(l)}$ and $m^{(l-1)}$ represent the lengths of the vectors in the l-th and $(l-1)$-th layers, respectively [231].



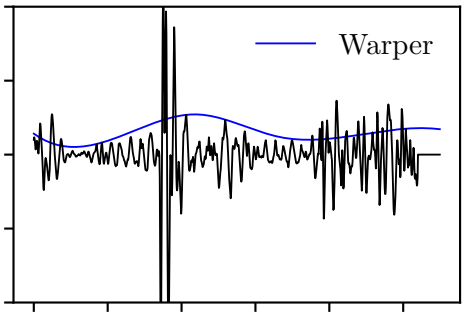
(a) x_i



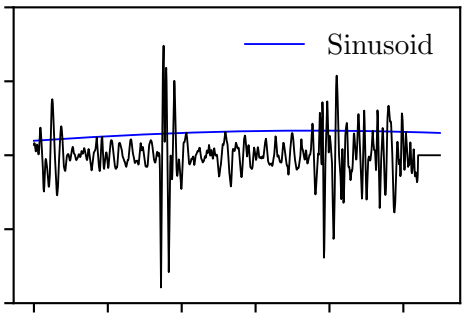
(b) x_j



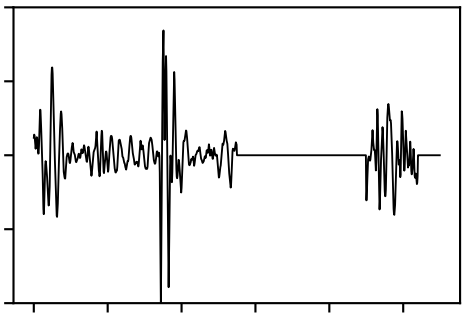
(c) Noise injection: $h_{\mathrm{NI}}(x_i)$



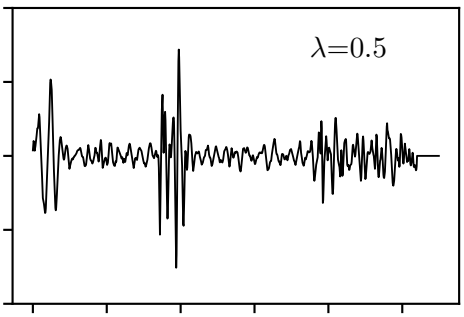
(d) Magnitude warping: $h_{\mathrm{W}}(x_i)$



(e) Respiratory scaling: $h_{\mathrm{S}}(x_i)$

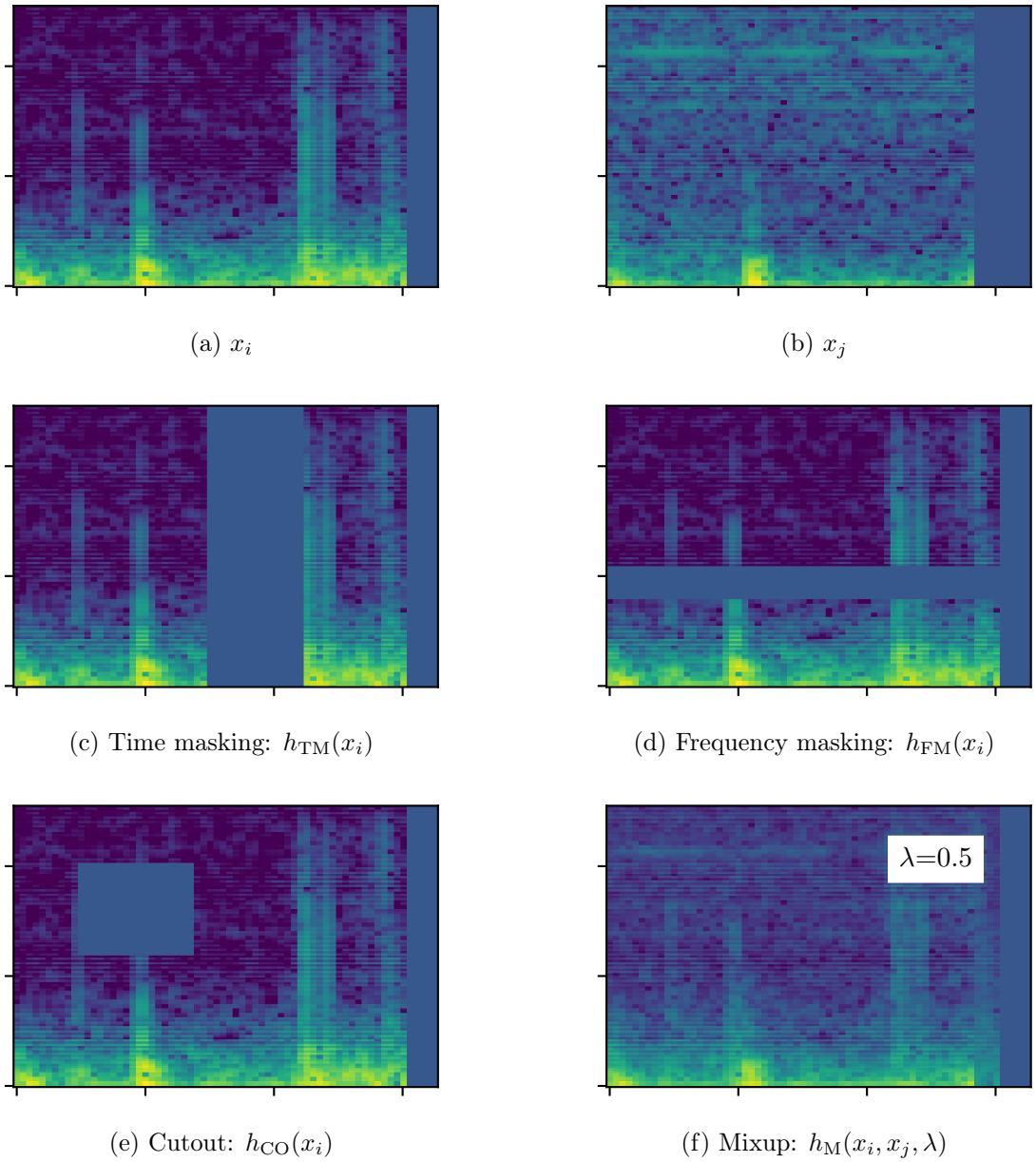


(f) Time masking: $h_{\mathrm{TM}}(x_i)$



(g) Mixup: $h_{\mathrm{M}}(x_i, x_j, \lambda)$

Figure 3.8: Visualization of baseline augmentation methods in the time-series domain. Note that Manifold Mixup is not shown here as it takes place in the latent space.



(a) x_i

(b) x_j

(c) Time masking: $h_{\mathrm{TM}}(x_i)$

(d) Frequency masking: $h_{\mathrm{FM}}(x_i)$

(e) Cutout: $h_{\mathrm{CO}}(x_i)$

(f) Mixup: $h_{\mathrm{M}}(x_i, x_j, \lambda)$

Figure 3.9: Visualization of baseline augmentation methods in the spectrogram domain. Note that Manifold Mixup is not shown here as it takes place in the latent space.

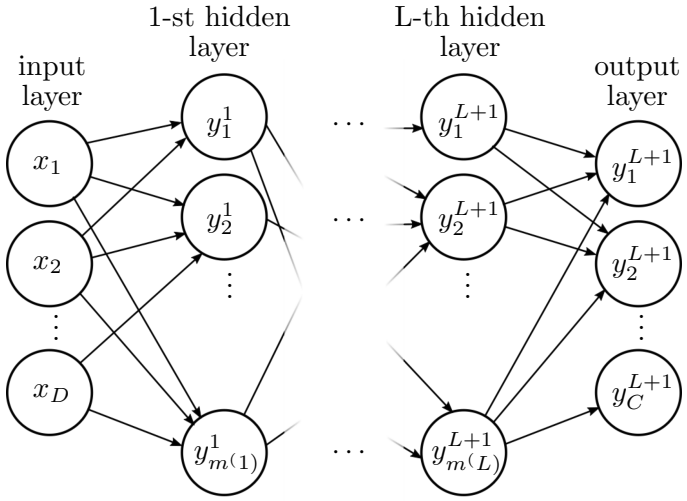


Figure 3.10: Netwrok graph of a multilayer perceptron with D input units, L hidden layers, and C output units. The l-th hidden layer consists of $m^{(l)}$ units.

The SoftMax activation function is crucial for ensuring that the output vector of a network's output layer represents a valid probability distribution over the classes or categories in the dataset. With the number of units in the output layer corresponding to the number of classes, the output vector $\hat{\boldsymbol{y}}$, comprising elements $\hat{y}_i = P(y = i|\boldsymbol{x})$, should be a unit vector, meaning each element $\hat{y}_i$ lies between 0 and 1, and the sum of all elements $\sum_i \hat{y}_i$ equals 1. To achieve this desired output vector $\hat{\boldsymbol{y}}$, the SoftMax activation function is typically employed at the output layer. The SoftMax function transforms the linear transformation $\boldsymbol{z} = \boldsymbol{W_{L+1}}\boldsymbol{y_L} + \boldsymbol{b_{L+1}}$ of the output layer, where L denotes the number of hidden layers in the network, into a valid probability distribution. Mathematically, the SoftMax function is defined as

$$\text{softmax}(\boldsymbol{z}) = \frac{\exp(z_i)}{\sum_{j=1}^{C}(z_j)}, \tag{3.51}$$

where C represents the number of units in the output layer [231].

A feedforward network accepts input $\boldsymbol{x}$ and produces output $\hat{\boldsymbol{y}}$ by propagating the information forward through the network's layers. During training, this process results in a scalar loss $J(\boldsymbol{\theta})$, which quantifies the discrepancy between the predicted output $\hat{\boldsymbol{y}}$ and the true target $\boldsymbol{y}$. The backpropagation algorithm allows the information to flow backward from the loss through the network to compute the gradient of the loss function with respect to the model parameters, denoted as $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$. This gradient indicates how much the loss would change with a slight adjustment to each parameter. An optimization algorithm, such as gradient descent, uses this gradient to perform learning by updating the model parameters, including weights and biases, to minimize the loss. The parameter updates are made in the direction that reduces the loss, iteratively refining the model's predictions. We used the cross-entropy loss function in our experiments. It measures the difference between the true and predicted probability distributions and is defined as

$$J(\boldsymbol{\theta}) = -\sum_{j=1}^{C} \hat{p}_j(\boldsymbol{x}, \boldsymbol{y}) \log p_j(\boldsymbol{y}|\boldsymbol{x}), \tag{3.52}$$

where $\hat{p}_j$ is true probability distribution and p_j is the model's predicted probability distribution for the j-th class [231].

The three most important hyperparameters that must be set before starting the network training are the batch size, the learning rate and the number of epochs. The *batch size* is the number of training examples that are processed in a single forward/backward pass. As the entire data set is usually too large to process at once, it is divided into batches. Smaller batches provide a more accurate gradient estimate but are computationally expensive, whereas larger batches are faster but less accurate. The *learning rate* is a hyper-parameter that controls the step size of the change in the model parameters with respect to the gradient of the loss function. Higher learning rates allow larger steps, which speeds up learning but risks overshooting the optimal values, whereas lower rates allow accurate updates but slow down learning. An *epoch* refers to a complete pass through the entire training dataset, with the network processing all batches once per epoch. The number of epochs determines how often the learning algorithm will work through the training data [231].

3.4.4.1 1D-CNN

The first model we employed is a 1-dimensional CNN. We refer to it as the "1D-CNN". It was designed to work with time-series data as input.

CNNs are specialised for processing data with grid-like topology, such as time-series data and images. Time-series data can be seen as a 1-dimensional grid of samples at regular intervals, while images are 2-dimensional grids of pixels. Unlike traditional NNs, CNNs use convolutional operations instead of matrix multiplication in their convolutional layers. Convolution between two discrete functions, x (input) and w (kernel or filter), is defined as

$$s(t) = (x * w)(t) = \sum_{-\infty}^{\infty} x(a)w(t-a). \tag{3.53}$$

The kernels, which are adapted during training, slide across the input data, performing element-wise multiplication and summing the results to produce a single output pixel. For instance, applying a 2×2 kernel to a 3×4 image results in a 2×3 output. Figure 3.11 illustrates this process, where the kernel slides over the height and width of the image. In one-dimensional CNNs, the kernel moves only in one direction, i.e., along the time axis. Kernels are crucial for feature extraction in CNNs, becoming specialised to detect edges, textures, colours, and complex patterns in images, or identifying trends, seasonal patterns, anomalies, and frequency components in time-series data. As the network deepens, the kernels recognise increasingly complex patterns [231].

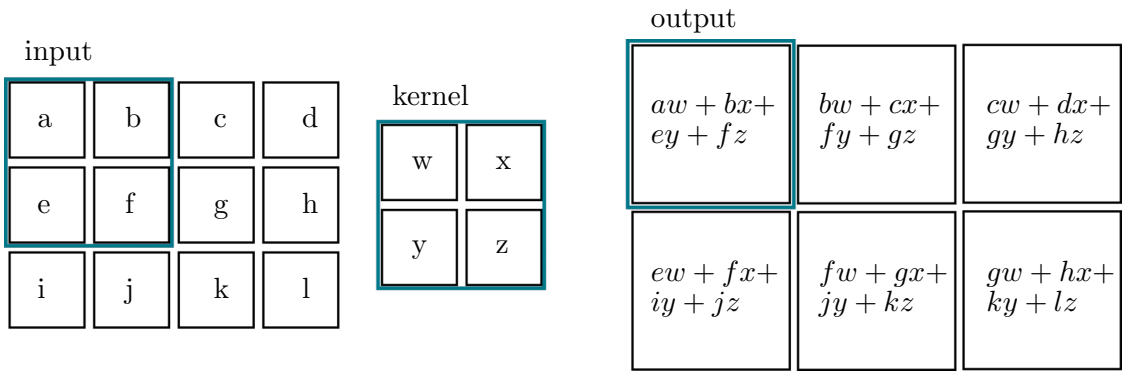


Figure 3.11: Example of a 2-dimensional convolution.

A convolutional layer typically comprises three stages. First, the layer performs a series of convolutions in parallel, where kernels slide over the input data to produce feature maps. Next, an activation function, such as ReLU, is applied to these feature maps to introduce non-linearity. Finally, the pooling function down-samples the data in terms of width and height while retaining the most important information. An example of a pooling function is max-pooling, which selects the maximum value within a rectangular neighborhood of the feature map. This process reduces the spatial dimensions of the data, decreases the computational load, and helps to make the network invariant to small translations of the input [231].

The architecture of the 1D-CNN used in our experiments was developed by Potes et al. [216]. It was developed specifically for the classification of heart sounds and was used in their entry for the PhysioNet/Cinc Challenge 2016, with which they won 1-st place in the competition. It consists of four identical, three-layer, 1-dimensional CNNs and a dense network, with each 1-dimensional CNN processing a distinct frequency band as input. The three layers of the CNNs are the input layer and the two convolutional layers. The first convolutional layer consists of 8 filters of size 5, ReLu activation, and max-pooling of size 2. The second convolutional layer consists of 4 filters of size 5, ReLu activation, a max-pooling of size 2, and a dropout of 0.25. The outputs of the four CNNs are flattened and fed into a dense network with 20 hidden neurons and a dropout of 0.5, and two output neurons to determine the probabilities for each of the two class labels. The model consists of approximately 200 thousand learnable parameters. The architecture of the model is shown in Figure 3.12

3.4.4.2 1D-ResNet

The second model we employed is a 1-dimensional residual network. We refer to it as the "1D-ResNet". It was designed to work with time-series data as input.

Residual networks (ResNets) are a type of deep NN architecture designed to address the problem of vanishing and exploding gradients in very deep networks. These issues can make training deep networks challenging, as the gradients used for updating the weights can become extremely small or large, leading to ineffective learning. The key innovation in ResNets is the introduction of residual connections, or skip connections, which allow the network to learn residual functions relative to the layer inputs. In a residual block, the output of one layer is added directly to the output of a deeper layer within the block, bypassing the intermediate layers. This approach is represented by the equation

$$y = F(x) + x,$$

where x is the input to the residual block, $F(x)$ represents the function composed of the layers within the block, and the term $+x$ denotes the skip connection. This skip connection performs an identity mapping that links the input of the subnetwork to its output. An example of a residual block can be seen in Figure 3.13. The architecture of ResNet is built by stacking multiple residual blocks together. This design helps in mitigating the degradation problem, where adding more layers to a deep network can degrade its performance [271].

The 1D-ResNet consists a total of eight 1-dimensional convolutional layers. Each convolutional layer is followed by a batch normalisation and ReLU activation. The convolutional layers are organised into two stages. The first stage includes a layer with 64 filters, a layer with 128 filters and max-pooling of size 2, and a residual block comprising two layers with 128 filters. The second stage includes a layer with 256 filters and max-pooling of size 2, a layer with 512 filters and max-pooling of size 2, and a residual block comprising two

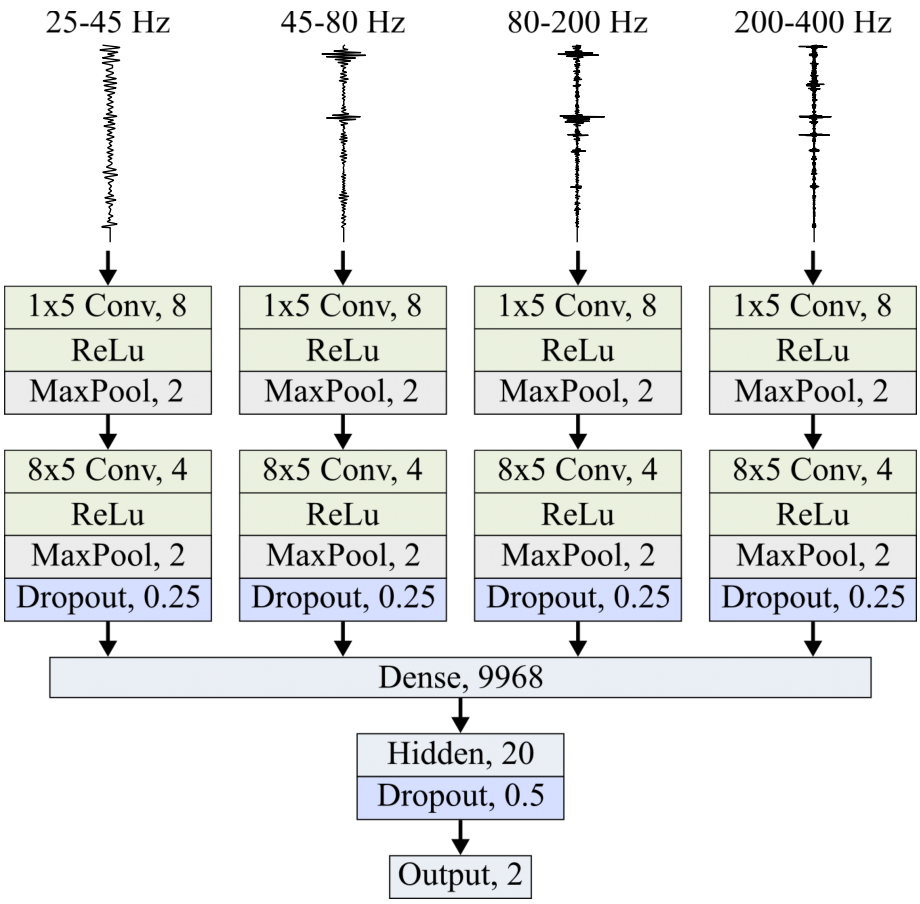


Figure 3.12: Architecture of 1D-CNN.

layers with 512 filters. The second residual block is followed by a max-pooling of size 4, a dense network with 39936 neurons, and two output neurons. The model consists of approximately 2.3 million learnable parameters. We experimented with varying the number of filters in the convolutional layers and the number of neurons in the dense layer, testing architectures ranging from 50 thousand to 9 million learnable parameters. However, no configuration demonstrated consistently superior performance, so we opted for the default 2.3 million-parameter model, as it is a standard architecture. The architecture of the model is shown in Figure 3.14.

3.4.4.3 2D-ResNet

The third model is a 2-Dimensional ResNet, referred to as the "2D-ResNet", designed to work with spectrogram data as input. Its structure mirrors that of the 1D-ResNet (see Figure 3.14), except that the 1-dimensional convolution and max-pooling layers are replaced with their 2-dimensional counterparts. The kernel sizes of the convolutions are 3×3. Additionally, due to the different input data sizes, the number of neurons in the dense layers was reduced to 8192. The model consists of approximately 6.6 million learnable parameters.

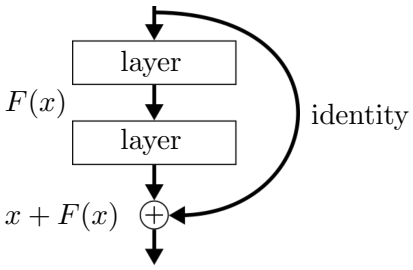


Figure 3.13: Residual block: the residual connection skips two layers.

3.4.5 Data Partitioning

To evaluate the effect of augmentation on datasets of varying sizes, we partitioned our training set into smaller, intersecting subsets, stratified by class label. These subsets are denoted as $\mathcal{N}_f^{seed}$, where f represents the fraction of our whole training data, $f \in [0, 1]$, and the *seed* represents a random-number-generator (RNG) seed that samples the recordings to be included in the subset. The use of a data-sampling seed ensures that the process of selecting the subset can be reproduced exactly, thereby guaranteeing consistency and repeatability in the subset's composition. It holds that $\mathcal{N}_{1.0}^{seed_i} = \mathcal{N}_{1.0}^{seed_j}$ for all $seed_i, seed_j \in \mathbb{N}$ as there is only one way of selecting all of the training data. In addition, the random seed influenced the shuffling of the data during model training, thereby introducing variability of the experimental results with $f = 1.0$

For each selected training-data fraction f, we performed a series of experiments to obtain robust results:

- Time-series domain: For each f, we conducted a total of $\frac{5}{f}$ experiments. This approach ensured that the number of PCGs included in these experiments equaled five times the number of PCGs in the whole training set. This extensive sampling helped mitigate randomness and minimized potential biases, and aimed to yield representative results that accurately reflected underlying data trends and patterns.

- Spectrogram domain: For each f, we conducted a total of $\frac{3}{f}$ experiments. We found that in the spectrogram domain, the results were more consistent with respect to different RNG data-sampling seeds and thus converged faster.

It is important to note that we ran significantly more experiments for smaller values of f than for larger values. This is because there are many more ways to select a smaller subset of data than there are ways to select larger subsets, making the results for smaller fractions much more variable.

3.4.6 Experimental Setup

For model optimization, we employed the Adam algorithm [272], and selected the cross-entropy loss function J. Additionally, we implemented a one-cycle learning-rate scheduler [273], which dynamically adjusts the learning rate during the training. This scheduler starts with an initial small value, increases to a maximum learning rate α, and then decreases back to a very small value by the end of training. The graph depicting learning rate changes during training with a one-cycle scheduler is shown Figure 3.15.

Data augmentation was applied dynamically during model training, with new instances generated "on-the-fly" for each training batch. Rather than pre-generating and adding new

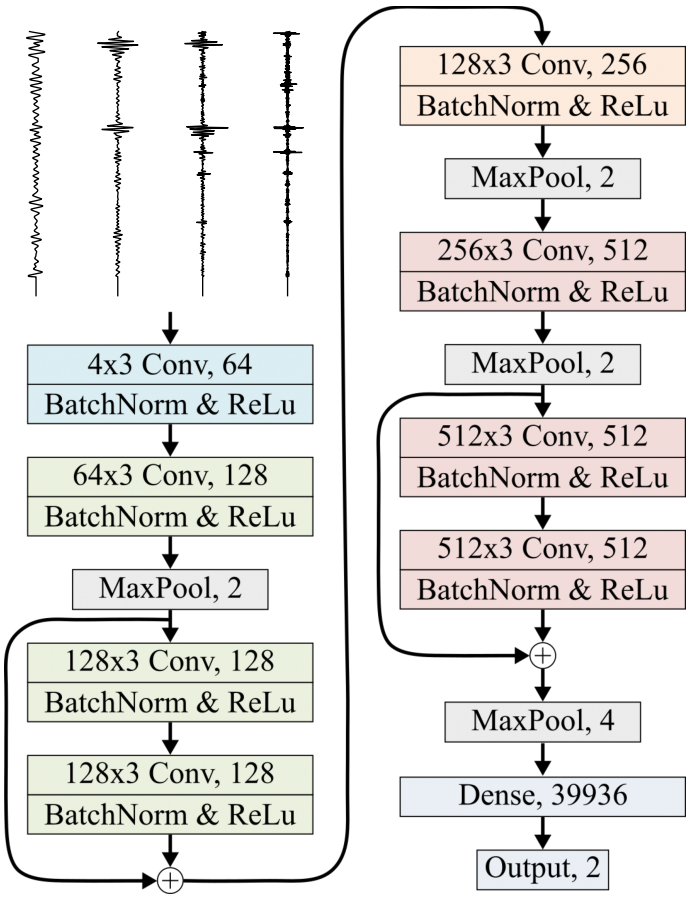


Figure 3.14: Architecture of 1D-ResNet.

instances to the training set, this approach randomly decides for each batch whether to use original data or to generate augmented data that replaces the original batch. This method is a standard practice in data augmentation [262], [265], as it allows the model to be trained in a standard manner while continuously introducing variation, enhancing robustness without increasing dataset size upfront. The model-training pipeline is shown in the Algorithm 3.3. If a batch underwent augmentation (this was determined by the augmentation probability parameter, μ), the whole batch was replaced with an augmented version. When an augmentation technique that requires two original instances as the input was employed, a one-to-one mapping function $\phi : \mathcal{B} \rightarrow \mathcal{B}$ was used to determine the pairs for the augmentation $h(x_i, \phi(x_i) = x_j)$. The mapping function randomly mapped each instance to an instance with the same class label, ensuring $y_i = y_j$ and preserving the labels. Each instance of $\mathcal{B}$ played a role of x_i and x_j inputs to the augmentation function $h(x_i, x_j)$ exactly once. Consequently, the augmented batch had the same size as the original batch, and no original instance was over- or under-represented. If an augmentation method that takes a single instance input was employed, method $h(x_i, x_j)$ in Algorithm 3.3 was replaced with $h(x_i)$, and no mapping was needed.

After the model was trained, it was tested on the test data. The model predicted the class probabilities for each heartbeat. The predictions for the beats in the same recording were averaged to determine the class probabilities for the entire recording. The higher probability then determined the predicted class label. Notably, the performance metrics calculated at the beat level were closely aligned with those calculated at the recording level,

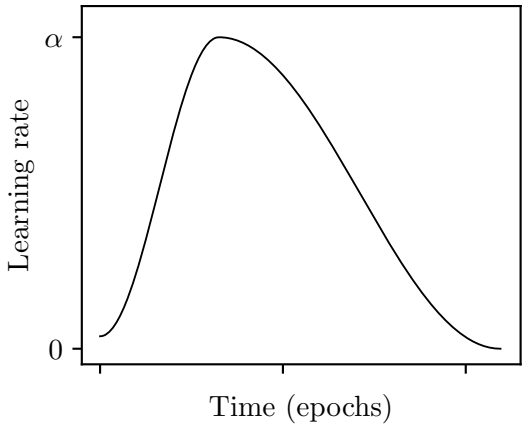


Figure 3.15: Learning rate progression during training using the one-cycle learning-rate scheduler.

indicating consistency in model accuracy across both levels of evaluation. More advanced methods for this task, like meta-learning Markov models, are considered for future work.

3.4.7 Implementation Details

We trained the models on four NVIDIA GeForce RTX 2080 GPUs. We used training-data fractions of 1.5%, 5.2%, 10%, 20%, 30%, 40%, 60%, 80%, and 100%, which correspond to 10, 30, 56, 112, 168, 222, 334, 444, and 554 PCG recordings, respectively. For each fraction, we conducted the following number of experiments in the time-series domain: 333, 100, 50, 25, 16, 12, 8, 6, and 5, respectively. In the spectrogram domain, we ran 200, 60, 30, 15, 10, 8, 5, 4, and 3 experiments, respectively, for each fraction.

As the core experiment was already considerably time-consuming, it was not feasible to perform further fine-tuning, and we fixed all of the hyper-parameters in advance based on initial experimentation conducted on the full training set of 554 PCGs. We fixed the batch size to $\nu = 64$, epoch number to $\epsilon = 50$, and maximal learning rate of the one-cycle learning-rate scheduler to $\alpha = 0.01$. The settings were reasonably close to optimal for all NN models, with all the experiments successfully converging with little to no over-fitting. The augmentation probabilities (μ) were determined for each f and each method. For low fs (1.5%, 5.2%, 10%), the probabilities were set to 1.0, while for high fs (80%, 100%) they were set to 0.2. Values in between were adjusted accordingly with a granularity of 0.2, which was determined separately for each method by initial experimentation. To reproduce the experiments, we provided the instructions in Appendix A.1 and A.3.

The methods employed include parameters that had to be determined. For PCGmix(+), Mixup, and Manifold Mixup we sampled the mixing coefficient λ from a uniform distribution, $\lambda \sim U(0, 1)$. For noise injection we used a random SNR between $\mathrm{SNR}_{\mathrm{dB,min}} = 25\,\mathrm{dB}$ and $\mathrm{SNR}_{\mathrm{dB,max}} = 40\,\mathrm{dB}$. Fo the cubic spline used in the magnitude warping and PCGmix+, we used $k_W = 4$ knots with amplitudes sampled from Gaussian distribution with $\sigma_W = 0.2$ and $\mu_W = 1$. For respiratory scaling, we sampled the respiration rate from an interval with $\omega_{S,\mathrm{min}} = \frac{12}{60\,\mathrm{s}}$ and $\omega_{S,\mathrm{max}} = \frac{18}{60\,\mathrm{s}}$, covering the most common respiration rates. The phase was randomly sampled from an interval between 0 to 2π, $\phi_S \in (0, 2\pi)$. For time masking, we limited the masked region to a single interval of up to $p_{\mathrm{TM,\,max}} = 20\%$ of a heartbeat's length. For frequency masking, we limited the masked region to a single interval of up to $p_{\mathrm{FM,\,max}} = 20\%$ of a heart-

Algorithm 3.3: Neural network model training with data augmentation.

Input: $\mathcal{N}$, training data; M, model; J, loss function; ϵ, epoch number; ν, batch size; α, learning rate; h, augmentation method; μ, augmentation probability; ϕ, mapping function

Output: θ, model parameters (weights and biases)

$\theta \leftarrow$ randomly initialize all parameters in M
$\kappa \leftarrow$ calculate the number of batches of size ν per each epoch
for $e \leftarrow 1$ **to** ϵ **do**
  $\left\{\mathcal{B}^b = \left\{(x_i^b, y_i^b)\right\}_{i=1}^{\nu}\right\}_{b=1}^{\kappa} \leftarrow$ randomly split $\mathcal{N}$ into κ batches
  for $b \leftarrow 1$ **to** κ **do**
    $r \leftarrow$ random value sampled from $U(0, 1)$
    if $\mu > r$ **then**
      $\mathcal{B}_{\text{aug}}^b \leftarrow$ initialize empty augmented batch
      for $i \leftarrow 1$ **to** ν **do**
        $x_j^b \leftarrow \phi(x_i^b)$ // find the same label pairing instance from the same batch
        $x_{\text{aug},i}^b \leftarrow h(x_i^b, x_j^b)$ // generate a new instance
        $\mathcal{B}_{\text{aug}}^b[i] \leftarrow (x_{\text{aug},i}^b, y_i^b)$ // add the new instance to the augmented batch
      end for
      $\mathcal{B}^b \leftarrow \mathcal{B}_{\text{aug}}^b$ // replace the original batch with the augmented one
    end if
    output $\leftarrow M\left(\{x_i^b\}_{i=1}^{\nu}\right)$
    loss $\leftarrow J(\text{output}, \{y_i^b\}_{i=1}^{\nu})$
    loss.BackwardPropagation()
    $\theta \leftarrow$ update weights using α
  end for
end for

beat's frequency bins. For Cutout, we limited the masked region along both axis equally with $p_{\text{CO,TM, max}} = p_{\text{CO,FM, max}} = 25\%$, meaning the maximal possible mask rectangle equaled 6.25% of the total instance. The selected parameter values are also gathered in Appendix A.3.1.

3.5 Performance Metrics

In section, we describe the classification metrics we used to evaluate our results. Let's start with the confusion matrix. It provides a detailed view of the model's performance and helps identify errors. Each row represents instances of an actual class, and each column represents instances of a predicted class. For a binary classification, the confusion matrix consists of four values:

- True positive (TP): the number of instances correctly predicted as positive;
- False positive (FP): the number of instances incorrectly predicted as positive;
- False negative (FN): the number of instances incorrectly predicted as negative;
- True negative (TN): the number of instances correctly predicted as negative.

Accuracy, which is probably the most used performance metric in classification tasks, measures the percentage of all instances that were correctly classified. It is calculated as

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN}. \tag{3.54}$$

However, accuracy is not suitable for imbalanced datasets as it tends to favour the majority class. In addition, it does not distinguish between the numbers of correctly classified examples of different classes. Therefore, we look at some other common metrics.

Precision and recall asses how well the model performs for each class and are thus often used in multi-class (more than two classes) classification tasks. Precision indicates the percentage of true positives among all predicted positives. It is crucial for assessing the quality of the positive predictions made by the model. It is computed as

$$\text{Precision} = \frac{TP}{TP + FP}. \tag{3.55}$$

Recall, also known as sensitivity or the true positive rate (TPR), measures the percentage of actual positives that were correctly identified by the model. It is important for understanding how well the model detects positive instances. It is defined as

$$\text{Recall} = \frac{TP}{TP + FN}. \tag{3.56}$$

The F1-score is the harmonic mean of precision and recall, providing a single metric that balances both metrics. This score is particularly useful when dealing with unbalanced datasets. It is defined as

$$\text{F1-score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{3.57}$$

Specificity, or the true negative rate (TNR), measures the proportion of actual negatives that were correctly identified, highlighting how well the model distinguishes negative instances. It is computed as

$$\text{Specificity} = \frac{TN}{TN + FP}. \tag{3.58}$$

The false positive rate (FPR) denotes the proportion of negative instances that were incorrectly classified as positive, whereas the false negative rate (FNR) indicates the proportion of positive instances that were incorrectly classified as negative. They are computed as

$$\text{FPR} = \frac{FP}{FP + TN} \tag{3.59}$$

and

$$\text{FNR} = \frac{FN}{TP + FN}. \tag{3.60}$$

The connection between sensitivity and specificity can be graphically represented by the receiver operating characteristic (ROC) curve, which plots the TPR against the FPR at various classification thresholds. The area under the curve (AUC) of the ROC quantifies the overall performance of the model. A larger AUC indicates better performance. ROC AUC is a standard metric in fields such as medical diagnostics.

We also introduce a novel metric called the Apparent Data Size Increase (ADSI) that we used to evaluate the performance of data-augmentation techniques. It takes into account the results of all the data-subset sizes with which the experiments were performed and measures how much the dataset's size appears to increase due to the employment of augmentation compared to the no augmentation (vanilla). To compute the ADSI factor

of an augmentation method for the training-data fraction f, we identify the data fraction, f_0, at which the vanilla achieves the same performance as that of the method at f. We define ADSI as

$$\text{ADSI}(f) = \frac{f_0}{f}. \tag{3.61}$$

An ADSI factor value of x denotes that for a given amount of training data f, the augmentation method achieves the same performance as the vanilla when trained on an amount of training data equal to xf. ADSI complements the conventional performance metric(s) by providing a more nuanced assessment of the method's performance in the context of experiments with datasets of different sizes. It indicates how much larger the original dataset would need to be for the vanilla approach to reach the same level of performance as the augmentation method. It is important to note that some results achieved with the augmentation method can lie outside the results range achieved by vanilla, in which cases the ADSI is undefined. Illustrative example of how ADSI is computed is shown in Figure 3.16.

