POLITECNICO DI TORINO

Master's Degree in Data Science and Engineering

Master's Degree Thesis

Deep Learning Analysis of ECG Signals for Early Myocardial Infarction Detection

Supervisors

Ing. Vincenzo Randazzo

Company Tutor

Ing. Luca Manunta

Candidate

Alessandro Masala

A.A. 2023/2024

Preface

The preface to this thesis announces the trip into the world of cardiac health monitoring via machine learning and electrocardiogram (ECG) analysis. In an era characterized by the proliferation of artificial intelligence and its integration into numerous sectors, the medical industry stands out as a beacon of innovation and revolution. In this setting, the diagnosis of myocardial infarction, a serious cardiac ailment, emerges as a focus for research and development efforts to improve diagnostic capabilities. The following pages provide a thorough examination of the construction of a machine learning pipeline designed for the identification of myocardial infarction using ECG data. The journey is divided into painstaking steps, beginning with ECG data preparation and ending with sophisticated neural network training, all of which are carefully coordinated. The use of the PTB-XL Database, a repository teeming with significant clinical data that serves as the foundation for our scientific efforts, is vital to our undertaking. Powered by Python and TensorFlow's versatility and resilience, our implementation aims to bridge the gap between theory and practice, opening the way for tangible advances in cardiac health monitoring. The diverse range of neural network topologies exemplifies the spirit of innovation and exploration that drives our pursuit for excellence. Each architecture represents a node in the intricate web of possibilities, providing distinct insights and opportunities for growth in the field of automated myocardial infarction diagnosis. As we embark on our adventure, let us approach the challenges ahead with unflinching determination and unbounded curiosity. We aim to not only improve the art of detecting myocardial infarctions, but also to push the frontiers of what is possible in the field of cardiac health monitoring. May this prelude serve as a compass, leading us through the maze of discovery and invention to a future in which healthcare knows no bounds and every heartbeat echoes with the promise of hope and healing.
With deep anticipation,

Alessandro Masala

Abstract

This thesis investigates the development of a robust machine learning pipeline for the early detection of myocardial infarction (MI) using electrocardiogram (ECG) data. Recognizing MI promptly is vital for effective treatment, potentially saving lives by reducing the mortality and morbidity associated with this condition. The research leverages the comprehensive PTB-XL dataset, which includes a vast array of clinical ECG recordings annotated by medical professionals, making it an ideal resource for training and validating the proposed neural network models. The methodology encompasses several stages, starting with the preprocessing of ECG data to enhance signal quality and remove noise and artifacts. Various neural network architectures were explored, including Convolutional Neural Networks (CNNs), and LSTM networks, to determine the most effective model for ECG analysis. The study employed data augmentation techniques such as noise addition, time warping, and signal shifting to address the issue of overfitting and improve the generalizability of the models. Training was executed using Python and TensorFlow, with an emphasis on optimizing the neural networks through meticulous hyperparameter tuning and the application of advanced optimizers like Adam and SGD. The effectiveness of different loss functions and learning rate schedulers was also evaluated to enhance model training dynamics. The models exhibited high accuracy and precision in detecting MI from ECG signals. The LSTM model, in particular, showed a significant improvement in performance, achieving an accuracy of 96%, under augmented data conditions. These results highlight the potential of advanced neural networks in the automatic detection of cardiac event. The machine learning pipeline developed in this thesis marks an advancement in the automated detection of myocardial infarction using ECG data. The findings suggest that such models can be integrated into clinical settings to provide real-time, accurate assessments of MI, thereby facilitating prompt medical intervention. Future work will focus on ECG image analysis and introducing techniques such as fine-tuning to potentially increase the pipeline's utility in clinical settings.

Acknowledgments

It is my duty to dedicate this section of the document to the people who have contributed, with their tireless support, to its realization.

I would like to extend my heartfelt thanks to Dr. Vincenzo Randazzo, thesis advisor, for the support, advice, and collaboration in the completion of the thesis.

I thank S.T.E.P. S.R.L. for giving me the opportunity to carry out my thesis work in an interesting and dynamic environment, which allowed me to challenge myself and gain an experience that will be valuable for my future. In particular, I would like to thank Engineer Luca Manunta for guiding and supporting me during the most important phase of my academic journey.

I am infinitely grateful to my parents and my brother who have always supported me, endorsing every decision I made, starting from the choice of my academic path.

To my girlfiend Clara, I want to express my deepest gratitude for your love, support, and understanding throughout this journey. Your presence in my life brings me endless joy and comfort. Thank you for being my rock, my confidante, and my source of inspiration. Your love has made every moment brighter and every challenge easier to overcome. I am truly blessed to have you by my side.

A heartfelt thank you to everyone.

Contents

Chapter 1

Introduction

Cardiovascular diseases (CVDs) were the leading cause of deaths [1] until few years ago, when cancer surpassed them in high-income nations [2]. There are plenty of different CVDs, but one of the most critic is the Myocardial Infarction (MI), also known as heart attack [3]. Myocardial Infarction occurs when there is a sudden blockage of blood flow to the heart, due to the obstruction of a coronary artery by a blood clot. This obstruction cause an interruption of oxigen and nutrients to the heart, leading to a tissue damage or, in the most critical cases, to the death. This pathology can occur in various forms, particularly we can distinguish between ST-segment elevation myocardial infarction (STEMI) and Non-ST-elevation myocardial infarction (NSTEMI); the main difference is that the STEMI is a critical condition that needs quick actions, while the NSTEMI is harder to detect and the treatment is indeed different. The aim of this project is to identify STEMI cases based on electrocardiogram (ECG) analysis, since it is the most common detection system.