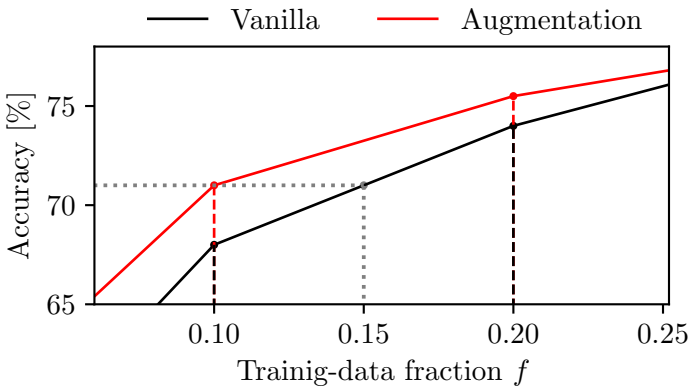


Figure 3.16: Illustration of how to compute the ADSI metric. At $f = 0.1$ the augmentation method achieves the same accuracy as vanilla at $f_0 = 0.15$, resulting in ADSI(0.1) = 1.5.

Chapter 4

Experimental Results

4.1 Detection of Decompensation Episodes of CHF

This section presents the results on detection of decompensation episodes of CHF using classical ML models and domain-specific features extracted from the PCG signals. The feature engineering, the models, and the experimental setup have been detailed in Sections 3.3.1, 2.5.3.1, and 3.3.2, respectively. Our main results have already been published in [274], following our preliminary study published in [275], however, this section includes additional analyses and material.

The UMCL dataset included 898 decompensated and 908 recompensated (1806 in total) PCG segments with 354 features. For details about the dataset, see Section 3.1.1. We used accuracy, precision, recall, F1-score, and ROC AUC as the evaluation metrics, with accuracy as the main metric of performance evaluation.

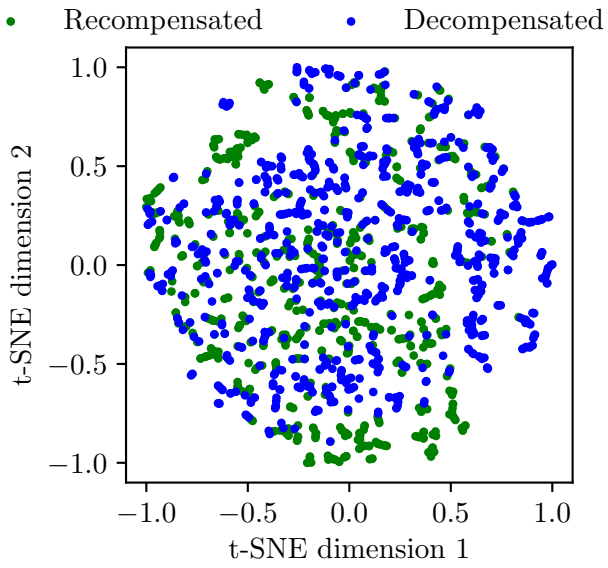


Figure 4.1: Visual representation of the UMCL dataset.

To better understand the underlying structure and relationships within our dataset, we applied t-SNE dimensionality reduction to the 177 extracted features (prior to feature smoothing), reducing them to two components. The resulting 2-dimensional representation is displayed in Figure 4.1. The shows that the two class labels do not form distinct

groups. Instead, smaller clusters of points are visible, corresponding to segments from the same patient. This observation suggests that while the features capture patient-specific characteristics, they do not clearly differentiate between the class labels.

4.1.1 Classification by the Experts

We established the baseline for our method based on classifications made by three cardiology experts. Each cardiologist independently listened to a representative subset of 12 PCG recordings, classifying them as either decompensated or recompensated. Importantly, no other clinical data on the CHF patients were available to the clinicians at that time. This differs from the typical clinical approach, where doctors have access to comprehensive patient information, including clinical data. The subset included three decompensated and three recompensated recordings from each of the two data acquisition setups. The results are given in Table 4.1.

Table 4.1: Results of classification of a representative subset of UMCL dataset by the medical experts. Classes 0 and 1 are recompensated and decompensated CHF phases, respectively. Ground truth labels were assigned by cardiologists at the time the recordings were captured.

PCG	Class	Expert 1	Expert 2	Expert 3	Accuracy [%]
1	1	0	1	0	33.3
2	0	1	1	1	0
3	1	1	0	0	33.3
4	1	1	1	1	100
5	0	0	0	0	100
6	0	0	0	1	66.6
7	0	1	1	1	0
8	1	1	0	0	33.3
9	1	0	1	0	33.3
10	0	0	0	1	66.6
11	0	0	0	0	1
12	1	0	1	0	33.3
Overall accuracy [%]		58	66.6	25	50

The experts' classification accuracies were 58%, 66.6%, and 25%, averaging at 50 ± 18%, which coincides with the dataset class distribution, meaning the cardiac auscultation alone contributes little to the experts' recognition of CHF decompensation episodes.

4.1.2 Evaluation of the Model's Performance

The models were evaluated in a subject-wise 10-fold CV. For removing outliers, a z-score threshold of $f_A = 2.2$ was employed, excluding segments with z-scores beyond this limit. Feature smoothing was performed using a window size of $w = 6$, which helped stabilize feature values. For the feature selection, the number of selected features was set to $k_b = 40$, ensuring that only the most impactful features were used in the analysis. Details about these parameters can be found in Sections 3.2.3 and 3.3.1.2. The results of the models' performance evaluation are shown in Table 4.2. They are given as mean ± standard deviation (SD), computed across the 10 folds.

All of the 10 implemented models outperform the baseline, while four of them outperform the baseline within the SD (not within the SD of the baseline, however). It is crucial

Table 4.2: Performance of the ML models on the UMCL dataset. Best values are in bold.

Classifier	Accuracy [%]	Precision [%]	Recall [%]	F1-score [%]	ROC AUC [%]
LR	**71.9 ± 14.7**	73.5 ± 16.5	**73.0 ± 22.2**	**71.4 ± 16.3**	**74.1 ± 18.0**
LGBM	69.7 ± 16.0	68.3 ± 28.8	63.0 ± 29.2	63.5 ± 26.6	70.6 ± 16.0
SVC	67.5 ± 15.7	**75.0 ± 22.7**	60.5 ± 31.2	61.8 ± 21.9	71.0 ± 15.5
RF	67.5 ± 17.6	63.0 ± 28.5	68.0 ± 34.6	62.7 ± 28.4	65.5 ± 22.9
DT	62.1 ± 14.8	60.0 ± 25.1	65.5 ± 29.0	60.1 ± 23.4	65.8 ± 16.3
GB	60.7 ± 16.4	60.7 ± 25.3	63.0 ± 27.0	59.4 ± 22.7	64.8 ± 16.8
XGB	60.7 ± 19.1	59.5 ± 26.8	60.5 ± 31.2	57.7 ± 25.9	67.0 ± 18.1
KN	58.5 ± 15.8	60.0 ± 19.3	57.7 ± 23.1	57.1 ± 18.3	61.0 ± 17.0
SGD	57.6 ± 6.9	45.0 ± 22.8	70.8 ± 40.7	54.0 ± 28.7	62.9 ± 13.0
GNB	54.2 ± 13.8	52.2 ± 20.5	55.2 ± 27.0	52.3 ± 21.2	56.6 ± 23.1

to note that the performance of the models is not directly comparable to that of the baseline, as the experts were assessed only on a small subset of the dataset. Additionally, while this subset is representative, its limited size presents challenges in drawing definitive conclusions about the true performance of the experts. The best performing model is LR, which achieved the highest accuracy, recall, F1-score, and ROC AUC. The confusion matrix of the LR is shown in Table 4.3. The ROC curves of the four most accurate models are shown in Figure 4.2.

Table 4.3: Confusion matrix of the LR model.

Actual label	Predicted label	
	Recompensated	**Decompensated**
Recompensated	26 (70.3%)	11 (29.7%)
Decompensated	10 (26.3%)	28 (73.7%)

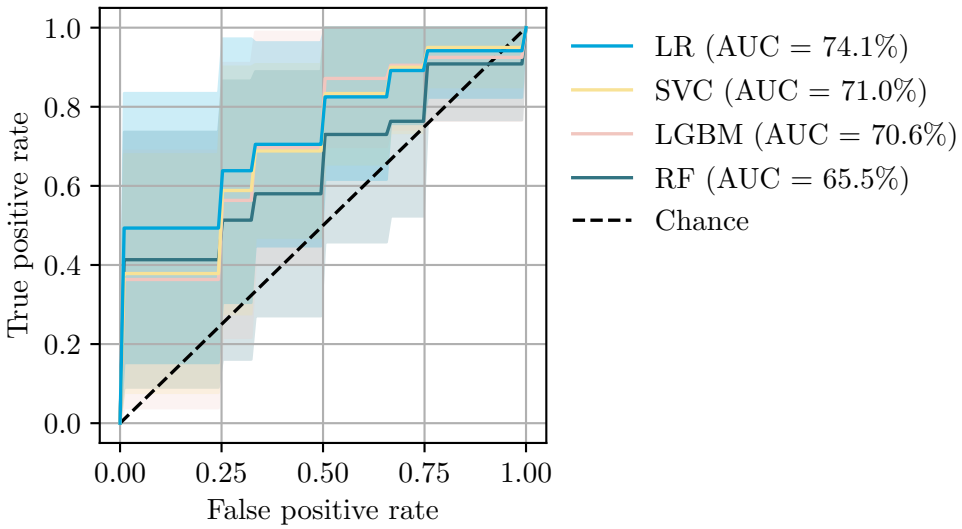


Figure 4.2: ROC curves of the most accurate four models. The coloured areas denote one SD.

To test whether the difference in the models' predictive accuracy was statistically significant, we performed a Wilcoxon signed-rank test. This non-parametric test is used to compare paired samples to assess if their population mean ranks differ, which is suitable in this context as it does not assume normality of the differences [276]. By computing the p-values for each pairwise comparison of models' accuracies across the 10 folds, we assessed the likelihood that any observed differences in performance occurred by chance. Thus, the p-values indicate whether the differences in models' accuracies are statistically significant. The results are given in Figure 4.3. The models are sorted based on their achieved accuracies. We see that models with comparable performance tended to yield similar predictions, as evidenced by the higher p-values along the diagonal of the table.

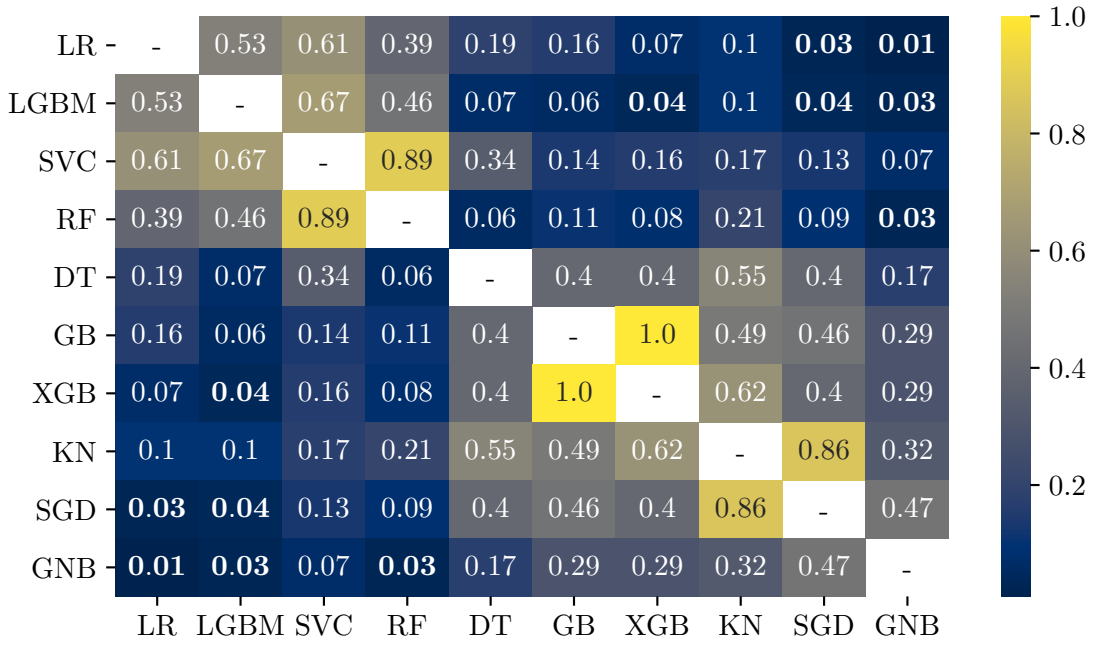


Figure 4.3: P-values from the Wilcoxon signed-rank test between the ML models.

The top 40 features selected from the entire dataset have already been shown in Table 3.5 (Section 3.3.1.2). The most important features are the time-domain features (10/40), the power spectral density features for different frequency bands (17/40), and the MFCCs (9/40). The most important heart sound seems to be diastole, as 21/40 of the most important features were extracted from diastole. From the medical perspective, this is reasonable, since the sounds produced in the diastole originate from the heart chambers refilling by blood. The heart of a CHF patient is more rigid than a healthy heart and will thus vibrate differently. Another observation is that several of the important features are related to heart rate. Again, this is relevant from the medical point of view, as patients in the decompensated phase have a faster pulse than those that are not decompensated. From an XAI perspective, identifying these key features provides insight into how the model interprets the heart sounds and supports its predictions, improving our understanding of the underlying medical phenomena and the model's decision-making process.

4.1.3 Model Fine-tuning

To enhance the overall performance of the models, we conducted grid-search fine-tuning as per Algorithm 3.1. The values of the parameters used in grid-search are given in Ap-

pendix A.2. The bar plots of the fine-tuned accuracies, along with the original results are shown in Figure 4.4. Although the performance of the models generally improved, the top performance remained practically unchanged.

Note that only the model parameters were fine-tuned using grid search;

Note that the fine-tuning using grid search was only applied for the model parameters and not also for the three model-independent parameters in our pipeline. Those parameters - discussed and analysed later in Section 4.1.5 - include the data pre-processing parameters f_A and w, and the number of selected features k. Their values were determined empirically and fixed in advance for all experiments.

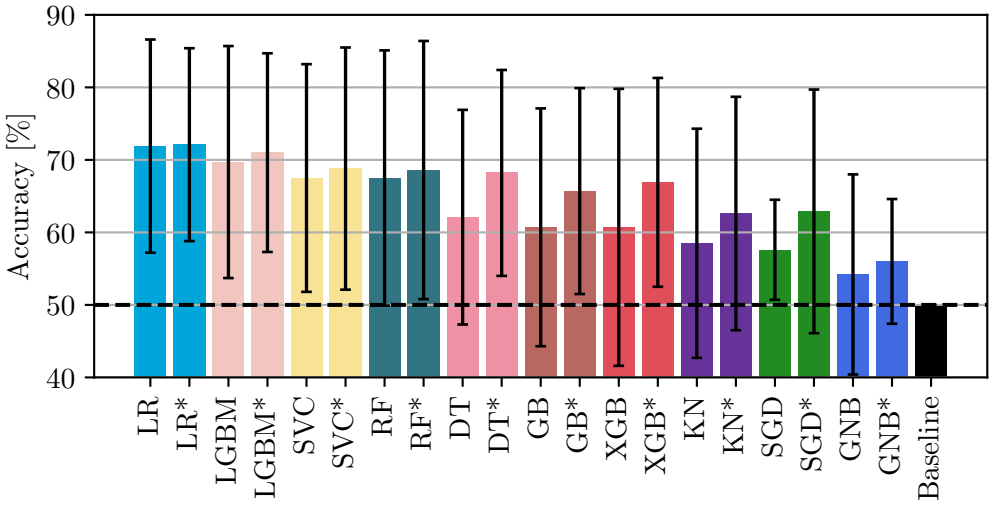


Figure 4.4: Accuracy of the models before and after (*) fine-tuning. Bars of the fine-tuned models are hatched.

4.1.4 PhysioNet Dataset Experiments

To test the robustness of our pipeline, we applied it to the PhysioNet dataset for classification between normal and abnormal PCGs. A robust pipeline can consistently and accurately classify across different datasets or conditions. Testing with a different dataset assesses whether the pipeline maintains its performance and reliability beyond the initial dataset, demonstrating its generalizability and stability. The PhysioNet dataset is described in Section 3.1.2. It consists of 554 train and 297 test recordings. The models were evaluated in a train/test split experiment using the same data processing and feature-engineering parameters ($f_A = 2.2$, $w = 6$, and $k_b = 40$) as in the UMCL experiments. To ensure reliable results, we ran five experiments, each time with a different RNG (random) seed that affected feature selection and the models' learning processes.

The model with the best accuracy was XGB, which achieved a score of 77.2 ± 1.1 %, significantly better than that of the majority, which was 52.2%. The following best performing models were LGBM, SVC, KN, GB, and RF with the accuracy scores of 76.7 ± 1.0 %, 76.4 ± 1.1 %, 75.8 ± 0.3 %, 75.7 ± 0.1 %, and 74.1 ± 1.0 %, respectively.

Interestingly, while LR achieved the highest performance on the UMCL dataset, it did not rank among the top models on the PhysioNet dataset. Similarly, models such as kNN, GB, and XGB that performed well on the PhysioNet dataset exhibited relatively poor results on the UMCL dataset. In contrast, LGBM and SVC emerged as the only consistently performing models, although this consistency may be attributed to random

chance. In the PhysioNet experiments, more complex models tended to demonstrate superior performance, which could be attributed to the complexity of the data (variations in recording equipment, patient demographics etc.).

Notably, the PhysioNet dataset comprises both normal and CVD subjects, while the UMCL dataset includes only CVD subjects, specifically two phases of CHF, both classified as "abnormal" recordings. Due to these fundamental differences, we opted not to test the UMCL-trained models on the PhysioNet dataset for cross-database generalization, although such an analysis is generally considered both interesting and informative.

4.1.5 Parameter Sensitivity Analysis

In this section, we delve into the sensitivity analysis of the three model-independent parameters in our pipeline:

- Threshold for removing bad segments: the f_A parameter determines the z-score for the interval of the envelope area of the segments to be retained. Segments outside this interval are removed because they are either segmented incorrectly, too noisy, or missing one of the two main heart sounds. For our final results, this was set $f_A = 2.2$ (see Section 3.2.3);
- Window size for smoothing the features: in our finals results, this was set to $w = 6$ (see Section 3.3.1.2);
- Number of selected features: in our final results, this was set to $k_b = 40$ (see Section 3.3.1.2).

We analysed the sensitivity of each parameter independently using the UMCL dataset by evaluating model performance across a range of values for that parameter, while keeping the other two parameters fixed at their values from the final results. This approach assumed that the parameters are independent, whereas for a more robust understanding of their combined effects, the parameters' mutual interactions should be explored. For each parameter, we created two graphs. In the first, the models' accuracies are plotted as functions of the parameter values. In the second graph, the mean accuracies of the best performing models (up to five) at each parameter value are plotted. The accuracies were averaged across the 10 folds of the CV.

4.1.5.1 Threshold for Removing Bad Segments

After the segmentation of PCGs, segments that were incorrectly segmented, excessively noisy, or missing fundamental heart sounds were removed. The fraction of removed segments was determined by parameter f_A as per Eq. (3.4). To analyse how sensitive the models are to this parameter, we checked their performance across different f_A values. The results are plotted in Figure 4.5.

As shown in Figure 4.5a, the overall pattern is unclear due to varying responses from the models. However, when focusing on the top X models for each parameter value and averaging their performance, the pattern becomes clearer, as illustrated in Figure 4.5b. At $f_A = 1.0$, around 27.3% of segments per PCG are removed, resulting in the lowest accuracy. As the z-score increases, accuracy improves, reaching its peak at $f_A = 2.2$, where only about 3.3% of segments are removed. At $f_A = 3.0$, almost no segments are removed, and while accuracy is higher than at $f_A = 1.0$, it does not surpass the peak at $f_A = 2.2$. Thus, $f_A = 2.2$ achieves the best balance, maximizing accuracy by effectively filtering out low-quality segments without discarding too many valid ones. It is important to note that the segments are removed on a per-PCG basis, which means the actual percentages of

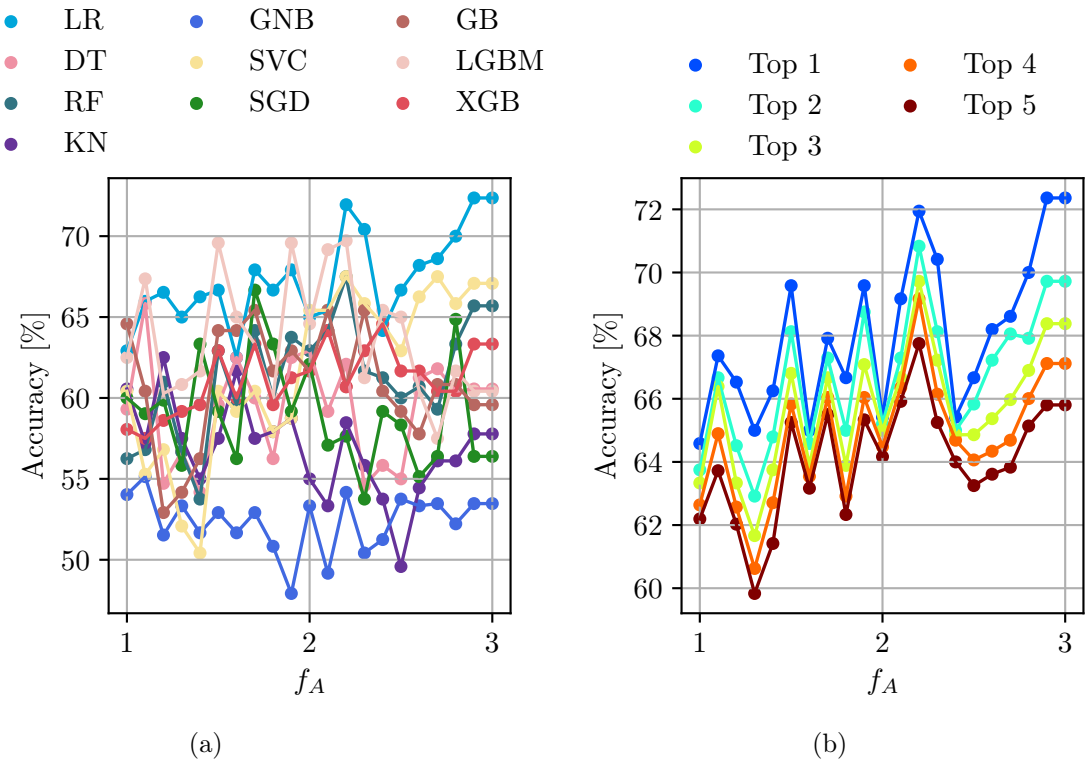


Figure 4.5: Accuracy vs. the fraction of bad segments to be removed f_A: (a) all models and (b) mean of the top models at each parameter value. The results are averaged across 10 CV folds.

removed segments differ slightly from the theoretical expectations based on the z-scores of a standard normal distribution. The expected removal rates based on these z-scores would be around 31.8% for a z-score of 1.0, around 4.6% for a z-score of 2.0, and around 0.4% for a z-score of 3.0. While removing poor-quality segments improves model performance, excessive removal can lead to insufficient data and potential overfitting, which may negatively impact model performance.

4.1.5.2 Features Smoothing Window Size

To smooth out the outliers, our final features were the mean and the SD of the features, calculated using a sliding window with a window size of $w = 6$ across the segments of each PCG. The sensitivity analysis of this parameter is given in Figure 4.6.

As the smoothing window size increases, the feature vectors of the segments become more and more similar. If the window size matches (or is larger than) the number of segments for a PCG, all of the segment features become the same as they are an average of the same set of segments. Increasing the window size thus reduces variability in the data. After some point of increasing the window size, the negative effect of reduced variability outweighs the positive effect of smoothing out the outliers.

4.1.5.3 Number of Selected Features

During training, we performed feature selection by computing the mutual information between the features and the target variable [185]. We experimentally determined that

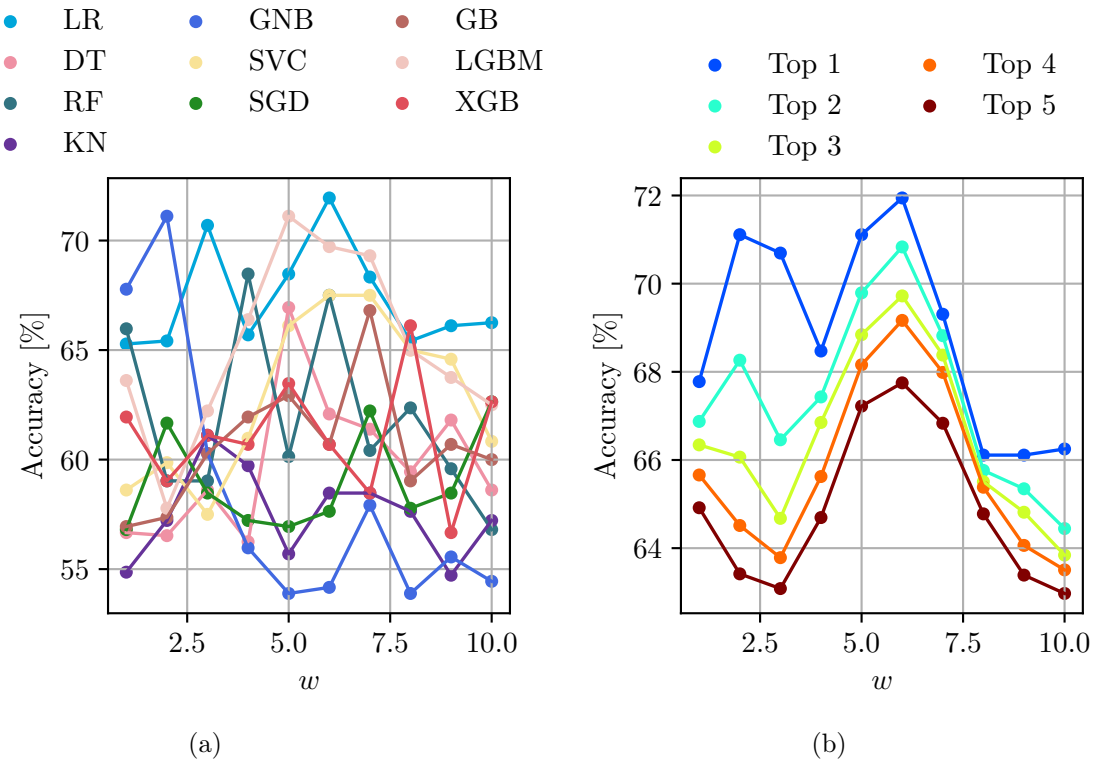


Figure 4.6: Accuracy vs. the size of the features smoothing window w: (a) all models and (b) mean of the top models at each parameter value. The results are averaged across 10 CV folds.

selecting the top $k_b = 40$ features according to this metric provided the best performance. This determination was made by observing the point at which increasing the number of features no longer resulted in performance improvements. To analyse the sensitivity of this parameter, we examined the performance of the models with different numbers of selected features. For this analysis, we iteratively added features, starting with the most important according to mutual information, and estimated performance using 10-fold CV. The results are plotted in Figure 4.7.

Initially, as the number of features increases, the models' accuracies improve, reaching an optimal performance around 40 features. Beyond this point, the accuracies begin to decline. This trend indicates that including up to $k_b = 40$ of the most informative features, as determined by mutual information, enhances the models' predictive power. However, adding more features beyond this threshold introduces noise and redundancy, leading to overfitting and increased variance, which in turn deteriorates the models' performances.

This section demonstrated the potential of detecting CHF decompensation episodes from heart sounds alone, with the best models performing similar to expert cardiologists. However, in practice, clinicians use additional data (e.g., ECG, PPG, clinical data). Our models were trained on extreme CHF cases, likely reflecting maximum achievable accuracy on these data. To build an effective detection system, additional PCG and multimodal data are needed. Since the UMCL dataset lacks this, methods like data augmentation are essential to enhance model performance.

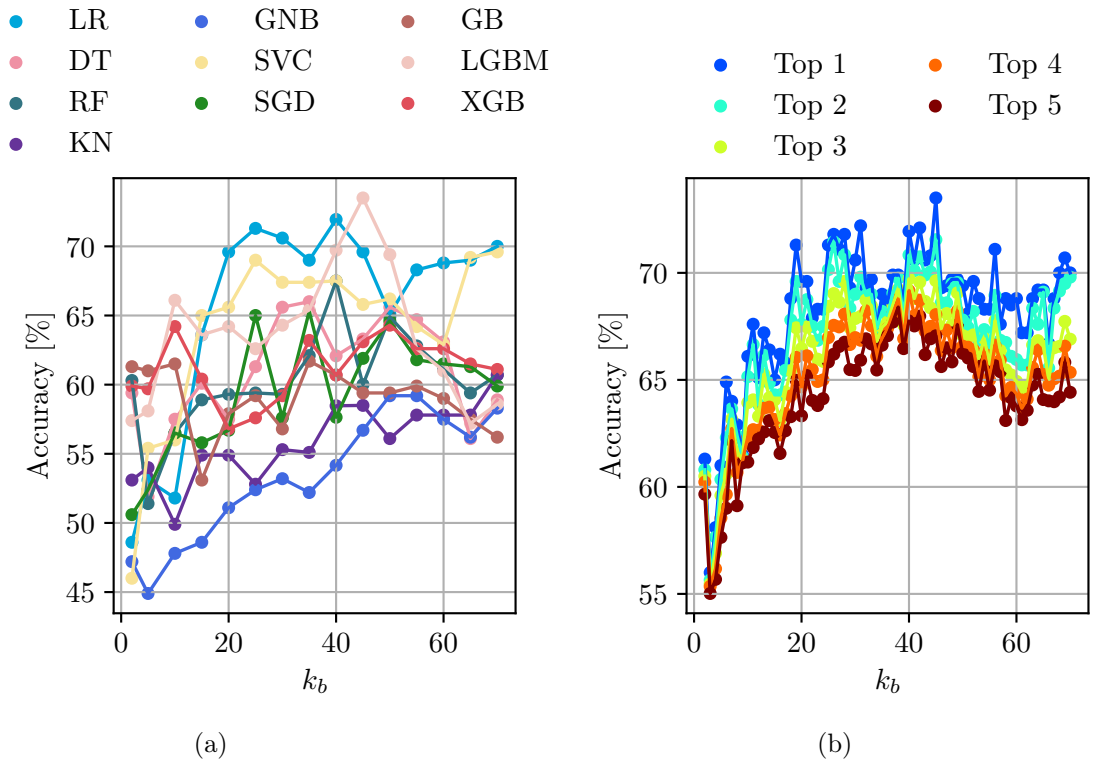


Figure 4.7: Accuracy vs. the number of predictor features k_b: (a) all models and (b) mean of the top models at each parameter value. The results are averaged across 10 CV folds.

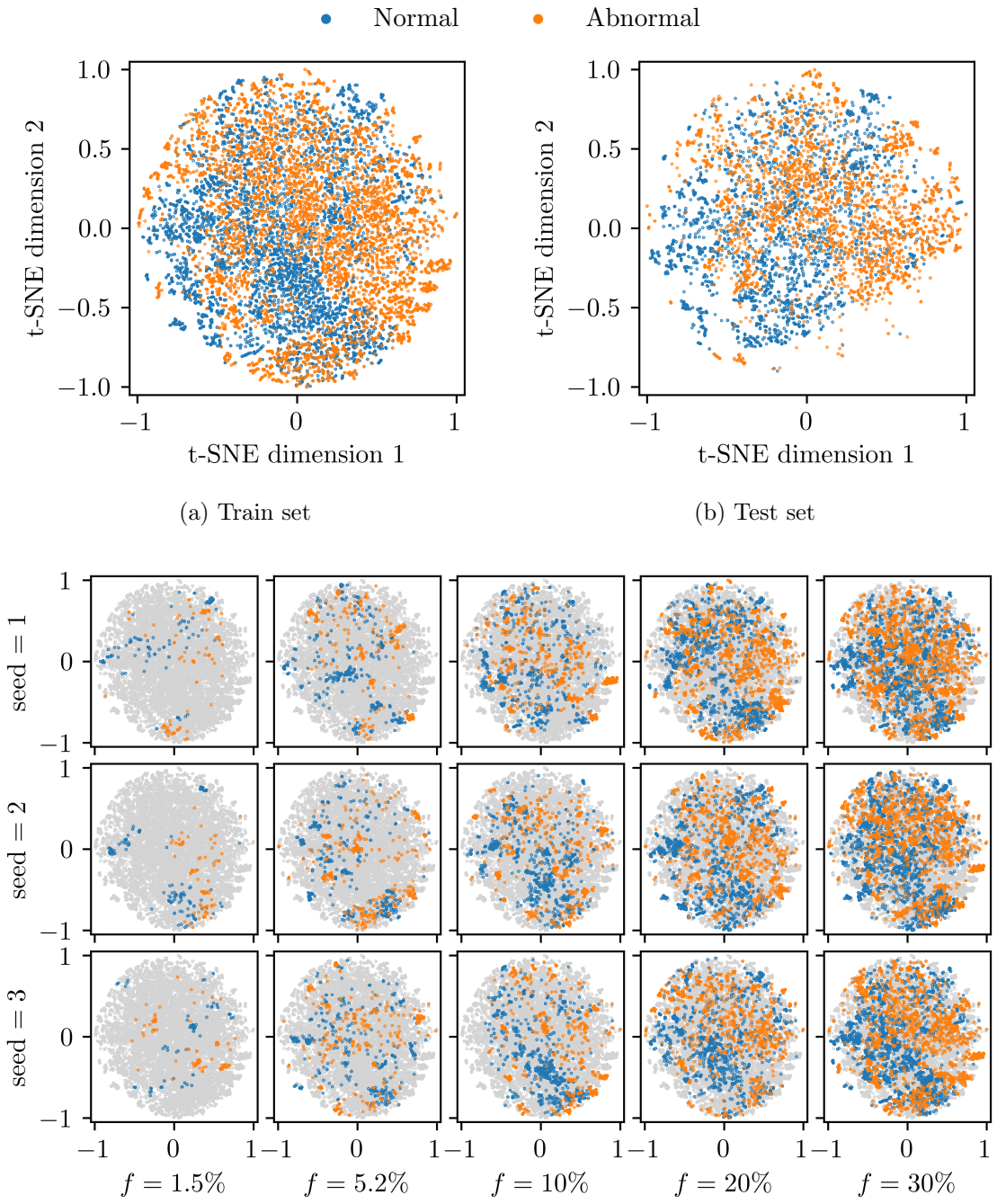
4.2 Heart-Sound Data Augmentation

This section provides a comprehensive analysis of the proposed heart-sound data-augmentation technique (presented in Section 3.4.2). It begins with results for both time-series and spectrogram domains, comparing our method with other augmentation strategies using subsets of the PhysioNet dataset of various sizes. We have already published our time-series results in [277]. After that, a parameter sensitivity analysis is conducted to evaluate how different settings for model training and the augmentation method impact performance. Next, the effect of constrained mapping—used for pairing data to generate new instances—on the reliability of generated instances is explored. The analysis continues with visualizations of feature exploration from augmentation methods and an investigation into out-of-manifold intrusion, where instances are generated in regions of the feature space where they should not be. Saliency maps are then visualized to gain insights into the explainability and interpretability of the NNs, followed by an examination of how leveraging saliency information can enhance the augmentation strategy. The section concludes with an assessment of computational complexity and an evaluation of the proposed augmentation method on the UMCL dataset.