1.1 Aim of the Project

In the past years, the development in deep learning methods applied to the medicine are noteworthy. The ECG classification systems in the last years showed results similar to the cardiologists for specific tasks [4][5], however the road to outperform the medical opinion is still long and tortuous [6]. Hence, the thesis aims to develop a machine learning pipeline for the fast and accurate detection of myocardial infarction through the analysis of electrocardiogram (ECG) data, having in mind as a primary goal the possibility of saving humal lives.

1.2 Methodology

The key steps of the projects are the preprocessing the ECG data and training neural networks to achieve accurate recognition of myocardial infarction using the PTB-XL dataset as the primary dataset. The implementation will be in Python and TensorFlow, using various neural networks architectures for training and testing the models. The objective is to create a codebase that is not only effective for the current study but also adaptable and reusable for future developments in cardiac health monitoring. An overview of the method can be seen in Figure 1.1, and represents the following steps:

1. Preprocessing
    - *Data Cleaning*: Remove any artifacts or noise from the raw ECG data to improve signal quality
    - *Annotation Handling*: Extract and structure the annotations provided in the PTB-XL dataset for effective utilization during model training
2. Split
    - *Train-Test-Val Split*: Split the dataset into training (80%), validation (10%), and testing (10%) subsets to ensure the model is trained and evaluated on separate data.
    - *Stratification*: Ensure that each subset has a proportional representation of different classes
3. Normalization
    - *Z-score Normalization*: Apply Z-score normalization to the features, scaling them to have a mean of 0 and a standard deviation of 1. This helps in speeding up the training process and achieving better convergence
    - *Handling Outliers*: Address any outliers in the data that might skew the normalization process
4. Neural Networks
    - *Model Architectures*: Experiment with different neural network architectures such as CNNs (Convolutional Neural Networks), RNNs (Recurrent Neural Networks), and hybrid models

- *Hyperparameter Tuning*: Optimize model parameters such as learning rate, batch size, number of layers, and neurons to improve performance

5. Evaluation

    - *Metrics*: Utilize metrics such as accuracy, precision, recall, and F1-score to evaluate the models
    - *Cross-validation*: Perform k-fold cross-validation to ensure robustness and generalizability of the model

6. Classification

    - *Predict New Data*: Use the trained model to predict myocardial infarction on unseen data

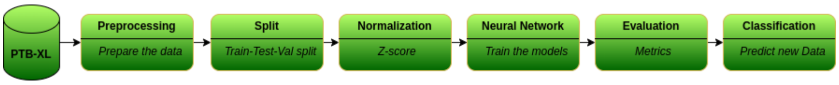

Figure 1.1: General overview of the methodology

Chapter 2

Electrocardiography

Electrocardiography (EGC) [7] is undoubtedly the most famous field in cardiology, providing useful information about the electrical activity of the heart. ECG had, during the years, the ability to completely change the understanding of the electrical impulses of the heart, providing valuable information about cardiac function, rhythm and structure.

2.1 The Heart

The ECG is crucial to understand the physiological processes of the heart. In fact, its electrical activity starts from specialized cells, that coordinates the rhythmic contraction and relaxation of the cardiac fibers. Moreover, this electrical activity is important for maintaining the heart's pumping function and allowing efficient blood circulation.

2.1.1 Structure and Physiology

The heart is specifically designed to pump blood thoughout the body and, along with it, the oxigen and nutrients. Composed primarily of muscle tissue, the heart consists of four chambers: two atria and two ventricles (Figure 2.1).

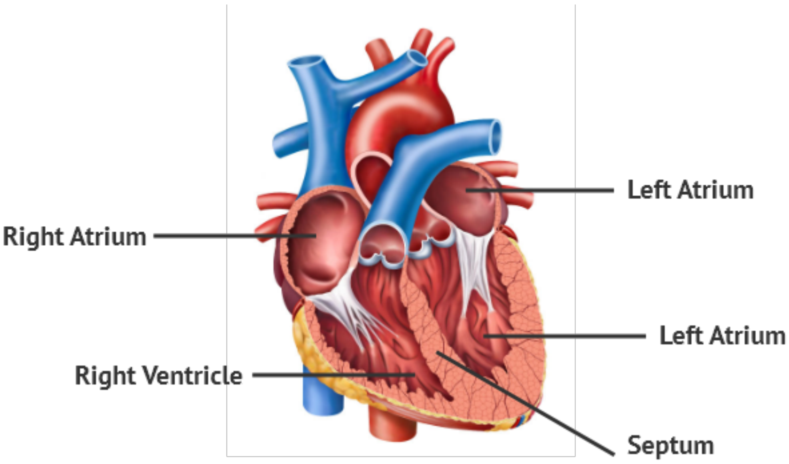

Figure 2.1: Heart Atria and Ventricules [8]

The heart structure [9] is directly linked to the underlying cardiovascular physiology [10]. The kernel of the heart is the myocardium, a specialized muscle tissue that generate the necessary force to pump the blood through the circulatory system. It is divided in two layers: the outer epicardium and the inner epicardium, that are responsible of the heart's contractile function. Between the epicardium layers there is the myocardial layer, that is made of cardiac muscle cells (cardiomyocytes), and is responsible of the rapid transmission of electrical signals and coordinated contraction. The heart's chambers are separated by septa, or walls, that allows the blood flow to stay undirectional. The internal septa divides the right and left atria, while the interventricular septum separates the right and left ventricles. Inside this complicated structure, it is also possible to find the valves, responsible for regulating blood flow within the heart. The valves are divided based on their functions, in particular the atrioventricular valves (AV), represented by the tricuspid and mitral, prevent backflow of blood from ventricles to atria during ventricular contraction. On the other hand, the semilunar valves, the pulmonary and aortic, prevent blood to flow back into the ventricules after contraction. Figure 2.2 visually show the heart complete structure.