The PhysioNet dataset used in our experiments included 7761 normal and 6994 abnormal PCG segments for training and 3664 normal and 3527 abnormal PCG segments for testing (see Section 3.1.2 for details). We used accuracy and ADSI as the main metrics of performance evaluation.

For the purpose of visualizing our train and test data, we extracted a total of 177 time, statistical, frequency, and wavelet features from each individual, normalized, 25 – 400 Hz

filtered heartbeat segment in the same way as described in Section 3.3.1. Subsequently, we employed the t-SNE dimensionality reduction technique [190] to condense the feature space into just two components. The resulting representations of the complete train and test sets, as well as 15 unique training data-sampling partitions ($\mathcal{N}_f^{seed}$), where $seed \in 1, 2, 3$ and training-data fraction $f \in 0.015, 0.052, 0.1, 0.2, 0.3$, are depicted in Figure 4.8.



(a) Train set

(b) Test set

(c) Examples of training data partitions

Figure 4.8: Visualization of the full (a) train and (b) test datasets and (c) 15 unique training data-sampling partitions of the PhysioNet dataset used in our experiments. Grey areas in (c) represent the complete train set.

As a side note, the smallest fractions, 0.015 and 0.052, correspond to 10 and 30 PCGs, respectively. These values were chosen so precisely because we can only use an integer number of recordings. The visualizations in Figure 4.8 demonstrate the inherent challenge of the classification task, particularly when working with smaller fractions of the training data.

The exact number of experiments we ran for each training-data fraction f is provided in Table 4.4 (for explanation of how these numbers were deterimined, see Section 3.4.5). Each experiment corresponds to a different random sampling of the training data. The table also lists the number of PCGs and the average number segments for each fraction. The detailed settings required to reproduce the experiments, including the RNG seeds used for data sampling, are given in the Appendix A.1 and A.3.

Table 4.4: Number of experiments, PCG recordings, and segments for each training-data fraction f.

Training-data fraction f	# Experiments (time series)	# Experiments (spectrogram)	# Recordings	# Heartbeats
1.5%	333	200	10	268 ± 60
5.2%	100	60	30	801 ± 96
10%	50	30	56	15119 ± 133
20%	25	15	112	2997 ± 175
30%	16	10	168	4512 ± 222
40%	12	8	222	5806 ± 203
60%	8	6	334	8925 ± 209
80%	6	5	444	11788 ± 143
100%	5	3	554	14755

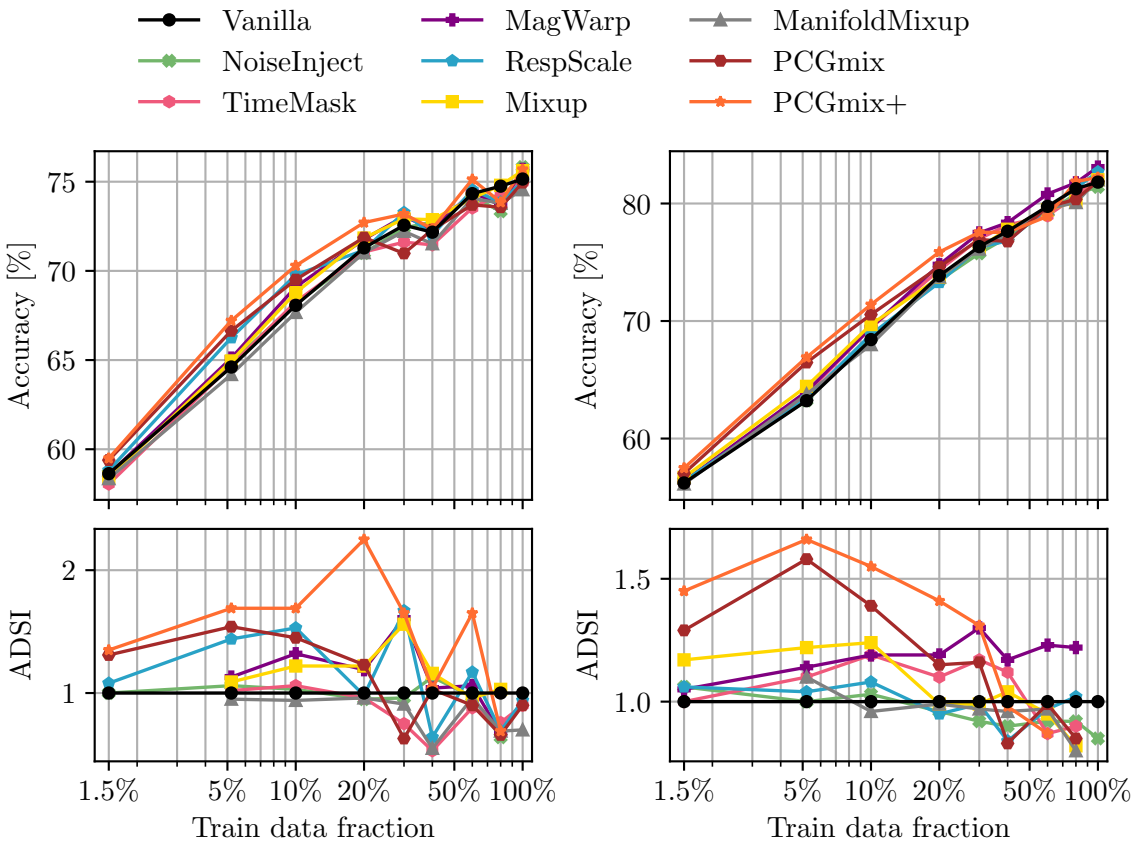
4.2.1 Time-Series Domain

In this series of experiments, we used data in the time-series domain (see Section 3.2.2) and employed 1-dimensional CNNs, the 1D-CNN and 1D-ResNet, with architectures detailed in Sections 3.4.4.1 and 3.4.4.2. We compared our data-augmentation method with the standard augmentation methods described in Section 3.4.3. Augmentation was performed during training by swapping the original batch with the augmented one as per Algorithm 3.3. The classification results for a selection of training-data fractions are shown in Table 4.5. In Figure 4.9, the accuracy and the ADSI for both the 1D-CNN and the 1D-ResNet are plotted as functions of the training-data fraction. The ADSI metric, introduced in Section 3.5, evaluates the performance of data-augmentation techniques by measuring how much the dataset's size appears to increase due to augmentation compared to no augmentation (vanilla), considering results across all data subset sizes.

Table 4.5 and Figure 4.9 indicate that PCGmix(+) excel when the amount of training data is the smallest (<30%). Demonstrating notable improvements over PCGmix, PCGmix+ consistently achieves higher accuracy. Moreover, it outperforms all other state-of-the-art approaches in this range of the amount of training data. The 1D-ResNet achieved higher overall accuracy than the 1D-CNN, particularly at larger training-data fractions, likely due to its residual connections that facilitate learning complex patterns in the data. While both PCGmix and PCGmix+ performed similarly in the 1D-ResNet, PCGmix+ slightly outperformed PCGmix in the 1D-CNN, with the effectiveness of PCGmix diminishing more rapidly as the training-data fraction increased.

Table 4.5: Results of the time-series-domain experiments. Best values are in bold.

		1D-CNN		1D-ResNet	
f	Method	Accuracy [%]	ADSI	Accuracy [%]	ADSI
1.5%	Vanilla (no aug.)	58.63 ± 5.80	-	56.22 ± 5.86	-
	NoiseInject	58.64 ± 5.66	1.00	56.40 ± 5.92	1.06
	TimeMask	58.05 ± 5.56	-	56.22 ± 5.69	1.00
	MagWarp	58.59 ± 5.72	-	56.35 ± 5.91	1.05
	RespScale	58.82 ± 5.99	1.08	56.38 ± 6.68	1.06
	Mixup	58.42 ± 5.86	-	56.69 ± 6.20	1.17
	ManifoldMixup	58.35 ± 5.75	-	56.13 ± 6.10	-
	PCGmix (ours)	59.39 ± 5.92	1.31	57.04 ± 6.21	1.29
	PCGmix+ (ours)	**59.49 ± 5.95**	**1.35**	**57.49 ± 6.07**	**1.45**
5.2%	Vanilla (no aug.)	64.61 ± 4.25	-	63.22 ± 4.94	-
	NoiseInject	64.84 ± 4.41	1.06	63.24 ± 5.01	1.00
	TimeMask	64.70 ± 4.17	1.02	63.81 ± 5.21	1.10
	MagWarp	65.11 ± 4.60	1.13	64.01 ± 5.42	1.14
	RespScale	66.25 ± 4.74	1.44	63.43 ± 6.42	1.04
	Mixup	64.95 ± 4.33	1.09	64.44 ± 5.82	1.22
	ManifoldMixup	64.19 ± 4.53	0.95	63.77 ± 4.95	1.10
	PCGmix (ours)	66.64 ± 4.04	1.54	66.47 ± 5.03	1.58
	PCGmix+ (ours)	**67.21 ± 4.46**	**1.69**	**66.91 ± 5.11**	**1.66**
10%	Vanilla (no aug.)	68.07 ± 2.78	-	68.42 ± 3.42	-
	NoiseInject	68.17 ± 2.69	1.03	68.58, ± 3.07	1.03
	TimeMask	68.26 ± 2.40	1.06	69.44 ± 3.63	1.19
	MagWarp	69.10 ± 2.65	1.32	69.43 ± 3.27	1.19
	RespScale	69.78 ± 2.79	1.53	68.86 ± 3.26	1.08
	Mixup	68.78 ± 2.56	1.22	69.70 ± 3.94	1.24
	ManifoldMixup	67.67 ± 2.79	0.94	67.99 ± 2.79	0.96
	PCGmix (ours)	69.51 ± 2.49	1.45	70.55 ± 2.88	1.39
	PCGmix+ (ours)	**70.29 ± 2.52**	**1.69**	**71.40 ± 2.59**	**1.55**
20%	Vanilla (no aug.)	71.29 ± 3.03	-	73.86 ± 3.10	-
	NoiseInject	71.00 ± 2.63	0.95	73.45 ± 3.32	0.96
	TimeMask	71.06 ± 2.64	0.96	74.33 ± 2.99	1.10
	MagWarp	71.77 ± 1.98	1.19	74.79 ± 3.26	1.19
	RespScale	71.16 ± 2.09	0.98	73.28 ± 2.67	0.95
	Mixup	71.84 ± 2.11	1.22	73.72 ± 3.00	0.99
	ManifoldMixup	71.02 ± 1.97	0.96	73.75 ± 3.11	0.99
	PCGmix (ours)	71.87 ± 2.25	1.23	74.61 ± 2.59	1.15
	PCGmix+ (ours)	**72.71 ± 2.07**	**2.25**	**75.87 ± 3.09**	**1.41**
30%	Vanilla (no aug.)	72.56 ± 1.52	-	76.33 ± 1.30	-
	NoiseInject	72.41 ± 1.54	0.96	75.76 ± 2.18	0.92
	TimeMask	71.61 ± 1.25	0.75	77.00 ± 2.09	1.17
	MagWarp	73.06 ± 1.31	1.61	77.50 ± 1.96	1.30
	RespScale	**73.27 ± 1.96**	**1.67**	76.28 ± 2.01	0.99
	Mixup	72.92 ± 1.89	1.56	76.24 ± 1.94	0.99
	ManifoldMixup	72.22 ± 1.73	0.91	76.12 ± 1.73	0.97
	PCGmix (ours)	70.98 ± 5.80	0.63	76.94 ± 2.38	1.16
	PCGmix+ (ours)	73.19 ± 1.39	1.65	**77.55 ± 2.71**	**1.31**



(a) 1D-CNN (b) 1D-ResNet

Figure 4.9: (top) Accuracy and (bottom) ADSI vs. training-data fraction in time-series-domain experiments: (a) 1D-CNN and (b) 1D-ResNet. In some cases, the accuracies achieved by the augmentation method are outside the accuracy range achieved by vanilla, leaving the ADSI undefined. The x-axis scale is logarithmic.

To check the statistical significance of the results, we performed a paired t-test to compare PCGmix+ with other methods. This test is suitable because the differences between the methods' performances were normally distributed. In Section 4.1.2, where this assumption could not be made, we used the Wilcoxon signed-rank test instead. The p-values of the tests are presented in Table 4.6. PCGmix+ has a clear advantage in scenarios with limited amounts of training data, where it provides both better performance and statistical robustness, as indicated by the small p-values. As the amount of training data increases, the performance differences between the methods become relatively smaller and statistically insignificant. The ADSI factor of the 1D-ResNet results shows that at 5.2%, 10%, and 20% of the training data, PCGmix+ achieves the same accuracy as the vanilla when trained on an amount of data that is 1.66, 1.55, and 1.41 times bigger, respectively. High standard deviations in the results indicate considerable diversity in the randomly selected subsets, which makes sense. A subset may consist predominantly of uninformative PCGs that cover only a small, unimportant or unrepresentative area of the feature space of the training data, resulting in poor performance. On the other hand, a subset may contain predominantly highly informative PCGs that cover a larger and more significant area of the feature space, resulting in high accuracy. Higher standard deviations are typically observed at lower fs, where the smaller subsets increase the probability of randomly selecting subsets

that are either completely informative or completely uninformative.

Table 4.6: P-values from the paired t-test between PCGmix+ and the other methods in 1D-ResNet experiments. Results with a p-value < 0.05 are in bold.

Other method	f				
	1.5%	5.2%	10%	20%	30%
Vanilla	**0.00**	**0.00**	**0.00**	**0.00**	**0.04**
NoiseInject	**0.00**	**0.00**	**0.00**	**0.00**	**0.01**
TimeMask	**0.00**	**0.00**	**0.00**	**0.00**	0.36
MagWarp	**0.00**	**0.00**	**0.00**	**0.02**	0.94
RespScale	**0.00**	**0.00**	**0.00**	**0.00**	0.05
Mixup	**0.00**	**0.00**	**0.00**	**0.00**	0.05
ManifoldMixup	**0.00**	**0.00**	**0.00**	**0.00**	**0.03**

Our empirical analysis shows that a trend in accuracy improvements appears to approach a plateau with the utilization of approximately 168 patient recordings, which equates to roughly 4500 heartbeats. After this threshold, any additional expansion in dataset size does not yield substantial improvements in the accuracy metrics. Consequently, the efficiency of the data-augmentation methods, including PCGmix(+), appears to diminish beyond this amount of training data.

In the following, we present additional analysis of the PCGmix+ method combined with the 1D-ResNet model, which was found to be the most effective pairing. We focus on training-data fractions of up to 30% as augmentation did not yield significant improvements beyond this point. Table 4.7 shows the confusion matrices of the PCGmix+, whereas Table 4.8 shows the precision, recall, F1-score, and ROC AUC metrics for all methods. Our analysis revealed that the differences in these metrics are statistically significant (for training-data fractions up to up to 30%), similar to accuracy. Notably, the ROC AUC metric exhibited the lowest p-values across all comparisons, indicating a strong level of significance. The confusion matrices in Table 4.7 indicate that abnormal recordings are correctly identified slightly more often than normal ones, with no class being over-prioritized, likely due to the training set being balanced. Our approach provides a superior ROC AUC compared to others, which is crucial in medical diagnostic systems for effective decision-making when trade-offs between sensitivity and specificity are critical.

Table 4.7: Aggregated and normalized confusion matrices of the PCGmix+ in 1D-ResNet experiments. The values are rounded to closest integer.

f	Actual label	Predicted label	
		Normal	**Abnormal**
5.2%	**Normal**	94 (64%)	52 (36%)
	Abnormal	46 (30%)	105 (70%)
10%	**Normal**	101 (69%)	45 (31%)
	Abnormal	40 (26%)	111 (74%)
20%	**Normal**	108 (74%)	38 (26%)
	Abnormal	33 (22%)	118 (78%)
30%	**Normal**	113 (77%)	33 (22%)
	Abnormal	34(23%)	118 (77%)

Table 4.8: F1-score, precision, recall, and ROC AUC in 1D-ResNet experiments. The mean and standard deviation of the results are reported. The best results are in bold.

f	Method	F1-score [%]	Precision [%]	Recall [%]	ROC AUC [%]
1.5%	Vanilla (no aug.)	55.96 ± 11.18	56.84 ± 6.24	58.12 ± 18.5	58.62 ± 7.74
	NoiseInject	56.46 ± 10.7	56.99 ± 6.2	58.66 ± 17.7	58.73 ± 7.99
	TimeMask	55.59 ± 9.98	57.5 ± 6.43	56.64 ± 16.96	58.69 ± 7.75
	MagWarp	56.11 ± 10.72	57.0 ± 6.35	57.82 ± 17.24	58.85 ± 7.72
	RespScale	55.36 ± 11.66	57.84 ± 7.69	56.76 ± 19.24	59.01 ± 8.92
	Mixup	56.05 ± 9.96	57.95 ± 6.98	56.76 ± 16.18	59.41 ± 8.29
	ManifoldMixup	56.32 ± 10.51	56.75 ± 6.63	58.56 ± 17.61	58.25 ± 8.0
	PCGmix (ours)	57.08 ± 10.27	58.06 ± 7.28	**58.85 ± 16.76**	59.88 ± 8.22
	PCGmix+ (ours)	**57.29 ± 9.98**	**58.44 ± 6.65**	58.61 ± 16.32	**60.44 ± 8.2**
5.2%	Vanilla (no aug.)	64.85 ± 6.08	63.12 ± 5.05	67.73 ± 10.84	68.23 ± 6.29
	NoiseInject	65.17 ± 5.86	62.92 ± 4.98	68.42 ± 9.94	67.79 ± 6.43
	TimeMask	64.65 ± 6.61	64.16 ± 5.29	66.08 ± 10.64	68.85 ± 6.62
	MagWarp	65.46 ± 6.62	63.86 ± 5.51	68.07 ± 10.91	68.83 ± 6.61
	RespScale	63.64 ± 8.17	64.16 ± 6.63	64.30 ± 12.64	68.23 ± 7.30
	Mixup	65.29 ± 6.93	64.75 ± 5.82	66.73 ± 10.96	69.48 ± 7.02
	ManifoldMixup	65.28 ± 6.29	63.66 ± 5.08	68.05 ± 11.05	68.39 ± 6.11
	PCGmix (ours)	67.48 ± 6.21	66.51 ± 5.22	**69.38 ± 10.48**	71.79 ± 5.96
	PCGmix+ (ours)	**67.75 ± 6.11**	**67.22 ± 5.60**	69.24 ± 10.28	**72.43 ± 6.10**
10%	Vanilla (no aug.)	69.95 ± 3.42	68.02 ± 4.20	72.58 ± 6.76	74.67 ± 3.62
	NoiseInject	70.01 ± 3.64	68.12 ± 3.72	72.58 ± 7.30	74.70 ± 3.03
	TimeMask	69.95 ± 4.03	70.17 ± 4.70	70.40 ± 7.45	75.87 ± 3.92
	MagWarp	70.60 ± 3.39	69.33 ± 4.36	72.48 ± 6.67	75.99 ± 3.89
	RespScale	69.73 ± 3.75	69.19 ± 4.22	71.05 ± 8.01	75.43 ± 3.62
	Mixup	70.77 ± 3.76	69.72 ± 4.83	72.30 ± 6.07	75.57 ± 4.06
	ManifoldMixup	69.64 ± 3.01	67.53 ± 4.05	72.56 ± 7.02	74.27 ± 3.52
	PCGmix (ours)	71.67 ± 3.15	70.42 ± 4.07	73.66 ± 7.12	77.54 ± 2.87
	PCGmix+ (ours)	**72.28 ± 2.81**	**71.55 ± 4.19**	**73.67 ± 6.58**	**78.53 ± 3.02**
20%	Vanilla (no aug.)	75.35 ± 2.89	72.49 ± 3.36	78.65 ± 4.51	80.64 ± 3.12
	NoiseInject	75.13 ± 2.93	71.94 ± 3.85	**78.83 ± 4.30**	80.20 ± 2.95
	TimeMask	75.09 ± 3.03	74.21 ± 3.55	76.24 ± 4.96	81.71 ± 2.72
	MagWarp	75.98 ± 3.05	73.81 ± 3.66	78.46 ± 4.39	82.12 ± 3.08
	RespScale	74.65 ± 2.58	72.15 ± 2.83	77.46 ± 3.87	81.12 ± 2.90
	Mixup	74.99 ± 3.25	72.67 ± 3.19	77.75 ± 5.81	80.92 ± 2.42
	ManifoldMixup	75.32 ± 2.92	72.30 ± 3.58	**78.83 ± 4.65**	80.61 ± 2.82
	PCGmix (ours)	75.64 ± 2.63	73.95 ± 3.15	77.70 ± 5.00	82.45 ± 2.52
	PCGmix+ (ours)	**76.65 ± 2.99**	**75.60 ± 3.80**	78.04 ± 5.03	**83.69 ± 2.52**
30%	Vanilla (no aug.)	77.79 ± 1.26	74.39 ± 1.57	**81.58 ± 2.4**	84.41 ± 1.39
	NoiseInject	77.12 ± 2.16	74.15 ± 2.21	80.42 ± 3.38	83.92 ± 1.59
	TimeMask	77.61 ± 2.18	76.82 ± 2.23	78.52 ± 3.51	84.69 ± 1.39
	MagWarp	**78.57 ± 2.13**	76.13 ± 1.59	81.25 ± 3.57	85.32 ± 1.45
	RespScale	77.43 ± 1.94	75.09 ± 2.71	80.09 ± 3.6	84.68 ± 2.06
	Mixup	77.22 ± 2.0	75.36 ± 2.3	79.3 ± 3.6	84.26 ± 1.52
	ManifoldMixup	77.44 ± 1.77	74.49 ± 1.79	80.71 ± 2.88	84.37 ± 1.5
	PCGmix (ours)	77.34 ± 2.54	77.25 ± 2.49	77.52 ± 3.84	85.31 ± 1.77
	PCGmix+ (ours)	77.86 ± 2.85	**78.06 ± 2.95**	77.77 ± 3.97	**86.14 ± 2.18**

4.2.2 Spectrogram Domain

In this series of experiments, we used data in the spectrogram domain (see Section 3.2.2) and employed 2-dimensional CNN, the 2D-ResNet, with architecture detailed in Section 3.4.4.3. The classification results for a selection of training-data fractions are shown in Table 4.9. In Figure 4.10, the accuracy and the ADSI for the 2D-ResNet are plotted as functions of the training-data fraction.

Table 4.9: Results of the spectrogram domain experiments.

		2D-CNN	
f	Method	Accuracy [%]	ADSI
1.5%	Vanilla (no aug.)	56.87 ± 5.64	-
	FreqMask	56.67 ± 5.49	-
	TimeMask	56.79 ± 5.54	-
	Cutout	56.76 ± 5.64	-
	Mixup	57.67 ± 5.71	1.22
	ManifoldMixup	56.69 ± 5.53	-
	PCGmix (ours)	**58.55 ± 6.48**	**1.46**
5.2%	Vanilla (no aug.)	65.79 ± 4.55	-
	FreqMask	65.5 ± 5.30	0.98
	TimeMask	66.72 ± 4.97	1.14
	Cutout	66.21 ± 4.90	1.06
	Mixup	67.78 ± 4.53	1.29
	ManifoldMixup	66.23 ± 4.85	1.06
	PCGmix (ours)	**68.16 ± 4.03**	**1.34**
10%	Vanilla (no aug.)	72.14 ± 2.72	-
	FreqMask	71.83 ± 2.54	0.98
	TimeMask	72.26 ± 2.78	1.03
	Cutout	72.59 ± 2.91	1.11
	Mixup	73.22 ± 2.69	1.27
	ManifoldMixup	72.74 ± 2.74	1.15
	PCGmix (ours)	**74.03 ± 2.12**	**1.46**
20%	Vanilla (no aug.)	76.21 ± 3.36	-
	FreqMask	76.68 ± 3.06	1.06
	TimeMask	76.79 ± 3.61	1.08
	Cutout	76.34 ± 3.58	1.02
	Mixup	**76.88 ± 2.82**	**1.09**
	ManifoldMixup	76.32 ± 3.08	1.01
	PCGmix (ours)	76.79 ± 3.24	1.08
30%	Vanilla (no aug.)	79.97 ± 2.77	-
	FreqMask	79.8 ± 1.69	0.98
	TimeMask	79.87 ± 1.69	0.99
	Cutout	79.36 ± 1.84	0.95
	Mixup	**80.77 ± 1.41**	**1.28**
	ManifoldMixup	79.12 ± 2.01	0.92
	PCGmix (ours)	80.0 ± 2.06	1.01

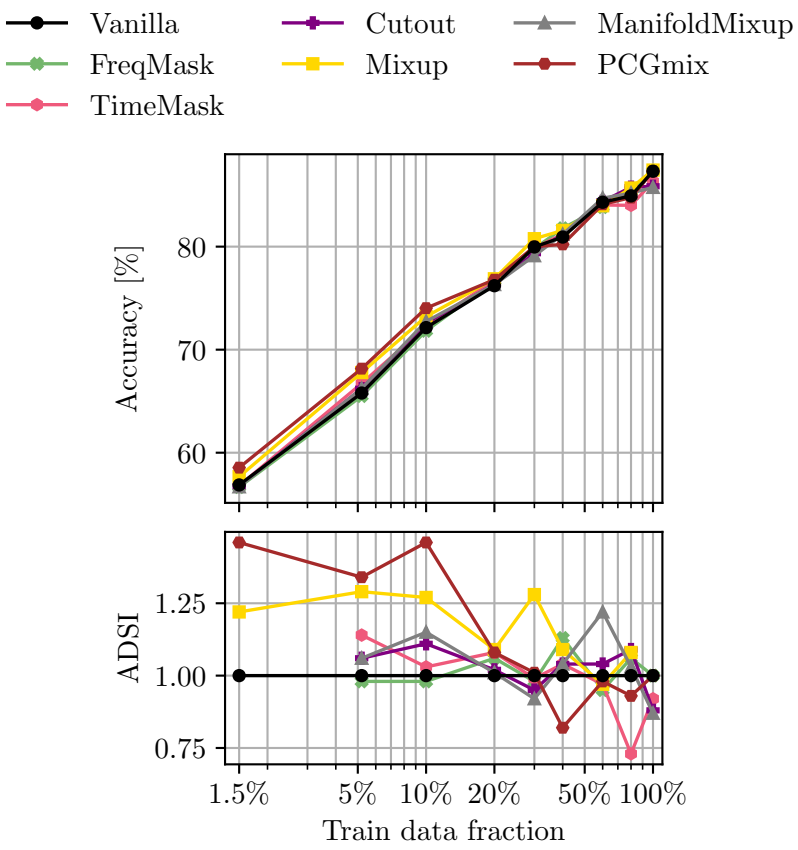


Figure 4.10: (top) Accuracy and (bottom) ADSI vs. training-data fraction in spectrogram domain experiments using the 2D-ResNet. In some cases, the accuracies achieved by the augmentation method are outside the accuracy range achieved by vanilla, leaving the ADSI undefined. The x-axis scale is logarithmic.

The 2D-ResNet with spectrograms achieved higher accuracies than the 1D-ResNet in the time-series-domain experiments at higher training-data fractions. However, the overall results were very similar to those from the time-series experiments. In particular, PCGmix performed the best when small amounts of training data were available ($f < 20\%$). Since we transformed the data into spectrograms before augmentation, we did not use PCGmix+, which involves time-series magnitude warping, in the spectrogram domain. If we had performed augmentation first and then transformed the data into spectrograms, we could have used PCGmix+ with spectrograms. This might have impacted the results at $f = 20\%$ and $f = 30\%$, where PCGmix+ is overall the most dominant among all augmentations, as per the time-series domain.

4.2.3 Parameter Sensitivity Analysis

In this section, we analyse the sensitivity of three of the parameters that were fixed during our experimentation, with the focus on our method. The three parameters are:

- Number of training epochs ϵ: this parameter was fixed at $\epsilon = 50$ all of the experiments;
- Augmentation probability μ: this parameter was determined separately for each augmentation method and f;

- Mixing coefficient λ distribution: this parameter affected only methods that combined two instances to create a new one (Mixup, Manifold Mixup, and PCGmix(+)).

We analysed one parameter at the time, meaning that all others were fixed at the reported values used in our experiments. Due to huge time costs of the experiments, parameters μ and λ distribution were analysed only for our method. We focused on PCGmix+ in the 1D-ResNet experiments.

The other parameters in our experiments that are not analysed are the batch size and the baseline models' parameters. We found that smaller batch sizes work better, so we fixed it at $\nu = 64$ for the best trade-off between performance and speed. For the baselines, their parameters were determined through initial experimentation (for exact values, see Appendix A.3.1).

4.2.3.1 Number of Training Epochs

The number of training epochs and the learning rate are interconnected parameters. We found that because we used the one-cycle-learning-rate scheduler, changing the maximal learning rate α did not affect results as much as changing the number of training epochs did. Therefore, we left the maximal learning rate at $\alpha = 0.01$ and evaluated model performance across a range of values for ϵ. We made this analysis for training-data fraction of 10% using the no-augmentation (vanilla) approach. The results are shown in Figure 4.11.

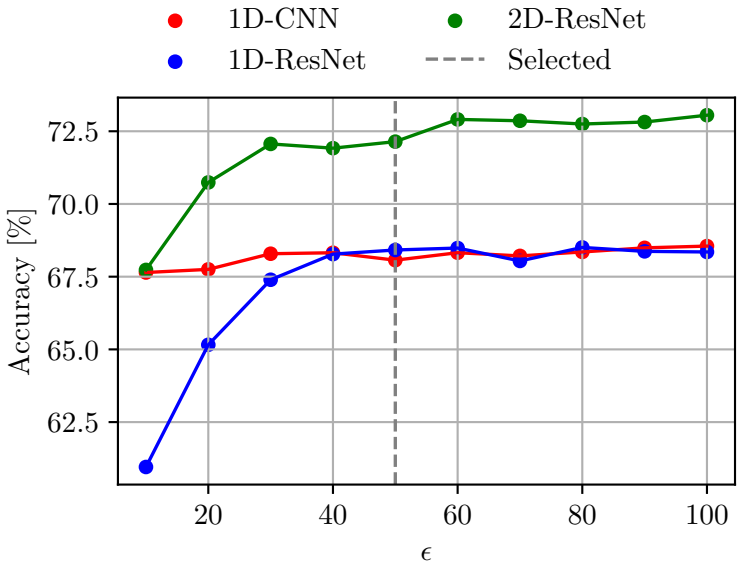


Figure 4.11: Accuracy as a function of the number of training epochs ϵ for $f = 10\%$ using vanilla.

The selected $\epsilon = 50$ turned out to be somewhat close to optimal for all models in terms of trade-off between reaching convergence and overtuning.

4.2.3.2 Augmentation Probability

We analysed augmentation probability for the PCGmix+ in 1D-ResNet experiments. For each training-data fraction f, we tested a range of augmentation probabilities: 0.2, 0.4, 0.6, 0.8, and 1.0. The results are shown in Figure 4.12.

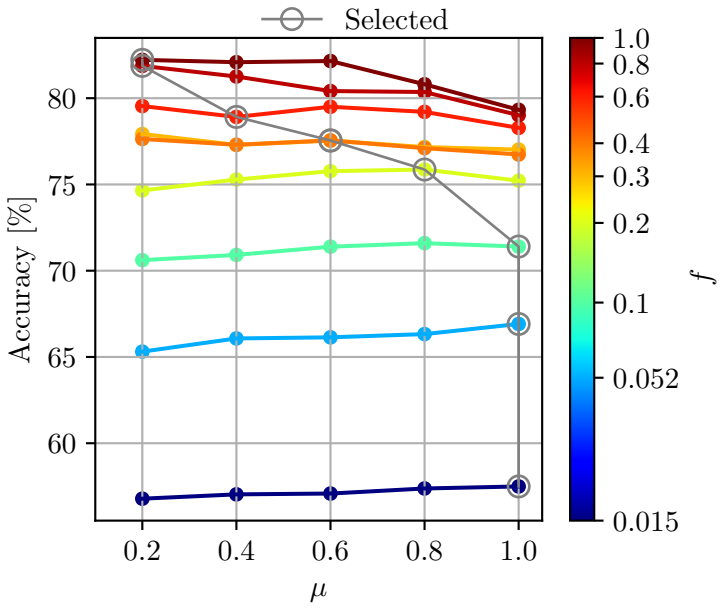


Figure 4.12: Accuracy as a function of the augmentation probability μ of PCGmix+ using 1D-ResNet for different training-data fractions.

The selected augmentation probabilities are marked with black circles. We found that using high probabilities ($\mu = 1.0$) for low fractions and low probabilities ($\mu = 0.2$) for high fractions, while gradually decreasing the probabilities for intermediate fractions, worked reasonably well.

4.2.3.3 Mixing Coefficient Distribution

In our experiments, the mixing coefficient λ was sampled from the beta distribution, $\lambda \sim \text{Beta}(\alpha, \alpha)$ (see Section 3.4.2). Typically, we set α to one, resulting in the beta function being equal to the uniform distribution, $\lambda \sim U(0, 1)$. However, to explore the effects of different mixing coefficient distributions on PCGmix(+), we also analysed cases with α values between 0.05 and 10.

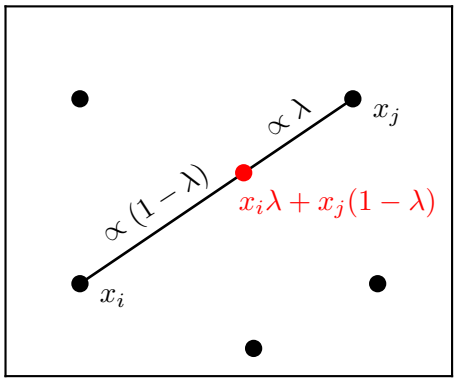


Figure 4.13: Illustration of generating a new instance by interpolation using the mixing coefficient λ. The new instance is positioned in the feature space along the line connecting the original instances. Note: the symbol $\propto$ means "proportional to" and is not the "alpha".

The mixing coefficient λ determines the proportions of the instances in the mixing pair that contribute to the generated instance. It defines the position in the feature space between the mixing pair where the new instance is generated (see illustration in Figure 4.13). In our method, this is an approximation since the interpolation is performed at the heart state level rather than at the individual instance level, as per Eq. (3.27).

When α is between zero and one, $0 \leq \alpha \leq 1$, the distribution prioritizes values closer to zero and one, leading to peaks near the extremes. This means that when interpolating, the generated points are closer to either the first or the second instance. Conversely, when α is larger than one, $\alpha > 1$, the distribution prioritizes values closer to 0.5. In this scenario, the generated points tend to be in the middle between the two instances. These differences in distribution shapes are depicted in Figure 4.14a, highlighting the varying interpolation behaviours based on the choice of α. The accuracy results for the PCGmix(+) as a function of α for training-data fraction of 10% in the in 1D-ResNet experiments are shown in 4.14b.