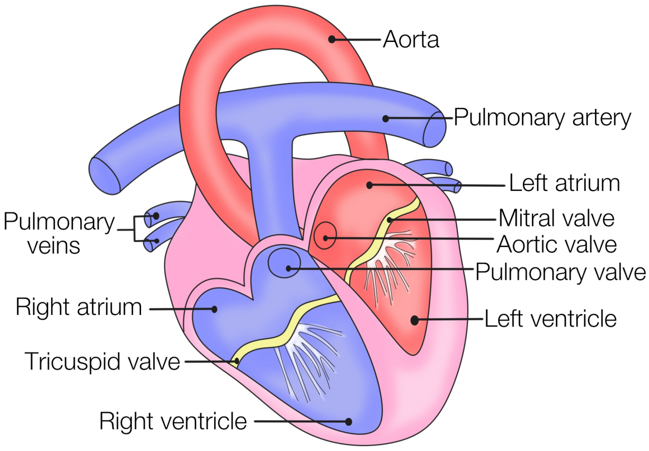

Figure 2.2: Heart complete structure [11]

2.2 Electrocardiograms

ECG [12] analysis, is one of the most important task for cardiologists since it gives them the possibility to detect cardiac abnormalities in real-time. Furthermore, this allow fast interventions and treatment strategies in order to increase the survival possibility of the patients. Moreover, continuous ECG monitoring can facilitate the detection of transient or intermittent abnormalities. Finally, ECG plays a crucial role in invasive cardiac procedures, ensuring safety and efficacy. Even if ECG is not the most recent techniques, it is indispensable in modern cardiology practice, significantly contributing to the diagnosis and management of cardiovascular diseases.

2.2.1 Wave Strucure and Description

The electrocardiogram (ECG) waveform [13] is a visual representation of the electrical activity of the heart. Understanding the structure of the ECG waveform is essential to understand how the abnormalities are detected and handled. Figure 2.3 provide a visual representation of a standard wave.

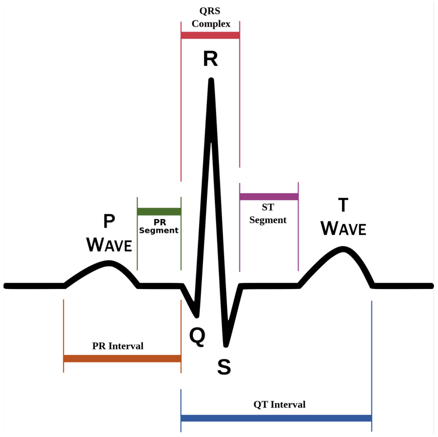

Figure 2.3: Standard ECG wave [14]

The cardiac cycle starts with a P wave, which represents atrial depolarization. Moreover, this is the reflection of the electrical impulses flowing through the atria as they contract to pump the blood flow into the ventricles. After the P wave, it normally appears the QRS complex, that represents ventricular depolarization and, particularly, the spread of electrical activity through the ventricles, leading to their contraction and subsequent ejection of blood into the pulmonary artery and aorta. Then the ST segment, that appears as a flat, isoelectric line that represents ventricular depolarization and repolarization. In detecting the myocardial infarction the ST segment is crucial, since every modification of this segment may indicate myocardial infarction, ischemia, or injury. Finally the T wave emerges, representing the ventricular repolarization.

2.2.2 Electrodes and Leads Placement

In a standard 12-lead electrocardiogram (ECG) [15], electrodes are strategically placed (as shown in Figure 2.4) on specific locations of the patient's body to capture electrical signals originating from the heart. This specific placement procedure is made to visualize the heart's electrical activity from different angles and perspectives. The leads can be divided in two main groups: the limb leads (RA, LA, RL, LL) that provide information from the frontal plane, and the precordial leads (V1-V6) that provide information of the horizontal plane. The two groups of leads allow the cardiologists to have a complete overview of the electrical activity of the heart.

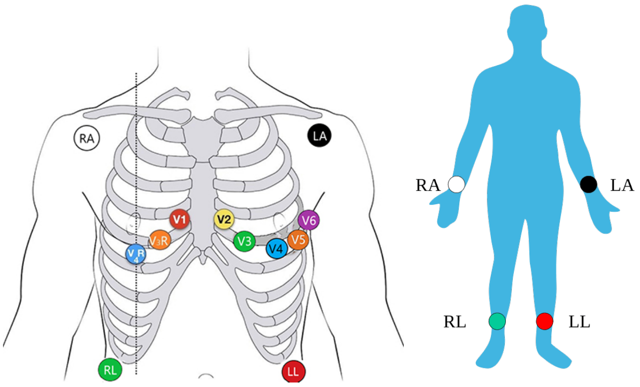

Figure 2.4: 12-lead ECG Placement [16][17]