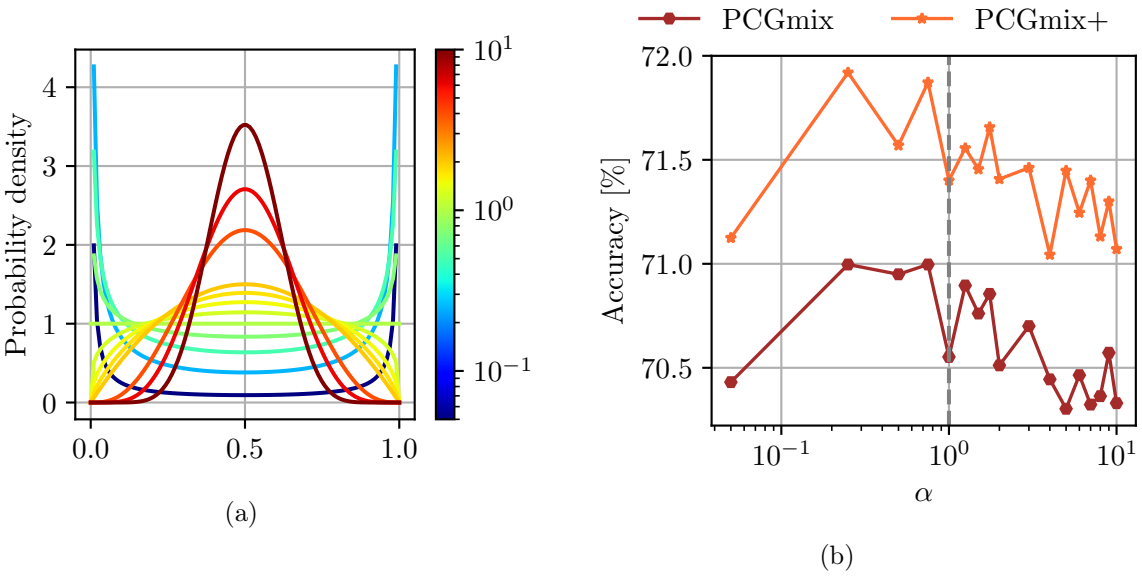


Figure 4.14: Mixing coefficient distribution analysis: (a) beta function probability density for different α values and (b) accuracy of the PCGmix(+) as a function of α for training-data fraction of 10% in the in 1D-ResNet experiments. The grey vertical line in (b) indicates $\alpha = 1.0$, which was the value used in our experiments.

When the sampling distribution is close to uniform, PCGmix(+) performs the best. When α is very small, the generated instances closely resemble the original data. Conversely, when α is high, the generated instances are consistently near the midpoint between the original instances. In both extremes, the variability of the augmented instances is limited, which is likely the reason for reduced performance.

4.2.4 Impact of Constrained Mapping on Reliability of Generated Instances

The original mapping function used in our experiments, ϕ, mapped each instance to an instance with the same class label (see Algorithm 3.3). This mapping function had no additional restrictions, allowing the mixing of instances with any characteristics. In this section, we analyse whether imposing constraints on the mapping function to mix only

similar instances increases the reliability of the generated instances by measuring their performance compared to the original approach. We have previously already conducted a similar analysis that was published in [278]. Due to the significant time costs of the experiments, these constrained mapping functions were implemented exclusively in combination with the PCGmix+ method.

Before proceeding, let us first discuss the concept of mixing diversity, as per Definition 4.1, which will be important in the following analysis.

Definition 4.1 (Mixing Diversity). *Mixing diversity* is the number of distinct pairs of original instances that can be mixed to generate new instances, where each pair $x_i \to x_j$ is considered different from $x_j \to x_i$.

For a set of n instances where all instances are allowed to mix, the mixing diversity N can be computed simply as

$$N = n(n-1). \tag{4.1}$$

This formula represents the total number of unique pairs, considering that the pair $x_i \to x_j$ is different from $x_j \to x_i$. The visual representation of our proposed method in Figure 3.6 illustrates why these two pairs are indeed different.

In our approach, the default mapping function ϕ is constrained to allow mixing only between instances with the same class label. This constraint divides all instances into two groups, each containing instances of a single class. Consequently, the number of valid mixing pairs is reduced, as mixing can only occur within these groups.

In general, for a mapping function with constraints that splits the instances into $k = 1, 2, \ldots, m$ groups of sizes n_k, within which the instances can mix, the mixing diversity becomes

$$N = \sum_{k=1}^{k=m} n_k(n_k - 1), \quad \text{where} \sum_{k=1}^{k=m} n_k = n. \tag{4.2}$$

Figure 4.15 illustrates the dependence of mixing diversity on the number of groups k. The plot shows the diversity values for different k, assuming all groups are of equal size, $n_k = \frac{n}{k}\ \forall k \in \{1 \ldots, m\}$, for simplicity.

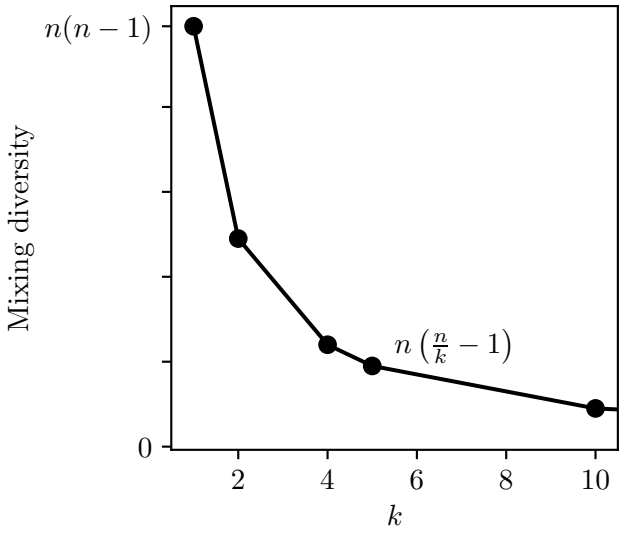


Figure 4.15: Dependence of mixing diversity on the number of groups k formed by constraints of the mapping function, with the assumption that all groups are of equal size for simplicity.

4.2.4.1 Mix Heartbeats With the Same CVD

To check if mixing instances with the same CVD increases reliability of generated instances, we implemented a mapping function ϕ_{CVD} that not only preserved the original class labels by mapping each instance to an instance with the same label, but also ensured that instances representing abnormal subjects were mapped to counterparts with the same CVD diagnosis. Overall, 52 (9%) of the training data PCGs did not have a specified CVD and were treated as having a separate CVD diagnosis from others. The results for $f \leq 30\%$ are shown in Table 4.10.

Table 4.10: Accuracy comparison of ϕ and ϕ_{CVD} in combination with PCGmix+.

f	1D-CNN			1D-ResNet		
	ϕ	ϕ_{CVD}	p-value	ϕ	ϕ_{CVD}	p-value
1.5%	**59.49 ± 5.95**	59.40 ± 5.87	0.48	57.49 ± 6.07	**57.63 ± 6.46**	0.56
5.2%	**67.21 ± 4.46**	66.99 ± 4.63	0.40	**66.91 ± 5.11**	66.35 ± 5.18	**0.03**
10%	70.29 ± 2.52	**70.51 ± 2.35**	0.36	71.40 ± 2.59	**71.78 ± 3.19**	0.23
20%	72.71 ± 2.07	**72.92 ± 1.93**	0.54	**75.87 ± 3.09**	75.07 ± 3.08	**0.01**
30%	**73.19 ± 1.39**	72.98 ± 1.95	0.64	77.55 ± 2.71	**78.03 ± 2.24**	0.33

The large p-values from the paired t-test where ϕ_{CVD} outperforms ϕ show that the differences in results are statistically insignificant and that mixing instances having different CVDs likely does not introduce any bias during model training.

4.2.4.2 Mix Heartbeats Recorded Under the Same Experimental Setup

To verify whether imposing a constraint to only allow mixing of instances recorded under the same experimental setups (recording equipment, signal quality, recording environments, body positions, and diagnostic methods) increases reliability, we implemented a mapping function, denoted as ϕ_{setup}. This function not only preserved the original class labels but also ensured that instances were mixed only with counterparts from the same dataset among the six datasets that compose the PhysioNet database. The results for $f \leq 30\%$ are shown in Table 4.11.

Table 4.11: Accuracy comparison of ϕ and ϕ_{setup} in combination with PCGmix+.

f	1D-CNN			1D-ResNet		
	ϕ	ϕ_{setup}	p-value	ϕ	ϕ_{setup}	p-value
1.5%	**59.49 ± 5.95**	59.37 ± 5.98	0.35	**57.49 ± 6.07**	57.05 6.12	**0.05**
5.2%	**67.21 ± 4.46**	66.76 ± 4.29	0.07	**66.91 ± 5.11**	65.86 5.14	**0.00**
10%	**70.29 ± 2.52**	70.06 ± 2.55	0.38	**71.40 ± 2.59**	71.38 2.62	0.92
20%	**72.71 ± 2.07**	72.11 ± 1.86	0.16	**75.87 ± 3.09**	75.31 2.87	0.14
30%	**73.19 ± 1.39**	71.57 ± 5.93	0.29	**77.55 ± 2.71**	77.27 2.16	0.58

Mapping ϕ_{setup} does not improve performance over ϕ. This suggests that generating instances by mixing heartbeats recorded under different experimental setups most likely results in realistic instances.

4.2.4.3 Mix Heartbeats From the Same Subject

We also implemented mapping function that only mixed instances from the same PCG, ϕ_{PCG}. This mapping mixed instances from the same subject, ensuring the same CVD, recording settings, and class label. The results for $f \leq 30\%$ are shown in Table 4.12.

Table 4.12: Accuracy comparison of ϕ and ϕ_{PCG} in combination with PCGmix+.

f	1D-CNN			1D-ResNet		
	ϕ	ϕ_{PCG}	p-value	ϕ	ϕ_{PCG}	p-value
1.5%	**59.49 ± 5.95**	59.30 ± 5.86	0.19	**57.49 ± 6.07**	56.84 ± 5.92	**0.01**
5.2%	**67.21 ± 4.46**	66.03 ± 4.54	**0.00**	**66.91 ± 5.11**	64.87 ± 5.59	**0.00**
10%	**70.29 ± 2.52**	69.37 ± 2.61	**0.00**	**71.40 ± 2.59**	70.51 ± 3.53	**0.02**
20%	**72.71 ± 2.07**	71.41 ± 2.40	**0.01**	**75.87 ± 3.09**	74.68 ± 3.21	**0.01**
30%	73.19 ± 1.39	**73.34 ± 1.95**	0.75	**77.55 ± 2.71**	76.94 ± 1.91	0.17

The ϕ_{PCG} function is subject to many constraints, which substantially reduces the mixing diversity of the generated instances as shown in Figure 4.16. This is most likely the reason for its significantly reduced performance compared to ϕ.

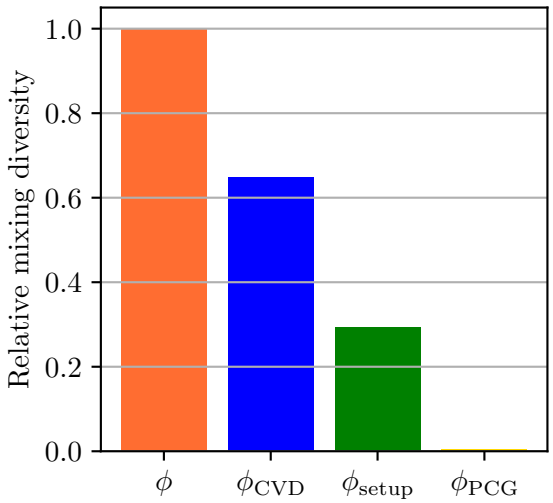
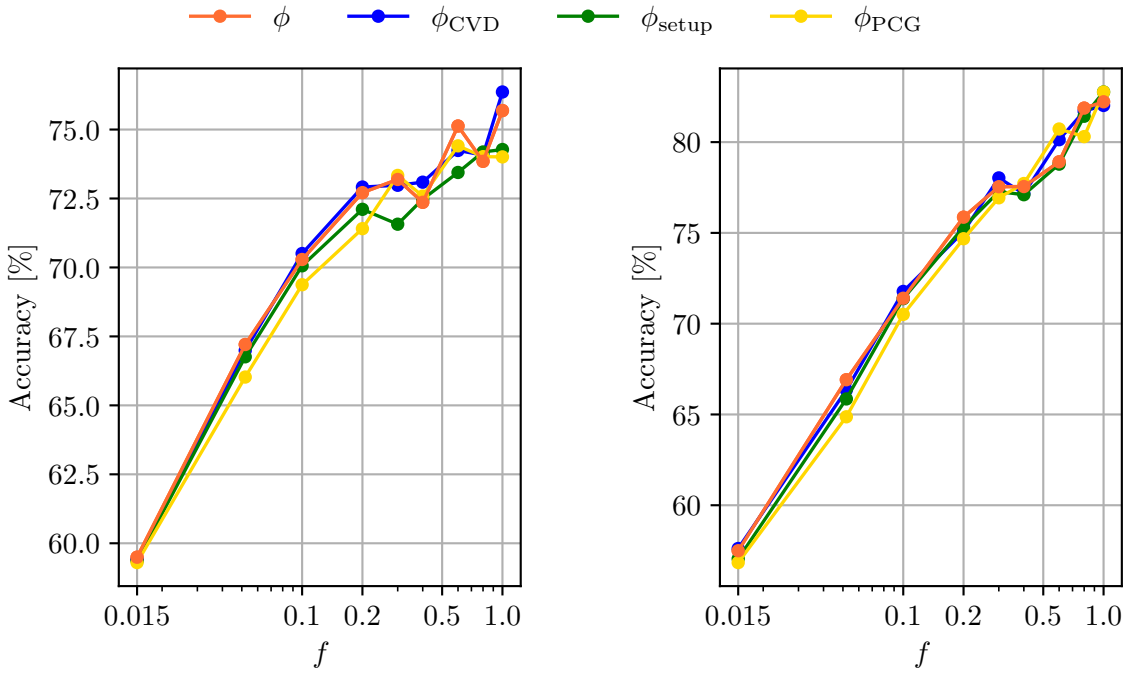


Figure 4.16: Theoretical mixing diversity of the analysed mapping functions relative to the original. The diversity of ϕ_{PCG} is about 0.5% of that of ϕ. Mapping function ϕ allows mixing between instances of the same class label, whereas the others impose additional constraints: ϕ_{CVD} mixes the same CVDs, ϕ_{setup} mixes the same recording setups, and ϕ_{PCG} mixes the same subjects.

Figure 4.16 shows the mixing diversity of the mapping functions, computed based on the entire training dataset using Eq. (4.2). In practice, the number of distinct mixing pairs selected is lower than the theoretical mixing diversity for several reasons. First, mixing pairs are selected randomly within the constrained group, meaning not all possible pairs are guaranteed to be chosen. Second, the shuffling of batches affects the selection of pairs, as pairs are generated separately for each batch. Third, for random subsets, the specific recordings sampled influence the diversity since theoretical values are based on the entire training data. Fourth, because pairs are selected randomly, it is possible for an instance to be mapped to itself, further reducing the effective mixing diversity.

We conclude that although a mapping function with additional constraints can potentially reduce the bias of generating unrealistic instances, this positive effect is outweighed by the negative effect of reducing diversity in the training data. The experiments confirm the reliability of generating examples by mixing instances with different types of CVDs and/or with different experimental setups. The results of different mapping functions for all training-data fractions are shown in Figure 4.17.



(a) 1D-CNN (b) 1D-ResNet

Figure 4.17: Accuracy vs. training-data fraction for different mapping functions in combination with PCGmix+.

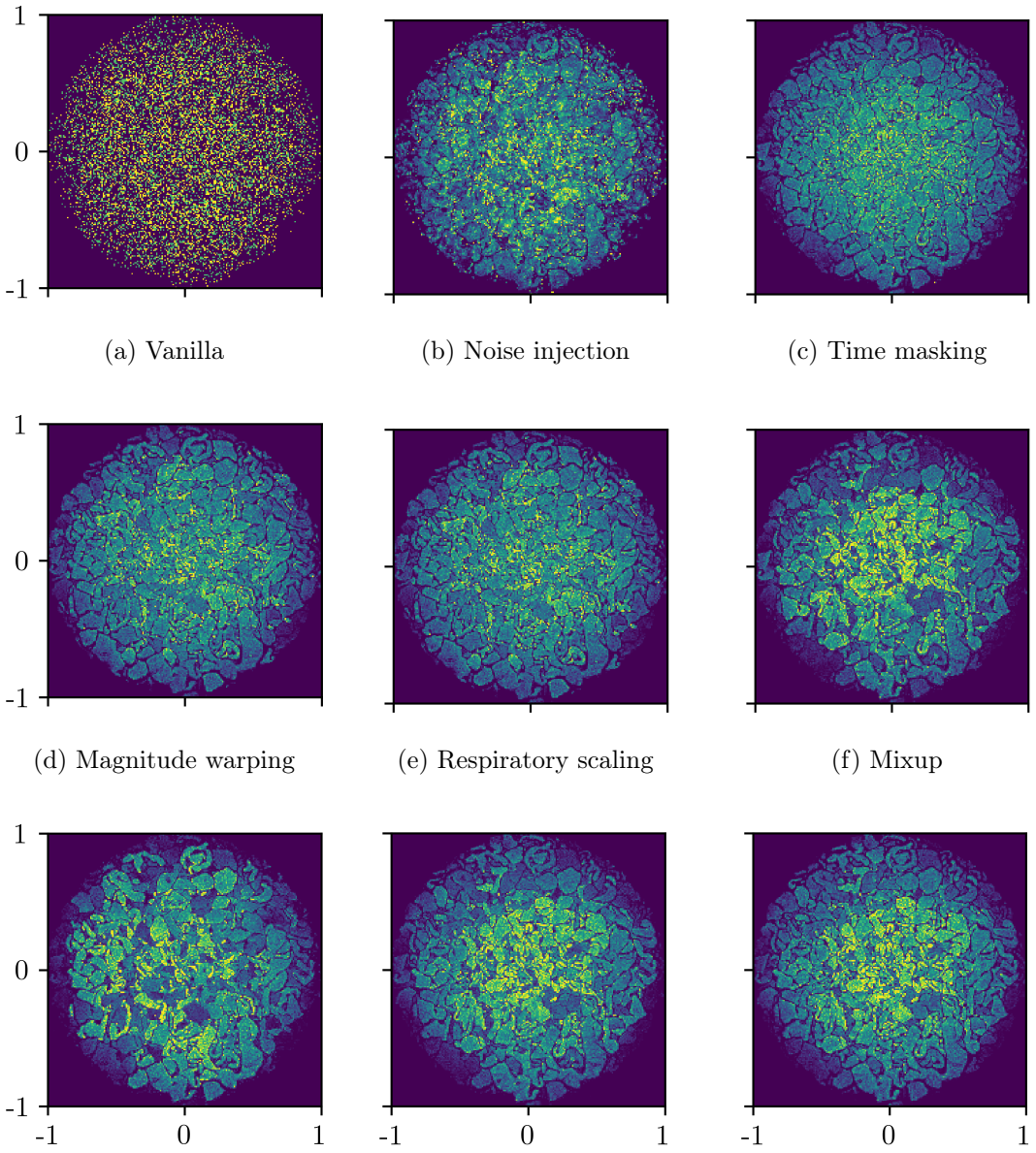
4.2.5 Feature Space Exploration

When generating new training instances, it is important to ensure that they exhibit a high degree of variability and at the same time remain realistic. Instances that are too similar to the original instances do not improve the robustness of the model, while instances that differ too much from the real data introduce noise into the model.

To gain insights into how different data-augmentation strategies affect the feature space coverage of the NN model, we performed an analysis that involved extracting the model's latent space features. This analysis was conduced in the time-domain since more augmentation strategies were implemented there instead of in the spectrogram domain. The latent space probability distribution of the original and augmented instances from different methods for is shown in Figure 4.18.

First, we trained a NN model on the entire training dataset to acquire a latent space representation of our data. For this purpose, a CNN ResNet architecture as described in [279] and implemented by [280] was used. This model was selected because it has only 128 units in the hidden layer before the output, resulting in a 128-dimensional latent space. In comparison, the 1D-CNN and 1D-ResNet models used in our experiments have 9968 and 39936 hidden units, respectively, which would create an enormous computational burden

when reducing the latent space features to two dimensions for visualization purposes. The latent space model was trained for 10 epochs with a batch size of 32 and a one-cycle learning-rate scheduler with a maximum learning rate of 0.00089 on the entire training set ($f = 1.0$).



(a) Vanilla (b) Noise injection (c) Time masking

(d) Magnitude warping (e) Respiratory scaling (f) Mixup

(g) Manifold Mixup (h) PCGmix (i) PCGmix+

Figure 4.18: Probability distributions of latent space features extracted from the pre-trained NN model from the $f = 10\%$ experiments, illustrating feature space coverage: (a) original training instances and (b)-(i) augmented instances using various augmentation methods. The x and the y axis correspond to the first and the second t-SNE component, respectively.

Subsequently, we employed this trained model as a latent space feature extractor during the experiments ($f = 10\%$). The instances generated by the augmentation methods were

used as input to our independent latent space extractor model, saving the corresponding features. For comparison, the features obtained from the no-augmentation (vanilla) method were also saved.

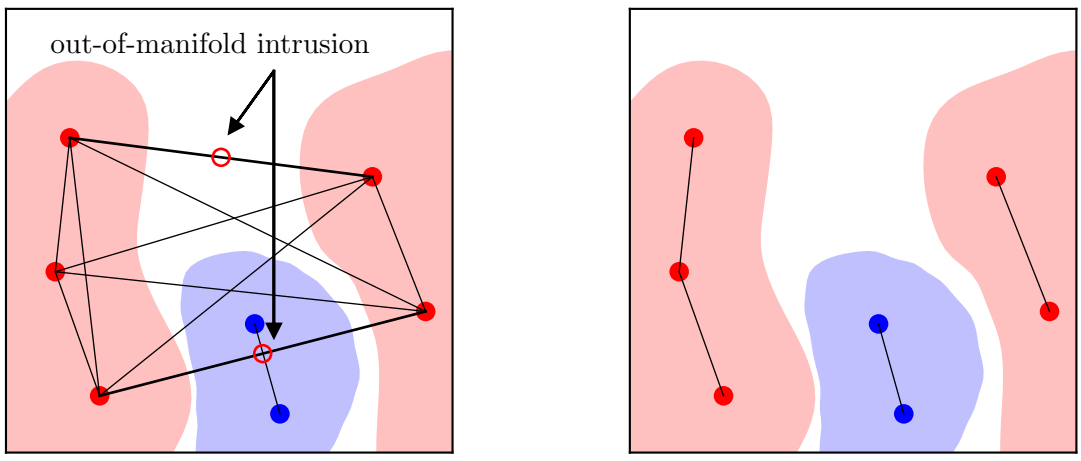
The vanilla probability distribution in Figure 4.18a represents the real-world instances and forms the real-world data manifolds. The empty holes in the feature space that are not covered by any instances may or may not correspond to realistic instances. If an instance is generated in a region that does not represent realistic instances, it represents an out-of-manifold intrusion [281], which introduces noise into the training data and reduces the model's generalization ability. Ideally, generated instances would cover all areas of the real-world data manifolds without extending beyond them.

Methods that transform a single instance (such as noise injection, time masking, magnitude warping, and respiratory scaling) have the probability densities distributed more smoothly across the NN feature space, potentially including many out-of-manifold instances. In contrast, methods that mix two original instances to generate a new one (like Mixup, Manifold Mixup, and PCGmix(+)) form smaller, higher probability clusters with more distinct borders between different regions in the NN feature space, potentially indicating the edges of the manifolds.

4.2.6 Investigating Out-of-manifold Intrusion

In the previous section, we visualized feature space coverage of the augmentation methods and introduced the concept of out-of-manifold intrusion, where instances are generated in the regions of the feature space where they should not be. In this section, we analyse to what extent our method suffers from this issue.

There are two types of out-of-manifold intrusion, both depicted in Figure 4.19a. The coloured areas correspond to the manifolds of the real-world instances where each colour represents a distinct class. In the first case, an instance is generated in a region where there are no realistic instances. In the second case, an instance is generated within the manifold of the opposite class.

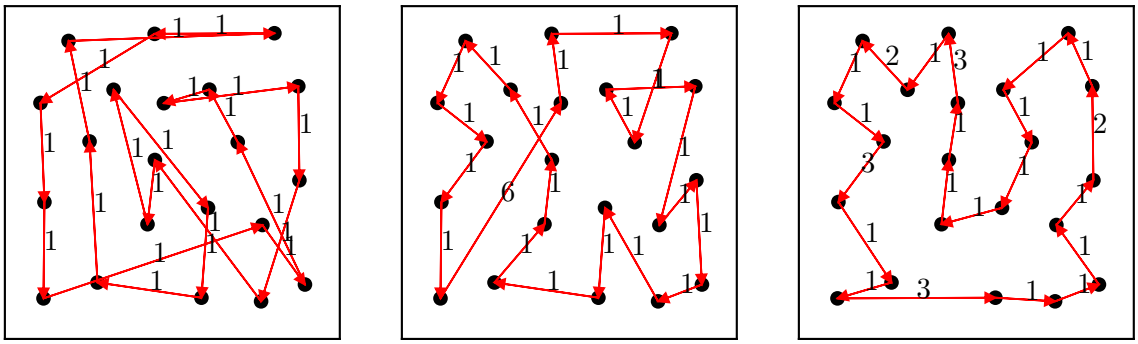


(a) (b)

Figure 4.19: Illustrations depicting (a) two types of out-of-manifold intrusion, where instances mix regardless of their mutual distance, leading to the generation of instances outside of the real-world data manifolds or within the manifold of the opposite class, and a (b) constraint allowing only nearby instances to mix, mitigating manifold intrusion.

To investigate the effect of manifold intrusion in our method, we implemented a constraint in the mapping function to only mix nearby instances (see Figure 4.19b). First, we extracted latent space features of the training data (as described in Section 4.2.5). During each training iteration (batch), we computed pairwise Euclidean distances between instances for each class label. For each instance, the others were assigned ranks r based on their distance to that instance (nearest instance: $r = 1$, second nearest: $r = 2$, etc.). These ranks were then adjusted based on the amount of nearest neighbors parameter κ as follows: the closest κ instances were assigned a new rank $\hat{r} = 1$, while others were assigned a new rank as $\hat{r} = r - \kappa + 1$. For example, with $\kappa = 5$, the closest five instances ($r \leq \kappa$) were assigned $\hat{r} = 1$, the 6-th closest instance ($r = 6$) was assigned $\hat{r} = 2$, etc. Next, we solved a traveling salesman problem (TSP) [282] for each class label group. TSP is a combinatorial optimization problem that seeks the route with minimal total travel cost while visiting each city exactly once and returning to the origin. In our context, instances represent cities and the cost to travel from x_i to x_j equals the nearest neighbor rank of x_j to x_i, $\hat{r}(x_i \rightarrow x_j)$. Note that this is an asymmetric TSP, as the costs $\hat{r}(x_i \rightarrow x_j)$ and $\hat{r}(x_j \rightarrow x_i)$ are generally not the same.

By adjusting κ, we control the amount of nearest neighbors each instance can mix with. This is a soft constraint: if the nearest κ instances have already been "visited," the instance will mix with one of the higher rank $\hat{r}$. At one extreme, with $\kappa = 1$, instances are soft-constrained to mix only with their nearest neighbor. At the other extreme, with κ set to a value equal to or larger than the batch size minus one ($\kappa \geq \nu - 1$), instances can mix freely without constraints, equivalent to the default setting. Illustrations of examples of TSP solutions for different numbers of nearest neighbors are shown in Figure 4.20.



(a) $\kappa = 12$ (b) $\kappa = 6$ (c) $\kappa = 2$

Figure 4.20: Illustrations of TSP solutions for different amounts of nearest neighbors. The numbers denote the ranks $\hat{r}(x_i \rightarrow x_j)$ of the connections (costs). Note that TSP solutions are cyclic, meaning each instance acts as both x_i and x_j exactly once.

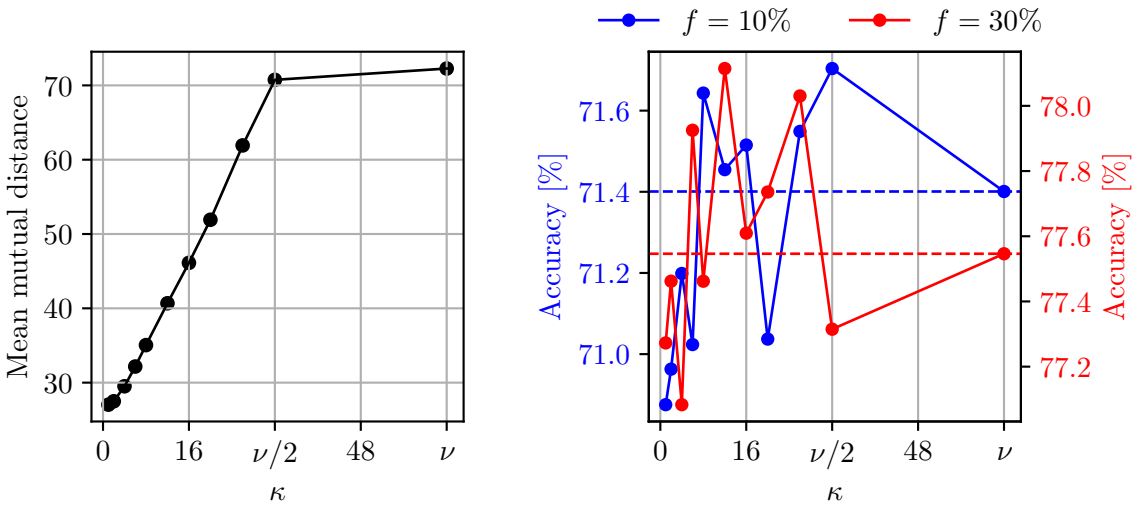
We employed a highly efficient symmetric TSP solver from [283] to initially solve the TSP as if it was symmetric. We then used that solution as the initial guess for the asymmetric TSP solver obtained from [284]. This approach enabled us to obtain near-optimal solutions in a significantly reduced computation time. The results for different numbers of nearest neighbors for PCGmix+ in the $f = 10\%$ and $f = 30\%$ 1D-ResNet experiments is shown in Figure 4.21

Mixing diversity (defined and explained in Section 4.2.6), indicating how many different

mixing pairs can be created, in this case for each batch becomes

$$N = 2\frac{\nu}{2}\kappa, \quad \text{where } \kappa \leq \frac{\nu}{2} - 1, \tag{4.3}$$

showing a linear dependence with κ. The value 2 in Eq. (4.3) corresponds to the number of classes. This equation is an approximation and differs from practical scenarios in two ways. First, it assumes a hard constraint for allowing each instance to mix with the closest κ, rather than the soft constraint used in the TSP case. Second, it presumes that the batch is perfectly balanced with respect to the classes, which, while generally expected, is not always guaranteed.



(a) (b)

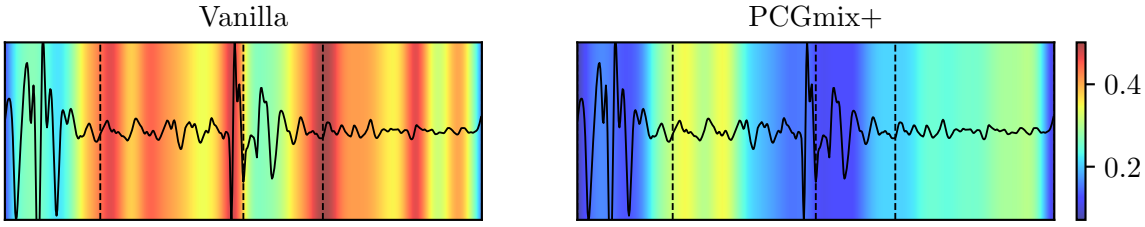
Figure 4.21: Investigating out-of-manifold intrusion of PCGmix+ (1D-ResNet): (a) mean mutual distance between mixed pairs and (b) accuracy vs. the amount of nearest neighbors. The horizontal dashed lines in (b) represent $\kappa = \nu$ which equals to no constraints.

The experiments ($\nu = 64$) depicted in Figure 4.21b suggest that our method might be affected from out-of-manifold intrusion instance generation, however the differences are not statistically significant. When restricted to only mix nearby instances, solutions with κ between 8 and 26 tend to perform better than those with no restrictions ($\kappa = \nu$). Solutions where $\kappa < 8$ likely suffer from greatly reduced mixing diversity. However, overall results are somewhat inconsistent, making it difficult to draw firm conclusions based on these limited experiments. Additional tests will be required to validate these initial observations and confirm our assumptions more robustly.

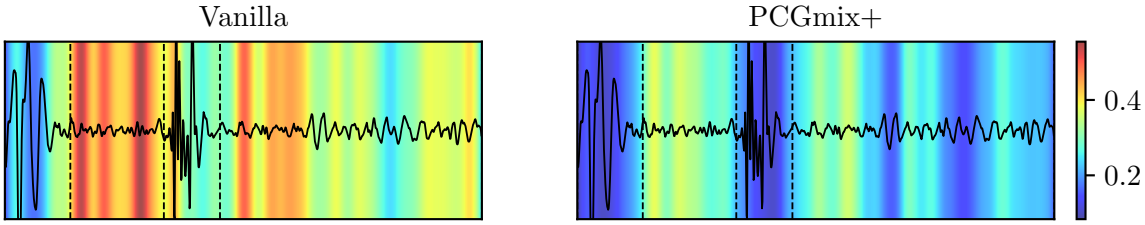
Since we only mix instances of the same class labels, solutions at $\kappa = \nu/2 - 1$ already have no restrictions for batches that are perfectly balanced. Thus, there is minimal difference in mean mutual distance between solutions at $\kappa = \nu/2 - 1$ and $\kappa = \nu$.

4.2.7 Saliency Map Visualizations

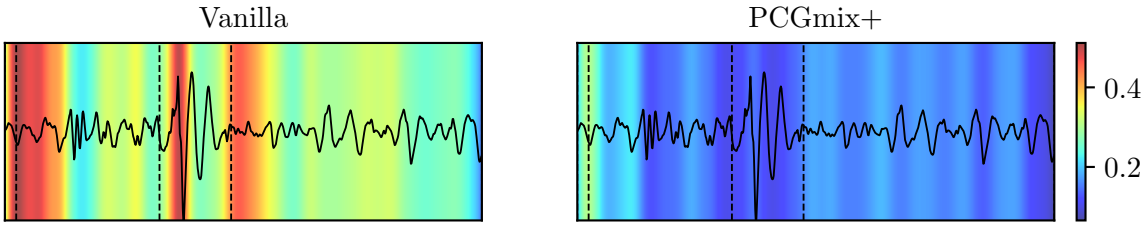
NNs are often considered "black boxes" because their decision-making process is not transparent. However, techniques like saliency maps can provide insights into the NN's decision-making process by highlighting the input data areas that most influence the model's decisions. Overview of XAI is given in Section 2.7.



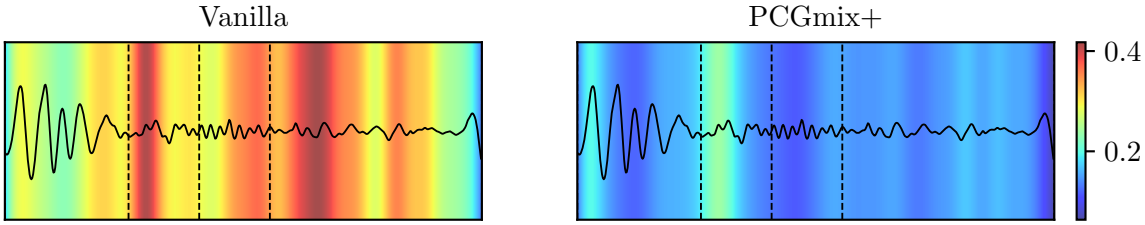
(a) Normal



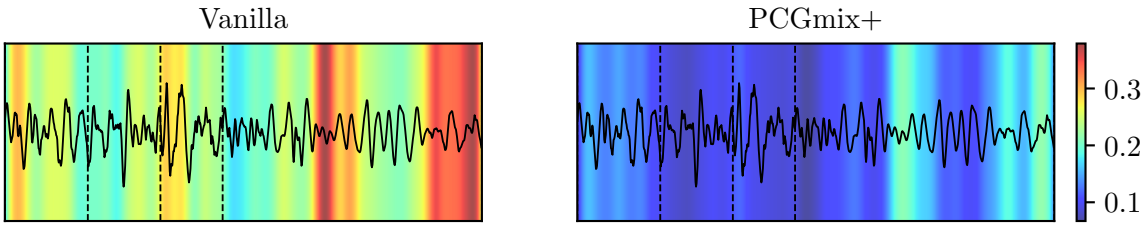
(b) Innocent or benign murmurs



(c) Coronary artery disease



(d) Mitral regurgitation



(e) Aortic disease

Figure 4.22: Saliency maps of (a) normal and (b)–(d) abnormal PCGs for the (left) no-augmentation approach and (right) PCGmix+.

There are various methods of extracting the saliency maps from NNs, all of which have been show to work well with image data inputs [285]–[289]. In this study, we followed the approach of Simonyan et al. [289] and extracted saliency maps by computing the gradients of the output with respect to each data point of the input. Data points with higher gradients have higher impact on the model's output, indicating their importance in classifying heart sounds. Saliency maps reveal regions that the NN pays more attention to, similar to how humans make decisions based on certain parts of a PCG.

We extracted the saliency maps of the time-series inputs and then summed the saliencies of the frequency-channels into a single channel. The visualizations are given in Figure 4.22. Each visualization was computed as average saliency map across all trained models from experiments with the same method and f combination (for $f = 10\%$, each saliency was averaged across 50 maps, as we ran trials with 50 RNG seeds for each method for this f). The maps were smoothed using a Gaussian filter.

In the vanilla approach, the saliency maps exhibit very strong, high values, indicating that the model over-focuses on certain parts of the PCGs. This can lead to an over-reliance on specific features, potentially missing other relevant information in the signal. In contrast, when using the PCGmix+ method, the saliency maps show that the model has more evenly distributed attention across the entire signal. This balanced view suggests that the model is less prone to over-focusing on specific regions and instead evaluates the PCGs more comprehensively. Despite this more balanced attention, the model still identifies and understands the crucial characteristics necessary for making accurate decisions.

From the perspective of XAI, understanding which parts of the heart-sound signals the NN considers important allows clinicians to gain confidence in the model's decisions, ensuring that critical diagnostic features are not overlooked.

4.2.8 Utilization of Saliency Information

In the previous section, we showed how to gain insights into the model's decision-making by extracting saliency maps. In this section, we investigate the utilization of saliency information to improve the augmentation strategy. Originally, our method interpolates the beginnings of heartbeat states (see Figure 3.6), which can distort important regions of the signal, potentially rendering them unrecognisable for decision-making purposes. This issue arises when two critical regions, such as abnormal sounds, are interpolated together, resulting in a loss of clarity. Additionally, our method may trim essential regions when it shortens the second signal's heartbeat state if it is longer than the corresponding state of the first signal.

By utilizing saliency information from NNs, we aim to improve this process. Specifically, saliency-guided optimization can help in shifting heartbeat states and controlling which parts of the states are interpolated, preserving crucial regions for decision-making. Mathematically, this involves optimizing M and S matrices in Eq. (3.27) to improve the quality of the augmented instances and thereby the model performance.

We tested two variations of how the parts of the heartbeat states get selected for the interpolation using the saliency information. We refer to them as the *MaxSum* and the *MaxEnv* approach. Both are mathematically formulated in the following. Let s_i be the 1-dimensional saliency map of the signal x_i and let s_i^{st}, where st $\in$ {S1, Sys., S2, Dia.}, be the saliency map of the corresponding heart state with length l_i^{st}. The padding function $\text{pad}(s, l)$ pads s with l zeros on the right-hand side. The rolling function $\text{roll}(s^{\text{pad}}, d)$ rolls the elements of s^{pad} to the right by d positions, where the elements that roll past the last position are re-introduced at the first.

For each approach, there are two cases that need to be treated, depending on which of the two states is longer. Let us start with the MaxSum approach. In the case where

$l_i^{\text{st}} > l_j^{\text{st}}$, the optimization function can be written as

$$\underset{d}{\text{maximize}} \sum_{t=1}^{d} s_i^{\text{st}}(t) + \sum_{t=d+1}^{d+l_j^{\text{st}}} \left(s_i^{\text{st}}(t)\lambda + s_j^{\text{st, roll}}(d,t)(1-\lambda) \right) + \sum_{t=d+l_j^{\text{st}}+1}^{l_i^{\text{st}}} s_i^{\text{st}}(t), \tag{4.4}$$

where $s_j^{\text{st, roll}}(d) = \text{roll}(\text{pad}(s_j^{\text{st}}, l_i^{\text{st}} - l_j^{\text{st}}), d)$. In the case where $l_i^{\text{st}} < l_j^{\text{st}}$, optimization follows as

$$\underset{d}{\text{maximize}} \sum_{t=d+1}^{d+l_i^{\text{st}}} \left(s_i^{\text{st, roll}}(d,t)\lambda + s_j^{\text{st}}(t)(1-\lambda) \right), \tag{4.5}$$

where $s_i^{\text{st, roll}}(d) = \text{roll}(\text{pad}(s_i^{\text{st}}, l_j^{\text{st}} - l_i^{\text{st}}), d)$.

In both cases, the shift $d \in \{0, 1, \ldots, |l_i^{\text{st}} - l_j^{\text{st}}|\}$ determines by how many positions to the right the beginning of the shorter state has to be shifted relative to the beginning of the longer state before the interpolation. The case with $d = 0$ corresponds to our original (default) approach.

For the MaxEnv approach, the optimization function in the case where $l_i^{\text{st}} > l_j^{\text{st}}$ can be written as

$$\underset{d}{\text{maximize}} \sum_{t=1}^{d} s_i^{\text{st}}(t) + \sum_{t=d+1}^{d+l_j^{\text{st}}} \max\{s_i^{\text{st}}(t), s_j^{\text{st, roll}}(d,t)\} + \sum_{t=d+l_j^{\text{st}}+1}^{l_i^{\text{st}}} s_i^{\text{st}}(t). \tag{4.6}$$

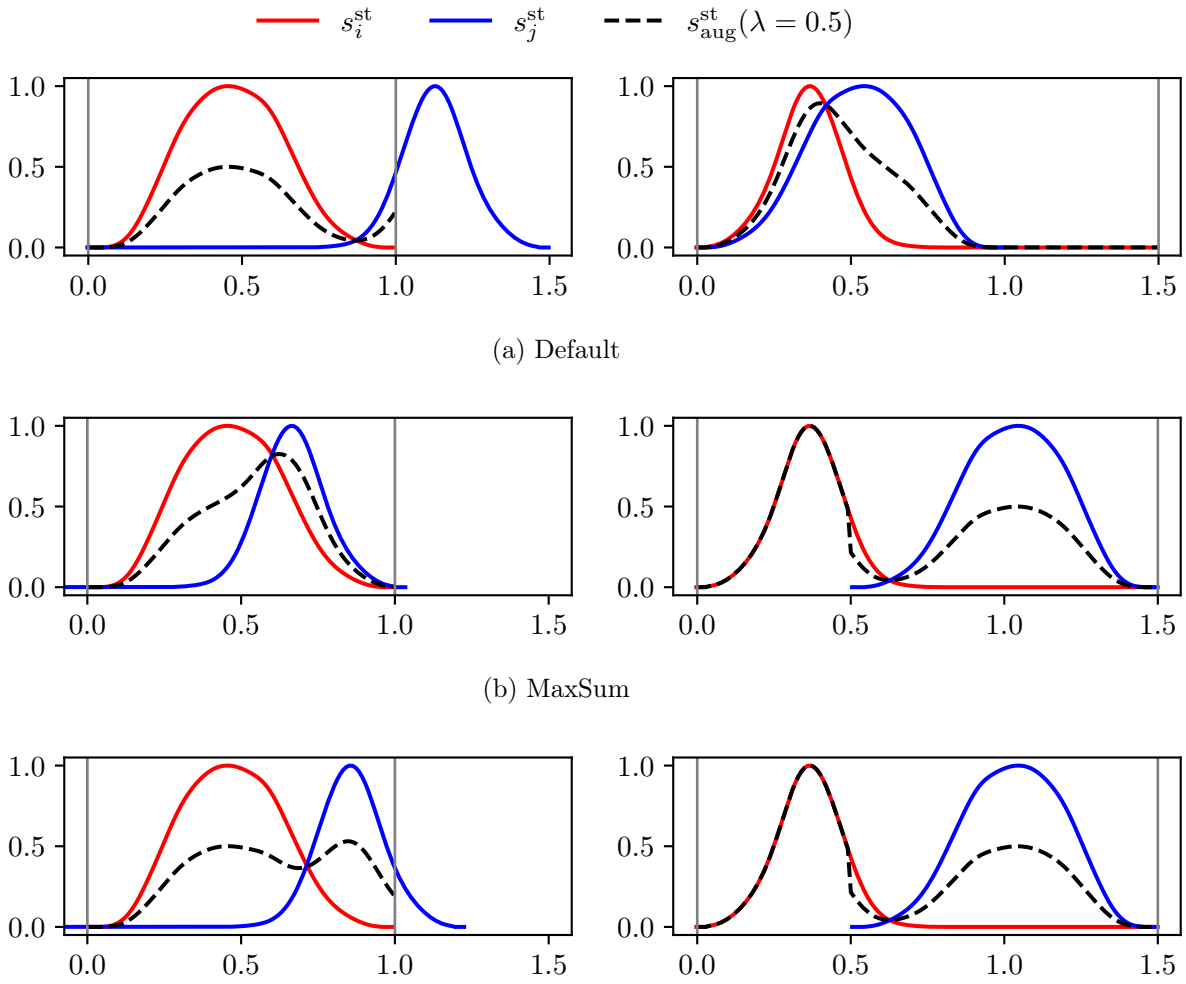
In the case where $l_j^{\text{st}} > l_i^{\text{st}}$, the optimization function is

$$\underset{d}{\text{maximize}} \sum_{t=d+1}^{d+l_i^{\text{st}}} \max\{s_i^{\text{st, roll}}(d,t), s_j^{\text{st}}(t)\}. \tag{4.7}$$

The illustrations of both approaches along with the default approach are depicted in Figure 4.23. The vertical grey lines indicate the beginning and end of the generated instance. The red and blue lines represent the state saliencies for the first and second instances, respectively. The black dashed line illustrates the interpolated saliency resulting from the combination of these two states.

In the MaxSum case, the second instance, x_j, is shifted so that the most salient parts are included in the interpolation process. In Figure 4.23b, this is represented by shifting the blue line so that the area under the black dashed line, representing the saliency of the generated instance, is maximized.

In the MaxEnv case, the selection of parts is based on maximizing the sum of the highest values at each index across the saliency maps of the two signals. Here, blue line is shifted such that the area under an imagined envelope encompassing both the red and blue lines is maximized (see Figure 4.23c). This ensures that the most prominent features of each signal are preserved (have minimal overlap) in the interpolated result.



(a) Default

(b) MaxSum

(c) MaxEnv

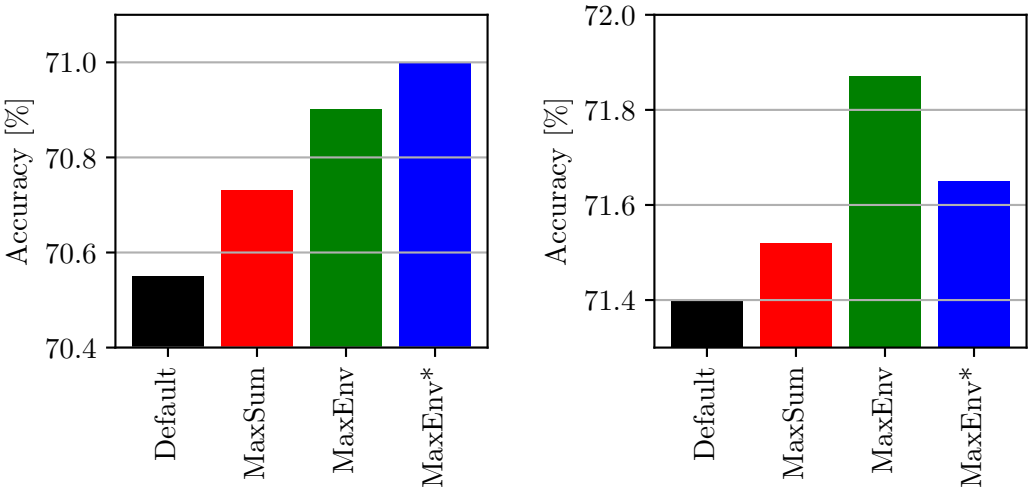
Figure 4.23: Comparison of default and saliency-guided augmentation strategies. The left column illustrates an example where $l_i^{\text{st}} < l_j^{\text{st}}$, whereas the right column shows an example where $l_i^{\text{st}} > l_j^{\text{st}}$. The length of the augmented state is determined by l_i^{st}, as indicated by the grey vertical lines.

We tested the salinecy-guided approaches in 1D-ResNet experiments with $f = 10\%$ for both PCGmix and PCGmix+. During training, saliency maps were extracted from the no-augmentation vanilla model that was pretrained on the same training set (same f and data selection RNG seed). Additionally, for the MaxEnv approach, the PCGmix(+) pretrained models were also used as the saliency-map-extraction models, denoted as MaxEnv*. Each maximization was solved by iterating over all possible shifts to find the optimal solution. The results are given in Table 4.13 and in Figure 4.24.

The results indicate that using saliency information when deciding which parts of the heartbeat states to interpolate in our data-augmentation method can improve the model performance. However, the improvements observed in the conducted experiments are not statistically significant. Furthermore, our approach has limitations. The computed saliency maps may not be optimal, and other established methods, such as GradCAM [285], should be investigated. Additionally, the margin for improvement is small when the heartbeat states are similar in length, leaving limited room for shifting. Moreover, this approach

Table 4.13: Saliency-guided augmentation strategies in $f = 10\%$ 1D-ResNet experiments.

	PCGmix	P-value	PCGmix+	P-value
Default	70.55 ± 2.88	-	71.40 ± 2.59	-
MaxSum	70.73 ± 2.65	0.58	71.52 ± 2.79	0.69
MaxEnv	70.90 ± 2.80	0.24	**71.87 ± 2.70**	0.17
MaxEnv*	**71.00 ± 2.94**	0.14	71.65 ± 2.71	0.31



(a) PCGmix (b) PCGmix+

Figure 4.24: Saliency-guided augmentation strategies in $f = 10\%$ 1D-ResNet experiments.

is computationally expensive, requiring a pretrained model and the extraction of saliency maps, which results in two additional forward/backward passes for each training iteration. Note that in theory, saliency maps can be extracted from the model being trained without the need for a pretrained model, however, the saliency information would not be reliable, especially in the early stages of model training, where performance is typically low. Thus, only after addressing these limitations can saliency-guided approaches fully realize their potential in improving model performance.

4.2.9 Analysis of Computational Complexity

So far we have seen two approaches of how our method can potentially be improved. In the first case, a TSP was solved to restrict the mapping function to only mix nearby instances to prevent out-of-manifold intrusion instance generation, and in the second case, salieny maps were extracted from the pretrained model to optimize the interpolation part of generating new instances.

This section analyses the computational complexity of the two potential improvements in comparison with the original approach. We provide insight into both theoretical and experimental complexity for the three approaches: the default PCGmix+ method, a variant where only the κ closest instances are allowed to mix to prevent out-of-manifold intrusion, and a MaxEnv saliency-guided approach, as shown in Table 4.14. The experimental results are derived from the $f = 10\%$ 1D-ResNet experiments, conducted on a setup with four NVIDIA GeForce RTX 2080 GPUs and two Intel Xeon Gold 5122 CPUs.

Table 4.14: Computational complexity of our method. The experimental results are derived from the $f = 10\%$ 1D-ResNet PCGmix+ experiments. The results are averaged over 50 experiments.

	Default	Mix κ closest	Saliency-guided
Uses pretrained model		✓	✓
Latent space features extraction		✓	
Solves TSP		✓	
Saliency maps extraction			✓
Forward passes per iteration	1	3	3
Backward passes per iteration	1	2	3
Time per experiment	167 ± 14 s	280 ± 26 s	635 ± 80 s

The forward/backward passes per training iteration rows indicate the number of forward/backward passes required per training iteration, relative to the default approach. In standard model training, each iteration involves a single forward pass and a single backward pass. When using a pretrained model, the passes are effectively doubled, as the data undergoes forward and backward passes both during the initial training of the pretrained model and in the current training iteration. Similarly, for tasks such as computing gradients for saliency map extraction, both forward and backward passes are required, leading to a tripling of the forward and backward passes per iteration compared to the default approach. This increased number of passes reflects the additional computational complexity associated with these methods.

It is important to note that the compute time is influenced by other processes running concurrently with the experiments. These additional processes can vary and may significantly impact the overall experiment duration. Therefore, the reported times should be considered approximate and may not fully reflect the exact computational requirements of each method.

Although the training time for complex methods can be considerably longer due to the additional computational demands, this is generally not a major problem for the final model used in practice. Once a model is trained and used for inference or deployment, training time no longer matters. However, during the development phase, including parameter optimization and experimentation, the increased computational costs can be a significant hurdle.

4.2.10 UMCL Experiments

The goal of the final step of our analysis was straightforward: to verify the effectiveness of our augmentation method on the UMCL dataset for detection of decompensation episodes. To achieve this, we trained NNs using both time-series data and spectrograms as inputs. Unfortunately, the performance of the NN models was unsatisfactory, achieving around 60% accuracy in a 10-fold CV, using exactly the same splits as in the classical ML approach. This accuracy was significantly lower than that of the best classical model, which exceeded 70%.

Based on these results, we adopted a modified approach: we retrained a time-series NN using our augmentation method, with an added step. In each training iteration, we extracted classical features from the augmented instances, as described in Section 3.3.1, and saved them. The models were trained for 20 epochs, resulting in each CV split having 20 times the amount of the original from the augmented instances in the form of classical features. This strategy was primarily a time-saver since the NN training with augmentation

was already implemented. Theoretically, we could have simply applied the augmentation outside the neural network and then extracted and saved the features directly.

We could not use the default unconstrained mapping function, ϕ, to find the mixing pairs because the classical ML pipeline required smoothing the features of heartbeats from the same PCG to compute the final mean and SD features for the models. If we were to mix all instances (of the same label), it would not be clear to which patient the new instance belongs, and feature smoothing would not be possible. Since feature smoothing was a very important part of the classical ML approach, we instead employed ϕ_{PCG}, which only mixed instances from the same PCG. Thus, the augmented instances were assigned to the same PCGs as the original instances.

The classical ML experiments were then conducted iteratively. Starting with only the original data, 1/20 of the augmented data was incrementally added in each experiment iteration. In each iteration, we followed the same pipeline as our classical approach. First, bad segments were removed based on the envelope of their signal using the z-score parameter f_A. Then, the features of the heartbeats within the same PCG were smoothed using a sliding window of size w to create 2×177 mean and SD features. Finally, the experiments were ran using feature selection.

We observed that using low values for f_A leads to poor performance with augmented instances, so we set it to 3. The number of selected features was set to $k_b = 40$, and the sliding window size was set to $w = 4$. The results are given in Figure 4.25.

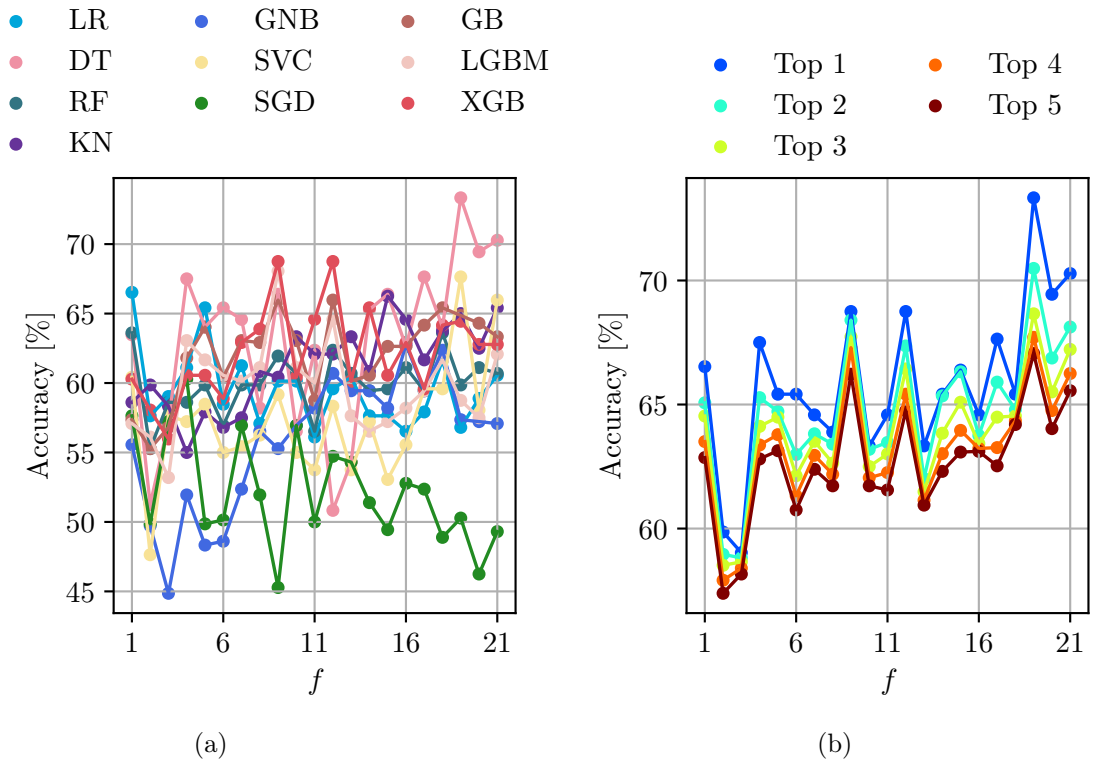


Figure 4.25: Data-augmentation results from UMLJ classical ML experiments: (a) all models and (b) mean of the top models at each f. The results are averaged across 10 CV folds. Training-data fraction ranges from 1 (original data only) to 21 (original data plus 20 times the original data in augmented instances). The augmented data were generated using PCGmix.

As shown in Figure 4.25a, the overall pattern is unclear due to varying responses from the models. However, when focusing on the top X models for each amount of added augmented data f and averaging their performance, the pattern becomes clearer, as illustrated in Figure 4.25b. Adding augmented data can potentially enhance the detection of decompensation episodes in CHF patients. However, the results fluctuate, making it challenging to conclude that additional data will consistently guarantee improvements, even though a general upward trend is noted.

In this analysis, the suboptimal parameter selection compared to the original classical ML approach (see Section 4.1.2) had to be used. Consequently, the no-augmentation results ($f = 1$ in Figure 4.25) were worse than the original, and the results with the augmented data ($f = 21$) did not surpass the original approach with optimal parameters and no augmentation. The best model achieved slightly above 70% accuracy, similar to the best performance in the original approach. Thus, while data augmentation can improve performance, selecting optimal parameters in the classical ML approach is equally effective.

It is also important to note that both classes in the UMCL dataset represent abnormal heart sounds corresponding to different phases of CHF. Therefore, the augmentation strategy, originally developed on a dataset with normal and abnormal classes, might not have transferred optimally to the UMCL dataset. Overall, our findings indicate that additional original data is required to improve the detection of decompensation episodes.

Currently, we are in the process of collecting data for a continuation of the study presented in this thesis. In this follow-up study, CHF patients record a variety of data modalities every two days, including ECG and PCG using the EkoDuo electronic stethoscope (see Figure 3.2b), PPG, weight, and a questionnaire about their overall well-being.

Chapter 5

Discussion and Conclusions

In this chapter, we present a comprehensive discussion of the two-part study outlined in this thesis: the detection of decompensation episodes of CHF, and the development of a heart-sound data-augmentation technique for heart-sound classification with limited data. First, our key findings are summarised. After that, the scientific contributions of our research are outlined. This is followed by the hypotheses analysis and discussion about the limitations. Finally, directions for future work are detailed.

To classify decompensation episodes in CHF using the UMCL dataset of heart-sound recordings from the 37 CHF patients, 10 classical ML methods were evaluated. Domain-specific predictor features were extracted from the four states of heart sounds. All models outperformed the classification performed independently by three cardiology experts, which averaged at 50%. Logistic regression proved to be the best model in terms of accuracy, reaching $71.9 \pm 14.7\%$. Power spectral density features, time domain features, and Mel-frequency cepstrum coefficients were found to be the most important predictors. Most of these features were extracted from diastole. From the medical perspective, this is reasonable, since the sounds produced in the diastole originate from the heart chambers refilling by blood. The heart of a CHF patient is more rigid than a healthy heart and will thus vibrate differently. Another finding is that several of the important features are related to heart rate. Again, this is relevant from the medical point of view, as patients in the decompensated phase have a faster pulse than those that are not decompensated. Our pipeline was also tested on the PhysioNet public dataset, which includes both normal and abnormal heart sounds, where it achieved results comparable to the NN approach discussed in the data augmentation section of our study (results at $f = 100\%$).

Our results are not directly comparable to those of related studies [195], [205], [206], [228]–[230] due to variations in datasets, subject demographics, and experimental methodologies (see Table 2.3). The higher classification accuracies reported in these studies may be influenced by their specific dataset compositions and design choices. For instance, Liu et al. [205], Gjoreski et al. [228], and Zheng et al. [195] focus on binary classification tasks between healthy individuals and those with CHF. This scenario presents a clearer distinction than the more complex classification task in our study, which classifies between decompensated and recompensated CHF phases. This latter task is inherently more challenging, as both phases involve CHF patients exhibiting similar abnormal characteristics. Additionally, potential dataset imbalances in studies such as [228] and [195] could artificially inflate classification accuracies. Although Gjoreski et al. [206] also address the distinction between decompensated and recompensated CHF phases, our study employs an expanded dataset and different experimental settings, further limiting direct comparability.

In view of early detection of decompensation episodes of CHF and thus preventing decompensation from occurring or progressing to hospitalization, our results are promising,

as they demonstrate that classical ML algorithms perform comparably to human experts based solely on heart sounds. The UMCL dataset captures acoustic anomalies characteristic of severe decompensation episodes, focusing on distinct differences in heart vibrations and rate between decompensated and recompensated phases during acute hospital admissions. However, since this dataset excludes milder decompensation phases, early-stage CHF, and different CHF subtypes, the model's efficiency is primarily relevant to detecting acute CHF decompensation in advanced cases. To adapt this system for early CHF deterioration detection and prevention of decompensation, additional patient data from daily or weekly self-recordings and self-assessments is essential.

Recently, various approaches to automatic detection of heart disease have been successfully implemented for numerous dataset modalities such as clinical features, images, and ECG [290]. Although the reported accuracies are very decent, some data are very difficult and/or expensive to obtain. Our plan for our system to support patients with CHF is to incorporate data that are easy to obtain and relatively inexpensive, such as clinical data, self-reported data, ECG data, daily activity data, and possibly others, which are currently in the process of being collected. In addition, our models could be integrated into a virtual coaching system [291] that tracks the patient's cardiac status and overall well-being and promotes medication use and/or physical activity to prevent deterioration of the condition.

Regarding XAI, our models currently offer only partial transparency. While some features, such as BPM and heart-sound state durations, are intuitive and interpretable, others, like MFCCs and PSD, are complex and less intuitive. Refining the feature engineering process to prioritize clinically interpretable characteristics, using only intuitive features as model inputs where possible would improve explainability. Additionally, presenting feature importance and the relative weight of each feature can help clinicians understand how the model makes decisions. Explainability could further be enhanced by showing examples of similar heart sounds from past patients in either decompensated or recompensated states, allowing experts to compare and contextualize the model's reasoning. Finally, confidence scoring and an error analysis log could inform clinicians about the model's certainty and instances prone to misclassification, ultimately enhancing the model's reliability as a decision-support tool in clinical practice.

The study on heart-sound data augmentation, conducted using the publicly available PhysioNet heart-sound database, indicates that our heart-sound data-augmentation methods, PCGmix(+), outperform the comparable state-of-the-art methods in situations with limited training data (168 recordings or fewer). The time-series-domain results show that at 10, 30, 56, 112, and 168 PCG recordings, PCGmix+ achieves the same accuracy as the no-augmentation approach when trained on the amount of data that is 1.35–1.45, 1.66–1.69, 1.55–1.69, 1.41–2.25, and 1.31–1.65 times bigger, respectively. In the spectrogram domain, the corresponding improvements over the no-augmentation baseline are 1.46, 1.34, 1.46, 1.08, and 1.01.

Our empirical analysis indicates that accuracy improvements plateau with the use of approximately 168 patient recordings, or around 4500 heartbeats. Beyond this threshold, further expansion of the dataset does not significantly enhance model performance. Consequently, the efficiency of data-augmentation methods, including PCGmix(+), diminishes beyond this amount of training data.

In our analysis, we found that most of the standard (baseline) augmentation methods did not yield notable improvements in heart-sound classification, with the exception of magnitude warping, which demonstrated intermediate performance. While many studies on data augmentation for heart-sound classification report enhancements from these augmentation techniques, these studies generally apply multiple augmentations simultaneously (see Table 2.4). This approach makes it challenging to isolate the impact of individual aug-

mentations [207], [209], [213], [222], [235]–[238], [240]. However, using combined augmentations, as seen in related studies, may strengthen augmentation effects. This is consistent with our findings for PCGmix+, the best-performing method in our study, which is also the only method combining two augmentations.

Potentially, PCGmix(+) outperform other methods due to their specialisation in the heart-sound-data domain. Several variations of PCGmix were tested, indicating that on average, data augmentation is beneficial in terms of accuracy when there is a limited number of heartbeats. However, the presented version that carefully maintains the durations of the heart states performed the best. Also, the comparable methods that ignore the specifics of heartbeats, in general, perform worse. For example, Mixup, which employs linear interpolation like PCGmix, but does not preserve the durations of the heart states, performs significantly worse. In addition, PCGmix+ also uses a time-warping transformation that mimics different recording body positions making the synthetic heartbeats better reflect the clinical diversity in practice.

Our training data included recordings with diverse characteristics, encompassing variations in recording equipment, signal quality, environments, body positions, patient types, and diagnostic methods, robustly testing our algorithm in various real-world scenarios. Experiments investigating a mapping function, which maps pairs of instances to generate new ones, show that imposing additional constraints to mix only instances with similar characteristics can potentially reduce the bias of generating unrealistic instances, however, this positive effect is outweighed by the negative impact of reduced diversity in the training data. The experiments, therefore, confirm the reliability of generating examples by mixing instances with different types of CVDs and/or different experimental setups.

In our feature-space analysis of generated instances, we examined the impact of out-of-manifold intrusion: the generation of instances in unrealistic regions of feature space or within the feature manifold of an opposite class. To mitigate this, a constraint was implemented in our mapping function that restricts mixing to nearby instances and is controlled by a parameter κ. Our results show that moderate values of κ can improve performance by reducing intrusion, while very low values reduce mixing diversity and degrade performance.

In addition, we explored the use of saliency information to refine our augmentation strategy, aiming to preserve critical signal regions during augmentation. Two saliency-guided approaches, MaxSum and MaxEnv, were tested, both demonstrating performance improvements over the original method, with MaxEnv yielding the highest accuracy gains. While these results indicate that saliency information can enhance model performance, it is important to note that this approach is computationally intensive and would benefit from further optimization of saliency map extraction techniques. From an XAI perspective, these findings are promising, as they suggest the model likely focuses on relevant regions of the heart sound in its decision-making. Providing saliency maps to clinicians could thus enhance model interpretability, allowing experts to better contextualize and understand the model's reasoning.

When tested on the UMCL dataset of CHF patients in decompensated and recompensated phases, our method demonstrated potential for improving performance through data augmentation. However, optimising parameters for data processing and feature engineering in a classical ML pipeline was similarly effective. The limited success of our method on this dataset may stem from the fact that it includes only abnormal heart sounds representing different phases of CHF, whereas our augmentation strategy was originally developed using a dataset with both normal and abnormal classes.

Customized models for specific CVDs are crucial in clinical settings, considering nuances in individual patient groups and CVDs. Integrating our method into computer-aided

diagnosis systems, the performance of the CVD-detection models in the cases where there is limited data could be maximized. This would enable earlier intervention and personalized treatments, which would ultimately improve patient's quality of life and reduce healthcare costs.

5.1 Scientific Contributions

The main scientific contributions of this work can be summarised as follows.

Contribution 1: A novel method for augmenting heart-sound data, particularly useful when trained on small datasets.

The PCGmix method utilizes information about the heartbeat states in its instance generation process, making it, to the best of our knowledge, the first heart-sound-specific data-augmentation technique. This method meticulously interpolates heartbeat signals while preserving the durations of heart states, a fundamental diagnostic characteristic crucial for accurate CVD detection. PCGmix can be applied in both time-series- and spectrogram-data domains and functions as a NN training regularization technique, effectively addressing the issue of limited data availability.

PCGmix leverages the theoretical framework provided by studies on Mixup, which suggest that enforcing a linear combination of inputs simplifies decision boundaries and can reduce overfitting by discouraging overly complex models. This technique enhances the learning of rare features by blending them with more common ones, a significant advantage in heart-sound data where pathological features may be infrequent. By interpolating between multiple instances, PCGmix ensures that rare but important features are robustly represented in the training data.

Additionally, its upgraded version, PCGmix+, modifies the amplitudes across different parts of a PCG signal, simulating the effect of different recording body positions. This is consistent with clinical observations that certain abnormalities are best detected from specific recording positions. PCGmix+ therefore mimics these amplitude variations, enhancing the PCGmix-generated instances to better reflect the clinical diversity observed in practice.

Contribution 2: Applying and thoroughly validating the data-augmentation method on a variety of subsets of the PhysioNet/CinC Challenge 2016 heart-sound database for heart-sound classification using NNs.

The proposed augmentation method is applicable to any CVD problems involving PCG and/or ECG data. In this thesis, the method was rigorously tested using the PhysioNet/CinC Challenge 2016 heart-sound database, the largest and most diverse publicly available dataset for heart sounds.

The experiments utilized various subsets of this database, ranging from 333 experiments with random samplings of 10 PCGs (approximately 268 heartbeats) to 5 experiments with 554 PCGs (14755 heartbeats). The method was evaluated across both time-series and spectrogram data modalities.

To assess its effectiveness, the proposed method was compared against several baseline approaches, including no augmentation and standard data-augmentation techniques. The comparisons included modality-agnostic methods such as Mixup and Manifold Mixup, as well as time-series-domain techniques like noise injection, magnitude warping, respiratory scaling, and time masking. In the spectrogram domain, the comparison encompassed time masking, frequency masking, and Cutout.

Contribution 3: Demonstrating the performance that can be achieved in detecting decompensation in CHF patients using heart sounds alone.

Using the UMLC dataset, which includes heart-sound recordings from 37 CHF patients

in both decompensated and recompensated phases, we evaluated the accuracy of classifying these two phases. This study is, to the best of our knowledge, the first to specifically address the classification of decompensated versus recompensated CHF episodes using heart sounds alone, thereby indicating the performance limits achievable through heart-sound analysis.

Additional contributions of our study include the following.

Contribution 4: Extending the existing labelled dataset of heart sounds from CHF patients in recompensated and decompensated phases.