2.2.3 Functionality

Heart's functionality is a series of complex procedures that coordinates rhythmic contraction and drive blood circulation [18]. The electrical implulses travel among different sections of the heart; particularly, the origin is the sinoatrial (SA) node, also known as the heart's natural pacemaker [19]. Then the impulse moves into the atrioventricular (AV) node and finally reaches the ventricles. This flow coordinates the sequences of depolarization and repolarization phases of the cardiac cycles that are determined not only by the electrical impulses but also by a series of chemical procedures. Furthermore, in the depolarization phase, the membrane becomes less negative because of the influx of sodium ions into cardiac cells; this starts the contraction of the cardiac muscle fibers. Moreover, this procedure is visible using the ECG, and it is linked to the P wave (atrial depolarization). This procedure is followed by the ventricular depolarization, represented by the QRS complex on ECG, since the sodium ions flood into ventricular cells, triggering their contraction. After the depolarization phase, the repolarization occurs; represented by the T wave on ECG. In this case, potassium ions exit the cardiac cells restoring cardiac membrane potensial and preparing the heart for the next cycle of electrical activity. A representation of the cardiac cycle is represented in Figure 2.5, which enhances understanding and appreciation of cardiovascular physiology.

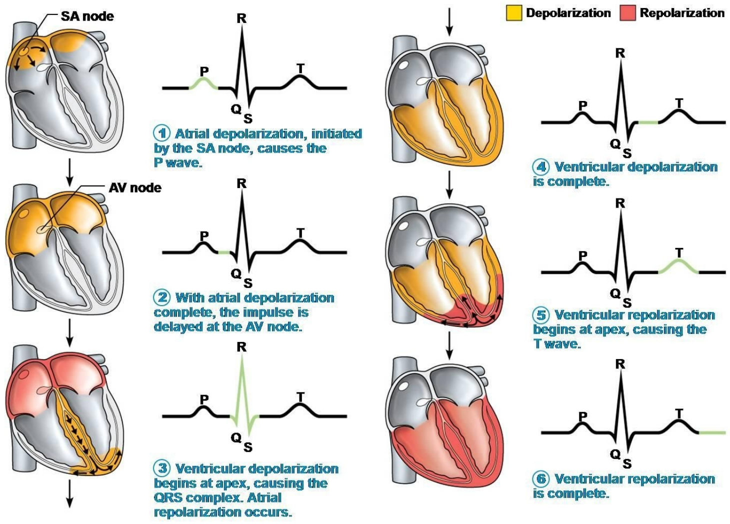
SA node
R
P
T
Q
S
① Atrial depolarization, initiated by the SA node, causes the P wave.
AV node
R
P
T
Q
S
② With atrial depolarization complete, the impulse is delayed at the AV node.
R
P
T
Q
S
③ Ventricular depolarization begins at apex, causing the QRS complex. Atrial repolarization occurs.
🟨 Depolarization
🟥 Repolarization
R
P
T
Q
S
④ Ventricular depolarization is complete.
R
P
T
Q
S
⑤ Ventricular repolarization begins at apex, causing the T wave.
R
P
T
Q
S
⑥ Ventricular repolarization is complete.

Figure 2.5: Polarization-Depolarization cycle and related ECG [20]

2.2.4 Interpretation

ECG interpretation [21] might be seem as a simple task, since it only requires the analysis of a wave; however, it requires a comprehensive understanding of cardiac physiology, electrical conduction pathways and waveform components. The analysis involves several key stages that allow the cardiologist to determine the heart health or the presence of abnormalities: shape, duration, amplitude, segment, rate, rhythm. The procedure consists in analyzing the regularity of the waves between consecutive cardiac cycles to detect abnormalities. Furthermore, irregularities in the different segments of the wave may indicate the presence of different types of abnormalities. Particularly, irregularities between two different R segments may indicate the presence of arrhythmias, while irregularities in the P segment may suggest atrial enlargement, conduction disturbances or ectopic atrial foci. Moreover, QRS or U abormalities may indicate electrolite imbalance or bundle branch blocks, while T segment problems may indicate ventricular hypertrophy or intercranial pathology. Lastly, the ST segment is evaluated for deviations from the baseline and it is the most important analysis for the objective of this project. In fact, the ST elevation may indicate myocardial infaction, myocardial ischemia or pericarditis, whereas the ST depression may suggest myocardial ischemia or digitalis effect.

Chapter 3

Myocardial Infarction

Myocardial Infarction (MI) [22], also known as heart attack, is a critical medical condition, characterized by sudden interruption of blood flow to a part of the heart, causing tissue damage or death. This condition is among the ones with the greates number of deaths worldwide, pointing out the necessity for prompt recognition, diagnosis and intervention. There are two types of myocardial infarction: STEMI with ST segment elevation and NSTEMI without ST segment elevation [23]. The distinction and correct diagnosis of the type of myocardial infaction is crucial in determining the intervention strategy. The focus of this project will be on STEMI cases, given their distinct clinical presentation, urgent need for reperfusion therapy, and potential for significant myocardial damage if left untreated.

3.1 ST-elevation Myocardial Infarction

STEMI [24] is characterized by ST segment elevation on the ECG analysis, representing complete occlusion of a coronary artery and imminent myocardial injury. This type of myocardial infarction has some common symptomps (Figure 3.1): chest pain, shortness of breath, nausea and diaphoresis. Therefore, a prompt intervation is required, usually through percutaneous coronary intervention (PCI) or thrombolytic therapy in order to restore blood flow to the heart and minimize cardiac damage.

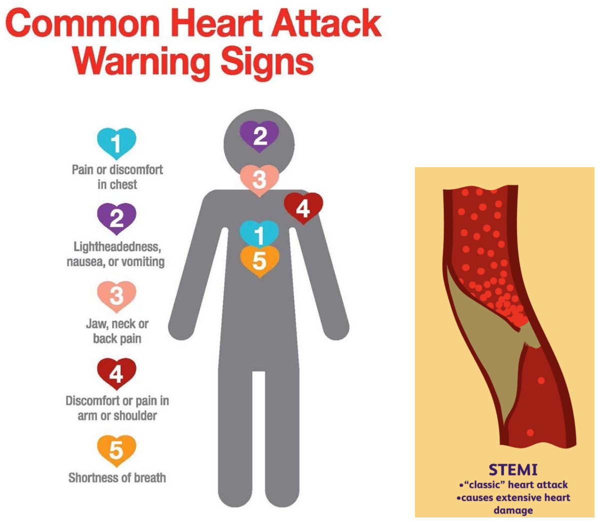
Common Heart Attack Warning Signs
1 Pain or discomfort in chest
2 Lightheadedness, nausea, or vomiting
3 Jaw, neck or back pain
4 Discomfort or pain in arm or shoulder
5 Shortness of breath
2
3
4
1
5
STEMI
•"classic" heart attack
•causes extensive heart damage

Figure 3.1: STEMI common symptoms and occlusion [25][26]

3.1.1 Etiology

The primary cause of STEMI [24] is the rapture of an atherosclerotic plaque within a coronary artery, that creates a thrombus that occludes blood flow (Figure 3.1). In the majority of STEMI cases, the thrombus is characterized by the accumulation of a plaque composed of cholesterol, fatty deposit or inflammatory cells.

3.1.2 Epidemiology

STEMI remains a leading cause of morbidity and mortality globally [27], accounting for a significant burden of cardiovascular disease. While the number of STEMI cases is decreased in high-income countries due to advancements in preventive strategies, it continues to represent a substantial public health challenge, particularly in low- and middle-income regions with limited access to healthcare resources.

3.1.3 Patophysiology

The pathophysiology of STEMI [28] involves the abrupt interruption of blood flow to a segment of the myocardium, leading to ischemia and subsequent necrosis if left untreated. The occlusion of a coronary artery results in an imbalance between myocardial oxygen supply and demand, triggering a series of events that end up in cellular injury and death.

3.1.4 Diagnosis

Diagnosing a STEMI case [29] mainly involves ECG interpretation, since it represents the most rapid way to identify ischemic changes and allow prompt intervention. The ECG representation of ST segment elevation (represented in Figure 3.2) is typically observed in at least two consecutive leads and is often linked by a reciprocal changes such as ST segment depression in leads opposite to the infarcted area.

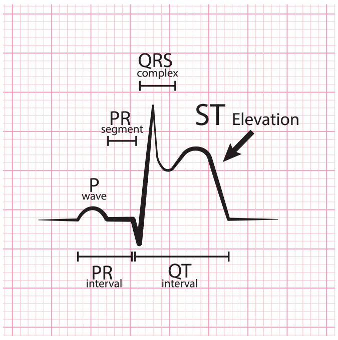

Figure 3.2: ST-segment elevation [30]

Once the STEMI is diagnosed, it is also possible to determine its location based on the on the lead that presents the irregularities. For instance, ST elevation in the anterior leads (V1-V4) suggests involvement in the left anterior descending (LAD) coronary artery, whereas elevation in the inferior leads (II, III, aVF) may indicate the occlusion of the right coronary artery (RCA) or left circumflex artery (LCx). Therefore, reciprocal changes (ST depression) in the leads V1-V3 with posterior ST elevation, suggest involvement of the posterior descending artery (PDA). The recognition time in STEMI is crucial for the survival of the patient and current guidelines suggest ECG analysis within 10 minutes from the patient arrivals for individuals presenting the inquired symptoms. Repeat ECGs may be performed at regular intervals to monitor for dynamic changes in the ST-segment and to assess response to treatment. In Figure 3.3 the visual representation of the evolution of a STEMI.

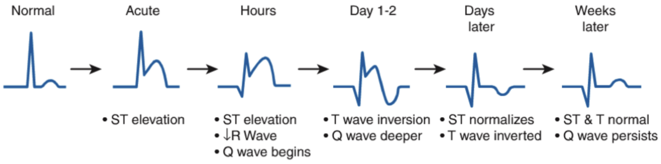

Figure 3.3: STEMI ECG evolution over time [31]

3.1.5 Treatment

Immediate reperfusion therapy [32] is the most commont treatment for STEMI, aimed at restoring blood flow to the ischemic myocardium and saving as much tissue as possible. Primary percutaneous coronary intervention (PCI) is the preferred method of reperfusion when available, as it offers the best outcomes compared to fibrinolytic therapy [33]. Other medical therapies such as antiplatelet agents, anticoagulants, beta-blockers, and statins are also administered to optimize outcomes and prevent recurrent ischemic events.

Chapter 4

Dataset

Neural Networks need a consistent number of records to be effective and to effectively outperform healthcare professionals; however, it is not a simple task to find a dataset composed of ECGs carefully annotated and organized that can be used to perform a machine learning pipeline. Therefore, the project analysis will focus on the PTB-XL dataset, which contains a multitude of ECGs annotated by professionals, that is one of the most complete dataset in terms of annotations and organization.