Within the scope of this study, the UMCL dataset has been extended by additional 16 CHF patients recorded in both decompensated and recompensaed phases. Overall, our UMCL dataset now consists of 75 PCGs, 37 and 38 for compensated and decompensated phases, respectively, and adds up to 29 min 15 s in length. Raw data is available upon request.

Contribution 5: A novel metric specialised for assessment of an augmentation method's performance in the context of experiments with datasets of different sizes.

The ADSI (Apparent Data Size Increase) defined in Section 3.5 represents a novel metric used for measuring the performance of augmentation methods in the context of experiments using data subsets of varying sizes. It indicates how much larger the original dataset would need to be for the no-augmentation approach to reach the same level of performance as the augmentation method.

5.2 Hypothesis Analysis

In this thesis, we investigated the following two hypotheses.

Hypothesis 1: Deep neural networks employing data-augmentation techniques on smaller heart-sound datasets perform comparably to those trained on larger datasets.

This hypothesis was **experimentally confirmed** through a series of experiments comparing the performance of models with and without data augmentation. Specifically, we evaluated the PCGmix(+) augmentation methods against the no-augmentation training on data subsets of varying sizes from the PhysioNet/CinC Challenge 2016 heart-sound database.

In the time-series domain, PCGmix+ trained on 10, 30, 56, 112, and 168 PCGs achieved the same performance as the no-augmentation approach when trained on the amount of data that is 1.35–1.45, 1.66-1.69, 1.55–1.69, 1.41–2.25, and 1.31–1.65 times bigger, respectively. In the spectrogram domain, PCGmix trained on 10, 30, 56, 112, and 168 PCGs achieved the same performance as the no-augmentation approach when trained on the amount of data that is 1.46, 1.34, 1.46, 1.08, and 1.01 times bigger, respectively.

These results confirm that data-augmentation techniques can effectively enhance performance of NNs in heart-sound classification tasks when working with limited datasets and provide performance similar to those trained on larger datasets.

Hypothesis 2: Machine-learning methods perform comparably to cardiology experts in detecting decompensation episodes in patients with chronic heart failure based solely on heart-sound analysis.

This hypothesis was **experimentally confirmed** by comparing the performance of a classical ML heart-sound analysis pipeline with the classifications made by three cardiology experts using the UMCL dataset.

In the experimental setup, each cardiologist independently analysed a representative subset of the UMCL dataset, classifying the heart sounds as either decompensated or recompensated. Notably, the cardiologists had no access to additional clinical data about

the patients, which contrasts with typical clinical practice where comprehensive patient information is available. The cardiologists' classification accuracies were 58%, 66.6%, and 25%, resulting in an average accuracy of 50%. Because the subset was small (12 PCGs), this is only an estimate of the experts' true performance.

In comparison, the ML models used in our classical pipeline, which employed domain-specific features manually extracted from heart-sound recordings, achieved classification accuracies ranging from 56.0% to 72.1%.

These results demonstrate the performance of ML methods that can be achieved in classifying decompensated and recompensated episodes of CHF based solely on heart sounds. Furthermore, they indicate that the performance of these models can match or potentially exceed that of cardiology experts, thereby validating the effectiveness of the proposed approach.

5.3 Limitations

The part of the study that deals with the detection of decompensation episodes of CHF has the following limitations. First, the UMCL dataset excludes milder decompensation phases and different CHF subtypes, so the model's efficiency is primarily relevant to detecting acute CHF decompensation in advanced cases. Second, the PCG is recorded when an individual is admitted to/discharged from the hospital. Ideally, the PCGs would be recorded on a daily/weekly basis with the intention of capturing the deterioration of the condition. Third, since we are using a dataset collected by ourselves, we cannot directly compare the accuracy of our method with related work, but only by testing it against a public dataset and the human experts. Fourth, the performance of our models is not directly comparable to that of the cardiology experts since the experts were assessed on a small subset of the dataset. Although this subset is representative, its limited size complicates definitive conclusions about the experts' true performance. Fifth, while the top-performing ML models show promising results, the inputs to these models include both explainable features and more complex features, such as MFCCs and PSD, that are not intuitive to humans. As a result, the models do not necessarily enhance the experts' decision-making process while auscultating the heart and function primarily as standalone decision-support tools.

The part of the study that deals with the development of heart-sound data-augmentation method has the following limitations. Firstly, the limited demographic information in our training data, specifically age and gender, constrains our ability to assess algorithm robustness across diverse patient demographics. Additionally, due to computational constraints, parameter fine-tuning, which could optimise model performance further, was not performed. While our experiments demonstrated that manually preset parameters were reasonably close to optimal for all NNs, fine-tuning approaches such as grid search on training data could enhance parameter optimisation. Furthermore, while the mapping functions mixing only instances diagnosed with the same CVD and/or under the same experimental setup did not yield improvements, we believe that finding a great mapping function holds the potential for enhancing results. Lastly, the two approaches for potential improvement of our method, the restriction of the mapping function to only mix nearby instances to prevent out-of-manifold intrusion instance generation, and the saliency-guided instance interpolation, were only tested in limited experimental settings. To robustly evaluate their performance and confirm their superiority over the original approach, more extensive and diverse experimentation using different NN models and data subset sizes is needed.

5.4 Future Work

In this thesis, we demonstrated that while ML algorithms perform similarly to or potentially better than cardiology experts, using heart-sound data alone is insufficient for the early detection of CHF decompensation. Therefore, we are currently collecting additional data to address this limitation through a telemedicine approach. In this follow-up study, CHF patients (so far, we have been able to include three patients) are recording various data modalities every two to three days, including ECG, PCG, PPG, weight, and a questionnaire about their overall well-being. This approach not only increases the amount of relevant data for decompensation detection but also aims to capture the point of deterioration more accurately, as data is collected every few days, rather than only upon hospitalization or release from the hospital, as in the current UMCL dataset.

The data currently being collected is automatically uploaded to the cloud and made available to clinicians for monitoring. The end goal of the follow-up study is to develop tools for the automatic analysis of data from various modalities, which would be deployed on the cloud. Once uploaded by the patient, the system would automatically analyse the data. If the system detects the deterioration of CHF, it would alert the patient or clinicians to make timely and appropriate changes to the medical therapy, thus preventing severe CHF decompensation or reducing its severity to avoid hospitalization. This approach has the potential to significantly improve patient outcomes and reduce the workload of medical staff.

The future goal for our data-augmentation technique is to extend it to various domains and problems. While we have extensively tested our method on a public heart-sound dataset, we aim to apply it to ECG data as well. ECG data, like heart sounds, follows cyclical patterns, making it a natural extension for our technique, provided that information about heart state locations is available. Additionally, PPG data is another possible application area, though identifying distinct heart-cycle states within PPG waveforms may be less apparent.

Using interpolation between two original instances to generate a new instance has been one of the most powerful data-augmentation techniques in the recent years. Our augmentation method enhances this approach by incorporating domain knowledge to specialise the interpolation process, thereby generating more realistic instances. This refinement has already demonstrated significant improvements in model performance within the heart-sound domain. Our goal is to extend this approach to other areas beyond CVDs, including both medical and non-medical fields. By leveraging domain-specific insights to tailor the interpolation process, our method could enhance model performance across a variety of datasets, making it a versatile tool for addressing a broad range of problems.

Appendix A

Reproducibility Details

A.1 PhysioNet Dataset

The PhysioNet/CinC Challenge 2016 database can be accessed at `https://archive.physionet.org/pn3/challenge/2016/`. Annotations and heartbeat segmentation (using the Springer's algorithm and hand corrected) can be accessed at `https://physionet.org/content/challenge-2016/1.0.0/#files-panel` (accessed August 2024).

A.1.1 Test Data

We used the recordings marked as the "validation" set as our test data. A few of the recordings from the "validation" set that do not include the corresponding segmentation files and are thus excluded in our study: e00001, e00032, e00039, e00044.

A.1.2 Train Data

The train set in experiments is a subset of the "train set" used in the Challenge. Those recordings include: a0059, a0101, a0103, a0104, a0110, a0115, a0120, a0122, a0123, a0126, a0130, a0132, a0133, a0134, a0140, a0141, a0142, a0143, a0148, a0149, a0152, a0153, a0154, a0155, a0158, a0160, a0161, a0165, a0166, a0169, a0170, a0171, a0172, a0178, a0179, a0180, a0181, a0183, a0184, a0185, a0187, a0188, a0189, a0193, a0195, a0196, a0197, a0199, a0204, a0208, a0210, a0211, a0212, a0213, a0214, a0221, a0224, a0227, a0228, a0229, a0231, a0235, a0236, a0238, a0240, a0241, a0242, a0243, a0245, a0246, a0248, a0250, a0252, a0253, a0254, a0257, a0261, a0264, a0266, a0267, a0268, a0269, a0270, a0271, a0274, a0276, a0278, a0283, a0285, a0287, a0288, a0289, a0290, a0291, a0293, a0294, a0297, a0298, a0299, a0301, a0302, a0304, a0306, a0309, a0310, a0311, a0312, a0313, a0320, a0323, a0325, a0326, a0328, a0329, a0331, a0332, a0334, a0335, a0336, a0337, a0339, a0340, a0342, a0346, a0347, a0348, a0351, a0352, a0353, a0355, a0358, a0359, a0361, a0366, a0368, a0371, a0374, a0381, a0384, a0385, a0386, a0389, a0391, a0393, a0396, a0401, a0404, a0405, a0406, a0407, a0408, a0409, b0110, b0112, b0129, b0132, b0134, b0142, b0143, b0156, b0157, b0161, b0166, b0177, b0189, b0192, b0202, b0246, b0250, b0251, b0257, b0261, b0262, b0263, b0265, b0267, b0268, b0269, b0271, b0273, b0279, b0280, b0282, b0292, b0295, b0306, b0307, b0318, b0319, b0327, b0328, b0334, b0335, b0339, b0341, b0344, b0347, b0354, b0369, b0373, b0383, b0385, b0390, b0392, b0395, b0397, b0406, b0408, b0422, b0428, b0434, b0435, b0436, b0437, b0446, b0449, b0450, b0454, b0467, b0468, b0471, b0474, b0477, b0478, b0484, b0490, c0010, c0011, c0015, c0016, c0019, c0022, c0023, c0030, d0010, d0012, d0014, d0015, d0016, d0017, d0018, d0019, d0020, d0021, d0022, d0023, d0024, d0025, d0027, d0028, d0029, d0030, d0031, d0032, d0033, d0034, d0035, d0036, d0037, d0038, d0039, d0040, d0041, d0042, d0043, d0045, d0046, d0047, d0048, d0049, d0050, d0051, d0052, d0053,

d0054, d0055, e00074, e00254, e00277, e00309, e00347, e00354, e00365, e00387, e00419, e00425, e00436, e00453, e00480, e00493, e00509, e00512, e00531, e00552, e00559, e00566, e00568, e00569, e00580, e00600, e00613, e00645, e00652, e00658, e00686, e00693, e00696, e00699, e00703, e00710, e00725, e00726, e00739, e00744, e00765, e00766, e00792, e00799, e00808, e00811, e00818, e00824, e00841, e00847, e00871, e00872, e00880, e00897, e00908, e00914, e00917, e00925, e00926, e00929, e00936, e00967, e00974, e00980, e00998, e01013, e01016, e01055, e01062, e01072, e01073, e01075, e01084, e01097, e01105, e01115, e01148, e01160, e01177, e01198, e01205, e01215, e01225, e01226, e01245, e01246, e01247, e01252, e01253, e01256, e01263, e01272, e01276, e01283, e01284, e01289, e01291, e01295, e01299, e01308, e01324, e01336, e01341, e01351, e01358, e01367, e01369, e01374, e01375, e01376, e01382, e01383, e01392, e01399, e01401, e01416, e01421, e01433, e01457, e01461, e01474, e01479, e01487, e01489, e01493, e01494, e01511, e01525, e01531, e01537, e01551, e01559, e01572, e01581, e01601, e01602, e01605, e01618, e01623, e01625, e01638, e01651, e01661, e01663, e01665, e01666, e01668, e01686, e01688, e01689, e01697, e01698, e01711, e01719, e01728, e01734, e01748, e01753, e01756, e01766, e01767, e01787, e01791, e01800, e01808, e01812, e01824, e01825, e01832, e01846, e01848, e01851, e01855, e01856, e01857, e01867, e01869, e01873, e01881, e01917, e01922, e01924, e01929, e01938, e01953, e01959, e01962, e01967, e01971, e01978, e02001, e02010, e02015, e02017, e02030, e02032, e02044, e02045, e02055, e02058, e02059, e02065, e02074, e02085, e02088, e02092, e02097, e02101, e02102, e02108, e02111, e02117, e02121, e02126, e02133, e02135, e02137, e02140, f0003, f0007, f0008, f0011, f0012, f0015, f0016, f0017, f0018, f0020, f0022, f0023, f0027, f0029, f0031, f0035, f0036, f0041, f0042, f0045, f0047, f0048, f0050, f0052, f0054, f0056, f0060, f0063, f0064, f0065, f0066, f0067, f0068, f0069, f0070, f0071, f0072, f0075, f0076, f0078, f0079, f0080, f0081, f0082, f0083, f0085, f0090, f0091, f0092, f0093, f0096, f0097, f0098, f0099, f0100, f0101, f0103, f0106, f0109, f0112, f0113, and f0114.

A.1.3 Data Partitioning

Using the alphabetically ordered list of train recordings `train_wav` (see Appendix A.1.2) and the the corresponding labels `train_label` (0 for normal, 1 for abnormal), the dataset partition for fraction (f) `n_fraction` and RNG seed `seed_data` can be reproduced using the following Python 3.8 code snippet.

```
import numpy as np # version 1.23.5
import random

dataset_map = {"a":0, "b":1, "c":2, "d":3, "e":4, "f":5}
wavs = [[] for i in range(12)]
wavs_flat = []
for wav, label in zip(train_wav, train_label):
        if wav not in wavs_flat:
                dataset_letter = wav[0]
                idx = dataset_map[dataset_letter] + 6*label
                wavs[idx].append(wav)
                wavs_flat.append(wav)
wavs_nfrac_flat_0 = [wav for sublist in wavs[:6] for wav in sublist]
wavs_nfrac_flat_1 = [wav for sublist in wavs[6:] for wav in sublist]
wavs_nfrac_flat_0 = sorted(wavs_nfrac_flat_0)
wavs_nfrac_flat_1 = sorted(wavs_nfrac_flat_1)
random.Random(seed_data).shuffle(wavs_nfrac_flat_0)
random.Random(seed_data).shuffle(wavs_nfrac_flat_1)
```

```
n_frac_wavs_label = int(np.ceil(n_fraction*len(list(set(train_wav)))/2))
wavs_nfrac_flat_0 = wavs_nfrac_flat_0[:n_frac_wavs_label]
wavs_nfrac_flat_1 = wavs_nfrac_flat_1[:n_frac_wavs_label]
wavs_nfrac_flat = wavs_nfrac_flat_0 + wavs_nfrac_flat_1
wavs_nfrac_flat = np.sort(wavs_nfrac_flat)
indices = [i for i, wav in enumerate(train_wav) if wav in wavs_nfrac_flat]
train_data = train_data[indices]
train_label = train_label[indices]
```

The values of `seed_data`s used in our experiments are:

- 1.5%: `n_fraction = 0.015` and `seed_datas = np.arange(1001001, 1001334, 1)` (333 experiments);
- 5.2%: `n_fraction = 0.052` and `seed_datas = np.arange(1005001, 1005101, 1)` (100 experiments);
- 10%: `n_fraction = 0.1` and `seed_datas = np.arange(1010001, 1010051, 1)` (50 experiments);
- 20%: `n_fraction = 0.2` and `seed_datas = np.arange(1020001, 1020026, 1)` (25 experiments);
- 30%: `n_fraction = 0.3` and `seed_datas = np.arange(1030001, 1030017, 1)` (16 experiments);
- 40%: `n_fraction = 0.4` and `seed_datas = np.arange(1040001, 1040013, 1)` (12 experiments);
- 60%: `n_fraction = 0.6` and `seed_datas = np.arange(1060001, 1060009, 1)` (8 experiments);
- 80%: `n_fraction = 0.8` and `seed_datas = np.arange(1080001, 1080007, 1)` (6 experiments);

For the spectrogram experiments, where fewer experiments were conducted, only the first subset of `seed_data` were taken used (e.g., for $f = 1.5\%$, the 200 subsets were selected with `seed_datas = np.arange(1001001, 1001201, 1)`).

A.2 Parameters of the Classical ML Models

The default values of the classical ML models' parameters, along with the values used in the grid-search fine-tuning experiments are given in Table A.1.

Table A.1: Classical ML models' parameters: default values and grid-search fine-tuning values. Definitions of the parameters can be found in Scikit and lightgbm libraries' documentations [260], [261]. Function linspace(*start*, *end*, *num*) creates *num* evenly spaced values in the interval [*start*, *end*], whereas logspace(*start*, *end*, *num*) creates *num* evenly spaced values on a log scale in the interval [10^{start}, 10^{end}].

Model	Parameter	Default value	Grid-search values
DT	*criterion*	`"gini"`	`["gini", "entropy"]`
	splitter	`"best"`	`["best", "random"]`
	min_samples_split	`2`	`linspace(2, 4, 3)`
	max_features	`None`	`["auto", "sqrt", "log2"]`
RF	*n_estimators*	`100`	`linspace(80, 120, 3)`
	criterion	`"gini"`	`["gini"]`
	min_samples_split	`2`	`[2]`
	max_features	`"sqrt"`	`["sqrt"]`
GB	*loss*	`"deviance"`	`["deviance", "exponential"]`
	learning_rate	`0.1`	`[0.01, 0.025, 0.05, 0.075, 0.1]`
	n_estimators	`100`	`linspace(80, 120, 3)`
	min_samples_split	`2`	`[2`
	max_features	`None`	`["auto", "sqrt", "log2"]`
XGB	*learning_rate*	`0.1`	`linspace(0.1, 0.2, 3)`
	n_estimators	`100`	`linspace(80, 120, 3)`
	max_depth	`3`	`linspace(1, 10, 10)`
LGBM	*learning_rate*	`0.1`	`[0.1, 0.15]`
	n_estimators	`100`	`linspace(60, 140, 3)`
SVM	*C*	`1`	`linspace(0.5, 1.5, 11)`
	gamma	`"scale"`	`["auto"]`
kNN	*n_neighbors*	`5`	`linspace(1, 9, 5)`
	weights	`"uniform"`	`["uniform", "distance"]`
	metric	`"minkowski"`	`["euclidean", "manhattan", "minkowski"]`
GNB	*var_smoothing*	`10^-9`	`logspace(0,-9, 100)`
LR	*solver*	`"lbfgs"`	`["lbfgs"]`
	penalty	`"l2"`	`["l1", "l2", "elasticnet"]`
	C	`1`	`linspace(0.1, 1.0, 11)`
	max_iter	`100`	`linspace(80, 120, 3)`
SGD	*loss*	`"hinge"`	`["log"]`
	penalty	`"l2"`	`["l2", "l1", "elasticnet"]`
	alpha	`10^-4`	`logspace(-1,-5, 50)`

A.3 Heart Sound Data Augmentation

A.3.1 Parameters of the Methods

The explanation of the parameters can be found in Sections 3.4.2 and 3.4.3. Table A.2 provides the values used in our experiments.

Table A.2: Data augmentation methods' parameters used in our experiments.

Method	Domain	Parameter	Value
NoiseInject	Time-series	$\mathrm{SNR}_{\mathrm{dB,min}}$	25 dB
		$\mathrm{SNR}_{\mathrm{dB,max}}$	40 dB
TimeMask	Both	$p_{\mathrm{TM,\ max}}$	20%
FreqMask	Spectrogram	$p_{\mathrm{FM,\ max}}$	20%
Cutout	Spectrogram	$p_{\mathrm{CO,TM,\ max}}$	25%
		$p_{\mathrm{CO,FM,\ max}}$	25%
MagWarp	Time-series	k_W	4
		σ_W	0.2
RespScale	Time-series	$\omega_{S,\mathrm{min}}$	12/60 s
		$\omega_{S,\mathrm{max}}$	18/60 s
Mixup	Both	λ	U(0, 1)
ManifoldMixup	Both	λ	U(0, 1)
PCGmix	Both	λ	U(0, 1)
PCGmix+	Time-series	λ	U(0, 1)
		k_W	4
		σ_W	0.2

A.3.2 Augmentation Probabilities

The augmentation probabilities used in our experiments are given in Table A.3

Table A.3: Augmentation probabilities of the methods used in our experiments for all training-data fractions.

	f								
Method	1.5%	5.2%	10%	20%	30%	40%	60%	80%	100%
NoiseInject	1.0	1.0	1.0	1.0	0.8	0.6	0.4	0.2	0.2
TimeMask	1.0	1.0	1.0	0.8	0.6	0.6	0.4	0.2	0.2
MagWarp	1.0	1.0	1.0	1.0	1.0	1.0	0.8	0.4	0.4
RespScale	1.0	1.0	1.0	0.8	0.6	0.6	0.2	0.2	0.2
Mixup	1.0	1.0	1.0	0.8	0.6	0.4	0.2	0.2	0.2
ManifoldMixup	1.0	1.0	1.0	1.0	0.6	0.6	0.2	0.2	0.2
PCGmix	1.0	1.0	1.0	0.8	0.6	0.6	0.4	0.2	0.2

References

[1] S. S. Martin, A. W. Aday, Z. I. Almarzooq, *et al.*, "2024 heart disease and stroke statistics: A report of US and global data from the American Heart Association," *Circulation*, vol. 149, no. 8, Feb. 2024. DOI: 10.1161/cir.0000000000001209.

[2] M. Lindstrom, N. DeCleene, H. Dorsey, *et al.*, "Global burden of cardiovascular diseases and risks collaboration, 1990-2021," *Journal of the American College of Cardiology*, vol. 80, no. 25, pp. 2372–2425, Dec. 2022. DOI: 10.1016/j.jacc.2022.11.001.

[3] World Health Organization. "Cardiovascular diseases (CVDs)." (2021), [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/cardiovascular-diseases-(cvds) (visited on 04/02/2024).

[4] Noncommunicable Diseases Alliance. "Cardiovascular diseases (CVD)." (2021), [Online]. Available: https://ncdalliance.org/why-ncds/ncds/cardiovascular-diseases (visited on 04/02/2024).

[5] T. A. McDonagh, M. Metra, M. Adamo, *et al.*, "2021 ESC Guidelines for the diagnosis and treatment of acute and chronic heart failure," *European Heart Journal*, vol. 42, no. 36, pp. 3599–3726, Aug. 2021. DOI: 10.1093/eurheartj/ehab368.

[6] A. A. Bhurane, M. Sharma, R. San-Tan, and U. R. Acharya, "An efficient detection of congestive heart failure using frequency localized filter banks for the diagnosis with ECG signals," *Cognitive Systems Research*, vol. 55, pp. 82–94, Jun. 2019. DOI: 10.1016/j.cogsys.2018.12.017.

[7] C. Liu, D. Springer, Q. Li, *et al.*, "An open access database for the evaluation of heart sound algorithms," *Physiological Measurement*, vol. 37, no. 12, pp. 2181–2213, Nov. 2016. DOI: 10.1088/0967-3334/37/12/2181.

[8] G. Images. "Istock." (2024), [Online]. Available: https://www.istockphoto.com (visited on 05/28/2024).

[9] P. Libby and P. Theroux, "Pathophysiology of coronary artery disease," *Circulation*, vol. 111, no. 25, pp. 3481–3488, Jun. 2005. DOI: 10.1161/circulationaha.105.537878.

[10] S. Nathaniel, "Aortic stenosis: An update," *World Journal of Cardiology*, vol. 2, no. 6, p. 135, 2010. DOI: 10.4330/wjc.v2.i6.135.

[11] M. T. El Hussein, D. Job, and J. Hakkola, "Aortic regurgitation: Review of current management," *Dimensions of Critical Care Nursing*, vol. 43, no. 2, pp. 80–86, Mar. 2024. DOI: 10.1097/dcc.0000000000000628.

[12] E. M. Isselbacher, O. Preventza, J. Hamilton Black, *et al.*, "2022 ACC/AHA guideline for the diagnosis and management of aortic disease: A report of the american heart association/american college of cardiology joint committee on clinical practice guidelines," *Circulation*, vol. 146, no. 24, Dec. 2022. DOI: 10.1161/cir.0000000000001106.

[13] Y. Chandrashekhar, S. Westaby, and J. Narula, "Mitral stenosis," *The Lancet*, vol. 374, no. 9697, pp. 1271–1283, Oct. 2009. DOI: 10.1016/s0140-6736(09)60994-6.

[14] A. D. Maslow and A. Poppas, "Primary mitral valve regurgitation: Update and review," *Global Cardiology Science and Practice*, vol. 2017, no. 1, May 2017. DOI: 10.21542/gcsp.2017.3.

[15] Y. Deng, J. Liu, S. Wu, *et al.*, "Arrhythmic mitral valve prolapse: A comprehensive review," *Diagnostics*, vol. 13, no. 18, p. 2868, Sep. 2023. DOI: 10.3390/diagnostics13182868.

[16] R. E. Klabunde, *Cardiovascular Physiology Concepts*, 2nd ed. Philadelphia, PA: Lippincott Williams and Wilkins, Sep. 2011.

[17] T. A. McDonagh, M. Metra, M. Adamo, *et al.*, "2023 Focused Update of the 2021 ESC Guidelines for the diagnosis and treatment of acute and chronic heart failure," *European Heart Journal*, vol. 44, no. 37, pp. 3627–3639, Aug. 2023. DOI: 10.1093/eurheartj/ehad195.

[18] B. Bozkurt, A. J. Coats, H. Tsutsui, *et al.*, "Universal definition and classification of heart failure: a report of the Heart Failure Society of America, Heart Failure Association of the European Society of Cardiology, Japanese Heart Failure Society and Writing Committee of the Universal Definition of Heart Failure: Endorsed by the Canadian Heart Failure Society, Heart Failure Association of India, Cardiac Society of Australia and New Zealand, and Chinese Heart Failure Association," *European Journal of Heart Failure*, vol. 23, no. 3, pp. 352–380, Mar. 2021. DOI: 10.1002/ejhf.2115.

[19] S. L. James, D. Abate, K. H. Abate, *et al.*, "Global, regional, and national incidence, prevalence, and years lived with disability for 354 diseases and injuries for 195 countries and territories, 1990–2017: a systematic analysis for the Global Burden of Disease Study 2017," *The Lancet*, vol. 392, no. 10159, pp. 1789–1858, Nov. 2018. DOI: 10.1016/s0140-6736(18)32279-7.

[20] A. Groenewegen, F. H. Rutten, A. Mosterd, and A. W. Hoes, "Epidemiology of heart failure," *European Journal of Heart Failure*, vol. 22, no. 8, pp. 1342–1356, Jun. 2020. DOI: 10.1002/ejhf.1858.

[21] W. Lesyuk, C. Kriza, and P. Kolominsky-Rabas, "Cost-of-illness studies in heart failure: A systematic review 2004–2016," *BMC Cardiovascular Disorders*, vol. 18, no. 1, May 2018. DOI: 10.1186/s12872-018-0815-3.

[22] I. Ford, M. Robertson, M. Komajda, *et al.*, "Top ten risk factors for morbidity and mortality in patients with chronic systolic heart failure and elevated heart rate: The SHIFT risk model," *International Journal of Cardiology*, vol. 184, pp. 163–169, Apr. 2015. DOI: 10.1016/j.ijcard.2015.02.001.

[23] E. E. Tripoliti, T. G. Papadopoulos, G. S. Karanasiou, K. K. Naka, and D. I. Fotiadis, "Heart failure: Diagnosis, severity estimation and prediction of adverse events through machine learning techniques," *Computational and Structural Biotechnology Journal*, vol. 15, pp. 26–47, 2017. DOI: 10.1016/j.csbj.2016.11.001.

[24] T. Sadad, S. A. C. Bukhari, A. Munir, A. Ghani, A. M. El-Sherbeeny, and H. T. Rauf, "Detection of cardiovascular disease based on PPG signals using machine learning with cloud computing," *Computational Intelligence and Neuroscience*, vol. 2022, D. Zhang, Ed., pp. 1–11, Aug. 2022. DOI: 10.1155/2022/1672677.

[25] A. S. Al Fahoum, A. O. Abu Al-Haija, and H. A. Alshraideh, "Identification of coronary artery diseases using photoplethysmography signals and practical feature selection process," *Bioengineering*, vol. 10, no. 2, p. 249, Feb. 2023. DOI: 10.3390/bioengineering10020249.

[26] C. Heinze, D. Sommer, U. Trutschel, and M. Golz, "Discrimination and relevance determination of heart rate variability features for the identification of congestive heart failure," in *2014 8th Conference of the European Study Group on Cardiovascular Oscillations (ESGCO)*, IEEE, May 2014. DOI: 10.1109/esgco.2014.6847598.

[27] S. G. Aydin, T. Kaya, and H. Guler, "Heart rate variability (HRV) based feature extraction for congestive heart failure," *International Journal of Computer and Electrical Engineering*, vol. 8, no. 4, pp. 272–279, 2016. DOI: 10.17706/ijcee.2016.8.4.272-279.

[28] G. Yang, Y. Ren, Q. Pan, *et al.*, "A heart failure diagnosis model based on support vector machine," in *2010 3rd International Conference on Biomedical Engineering and Informatics*, IEEE, Oct. 2010. DOI: 10.1109/bmei.2010.5639619.

[29] M. Kostkiewicz, "Myocardial perfusion imaging in coronary artery disease," *Cor et Vasa*, vol. 57, no. 6, e446–e452, Dec. 2015. DOI: 10.1016/j.crvasa.2015.09.010.

[30] L. Capotosto, F. Massoni, S. De Sio, S. Ricci, and A. Vitarelli, "Early diagnosis of cardiovascular diseases in workers: Role of standard and advanced echocardiography," *BioMed Research International*, vol. 2018, pp. 1–15, 2018. DOI: 10.1155/2018/7354691.

[31] A. Ammari, R. Mahmoudi, B. Hmida, R. Saouli, and M. H. Bedoui, "A review of approaches investigated for right ventricular segmentation using short-axis cardiac mri," *IET Image Processing*, vol. 15, no. 9, pp. 1845–1868, Mar. 2021. DOI: 10.1049/ipr2.12165.

[32] M.-L. Tan, Y. Su, C.-W. Lim, S. K. Selvaraj, L. Zhong, and R.-S. Tan, "A geometrical approach for automatic shape restoration of the left ventricle," *PLoS ONE*, vol. 8, no. 7, A. Talkachova, Ed., e68615, Jul. 2013. DOI: 10.1371/journal.pone.0068615.

[33] C. W. Lim, Y. Su, S. Y. Yeo, *et al.*, "Automatic 4D reconstruction of patient-specific cardiac mesh with 1-to-1 vertex correspondence from segmented contours lines," *PLoS ONE*, vol. 9, no. 4, A. Talkachova, Ed., e93747, Apr. 2014. DOI: 10.1371/journal.pone.0093747.

[34] W. Caesarendra, T. Triwiyanto, H. G. Ariswati, A. R. Masnulula, and N. M. Firdaus, "Development of cardiac monitor through carotid pulse, phonocardiography and electrocardiography," *Jurnal Teknokes*, vol. 15, no. 2, pp. 81–87, Jun. 2022, ISSN: 2407-8964. DOI: 10.35882/jteknokes.v15i2.132.

[35] P. S. Molcer, I. Kecskés, V. Delić, E. Domijan, and M. Domijan, "Examination of formant frequencies for further classification of heart murmurs," in *IEEE 8th International Symposium on Intelligent Systems and Informatics*, 2010, pp. 575–578. DOI: 10.1109/SISY.2010.5647147.

[36] A. Leatham, *Auscultation of the Heart and Phonocardiography*. Churchill Livingstone, 1975, ISBN: 9780443011559.

[37] E. L. Simon, P. J. Lecat, N. A. Haller, *et al.*, "Improved auscultation skills in paramedic students using a modified stethoscope," *The Journal of Emergency Medicine*, vol. 43, no. 6, pp. 1091–1097, Dec. 2012. DOI: 10.1016/j.jemermed.2012.01.048.

[38] C. Pinto, D. Pereira, J. Ferreira-Coimbra, J. Portugues, V. Gama, and M. Coimbra, "A comparative study of electronic stethoscopes for cardiac auscultation," in *2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, IEEE, Jul. 2017. DOI: 10.1109/embc.2017.8037392.

[39] D. Springer, L. Tarassenko, and G. Clifford, "Logistic regression-HSMM-based heart sound segmentation," *IEEE Transactions on Biomedical Engineering*, p. 1, 2015. DOI: 10.1109/tbme.2015.2475278.

[40] E. Kaniusas, "Acoustical signals of biomechanical systems," in *Biomechanical Systems Technology*. World Scientific Publishing Company, Nov. 2007, pp. 1–44. DOI: 10.1142/9789812771391_0001. [Online]. Available: http://dx.doi.org/10.1142/9789812771391_0001.

[41] S. Roccabianca, C. Figueroa, G. Tellides, and J. Humphrey, "Quantification of regional differences in aortic stiffness in the aging human," *Journal of the Mechanical Behavior of Biomedical Materials*, vol. 29, pp. 618–634, Jan. 2014. DOI: 10.1016/j.jmbbm.2013.01.026.

[42] A. K. Dwivedi, S. A. Imtiaz, and E. Rodriguez-Villegas, "Algorithms for automatic analysis and classification of heart sounds–a systematic review," *IEEE Access*, vol. 7, pp. 8316–8345, 2019. DOI: 10.1109/access.2018.2889437.

[43] J. Oliveira, F. Renna, P. D. Costa, *et al.*, "The CirCor DigiScope dataset: From murmur detection to murmur classification," *IEEE Journal of Biomedical and Health Informatics*, vol. 26, no. 6, pp. 2524–2535, Jun. 2022. DOI: 10.1109/jbhi.2021.3137048.

[44] B. S. Emmanuel, "A review of signal processing techniques for heart sound analysis in clinical diagnosis," *Journal of Medical Engineering & Technology*, vol. 36, no. 6, pp. 303–307, Jul. 2012. DOI: 10.3109/03091902.2012.684831.

[45] Wikimedia Commons, *Phonocardiograms from normal and abnormal heart sounds with pressure diagrams*, 2010. [Online]. Available: https://commons.wikimedia.org/wiki/File:Phonocardiograms_from_normal_and_abnormal_heart_sounds_with_pressure_diagrams.svg (visited on 06/27/2024).

[46] A. L. Goldberger, L. A. N. Amaral, L. Glass, *et al.*, "PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals," *Circulation*, vol. 101, no. 23, Jun. 2000. DOI: 10.1161/01.cir.101.23.e215.

[47] R. Judge and R. Mangrulkar. "The Open Michigan Heart Sound & Murmur Library (OMHSML)." (2015), [Online]. Available: http://www.med.umich.edu/lrc/psb/heartsounds/ (visited on 04/12/2024).

[48] P. Bentley, G. Nordehn, M. Coimbra, S. Mannor, and R. Getz. "The PASCAL Classifying Heart Sounds Challenge 2011 (CHSC2011)." (2011), [Online]. Available: https://istethoscope.peterjbentley.com/heartchallenge/index.html (visited on 04/12/2024).

[49] F. Dong, K. Qian, Z. Ren, *et al.*, "Machine listening for heart status monitoring: Introducing and benchmarking HSS–the heart sounds shenzhen corpus," *IEEE Journal of Biomedical and Health Informatics*, vol. 24, no. 7, pp. 2082–2092, Jul. 2020. DOI: 10.1109/jbhi.2019.2955281.