4.1 The PTB-XL Dataset

The PTB-XL dataset [34][35] is an exceptional resource for various research related to the cardiac health analysis. It contains a total of 21,799 records from 18,869 patients, containing clinical 12-lead ECGs [36]. One of the key features of the PTB-XL dataset is its comprehensive annotation process; in fact, each ECG record has been annotated by up to two cardiologists, who assign the diagnostic class, form and annotated the rhythm aspects. The annotations follow a standard, the SCP-ECG, that ensures clarity and consistency in interpretation. Furthermore, the dataset contains metadata regarding demographics, infarction characteristics and likelihoods for diagnostic class.

4.1.1 Data Acquisition

The data acquisition process described for the PTB-XL [36] dataset represents a robust and systematic approach to collect and organize electrocardiogram (ECG) data for research and clinical purposes. Firstly, the recording and storage of raw signal data in a proprietary compressed format highlight the importance of preserving data integrity while managing storage requirements efficiently. By providing the standard set of 12 leads with reference electrodes on the right arm, the dataset ensures consistency and compatibility with established ECG recording conventions, facilitating comparisons and analyses across different records. The inclusion of general metadata such as age, sex, weight, and height of patients adds valuable contextual information to the dataset; In fact, not only enables researchers to stratify and analyze the dataset based on patient characteristics but also allows for exploration of potential correlations between demographic factors and ECG findings. The annotation process of each ECG record has been made with a report string and converted into standardized SCP-ECG statements. Providing structured annotations, the dataset enables researchers to efficiently search for specific ECG characteristics and supports the development and validation of automated ECG interpretation algorithms. Moreover, the extraction of additional clinical information such as heart axis and infarction stadium enhances the clinical relevance of the dataset, enabling more comprehensive analyses of cardiac conditions and outcomes. The validation of records by cardiologists and technical experts underscores the commitment to ensuring the accuracy and reliability of the dataset. By subjecting a significant fraction of records to independent validation, potential errors or inconsistencies in the annotation process can be identified and addressed, thereby enhancing the overall quality and trustworthiness of the dataset. Additionally, validation by technical experts focusing on signal characteristics helps ensure that recorded signals are free from artifacts or anomalies that could confound analyses or interpretations.

4.1.2 Data Protection

The PTB-XL dataset demonstrates a particular concern to data privacy and security, [36] essential considerations in contemporary medical research, especially when dealing with sensitive patient information such as electrocardiogram (ECG) data. By assigning unique identifiers (ecg_id and patient_id) to ECGs and patients, the dataset ensures efficient data management while preserving patient anonymity. This practice aligns with best practices in data protection, allowing researchers to analyze the data without compromising patient confidentiality. Moreover, the pseudonymization of personal information in the metadata, including the names of validating cardiologists, nurses, and recording sites, underscores a commitment to safeguarding the identities of individuals involved in the data collection process. This approach minimizes the risk of unauthorized access or unintended disclosure of sensitive information, thereby maintaining patient trust and confidentiality. Furthermore, the random offset applied to ECG recording dates adds an additional layer of protection against re-identification, making it more challenging for adversaries to link specific records to individuals. This practice aligns with principles of data minimization and privacy by design, mitigating the risk of unintended disclosure and enhancing the overall security of the dataset. Adherence to the SCP-ECG standard for annotating ECG records ensures consistency and interoperability across the dataset, facilitating collaboration and comparison among researchers and healthcare professionals. By following established standards, the dataset promotes transparency and reproducibility in ECG analysis, thereby advancing the development of automated interpretation algorithms and diagnostic tools. In conclusion, the PTB-XL dataset sets a high standard for data privacy and security in medical research, incorporating measures to protect patient confidentiality while enabling valuable insights into cardiovascular health. By prioritizing privacy and adhering to established standards, the dataset serves as a valuable resource for advancing scientific understanding and improving patient care in the field of cardiology.

4.1.3 Data Description

The PTB-XL dataset represents a substantial and meticulously curated collection of clinical 12-lead ECG records, [36] offering insights into cardiac health and pathology across a diverse patient population. With 21,799 ECG records from 18,869 patients, the dataset provides a rich repository of cardiac data spanning a wide range of ages, genders, and clinical conditions. The distribution of patients by gender reflects a balanced representation, while the age distribution covers the entire spectrum from infancy to older adulthood, with a median age of 62 years and an interquartile range of 22 years. One of the remarkable features of the PTB-XL dataset is its comprehensive coverage of various cardiac pathologies alongside a significant proportion of healthy control samples. This diversity in pathology enriches the dataset and facilitates the exploration of complex relationships between different cardiac conditions and their corresponding ECG manifestations. The inclusion of conditions such as myocardial infarction, ST/T changes, conduction disturbances, and hypertrophy ensures that researchers have access to a wide array of clinical scenarios for analysis and interpretation. The waveform files stored in WaveForm DataBase (WFDB) format provide high-fidelity ECG data with 16-bit precision and a sampling frequency of 500Hz, ensuring detailed signal representation suitable for indepth analysis and algorithm development. Additionally, the availability of downsampled versions of the waveform data at a sampling frequency of 100Hz enhances accessibility and computational efficiency for users with varying computational resources. Metadata stored in ptbxl_database.csv offers a wealth of information essential for contextualizing and interpreting the ECG records. From demographic details such as age, sex, height, and weight to recording metadata including nurse, site, device, and recording date, the dataset provides a comprehensive framework for understanding the clinical context surrounding each ECG recording.

Chapter 5

Preprocessing

The strength of the PTB-XL dataset is not only the consistent number of records, but also in the diversity of pathologies and the inclusion of healthy control samples. With a distribution of diagnoses comprehending Normal ECG, Myocardial Infarction (MI), ST/T Change, Conduction Disturbance, and Hypertrophy, the dataset offers a complete overview over cardiac conditions. In this context, the focus will be on particular subsets of the data, such as normal and MI cases, for targeted analysis and algorithm development.

5.1 Dataset Inspection and Preparation

Dataset inspection and preparation are key stages in a machine learning pipeline, since the process may lead to accurate model predictions. Dataset inspection involves a comprehensive examination of the data's characteristics, including size, structure, quality, and distributions of variables. Therefore, dataset preparation techniques such as cleaning and transformtions ensure that the data is suitable for analysis and modeling tasks.

5.1.1 Demographic Characteristics

Characteristics such as sex, age, height, and weight are often referred to as demographic or biometric attributes. These attributes provide essential information about individuals within a dataset and are commonly used in various analyses. In this project, demographic characteristics play a crucial role in comprehending the impact of Myocardial Infarction (MI) on the population and in identifying and addressing potential outliers if they exist. Particularly, the focus is on the distributions of sex and age to gain insights into how MI affects different demographic segments. Moreover, investigating the distributions of sex and age allows us to identify any anomalies or outliers that may skew the analysis results,

Sex

Understanding the sex distribution in myocardial infarction (MI) is important for comprehending the epidemiology and clinical manifestations of this cardiovascular condition. The analysis of sex distribution provides insights into potential disparities in disease prevalence, risk factors, and outcomes between males and females. Moreover, the sex distribution in MI might lead to tailored prevention and intervention strategies Historically, studies have consistently shown that males have a higher prevalence of MI compared to females, particularly at younger ages [37]. This discrepancy in prevalence is often attributed to biological differences, including hormonal influences, genetic predispositions, and lifestyle factors such as smoking and dietary habits. Additionally, males tend to exhibit certain risk factors more prominently, such as hypertension and hypercholesterolemia, which contribute to the increased prevalence of MI in this demographic group. This discrepancy is also visible in the dataset and particularly in Figure 5.1.

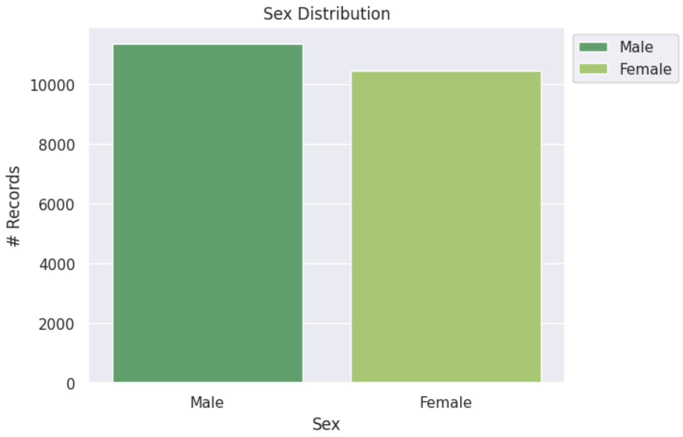

Figure 5.1: Sex distribution in the dataset.

Age

Age plays an important role in myocardial infarction (MI), with its incidence rising with the advance of the age. Analyzing age distribution within MI datasets offers insights into the epidemiology and risk factors associated with this cardiovascular condition. Research consistently demonstrates that MI becomes increasingly prevalent with age, with individuals over the age of 65 [38] facing a significantly higher risk compared to younger age groups. This age-related trend can be attributed to various factors, including cumulative exposure to cardiovascular risk factors, progressive arterial aging, and age-related changes in cardiac structure and function. Moreover, older adults often present with a higher prevalence of comorbidities such as hypertension, diabetes, and obesity, which further exacerbate their susceptibility to MI. Figure 5.2 shows the age distribution in the dataset showing how the age bucket between 61 and 80 is the most populous one.

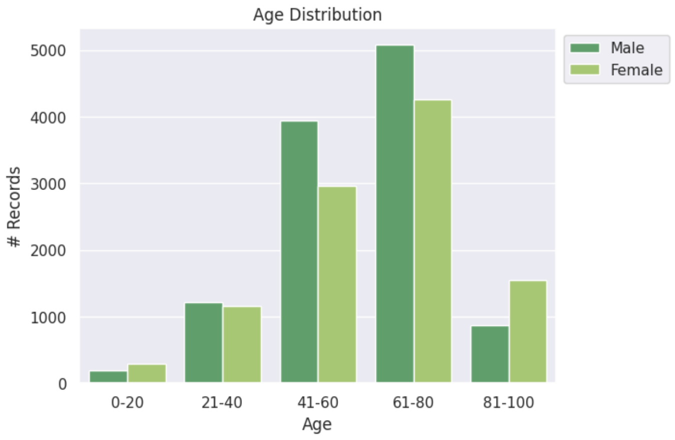

Figure 5.2: Age distribution in the dataset.