[50] J. Oliveira, F. Renna, T. Mantadelis, and M. Coimbra, "Adaptive sojourn time HSMM for heart sound segmentation," *IEEE Journal of Biomedical and Health Informatics*, vol. 23, no. 2, pp. 642–649, Mar. 2019. DOI: 10.1109/jbhi.2018.2841197.

[51] A. Spadaccini and F. Beritelli, "Performance evaluation of heart sounds biometric systems on an open dataset," in *2013 18th International Conference on Digital Signal Processing (DSP)*, IEEE, Jul. 2013. DOI: 10.1109/icdsp.2013.6622835.

[52] M. Cesarelli, M. Ruffo, M. Romano, and P. Bifulco, "Simulation of foetal phonocardiographic recordings for testing of FHR extraction algorithms," *Computer Methods and Programs in Biomedicine*, vol. 107, no. 3, pp. 513–523, Sep. 2012. DOI: 10.1016/j.cmpb.2011.11.008.

[53] A. Kazemnejad, P. Gordany, and R. Sameni, *EPHNOGRAM: A simultaneous electrocardiogram and phonocardiogram database*, 2021. DOI: 10.13026/TJTQ-5911.

[54] A. Kazemnejad, P. Gordany, and R. Sameni, "An open–access simultaneous electrocardiogram and phonocardiogram database," May 2021. DOI: 10.1101/2021.05.17.444563.

[55] Robert J. Hall Heart Sounds Laboratory. "Heart sounds podcast series." (2010), [Online]. Available: https://web.archive.org/web/20101226131759/http://texasheart.org/education/cme/explore/events/eventdetail_5469.cfm (visited on 04/25/2024).

[56] V. Nivitha Varghees and K. I. Ramachandran, "Effective heart sound segmentation and murmur classification using empirical wavelet transform and instantaneous phase for electronic stethoscope," *IEEE Sensors Journal*, vol. 17, no. 12, pp. 3861–3872, Jun. 2017. DOI: 10.1109/jsen.2017.2694970.

[57] C. Tuchinda and W. R. Thmopson, "Cardiac auscultatory recording database: Delivering heart sounds through the internet," in *Proc AMIA Symp*, IEEE, 2001, pp. 716–720.

[58] Yaseen, G.-Y. Son, and S. Kwon, "Classification of heart sound signal using multiple features," *Applied Sciences*, vol. 8, no. 12, p. 2344, Nov. 2018. DOI: 10.3390/app8122344.

[59] Q. Zhao, S. Geng, B. Wang, *et al.*, "Deep learning for heart sound analysis: A literature review," Sep. 2023. DOI: 10.1101/2023.09.16.23295653.

[60] S. Li, F. Li, S. Tang, and W. Xiong, "A review of computer-aided heart sound detection techniques," *BioMed Research International*, vol. 2020, pp. 1–10, Jan. 2020. DOI: 10.1155/2020/5846191.

[61] J. Chen, Z. Guo, X. Xu, G. Jeon, and D. Camacho, "Artificial intelligence for heart sound classification: A review," *Expert Systems*, vol. 41, no. 4, Jan. 2024. DOI: 10.1111/exsy.13535.

[62] M. Nabih-Ali, E.-S. A. El-Dahshan, and A. S. Yahia, "A review of intelligent systems for heart sound signal analysis," *Journal of Medical Engineering & Technology*, vol. 41, no. 7, pp. 553–563, Oct. 2017. DOI: 10.1080/03091902.2017.1382584.

[63] Z. Ren, Y. Chang, T. T. Nguyen, Y. Tan, K. Qian, and B. W. Schuller, *A comprehensive survey on heart sound analysis in the deep learning era*, 2023. arXiv: 2301.09362 [cs.SD].

[64] S. K. Ghosh, P. R. Nagarajan, and R. K. Tripathy, "Heart sound data acquisition and preprocessing techniques: A review," in *Advances in Healthcare Information Systems and Administration*. IGI Global, 2020, pp. 244–264. DOI: 10.4018/978-1-7998-2120-5.ch014.

[65] A. Williams and F. Taylor, *Electronic Filter Design Handbook, Fourth Edition*. McGraw Hill LLC, 2010, ISBN: 9780071491303.

[66] H. Alaskar, N. Alzhrani, A. Hussain, and F. Almarshed, "The implementation of pretrained AlexNet on PCG classification," in *Lecture Notes in Computer Science*. Springer International Publishing, 2019, pp. 784–794. DOI: 10.1007/978-3-030-26766-7_71.

[67] F. A. Khan, A. Abid, and M. S. Khan, "Automatic heart sound classification from segmented/unsegmented phonocardiogram signals using time and frequency features," *Physiological Measurement*, vol. 41, no. 5, p. 055 006, Jun. 2020. DOI: 10.1088/1361-6579/ab8770.

[68] A. Yadav, M. K. Dutta, C. M. Travieso, and J. B. Alonso, "Automatic classification of normal and abnormal PCG recording heart sound recording using fourier transform," in *2018 IEEE International Work Conference on Bioinspired Intelligence (IWOBI)*, IEEE, Jul. 2018. DOI: 10.1109/iwobi.2018.8464131.

[69] Q. Hu, J. Hu, X. Yu, and Y. Liu, "Automatic heart sound classification using one dimension deep neural network," in *Lecture Notes in Computer Science*. Springer International Publishing, 2021, pp. 200–208, ISBN: 9783030688844. DOI: 10.1007/978-3-030-68884-4_17.

[70] S. K. Ghosh, R. K. Tripathy, R. N. Ponnalagu, and R. B. Pachori, "Automated detection of heart valve disorders from the pcg signal using time-frequency magnitude and phase features," *IEEE Sensors Letters*, vol. 3, no. 12, pp. 1–4, Dec. 2019. DOI: 10.1109/lsens.2019.2949170.

[71] B. Ahmad, F. A. Khan, K. N. Khan, and M. S. Khan, "Automatic classification of heart sounds using long short-term memory," in *2021 15th International Conference on Open Source Systems and Technologies (ICOSST)*, IEEE, Dec. 2021. DOI: 10.1109/icosst53930.2021.9683975.

[72] S. A. Singh, T. G. Meitei, and S. Majumder, "Short PCG classification based on deep learning," in *Deep Learning Techniques for Biomedical and Health Informatics*. Elsevier, 2020, pp. 141–164. DOI: 10.1016/b978-0-12-819061-6.00006-9.

[73] F. Noman, S.-H. Salleh, C.-M. Ting, S. B. Samdin, H. Ombao, and H. Hussain, "A Markov-switching model approach to heart sound segmentation and classification," *IEEE Journal of Biomedical and Health Informatics*, vol. 24, no. 3, pp. 705–716, Mar. 2020. DOI: 10.1109/jbhi.2019.2925036.

[74] F. Noman, C.-M. Ting, S.-H. Salleh, and H. Ombao, "Short-segment heart sound classification using an ensemble of deep convolutional neural networks," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, May 2019. DOI: 10.1109/icassp.2019.8682668.

[75] J. Dastagir, F. A. Khan, M. S. Khan, and K. N. Khan, "Computer-aided phonocardiogram classification using multidomain time and frequency features," in *2021 International Conference on Artificial Intelligence (ICAI)*, IEEE, Apr. 2021. DOI: 10.1109/icai52203.2021.9445235.

[76] A. Bourouhou, A. Jilbab, C. Nacir, and A. Hammouch, "Heart sound signals segmentation and multiclass classification," *International Journal of Online and Biomedical Engineering (iJOE)*, vol. 16, no. 15, p. 64, Dec. 2020. DOI: 10.3991/ijoe.v16i15.16817.

[77] A. Meintjes, A. Lowe, and M. Legget, "Fundamental heart sound classification using the continuous wavelet transform and convolutional neural networks," in *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, IEEE, Jul. 2018. DOI: 10.1109/embc.2018.8512284.

[78] Z. Abduh, E. A. Nehary, M. A. Wahed, and Y. M. Kadah, "Classification of heart sounds using fractional Fourier transform based Mel-frequency spectral coefficients and stacked autoencoder deep neural network," *Journal of Medical Imaging and Health Informatics*, vol. 9, no. 1, pp. 1–8, Jan. 2019. DOI: 10.1166/jmihi.2019.2568.

[79] S. A. Singh, S. Majumder, and M. Mishra, "Classification of short unsegmented heart sound based on deep learning," in *2019 IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*, IEEE, May 2019. DOI: 10.1109/i2mtc.2019.8826991.

[80] S. Li, F. Li, S. Tang, and F. Luo, "Heart sounds classification based on feature fusion using lightweight neural networks," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–9, 2021. DOI: 10.1109/tim.2021.3109389.

[81] N. Ibrahim, N. Jamal, M. N. A.-H. Sha'abani, and L. F. Mahadi, "A comparative study of heart sound signal classification based on temporal, spectral and geometric features," in *2020 IEEE-EMBS Conference on Biomedical Engineering and Sciences (IECBES)*, IEEE, Mar. 2021. DOI: 10.1109/iecbes48179.2021.9398810.

[82] W. Zhang, J. Han, and S. Deng, "Heart sound classification based on scaled spectrogram and tensor decomposition," *Expert Systems with Applications*, vol. 84, pp. 220–231, Oct. 2017. DOI: 10.1016/j.eswa.2017.05.014.

[83] W. Zhang, J. Han, and S. Deng, "Heart sound classification based on scaled spectrogram and partial least squares regression," *Biomedical Signal Processing and Control*, vol. 32, pp. 20–28, Feb. 2017. DOI: 10.1016/j.bspc.2016.10.004.

[84] M. Banerjee and S. Majhi, "Multi-class heart sounds classification using 2D-convolutional neural network," in *2020 5th International Conference on Computing, Communication and Security (ICCCS)*, IEEE, Oct. 2020. DOI: 10.1109/icccs49678.2020.9277204.

[85] J. M.-T. Wu, M.-H. Tsai, Y. Z. Huang, *et al.*, "Applying an ensemble convolutional neural network with Savitzky–Golay filter to construct a phonocardiogram prediction model," *Applied Soft Computing*, vol. 78, pp. 29–40, May 2019. DOI: 10.1016/j.asoc.2019.01.019.

[86] P. T. Krishnan, P. Balasubramanian, and S. Umapathy, "Automated heart sound classification system from unsegmented phonocardiogram (PCG) using deep neural network," *Physical and Engineering Sciences in Medicine*, vol. 43, no. 2, pp. 505–515, Feb. 2020. DOI: 10.1007/s13246-020-00851-w.

[87] W. Zeng, J. Yuan, C. Yuan, Q. Wang, F. Liu, and Y. Wang, "A new approach for the detection of abnormal heart sound signals using TQWT, VMD and neural networks," *Artificial Intelligence Review*, vol. 54, no. 3, pp. 1613–1647, Jul. 2020. DOI: 10.1007/s10462-020-09875-w.

[88] M. V. Shervegar and G. V. Bhat, "Heart sound classification using gaussian mixture model," *Porto Biomedical Journal*, vol. 3, no. 1, e4, Aug. 2018. DOI: 10.1016/j.pbj.0000000000000004.

[89] B. Al-Naami, H. Fraihat, N. Y. Gharaibeh, and A.-R. M. Al-Hinnawi, "A framework classification of heart sound signals in PhysioNet Challenge 2016 using high order statistics and adaptive neuro-fuzzy inference system," *IEEE Access*, vol. 8, pp. 224 852–224 859, 2020. DOI: 10.1109/access.2020.3043290.

[90] S. Mallat, *A wavelet tour of signal processing*, en. San Diego, CA: Academic Press, Mar. 1998.

[91] S. K. Ghosh, R. K. Tripathy, and P. R N, "Evaluation of performance metrics and denoising of PCG signal using wavelet based decomposition," in *2020 IEEE 17th India Council International Conference (INDICON)*, IEEE, Dec. 2020. DOI: 10.1109/indicon49873.2020.9342464.

[92] Z. Dokur and T. Ölmez, "Heart sound classification using wavelet transform and incremental self-organizing map," *Digital Signal Processing*, vol. 18, no. 6, pp. 951–959, Nov. 2008. DOI: 10.1016/j.dsp.2008.06.001.

[93] M. N. Ali, E.-S. A. El-Dahshan, and A. H. Yahia, "Denoising of heart sound signals using discrete wavelet transform," *Circuits, Systems, and Signal Processing*, vol. 36, no. 11, pp. 4482–4497, Mar. 2017. DOI: 10.1007/s00034-017-0524-7.

[94] P. K. Jain and A. K. Tiwari, "An adaptive thresholding method for the wavelet based denoising of phonocardiogram signal," *Biomedical Signal Processing and Control*, vol. 38, pp. 388–399, Sep. 2017. DOI: 10.1016/j.bspc.2017.07.002.

[95] M. Rouis, A. Ouafi, and S. Sbaa, "Optimal level and order detection in wavelet decomposition for PCG signal denoising," *Biomedical Engineering / Biomedizinische Technik*, vol. 64, no. 2, pp. 163–176, May 2018. DOI: 10.1515/bmt-2018-0001.

[96] D. Gradolewski and G. Redlarski, "Wavelet-based denoising method for real phonocardiography signal recorded by mobile devices in noisy environment," *Computers in Biology and Medicine*, vol. 52, pp. 119–129, Sep. 2014. DOI: 10.1016/j.compbiomed.2014.06.011.

[97] A. Sbrollini, A. Strazza, M. Caragiuli, *et al.*, "Fetal phonocardiogram denoising by wavelet transformation: Robustness to noise," in *2017 Computing in Cardiology Conference (CinC)*, ser. CinC2017, Computing in Cardiology, Sep. 2017. DOI: 10.22489/cinc.2017.331-075.

[98] D. Agustika, S. Sumarna, A. Purwanto, and J. Astono, "Optimization of denoising technique of phonocardiography signal by using discrete wavelet transform," in *Proceedings of the International Conference of Science and Technology for the Internet of Things*, ser. ICSTI, EAI, 2019. DOI: 10.4108/eai.19-10-2018.2281369.

[99] H. Sun, W. Chen, and J. Gong, "An improved empirical mode decomposition-wavelet algorithm for phonocardiogram signal denoising and its application in the first and second heart sound extraction," in *2013 6th International Conference on Biomedical Engineering and Informatics*, IEEE, Dec. 2013. DOI: 10.1109/bmei.2013.6746931.

[100] J. Pedrosa, A. Castro, and T. T. V. Vinhoza, "Automatic heart sound segmentation and murmur detection in pediatric phonocardiograms," in *2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, IEEE, Aug. 2014. DOI: 10.1109/embc.2014.6944078.

[101] A. Quiceno, E. Delgado, M. Vallverd, A. Matijasevic, and G. Castellanos-Domnguez, "Effective phonocardiogram segmentation using nonlinear dynamic analysis and high-frequency decomposition," in *2008 Computers in Cardiology*, IEEE, Sep. 2008. DOI: 10.1109/cic.2008.4749002.

[102] P. Kumar Jain and A. Kumar Tiwari, "An adaptive method for shrinking of wavelet coefficients for phonocardiogram denoising," in *2016 IEEE International Conference on Digital Signal Processing (DSP)*, IEEE, Oct. 2016. DOI: 10.1109/icdsp.2016.7868503.

[103] N. Kouras, D. Boutana, and M. Benidir, "Wavelet based segmentation and time-frequency caracterisation of some abnormal heart sound signals," in *2012 24th International Conference on Microelectronics (ICM)*, IEEE, Dec. 2012. DOI: 10.1109/icm.2012.6471392.

[104] M. S. Grewal and A. P. Andrews, *Kalman Filtering: Theory and Practice Using MATLAB®*. Wiley, Jan. 2008, ISBN: 9780470377819. DOI: 10.1002/9780470377819.

[105] P. Yupapin, S. Sh Hussain, S.-H. Hadrina, *et al.*, "Acoustic cardiac signals analysis: A Kalman filter–based approach," *International Journal of Nanomedicine*, p. 2873, Jun. 2012. DOI: 10.2147/ijn.s32315.

[106] M. S. Nazemi, H. Hakimnejad, and Z. Azimifar, "PCG denoising using AR-based Kalman filter," in *2021 29th Iranian Conference on Electrical Engineering (ICEE)*, IEEE, May 2021. DOI: 10.1109/icee52715.2021.9544365.

[107] A. Almasi, M. Bagher Shamsollahi, and L. Senhadji, "Bayesian denoising framework of phonocardiogram based on a new dynamical model," *IRBM*, vol. 34, no. 3, pp. 214–225, Jun. 2013. DOI: 10.1016/j.irbm.2013.01.017.

[108] A. Gavrovska, M. Slavkovic, I. Reljin, and B. Reljin, "Application of wavelet and EMD-based denoising to phonocardiograms," in *International Symposium on Signals, Circuits and Systems ISSCS2013*, IEEE, Jul. 2013. DOI: 10.1109/isscs.2013.6651264.

[109] P. Ravina Manohar, M. Meshram, N. Dewangan, and R. Kumar, "Implementation of adaptive algorithm for PCG signal denoising," *IJIREEICE*, vol. 3, no. 4, pp. 33–42, Apr. 2015. DOI: 10.17148/ijireeice.2015.3408.

[110] S. Lahmiri and M. Boukadoum, "Biomedical image denoising using variational mode decomposition," in *2014 IEEE Biomedical Circuits and Systems Conference (BioCAS) Proceedings*, IEEE, Oct. 2014. DOI: 10.1109/biocas.2014.6981732.

[111] D.-H. Pham, S. Meignen, N. Dia, J. Fontecave-Jallon, and B. Rivet, "Phonocardiogram signal denoising based on nonnegative matrix factorization and adaptive contour representation computation," *IEEE Signal Processing Letters*, vol. 25, no. 10, pp. 1475–1479, Oct. 2018. DOI: 10.1109/lsp.2018.2865253.

[112] C. Lin and E. Hasting, "Blind source separation of heart and lung sounds based on nonnegative matrix factorization," in *2013 International Symposium on Intelligent Signal Processing and Communication Systems*, IEEE, Nov. 2013. DOI: 10.1109/ispacs.2013.6704646.

[113] F. Safara, S. Doraisamy, A. Azman, A. Jantan, and S. Ranga, "Wavelet packet entropy for heart murmurs classification," *Advances in Bioinformatics*, vol. 2012, pp. 1–6, Nov. 2012. DOI: 10.1155/2012/327269.

[114] M. K. Zia, B. Griffel, and J. L. Semmlow, "Robust detection of background noise in phonocardiograms," in *2011 1st Middle East Conference on Biomedical Engineering*, IEEE, Feb. 2011. DOI: 10.1109/mecbme.2011.5752082.

[115] A. Mondal, I. Saxena, H. Tang, and P. Banerjee, "A noise reduction technique based on nonlinear kernel function for heart sound analysis," *IEEE Journal of Biomedical and Health Informatics*, vol. 22, no. 3, pp. 775–784, May 2018. DOI: 10.1109/jbhi.2017.2667685.

[116] S. Mandal, K. Basak, K. M. Mandana, A. K. Ray, J. Chatterjee, and M. Mahadevappa, "Development of cardiac prescreening device for rural population using ultralow-power embedded system," *IEEE Transactions on Biomedical Engineering*, vol. 58, no. 3, pp. 745–749, Mar. 2011. DOI: 10.1109/tbme.2010.2089457.

[117] T. S. Sharan, R. Bhattacharjee, S. Sharma, and N. Sharma, "Evaluation of deep learning methods (DnCNN and U-Net) for denoising of heart auscultation signals," in *2020 3rd International Conference on Communication System, Computing and IT Applications (CSCITA)*, IEEE, Apr. 2020. DOI: 10.1109/cscita47329.2020.9137813.

[118] S. N. Ali, S. B. Shuvo, M. I. S. Al-Manzo, A. Hasan, and T. Hasan, "An end-to-end deep learning framework for real-time denoising of heart sounds for cardiac disease detection in unseen noise," *IEEE Access*, vol. 11, pp. 87 887–87 901, 2023. DOI: 10.1109/access.2023.3292551.

[119] D. Boutana, N. Kouras, and M. Benidir, "The empirical mode decomposition and the SVD for abnormal heart sound signals detection and time-frequency analysis," in *2nd IET International Conference on Intelligent Signal Processing 2015 (ISP)*, Institution of Engineering and Technology, 2015. DOI: 10.1049/cp.2015.1780.

[120] A. H. Salman, N. Ahmadi, R. Mengko, A. Z. R. Langi, and T. L. R. Mengko, "Performance comparison of denoising methods for heart sound signal," in *2015 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS)*, IEEE, Nov. 2015. DOI: 10.1109/ispacs.2015.7432811.

[121] A. H. Salman, N. Ahmadi, R. Mengko, A. Z. R. Langi, and T. L. R. Mengko, "Empirical mode decomposition (EMD) based denoising method for heart sound signal and its performance analysis," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 6, no. 5, p. 2197, Oct. 2016. DOI: 10.11591/ijece.v6i5.pp2197-2204.

[122] S. Ismail, I. Siddiqi, and U. Akram, "Localization and classification of heart beats in phonocardiography signals —a comprehensive review," *EURASIP Journal on Advances in Signal Processing*, vol. 2018, no. 1, Apr. 2018. DOI: 10.1186/s13634-018-0545-9.

[123] A. Moukadem, A. Dieterlen, N. Hueber, and C. Brandt, "A robust heart sounds segmentation module based on S-transform," *Biomedical Signal Processing and Control*, vol. 8, no. 3, pp. 273–281, May 2013. DOI: 10.1016/j.bspc.2012.11.008.

[124] M. Z. Othman and A. Khaleel, "Phonocardiogram signal analysis for murmur diagnosing using shannon energy envelop and sequenced DWT decomposition," *Journal of Engineering Science and Technology*, vol. 12, pp. 2393–2402, Sep. 2017.

[125] A. A. Sepehri, A. Gharehbaghi, T. Dutoit, A. Kocharian, and A. Kiani, "A novel method for pediatric heart sound segmentation without using the ECG," *Computer Methods and Programs in Biomedicine*, vol. 99, no. 1, pp. 43–48, Jul. 2010. DOI: 10.1016/j.cmpb.2009.10.006.

[126] N. Giordano and M. Knaflitz, "A novel method for measuring the timing of heart sound components through digital phonocardiography," *Sensors*, vol. 19, no. 8, p. 1868, Apr. 2019. DOI: 10.3390/s19081868.

[127] S. Sun, Z. Jiang, H. Wang, and Y. Fang, "Automatic moment segmentation and peak detection analysis of heart sound pattern via short-time modified hilbert transform," *Computer Methods and Programs in Biomedicine*, vol. 114, no. 3, pp. 219–230, May 2014. DOI: 10.1016/j.cmpb.2014.02.004.

[128] S. Sun, "An innovative intelligent system based on automatic diagnostic feature extraction for diagnosing heart diseases," *Knowledge-Based Systems*, vol. 75, pp. 224–238, Feb. 2015. DOI: 10.1016/j.knosys.2014.12.001.

[129] P. Sharma, S. A. Imtiaz, and E. Rodriguez-Villegas, "An algorithm for heart rate extraction from acoustic recordings at the neck," *IEEE Transactions on Biomedical Engineering*, vol. 66, no. 1, pp. 246–256, Jan. 2019. DOI: 10.1109/tbme.2018.2836187.

[130] S.-W. Deng and J.-Q. Han, "Towards heart sound classification without segmentation via autocorrelation feature and diffusion maps," *Future Generation Computer Systems*, vol. 60, pp. 13–21, Jul. 2016. DOI: 10.1016/j.future.2016.01.010.

[131] S. Yuenyong, A. Nishihara, W. Kongprawechnon, and K. Tungpimolrut, "A framework for automatic heart sound analysis without segmentation," *BioMedical Engineering OnLine*, vol. 10, no. 1, p. 13, 2011. DOI: 10.1186/1475-925x-10-13.

[132] V. N. Varghees and K. Ramachandran, "A novel heart sound activity detection framework for automated heart sound analysis," *Biomedical Signal Processing and Control*, vol. 13, pp. 174–188, Sep. 2014. DOI: 10.1016/j.bspc.2014.05.002.

[133] F. Plesinger, I. Viscor, J. Halamek, J. Jurco, and P. Jurak, "Heart sounds analysis using probability assessment," *Physiological Measurement*, vol. 38, no. 8, pp. 1685–1700, Jul. 2017. DOI: 10.1088/1361-6579/aa7620.

[134] A. Gharehbaghi, T. Dutoit, A. Sepehri, P. Hult, and P. Ask, "An automatic tool for pediatric heart sounds segmentation," in *2011 Computing in Cardiology*, 2011, pp. 37–40.

[135] S. Jabbari and H. Ghassemian, "Modeling of heart systolic murmurs based on multivariate matching pursuit for diagnosis of valvular disorders," *Computers in Biology and Medicine*, vol. 41, no. 9, pp. 802–811, Sep. 2011. DOI: 10.1016/j.compbiomed.2011.06.016.

[136] C. Ahlstrom, P. Hult, P. Rask, *et al.*, "Feature extraction for systolic heart murmur classification," *Annals of Biomedical Engineering*, vol. 34, no. 11, pp. 1666–1677, Oct. 2006. DOI: 10.1007/s10439-006-9187-4.

[137] A. Haghighi-Mood and J. Torry, "A sub–band energy tracking algorithm for heart sound segmentation," in *Computers in Cardiology 1995*, ser. CIC–95, IEEE. DOI: 10.1109/cic.1995.482711.

[138] H. Tang, T. Li, T. Qiu, and Y. Park, "Segmentation of heart sounds based on dynamic clustering," *Biomedical Signal Processing and Control*, vol. 7, no. 5, pp. 509–516, Sep. 2012. DOI: 10.1016/j.bspc.2011.09.002.

[139] Y.-L. Tseng, P.-Y. Ko, and F.-S. Jaw, "Detection of the third and fourth heart sounds using Hilbert-Huang transform," *BioMedical Engineering OnLine*, vol. 11, no. 1, Feb. 2012. DOI: 10.1186/1475-925x-11-8.

[140] B. Alexander, G. Nallathambi, and N. Selvaraj, "Screening of heart sounds using hidden markov and gammatone filterbank models," in *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, IEEE, Dec. 2018. DOI: 10.1109/icmla.2018.00237.

[141] C. Kwak and O.-W. Kwon, "Cardiac disorder classification by heart sound signals using murmur likelihood and hidden markov model state likelihood," *IET Signal Processing*, vol. 6, no. 4, p. 326, 2012. DOI: 10.1049/iet-spr.2011.0170.

[142] A. P. Kamson, L. Sharma, and S. Dandapat, "Multi-centroid diastolic duration distribution based HSMM for heart sound segmentation," *Biomedical Signal Processing and Control*, vol. 48, pp. 265–272, Feb. 2019. DOI: 10.1016/j.bspc.2018.10.018.

[143] E. Kay and A. Agarwal, "DropConnected neural networks trained on time-frequency and inter-beat features for classifying heart sounds," *Physiological Measurement*, vol. 38, no. 8, pp. 1645–1657, Jul. 2017. DOI: 10.1088/1361-6579/aa6a3d.

[144] C. Liu, D. Springer, and G. D. Clifford, "Performance of an open-source heart sound segmentation algorithm on eight independent databases," *Physiological Measurement*, vol. 38, no. 8, pp. 1730–1745, Aug. 2017. DOI: 10.1088/1361-6579/aa6e9f.

[145] M. Abdollahpur, A. Ghaffari, S. Ghiasi, and M. J. Mollakazemi, "Detection of pathological heart sounds," *Physiological Measurement*, vol. 38, no. 8, pp. 1616–1630, Jul. 2017. DOI: 10.1088/1361-6579/aa7840.

[146] M. Nabhan Homsi and P. Warrick, "Ensemble methods with outliers for phonocardiogram classification," *Physiological Measurement*, vol. 38, no. 8, pp. 1631–1644, Jul. 2017. DOI: 10.1088/1361-6579/aa7982.

[147] F. Renna, J. Oliveira, and M. T. Coimbra, "Deep convolutional neural networks for heart sound segmentation," *IEEE Journal of Biomedical and Health Informatics*, vol. 23, no. 6, pp. 2435–2445, Nov. 2019. DOI: 10.1109/jbhi.2019.2894222.

[148] S. E. Schmidt, C. Holst-Hansen, C. Graff, E. Toft, and J. J. Struijk, "Segmentation of heart sound recordings by a duration-dependent hidden markov model," *Physiological Measurement*, vol. 31, no. 4, pp. 513–529, Mar. 2010. DOI: 10.1088/0967-3334/31/4/004.

[149] H. Naseri and M. R. Homaeinezhad, "Detection and boundary identification of phonocardiogram sounds using an expert frequency-energy based metric," *Annals of Biomedical Engineering*, vol. 41, no. 2, pp. 279–292, Sep. 2012. DOI: 10.1007/s10439-012-0645-x.

[150] C. D. Papadaniil and L. J. Hadjileontiadis, "Efficient heart sound segmentation and extraction using ensemble empirical mode decomposition and kurtosis features," *IEEE Journal of Biomedical and Health Informatics*, vol. 18, no. 4, pp. 1138–1152, Jul. 2014. DOI: 10.1109/jbhi.2013.2294399.

[151] J. E. Guillermo, L. J. Ricalde Castellanos, E. N. Sanchez, and A. Y. Alanis, "Detection of heart murmurs based on radial wavelet neural network with kalman learning," *Neurocomputing*, vol. 164, pp. 307–317, Sep. 2015. DOI: 10.1016/j.neucom.2014.12.059.

[152] T.-E. Chen, S.-I. Yang, L.-T. Ho, *et al.*, "S1 and S2 heart sound recognition using deep neural networks," *IEEE Transactions on Biomedical Engineering*, vol. 64, no. 2, pp. 372–380, 2017. DOI: 10.1109/TBME.2016.2559800.

[153] Y. Zheng, X. Guo, and X. Ding, "A novel hybrid energy fraction and entropy-based approach for systolic heart murmurs identification," *Expert Systems with Applications*, vol. 42, no. 5, pp. 2710–2721, Apr. 2015, ISSN: 0957-4174. DOI: 10.1016/j.eswa.2014.10.051. [Online]. Available: http://dx.doi.org/10.1016/j.eswa.2014.10.051.

[154] V. Nigam and R. Priemer, "Accessing heart dynamics to estimate durations of heart sounds," *Physiological Measurement*, vol. 26, no. 6, pp. 1005–1018, Oct. 2005. DOI: 10.1088/0967-3334/26/6/010.

[155] S. Ari, P. Kumar, and G. Saha, "A robust heart sound segmentation algorithm for commonly occurring heart valve diseases," *Journal of Medical Engineering & Technology*, vol. 32, no. 6, pp. 456–465, Jan. 2008. DOI: 10.1080/03091900601015162.

[156] A. H. Salman, N. Ahmadi, R. Mengko, A. Z. R. Langi, and T. L. R. Mengko, "Automatic segmentation and detection of heart sound components S1, S2, S3 and S4," in *2015 4th International Conference on Instrumentation, Communications, Information Technology, and Biomedical Engineering (ICICI-BME)*, IEEE, Nov. 2015. DOI: 10.1109/icici-bme.2015.7401344.

[157] M. Z. Belmecheri, M. Ahfir, and I. Kale, "Automatic heart sounds segmentation based on the correlation coefficients matrix for similar cardiac cycles identification," *Biomedical Signal Processing and Control*, vol. 43, pp. 300–310, May 2018. DOI: 10.1016/j.bspc.2018.03.009.

[158] S. Sanei, M. Ghodsi, and H. Hassani, "An adaptive singular spectrum analysis approach to murmur detection from heart sounds," *Medical Engineering & Physics*, vol. 33, no. 3, pp. 362–367, Apr. 2011. DOI: 10.1016/j.medengphy.2010.11.004.

[159] S. Patidar and R. B. Pachori, "Segmentation of cardiac sound signals by removing murmurs using constrained tunable-Q wavelet transform," *Biomedical Signal Processing and Control*, vol. 8, no. 6, pp. 559–567, Nov. 2013. DOI: 10.1016/j.bspc.2013.05.004.

[160] P. K. Jain and A. K. Tiwari, "A robust algorithm for segmentation of phonocardiography signal using tunable quality wavelet transform," *Journal of Medical and Biological Engineering*, vol. 38, no. 3, pp. 396–410, Sep. 2017. DOI: 10.1007/s40846-017-0320-7.

[161] K. A. Babu, B. Ramkumar, and M. S. Manikandan, "S1 and S2 heart sound segmentation using variational mode decomposition," in *TENCON 2017 - 2017 IEEE Region 10 Conference*, IEEE, Nov. 2017. DOI: 10.1109/tencon.2017.8228119.

[162] R. Thomas, L. L. Hsi, S. C. Boon, and E. Gunawan, "Heart sound segmentation using fractal decomposition," in *2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, IEEE, Aug. 2016. DOI: 10.1109/embc.2016.7592153.

[163] A. Gavrovska, G. Zajić, V. Bogdanović, I. Reljin, and B. Reljin, "Identification of S1 and S2 heart sound patterns based on fractal theory and shape context," *Complexity*, vol. 2017, pp. 1–9, 2017. DOI: 10.1155/2017/1580414.

[164] Q. Liu, X. Wu, and X. Ma, "An automatic segmentation method for heart sounds," *BioMedical Engineering OnLine*, vol. 17, no. 1, Aug. 2018. DOI: 10.1186/s12938-018-0538-9.

[165] S. Kang, R. Doroshow, J. McConnaughey, and R. Shekhar, "Automated identification of innocent still's murmur in children," *IEEE Transactions on Biomedical Engineering*, vol. 64, no. 6, pp. 1326–1334, Jun. 2017. DOI: 10.1109/tbme.2016.2603787.

[166] R. Saraçoğlu, "Hidden markov model-based classification of heart valve disease with PCA for dimension reduction," *Engineering Applications of Artificial Intelligence*, vol. 25, no. 7, pp. 1523–1528, Oct. 2012. DOI: 10.1016/j.engappai.2012.07.005.

[167] H. M. Fahad, M. U. Ghani Khan, T. Saba, A. Rehman, and S. Iqbal, "Microscopic abnormality classification of cardiac murmurs using ANFIS and HMM," *Microscopy Research and Technique*, vol. 81, no. 5, pp. 449–457, Jan. 2018. DOI: 10.1002/jemt.22998.

[168] M. Zabihi, A. B. Rad, S. Kiranyaz, M. Gabbouj, and A. K. Katsaggelos, "Heart sound anomaly and quality detection using ensemble of neural networks without segmentation," in *2016 Computing in Cardiology Conference (CinC)*, 2016, pp. 613–616.

[169] G. Redlarski, D. Gradolewski, and A. Palkowski, "A system for heart sounds classification," *PLoS ONE*, vol. 9, no. 11, A. Talkachova, Ed., e112673, Nov. 2014. DOI: 10.1371/journal.pone.0112673.

[170] S. Das, S. Pal, and M. Mitra, "Acoustic feature based unsupervised approach of heart sound event detection," *Computers in Biology and Medicine*, vol. 126, p. 103 990, Nov. 2020. DOI: 10.1016/j.compbiomed.2020.103990.

[171] W. Wang, X. Yu, B. Fang, *et al.*, "Cross-modality LGE-CMR segmentation using image-to-image translation based data augmentation," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 20, no. 4, pp. 2367–2375, Jul. 2023. DOI: 10.1109/tcbb.2022.3140306.

[172] H. Tang, T. Li, and T. Qiu, "Noise and disturbance reduction for heart sounds in cycle-frequency domain based on nonlinear time scaling," *IEEE Transactions on Biomedical Engineering*, vol. 57, no. 2, pp. 325–333, Feb. 2010. DOI: 10.1109/tbme.2009.2028693.

[173] T. Fan, J. Zhu, Y. Cheng, Q. Li, D. Xue, and R. Munnoch, "A new direct heart sound segmentation approach using bi-directional GRU," in *2018 24th International Conference on Automation and Computing (ICAC)*, IEEE, Sep. 2018. DOI: 10.23919/iconac.2018.8749010.

[174] Y. Chen, J. Lv, Y. Sun, and B. Jia, "Heart sound segmentation via duration long–short term memory neural network," *Applied Soft Computing*, vol. 95, p. 106 540, Oct. 2020. DOI: 10.1016/j.asoc.2020.106540.

[175] T. Fernando, H. Ghaemmaghami, S. Denman, S. Sridharan, N. Hussain, and C. Fookes, "Heart sound segmentation using bidirectional LSTMs with attention," *IEEE Journal of Biomedical and Health Informatics*, vol. 24, no. 6, pp. 1601–1609, Jun. 2020. DOI: 10.1109/jbhi.2019.2949516.

[176] Y. Chen, Y. Sun, J. Lv, B. Jia, and X. Huang, "End-to-end heart sound segmentation using deep convolutional recurrent network," *Complex &; Intelligent Systems*, vol. 7, no. 4, pp. 2103–2117, Mar. 2021. DOI: 10.1007/s40747-021-00325-w.

[177] M. C. Bishop, *Pattern Recognition and Machine Learning* (Information Science and Statistics), en, 1st ed. New York, NY: Springer, Aug. 2006.

[178] M. Kuhn and K. Johnson, *Applied predictive modeling*, en, 1st ed. New York, NY: Springer, May 2013.

[179] S. Guido and A. C. Mueller, *Introduction to machine learning with python*. Sebastopol, CA: O'Reilly Media, Oct. 2016.

[180] D. Weisburd, C. Britt, D. B. Wilson, and A. Wooditch, "Measuring association for scaled data: Pearson's correlation coefficient," in *Basic Statistics in Criminology and Criminal Justice*. Springer International Publishing, 2020, pp. 479–530. DOI: 10.1007/978-3-030-47967-1_14.

[181] M. Hollander, D. A. Wolfe, and E. Chicken, *Nonparametric Statistical Methods* (Wiley Series in Probability and Statistics), en, 3rd ed. Hoboken, NJ: Wiley-Blackwell, Nov. 2013.

[182] A. Heinen and A. Valdesogo, "The Kendall and Spearman rank correlations of the bivariate skew normal distribution," *Scandinavian Journal of Statistics*, vol. 49, no. 4, pp. 1669–1698, Mar. 2022. DOI: 10.1111/sjos.12587.

[183] S. B. Shankar and U. Arcot, "ANOVA-based analysis using MATLAB for groundwater quality assessment," May 2023. DOI: 10.21203/rs.3.rs-2901343/v1.

[184] S. Shennan, "The chi-squared test," in *Quantifying Archaeology*. Elsevier, 1988, pp. 65–76, ISBN: 9780126398601. DOI: 10.1016/b978-0-12-639860-1.50009-2.

[185] A. Kraskov, H. Stögbauer, and P. Grassberger, "Estimating mutual information," *Physical Review E*, vol. 69, no. 6, Jun. 2004. DOI: 10.1103/physreve.69.066138.

[186] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society Series B: Statistical Methodology*, vol. 58, no. 1, pp. 267–288, Jan. 1996. DOI: 10.1111/j.2517-6161.1996.tb02080.x.

[187] A. E. Hoerl and R. W. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, vol. 42, no. 1, pp. 80–86, Feb. 2000. DOI: 10.1080/00401706.2000.10485983.

[188] A. Krizhevsky, V. Nair, and G. Hinton, "CIFAR-10 (Canadian Institute for Advanced Research)," [Online]. Available: http://www.cs.toronto.edu/~kriz/cifar.html.

[189] I. T. Jolliffe, *Principal Component Analysis* (Springer Series in Statistics), en, 2nd ed. New York, NY: Springer, Oct. 2002.

[190] L. van der Maaten and G. Hinton, "Viualizing data using t-SNE," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, Nov. 2008.

[191] A. Geron, *Hands-on machine learning with scikit-learn, keras, and TensorFlow*, 2nd ed. Sebastopol, CA: O'Reilly Media, Oct. 2019.

[192] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16, ACM, Aug. 2016. DOI: 10.1145/2939672.2939785.

[193] G. Ke, Q. Meng, T. Finley, *et al.*, "LightGBM: A highly efficient gradient boosting decision tree," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, *et al.*, Eds., vol. 30, Curran Associates, Inc., 2017.

[194] T. Hastie, R. Tibshirani, and J. H. Friedman, *The elements of statistical learning* (Springer series in statistics), en, 2nd ed. New York, NY: Springer, Dec. 2009.

[195] Y. Zheng, X. Guo, J. Qin, and S. Xiao, "Computer-assisted diagnosis for chronic heart failure by the analysis of their cardiac reserve and heart sound characteristics," *Computer Methods and Programs in Biomedicine*, vol. 122, no. 3, pp. 372–383, Dec. 2015, ISSN: 0169-2607. DOI: 10.1016/j.cmpb.2015.09.001.

[196] M. Tschannen, T. Kramer, G. Marti, M. Heinzmann, and T. Wiatowski, "Heart sound classification using deep structured features," in *2016 Computing in Cardiology Conference (CinC)*, ser. CinC2016, Computing in Cardiology, Sep. 2016. DOI: 10.22489/cinc.2016.162-186.

[197] F. Demir, A. Şengür, V. Bajaj, and K. Polat, "Towards the classification of heart sounds based on convolutional deep neural network," *Health Information Science and Systems*, vol. 7, no. 1, Aug. 2019, ISSN: 2047-2501. DOI: 10.1007/s13755-019-0078-0.

[198] L. Hussain, I. A. Awan, W. Aziz, *et al.*, "Detecting congestive heart failure by extracting multimodal features and employing machine learning techniques," *BioMed Research International*, vol. 2020, pp. 1–19, Feb. 2020, ISSN: 2314-6141. DOI: 10.1155/2020/4281243.

[199] N. E. Singh-Miller and N. Singh-Miller, "Using spectral acoustic features to identify abnormal heart sounds," in *2016 Computing in Cardiology Conference (CinC)*, 2016, pp. 557–560.

[200] M. Á. Goda and P. Hajas, "Morphological determination of pathological PCG signals by time and frequency domain analysis," in *2016 Computing in Cardiology Conference (CinC)*, 2016, pp. 1133–1136.

[201] M. N. Homsi, N. Medina, M. Hernandez, *et al.*, "Automatic heart sound recording classification using a nested set of ensemble algorithms," in *2016 Computing in Cardiology Conference (CinC)*, 2016, pp. 817–820.

[202] J. J. González Ortiz, C. P. Phoo, and J. Wiens, "Heart sound classification based on temporal alignment techniques," in *2016 Computing in Cardiology Conference (CinC)*, 2016, pp. 589–592.

[203] S. Vernekar, S. Nair, D. Vijaysenan, and R. Ranjan, "A novel approach for classification of normal/abnormal phonocardiogram recordings using temporal signal analysis and machine learning," in *2016 Computing in Cardiology Conference (CinC)*, 2016, pp. 1141–1144.

[204] I. Grzegorczyk, M. Soliński, M. Łepek, *et al.*, "PCG classification using a neural network approach," in *2016 Computing in Cardiology Conference (CinC)*, 2016, pp. 1129–1132.

[205] Y. Liu, X. Guo, and Y. Zheng, "An automatic approach using ELM classifier for HFpEF identification based on heart sound characteristics," *Journal of Medical Systems*, vol. 43, no. 9, Jul. 2019, ISSN: 1573-689X. DOI: 10.1007/s10916-019-1415-1.

[206] M. Gjoreski, A. Gradisek, B. Budna, M. Gams, and G. Poglajen, "Machine learning and end-to-end deep learning for the detection of chronic heart failure from heart sounds," *IEEE Access*, vol. 8, pp. 20 313–20 324, 2020, ISSN: 2169-3536. DOI: 10.1109/access.2020.2968900.

[207] M. Habijan, I. Galić, and A. Pizurica, "Heart sound classification using deep learning," in *2023 8th International Conference on Smart and Sustainable Technologies (SpliTech)*, IEEE, Jun. 2023. DOI: 10.23919/splitech58164.2023.10193726. [Online]. Available: http://dx.doi.org/10.23919/SpliTech58164.2023.10193726.

[208] J. Rubin, R. Abreu, A. Ganguli, S. Nelaturi, I. Matei, and K. Sricharan, "Classifying heart sound recordings using deep convolutional neural networks and mel:frequency cepstral coefficients," in *2016 Computing in Cardiology Conference (CinC)*, ser. CinC2016, Computing in Cardiology, Sep. 2016. DOI: 10.22489/cinc.2016.236-175.

[209] G. Zhou, Y. Chen, and C. Chien, "On the analysis of data augmentation methods for spectral imaged based heart sound classification using convolutional neural networks," *BMC Medical Informatics and Decision Making*, vol. 22, no. 1, Aug. 2022, ISSN: 1472-6947. DOI: 10.1186/s12911-022-01942-2.

[210] T. Nilanon, S. Purushotham, and Y. Liu, "Normal / abnormal heart sound recordings classification using convolutional neural network," in *2016 Computing in Cardiology Conference (CinC)*, ser. CinC2016, Computing in Cardiology, Sep. 2016. DOI: 10.22489/cinc.2016.169-535.

[211] D. Kucharski, D. Grochala, M. Kajor, and E. Kańtoch, "A deep learning approach for valve defect recognition in heart acoustic signal," in *Advances in Intelligent Systems and Computing*. Springer International Publishing, Sep. 2017, pp. 3–14, ISBN: 9783319672205. DOI: 10.1007/978-3-319-67220-5_1.

[212] X. Cheng, J. Huang, Y. Li, and G. Gui, "Design and application of a laconic heart sound neural network," *IEEE Access*, vol. 7, pp. 124 417–124 425, 2019, ISSN: 2169-3536. DOI: 10.1109/access.2019.2934827.

[213] T. Koike, K. Qian, B. W. Schuller, and Y. Yamamoto, "Transferring cross-corpus knowledge: An investigation on data augmentation for heart sound classification," in *2021 43rd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, IEEE, Nov. 2021. DOI: 10.1109/embc46164.2021.9629714.

[214] S. L. Oh, V. Jahmunah, C. P. Ooi, *et al.*, "Classification of heart sound signals using a novel deep wavenet model," *Computer Methods and Programs in Biomedicine*, vol. 196, p. 105 604, Nov. 2020, ISSN: 0169-2607. DOI: 10.1016/j.cmpb.2020.105604.

[215] A. Raza, A. Mehmood, S. Ullah, M. Ahmad, G. S. Choi, and B.-W. On, "Heartbeat sound signal classification using deep learning," *Sensors*, vol. 19, no. 21, p. 4819, Nov. 2019, ISSN: 1424-8220. DOI: 10.3390/s19214819.

[216] C. Potes, S. Parvaneh, A. Rahman, and B. Conroy, "Ensemble of feature:based and deep learning:based classifiers for detection of abnormal heart sounds," in *2016 Computing in Cardiology Conference (CinC)*, ser. CinC2016, Computing in Cardiology, Sep. 2016. DOI: 10.22489/cinc.2016.182-399.

[217] H. Ryu, J. Park, and H. Shin, "Classification of heart sound recordings using convolution neural network," in *2016 Computing in Cardiology Conference (CinC)*, 2016, pp. 1153–1156.

[218] B. Xiao, Y. Xu, X. Bi, *et al.*, "Follow the sound of children's heart: A deep-learning-based computer-aided pediatric chds diagnosis system," *IEEE Internet of Things Journal*, vol. 7, no. 3, pp. 1994–2004, Mar. 2020, ISSN: 2372-2541. DOI: 10.1109/jiot.2019.2961132.

[219] A. I. Humayun, S. Ghaffarzadegan, M. I. Ansari, Z. Feng, and T. Hasan, "Towards domain invariant heart sound abnormality detection using learnable filterbanks," *IEEE Journal of Biomedical and Health Informatics*, vol. 24, no. 8, pp. 2189–2198, Aug. 2020, ISSN: 2168-2208. DOI: 10.1109/jbhi.2020.2970252.

[220] A. I. Humayun, S. Ghaffarzadegan, Z. Feng, and T. Hasan, "Learning front-end filter-bank parameters using convolutional neural networks for abnormal heart sound detection," in *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, IEEE, Jul. 2018. DOI: 10.1109/embc.2018.8512578.

[221] B. Xiao, Y. Xu, X. Bi, J. Zhang, and X. Ma, "Heart sounds classification using a novel 1-d convolutional neural network with extremely low parameter consumption," *Neurocomputing*, vol. 392, pp. 153–159, Jun. 2020, ISSN: 0925-2312. DOI: 10.1016/j.neucom.2018.09.101.

[222] N. Baghel, M. K. Dutta, and R. Burget, "Automatic diagnosis of multiple cardiac diseases from PCG signals using convolutional neural network," *Computer Methods and Programs in Biomedicine*, vol. 197, p. 105 750, Dec. 2020, ISSN: 0169-2607. DOI: 10.1016/j.cmpb.2020.105750.

[223] T.-c. I. Yang and H. Hsieh, "Classification of acoustic physiological signals based on deep learning neural networks with augmented features," in *2016 Computing in Cardiology Conference (CinC)*, 2016, pp. 569–572.

[224] F. Li, M. Liu, Y. Zhao, *et al.*, "Feature extraction and classification of heart sound using 1d convolutional neural networks," *EURASIP Journal on Advances in Signal Processing*, vol. 2019, no. 1, Dec. 2019, ISSN: 1687-6180. DOI: 10.1186/s13634-019-0651-3.

[225] G. D. Clifford, C. Liu, B. Moody, *et al.*, "Recent advances in heart sound analysis," *Physiological Measurement*, vol. 38, no. 8, E10–E25, Aug. 2017. DOI: 10.1088/1361-6579/aa7ec8.

[226] D. K. Plati, E. E. Tripoliti, A. Bechlioulis, *et al.*, "A machine learning approach for chronic heart failure diagnosis," *Diagnostics*, vol. 11, no. 10, p. 1863, Oct. 2021, ISSN: 2075-4418. DOI: 10.3390/diagnostics11101863.

[227] Y. Zheng and X. Guo, "Identification of chronic heart failure using linear and nonlinear analysis of heart sound," in *2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, IEEE, Jul. 2017. DOI: 10.1109/embc.2017.8037877.

[228] M. Gjoreski, M. Simjanoska, A. Gradisek, A. Peterlin, M. Gams, and G. Poglajen, "Chronic heart failure detection from heart sounds using a stack of machine-learning classifiers," in *2017 International Conference on Intelligent Environments (IE)*, IEEE, Aug. 2017. DOI: 10.1109/ie.2017.19.

[229] S. Gao, Y. Zheng, and X. Guo, "Gated recurrent unit-based heart sound analysis for heart failure screening," *BioMedical Engineering OnLine*, vol. 19, no. 1, Jan. 2020, ISSN: 1475-925X. DOI: 10.1186/s12938-020-0747-x.

[230] Y. Zheng, X. Guo, Y. Wang, J. Qin, and F. Lv, "A multi-scale and multi-domain heart sound feature-based machine learning model for acc/aha heart failure stage classification," *Physiological Measurement*, vol. 43, no. 6, p. 065 002, Jun. 2022, ISSN: 1361-6579. DOI: 10.1088/1361-6579/ac6d40.

[231] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[232] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, *et al.*, *Generative adversarial networks*, 2014. DOI: 10.48550/ARXIV.1406.2661.

[233] D. P. Kingma and M. Welling, *Auto-encoding variational bayes*, 2013. DOI: 10.48550/ARXIV.1312.6114.

[234] B. K. Iwana and S. Uchida, "An empirical survey of data augmentation for time series classification with neural networks," *PLOS ONE*, vol. 16, no. 7, F. Schwenker, Ed., e0254841, Jul. 2021, ISSN: 1932-6203. DOI: 10.1371/journal.pone.0254841.

[235] A. Mishra, J. Q. Yip, and E. S. Chng, "Codec data augmentation for time-domain heart sound classification," in *2023 10th International Conference on Advanced Informatics: Concept, Theory and Application (ICAICTA)*, IEEE, Oct. 2023. DOI: 10.1109/icaicta59291.2023.10390292.

[236] Y. Jeong, J. Kim, D. Kim, J. Kim, and K. Lee, "Methods for improving deep learning-based cardiac auscultation accuracy: Data augmentation and data generalization," *Applied Sciences*, vol. 11, no. 10, p. 4544, May 2021, ISSN: 2076-3417. DOI: 10.3390/app11104544.

[237] Y. Al-Issa and A. M. Alqudah, "A lightweight hybrid deep learning system for cardiac valvular disease classification," *Scientific Reports*, vol. 12, no. 1, Aug. 2022, ISSN: 2045-2322. DOI: 10.1038/s41598-022-18293-7.

[238] ". Lu, J. Beatriz Yip, T. Steigleder, *et al.*, "A lightweight robust approach for automatic heart murmurs and clinical outcomes classification from phonocardiogram recordings," in *2022 Computing in Cardiology Conference (CinC)*, ser. CinC2022, Computing in Cardiology, Dec. 2022. DOI: 10.22489/cinc.2022.165.

[239] T. DeVries and G. W. Taylor, *Improved regularization of convolutional neural networks with Cutout*, 2017. DOI: 10.48550/ARXIV.1708.04552.

[240] M. H. Asmare, F. Woldehanna, L. Janssens, and B. Vanrumste, "Rheumatic heart disease detection using deep learning from spectro-temporal representation of unsegmented heart sounds," in *2020 42nd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, IEEE, Jul. 2020. DOI: 10.1109/embc44109.2020.9176544.

[241] Littmann Electronic Stethoscopes 3M United States. "Littmann electronic stethoscopes." (2024), [Online]. Available: https://www.littmann.com/3M/en_US/p/ (visited on 05/28/2024).

[242] Eko. "Eko DUO ECG + digital stethoscope." (2024), [Online]. Available: https://www.ekohealth.com/products/core-500-digital-stethoscope?variant=39999867879520 (visited on 05/28/2024).

[243] Baron Medical Supply Inc., *3M littmann advance electronic stethoscope model 3200, choose color*. [Online]. Available: https://www.baronmedical.com/3m-littmann-advance-electronic-stethoscope-model-3200-choose-color/ (visited on 06/27/2024).

[244] G. D. Clifford, C. Liu, D. B. Springer, *et al.*, "Classification of normal/abnormal heart sound recordings: The PhysioNet/Computing in Cardiology Challenge 2016," *2016 Computing in Cardiology Conference (CinC)*, pp. 609–612, 2016.

[245] S. McGee, *Evidence-based physical diagnosis*, en, 5th ed. Philadelphia, PA: Elsevier - Health Sciences Division, Aug. 2021, pp. 327–332.

[246] *MATLAB version 9.10.0 (R2021a)*, The Mathworks, Inc., Natick, Massachusetts, 2021.

[247] J. Robert, M. Webbie, *et al.*, *Pydub*, 2018. [Online]. Available: http://pydub.com/.

[248] C. Shannon, "Communication in the presence of noise," *Proceedings of the IRE*, vol. 37, no. 1, pp. 10–21, Jan. 1949. DOI: 10.1109/jrproc.1949.232969.

[249] B. McFee, M. McVicar, D. Faronbi, *et al.*, *Librosa/librosa: 0.10.2.post1*, 2024.

[250] J. O. Smith, *Mathematics of the Discrete Fourier Transform (DFT)*. W3K Publishing, 2007, ISBN: 978-0-9745607-4-8. [Online]. Available: %7Bhttp://www.w3k.org/books/%7D.

[251] P. Virtanen, R. Gommers, T. E. Oliphant, *et al.*, "SciPy 1.0: Fundamental algorithms for scientific computing in Python," *Nature Methods*, vol. 17, no. 3, pp. 261–272, Feb. 2020, ISSN: 1548-7105. DOI: 10.1038/s41592-019-0686-2.

[252] C. R. Harris, K. J. Millman, S. J. van der Walt, *et al.*, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 2020, ISSN: 1476-4687. DOI: 10.1038/s41586-020-2649-2.

[253] T. Giannakopoulos and A. Pikrakis, *Introduction to Audio Analysis: A MATLAB Approach*. Elsevier, 2014, ISBN: 9780080993881. DOI: 10.1016/c2012-0-03524-7.

[254] H. Zhang, S. Krooswyk, and J. Ou, "Chapter 2 - PCB design for signal integrity," in *High Speed Digital Design*, H. Zhang, S. Krooswyk, and J. Ou, Eds., Boston: Morgan Kaufmann, 2015, pp. 27–115, ISBN: 978-0-12-418663-7. DOI: https://doi.org/10.1016/B978-0-12-418663-7.00002-2.

[255] Z. K. Abdul and A. K. Al-Talabani, "Mel frequency cepstral coefficient and its applications: A review," *IEEE Access*, vol. 10, pp. 122 136–122 158, 2022. DOI: 10.1109/access.2022.3223444.

[256] D.-N. Jiang, L. Lu, H. Zhang, J. Tao, and L. Cai, "Music type classification by spectral contrast feature," *Proceedings. IEEE International Conference on Multimedia and Expo*, vol. 1, pp. 113–116, 2002.

[257] S. Dubnov, "Generalization of spectral flatness measure for non-Gaussian linear processes," *IEEE Signal Processing Letters*, vol. 11, no. 8, pp. 698–701, Aug. 2004, ISSN: 1070-9908. DOI: 10.1109/lsp.2004.831663.

[258] R. Merry, *Wavelet theory and applications: a literature study* (DCT rapporten), English. Technische Universiteit Eindhoven, 2005, DCT 2005.053.

[259] G. Lee, R. Gommers, F. Waselewski, K. Wohlfahrt, and A. O'Leary, "Pywavelets: A python package for wavelet analysis," *Journal of Open Source Software*, vol. 4, no. 36, p. 1237, Apr. 2019, ISSN: 2475-9066. DOI: 10.21105/joss.01237.

[260] F. Pedregosa, G. Varoquaux, A. Gramfort, *et al.*, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, Jan. 2012.

[261] G. Ke, Q. Meng, T. Finley, *et al.*, "LightGBM: A highly efficient gradient boosting decision tree," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, ser. NIPS'17, Long Beach, California, USA: Curran Associates Inc., 2017, pp. 3149–3157, ISBN: 9781510860964.

[262] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "Mixup: Beyond empirical risk minimization," in *International Conference on Learning Representations*, 2018.

[263] D. Zou, Y. Cao, Y. Li, and Q. Gu, *The benefits of mixup for feature learning*, 2023. arXiv: 2303.08433 [cs.LG]. [Online]. Available: https://arxiv.org/abs/2303.08433.

[264] T. T. Um, F. M. J. Pfister, D. Pichler, *et al.*, "Data augmentation of wearable sensor data for parkinson's disease monitoring using convolutional neural networks," in *Proceedings of the 19th ACM International Conference on Multimodal Interaction*, ser. ICMI '17, ACM, Nov. 2017. DOI: 10.1145/3136755.3136817.

[265] V. Verma, A. Lamb, C. Beckham, *et al.*, "Manifold Mixup: Better representations by interpolating hidden states," in *Proceedings of the 36th International Conference on Machine Learning*, K. Chaudhuri and R. Salakhutdinov, Eds., ser. Proceedings of Machine Learning Research, vol. 97, PMLR, Jun. 2019, pp. 6438–6447.

[266] K. Aggarwal and J. Srivastava, *Embarrassingly simple MixUp for time-series*, 2023. DOI: 10.48550/ARXIV.2304.04271.

[267] M. Arslan, M. Guzel, M. Demirci, and S. Ozdemir, "SMOTE and Gaussian noise based sensor data augmentation," in *2019 4th International Conference on Computer Science and Engineering (UBMK)*, IEEE, Sep. 2019. DOI: 10.1109/ubmk.2019.8907003.

[268] A. B. Carlson, *Communication systems: Solutions manual*, 2nd ed. New York, NY: McGraw-Hill, Dec. 1975.

[269] D. S. Park, W. Chan, Y. Zhang, *et al.*, "SpecAugment: A simple data augmentation method for automatic speech recognition," in *Interspeech 2019*, ser. interspeech_2019, ISCA, Sep. 2019. DOI: 10.21437/interspeech.2019-2680.

[270] T. DeVries and G. W. Taylor, "Improved regularization of convolutional neural networks with cutout," 2017.

[271] K. He, X. Zhang, S. Ren, and J. Sun, *Deep residual learning for image recognition*, 2015. DOI: 10.48550/ARXIV.1512.03385. [Online]. Available: https://arxiv.org/abs/1512.03385.

[272] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, San Diega, CA, USA, 2015.

[273] L. N. Smith and N. Topin, "Super-convergence: Very fast training of neural networks using large learning rates," in *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications*, T. Pham, Ed., Baltimore, United States: SPIE, May 2019.

[274] D. Susič, G. Poglajen, and A. Gradišek, "Identification of decompensation episodes in chronic heart failure patients based solely on heart sounds," *Frontiers in Cardiovascular Medicine*, vol. 9, Nov. 2022, ISSN: 2297-055X. DOI: 10.3389/fcvm.2022.1009821.

[275] D. Susič, G. Poglajen, and A. Gradišek, "Machine learning models for detection of decompensation in chronic heart failure using heart sounds," in *Workshops at 18th International Conference on Intelligent Environments (IE2022)*, 2022, pp. 340–349.

[276] W. J. Conover, *Practical Nonparametric Statistics* (Wiley Series in Probability and Statistics), en, 3rd ed. Nashville, TN: John Wiley & Sons, Dec. 1998.

[277] D. Susič, A. Gradišek, and M. Gams, "PCGmix: A data-augmentation method for heart-sound classification," *IEEE Journal of Biomedical and Health Informatics*, pp. 1–12, 2024, ISSN: 2168-2208. DOI: 10.1109/jbhi.2024.3458430.

[278] D. Susič and M. Gams, "Time-series cutmix data augmentation for heart sound classification using neural networks," in *Slovenian Conference on Artificial Intelligence: Proceedings of the 26th International Multiconference Information Society - IS*, 2023, pp. 7–10.

[279] X. Zou, Z. Wang, Q. Li, and W. Sheng, "Integration of residual network and convolutional neural network along with various activation functions and global pooling for time series classification," *Neurocomputing*, vol. 367, pp. 39–45, Nov. 2019, ISSN: 0925-2312. DOI: 10.1016/j.neucom.2019.08.023. [Online]. Available: http://dx.doi.org/10.1016/j.neucom.2019.08.023.

[280] I. Oguiza, *Tsai - a state-of-the-art deep learning library for time series and sequential data*, Github, 2023. [Online]. Available: https://github.com/timeseriesAI/tsai.

[281] H. Guo, Y. Mao, and R. Zhang, *MixUp as locally linear out-of-manifold regularization*, 2018. arXiv: 1809.02499 [cs.LG]. [Online]. Available: https://arxiv.org/abs/1809.02499.

[282] K. L. Hoffman, M. Padberg, and G. Rinaldi, "Traveling salesman problem," in *Encyclopedia of Operations Research and Management Science*. Springer US, 2013, pp. 1573–1578, ISBN: 9781441911537. DOI: 10.1007/978-1-4419-1153-7_1068.

[283] D. Shintyakov, *Tsp-solver2 0.4.1*, https://github.com/dmishin/tsp-solver, 2020.

[284] F. Goulart, *Python_tsp 0.4.1*, https://github.com/fillipe-gsm/python-tsp, 2024.

[285] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual explanations from deep networks via gradient-based localization," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 618–626. DOI: 10.1109/ICCV.2017.74.

[286] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, *Learning deep features for discriminative localization*, 2015. arXiv: 1512.04150 [cs.CV].

[287] L. Wang, H. Lu, X. Ruan, and M.-H. Yang, "Deep networks for saliency detection via local estimation and global search," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 3183–3192. DOI: 10.1109/CVPR.2015.7298938.

[288] R. Zhao, W. Ouyang, H. Li, and X. Wang, "Saliency detection by multi-context deep learning," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, Jun. 2015. DOI: 10.1109/cvpr.2015.7298731. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2015.7298731.

[289] K. Simonyan, A. Vedaldi, and A. Zisserman, *Deep inside convolutional networks: Visualising image classification models and saliency maps*, 2014. arXiv: 1312.6034 [cs.CV].

[290] A. Javeed, S. U. Khan, L. Ali, S. Ali, Y. Imrana, and A. Rahman, "Machine learning-based automated diagnostic systems developed for heart failure prediction using different types of data modalities: A systematic review and future directions," *Computational and Mathematical Methods in Medicine*, vol. 2022, M. Z. Asghar, Ed., pp. 1–30, Feb. 2022, ISSN: 1748-670X. DOI: 10.1155/2022/9288452. [Online]. Available: http://dx.doi.org/10.1155/2022/9288452.

[291] K. M. Tsiouris, V. D. Tsakanikas, D. Gatsios, and D. I. Fotiadis, "A review of virtual coaching systems in healthcare: Closing the loop with real-time feedback," *Frontiers in Digital Health*, vol. 2, Sep. 2020, ISSN: 2673-253X. DOI: 10.3389/fdgth.2020.567502. [Online]. Available: http://dx.doi.org/10.3389/fdgth.2020.567502.

Bibliography

Publications Related to the Thesis

Journal Articles

D. Susič, A. Gradišek, and M. Gams, "PCGmix: A data-augmentation method for heart-sound classification," *IEEE Journal of Biomedical and Health Informatics*, pp. 1–12, 2024, ISSN: 2168-2208. DOI: 10.1109/jbhi.2024.3458430.

D. Susič, G. Poglajen, and A. Gradišek, "Identification of decompensation episodes in chronic heart failure patients based solely on heart sounds," *Frontiers in Cardiovascular Medicine*, vol. 9, Nov. 2022, ISSN: 2297-055X. DOI: 10.3389/fcvm.2022.1009821.

Conference Papers

D. Susič and M. Gams, "Time-series cutmix data augmentation for heart sound classification using neural networks," in *Slovenian Conference on Artificial Intelligence: Proceedings of the 26th International Multiconference Information Society - IS*, 2023, pp. 7–10.

D. Susič, G. Poglajen, and A. Gradišek, "Machine learning models for detection of decompensation in chronic heart failure using heart sounds," in *Workshops at 18th International Conference on Intelligent Environments (IE2022)*, 2022, pp. 340–349.

Other Publications

Journal Articles

D. Susič, S. Syed-Abdul, E. Dovgan, J. Jonnagaddala, and A. Gradišek, "Artificial intelligence based personalized predictive survival among colorectal cancer patients," *Computer Methods and Programs in Biomedicine*, vol. 231, p. 107 435, Apr. 2023, ISSN: 0169-2607. DOI: 10.1016/j.cmpb.2023.107435.

S. Fajfer and D. Susič, "Colored scalar mediated nucleon decays to an invisible fermion," *Physical Review D*, vol. 103, no. 5, Mar. 2021, ISSN: 2470-0029. DOI: 10.1103/physrevd.103.055012.

V. Janko, N. Reščič, A. Vodopija, *et al.*, "Optimizing non-pharmaceutical intervention strategies against COVID-19 using artificial intelligence," *Frontiers in Public Health*, vol. 11, Feb. 2023, ISSN: 2296-2565. DOI: 10.3389/fpubh.2023.1073581.

D. Susič, L. Bombač Tavčar, M. Lučovnik, H. Hrobat, L. Gornik, and A. Gradišek, "Well-being forecasting in postpartum anemia patients," *Healthcare*, vol. 11, no. 12, p. 1694, Jun. 2023, ISSN: 2227-9032. DOI: 10.3390/healthcare11121694.

T. Kompara, J. Perš, D. Susič, and M. Gams, "A one-dimensional non-intrusive and privacy-preserving identification system for households," *Electronics*, vol. 10, no. 5, p. 559, Feb. 2021, ISSN: 2079-9292. DOI: 10.3390/electronics10050559.

D. Susič, J. Tomšič, and M. Gams, "Ranking effectiveness of non-pharmaceutical interventions against COVID-19: A review," *Informatica*, vol. 46, no. 4, Dec. 2022, ISSN: 0350-5596. DOI: 10.31449/inf.v46i4.4181.

Conference Papers

T. Šket, D. Susič, C. Galen, *et al.*, "Identifying bumblebee buzzes using neural networks," in *Slovenian Conference on Artificial Intelligence: Proceedings of the 26th International Multiconference Information Society - IS*, 2023, pp. 20–23.

Ž. Kolar, B. Erzar, N. Čelan, *et al.*, "Peak detection for classification of number of axles," in *Slovenian Conference on Artificial Intelligence: Proceedings of the 25th International Multiconference Information Society - IS*, 2022, pp. 19–22.

D. Susič, B. Erzar, N. Čelan, *et al.*, "Vehicle axle distance detection from time-series signals using machine learning," in *Slovenian Conference on Artificial Intelligence: Proceedings of the 25th International Multiconference Information Society - IS*, 2022, pp. 39–42.

D. Susič, L. Bombač Tavčar, H. Hrobat, L. Gornik, M. Lučovnik, and A. Gradišek, "Detection of postpartum anemia using machine learning," in *Pervasive Health and Smart Sensing: Proceedings of the 25th International Multiconference Information Society - IS*, 2022, pp. 40–43.

J. Tomšič, D. Susič, and M. Gams, "Effectiveness of non-pharmaceutical interventions in handling the COVID-19 pandemic: Review of related studies," in *Insieme Project Workshop: Proceedings of the 24th International Multiconference Information Society - IS*, 2021, pp. 46–51.

M. Luštrek, N. Reščič, V. Janko, *et al.*, "Forecasting trends and optimizing the intervention plans against the COVID-19 pandemic: The XPRIZE competition and beyond," in *Insieme Project Workshop: Proceedings of the 24th International Multiconference Information Society - IS*, 2021, pp. 52–55.

A. Marinko, M. Žaucer, D. Susič, and M. Gams, "Sensitivity of expected civilization longevity models," in *Cognitive Science: Proceedings of the 24th International Multiconference Information Society - IS*, 2021, pp. 22–25.

D. Susič, "Daily covid-19 deaths prediction for slovenia," in *Slovenian Conference on Artificial Intelligence: Proceedings of the 24th International Multiconference Information Society - IS*, 2021, pp. 43–46.

N. Reščič, V. Janko, D. Susič, *et al.*, "Finding efficient intervention plans against covid-19 : Second place at the XPRIZE pandemic response challenge," in *International Conference ETAI 2021*, 2021, pp. 139–143.

Biography

Candidate David Susič, born on December 2, 1995, obtained his Bachelor's degree in Physics from the Faculty of Mathematics and Physics at the University of Ljubljana in 2017. He continued his studies at the same faculty, earning his Master's degree in Physics in 2020. During his final year of the Master's program, he was honored with the Dean's Award, recognizing his excellence as one of the top students.

In 2020, Susič joined the Department of Intelligent Systems at the Jožef Stefan Institute and enrolled in the doctoral program of Information and Communication Technologies at the Jožef Stefan International Postgraduate School. His PhD research is conducted under the supervision of Asst. Prof. Anton Gradišek and co-supervision of Prof. Matjaž Gams. His research focuses on machine learning and artificial intelligence, addressing challenges in medicine, healthcare, and various other domains. Throughout his PhD studies, he actively participated in numerous projects at the Department of Intelligent Systems, contributing to a diverse range of publications.

A notable achievement during his PhD journey was being part of the team that secured second place in the XPRIZE $500k Pandemic Response Challenge, which took place during the COVID-19 pandemic in the years 2020 and 2021.