In order to find age outliers the violin plot, that is a powerful visualization tool, has been used. This plot combines the features of a box plot and a kernel density plot, providing a comprehensive representation of the distribution of ages among individuals with MI. In the context of MI research, age outliers may represent individuals who experience the condition at unusually young or old ages. Additionally, age outliers may warrant further investigation to assess their clinical significance, potential contributing factors, and implications for treatment and prevention strategies. Figure 5.3 represents the violin plot of the age distribution.

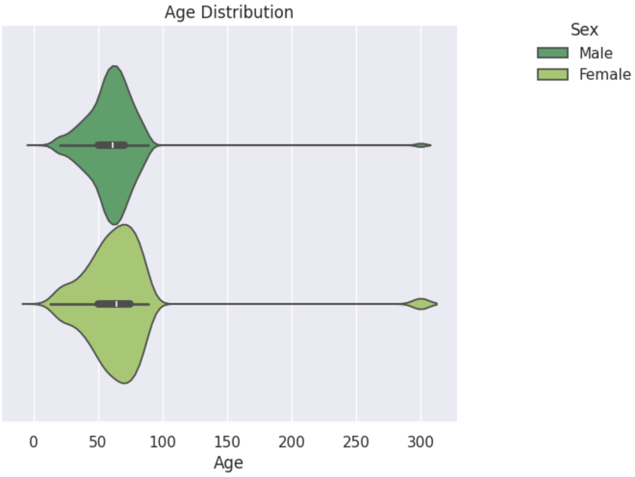

Figure 5.3: Violin plot of the distribution of the age.

It's evident from the violin plot that there are outliers within the dataset, notably individuals are aged over 300 years, an occurrence that defies biological plausibility. These extreme outliers, probably deriving from annotation mistakes, challenge the integrity of the dataset and underscore the importance of rigorous data validation and cleaning procedures. Therefore, these outliers has been removed from the dataset (Figure 5.4).

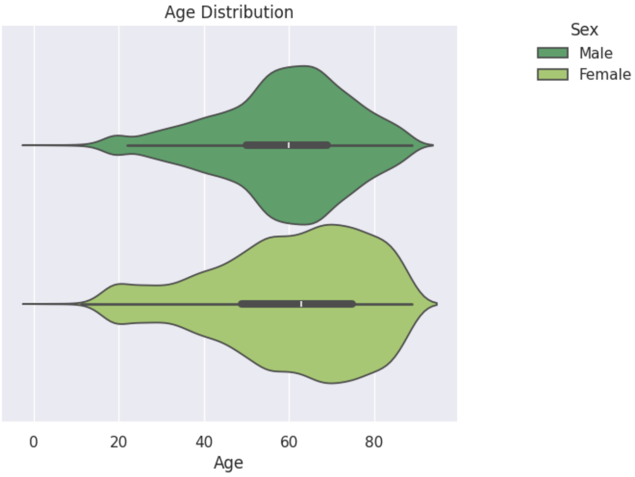

Figure 5.4: Violin plot of the age distribution after the removal of the outliers.

5.1.2 Electrodes Problems and Burst Noise

In the context of biomedical signal processing, records contaminated by electrode problems and burst noise often require careful consideration regarding their inclusion in the dataset. Due to the potential distortion and inaccuracies introduced by the noise, removing such records from the dataset is a common practice. By eliminating records affected by electrode problems and burst noise, researchers can mitigate the risk of erroneous conclusions, improve the quality of data-driven insights, and enhance the validity of research findings.

Electrodes Problems

Electrode problems encompass issues such as poor electrode-skin contact, electrode detachment, or electrode impedance mismatches, which can result in signal distortion, artifact contamination, and reduced signal quality. These problems often manifest as baseline drift, signal saturation, or irregular waveform shapes, compromising the accuracy and reliability of physiological measurements. As shown in Table 5.1 the number of records with some electrodes problems is really tiny. Therefore, the records are removed from the dataset.

No Problems	Malfunctions
21479	29

Table 5.1: Number of records with and without electrodes problems.

Burst Noise

Burst noise, refers to short-duration, high-amplitude spikes or disturbances superimposed on the signal, and might be caused by various sources, including electromagnetic interference, power line fluctuations, or electronic equipment malfunctions. In biomedical signal processing, burst noise can obscure important physiological information, introduce false alarms, and impair the interpretability of signals.

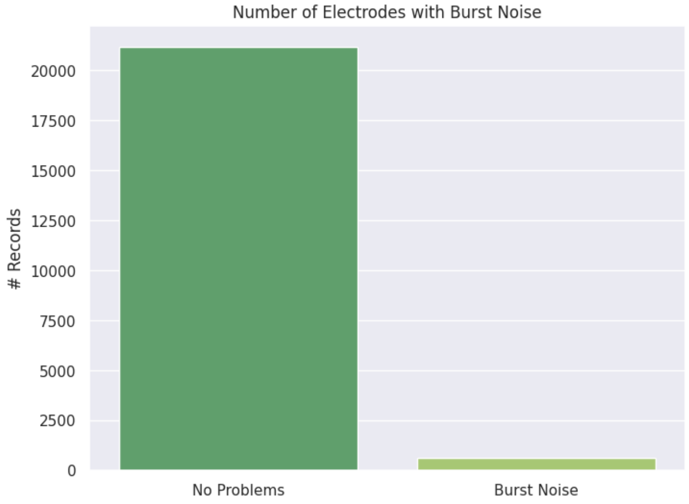

Figure 5.5: Number of records with and without burst noise.

As shown in Figure 5.5 and Table 5.2 the number of records with burst noise is not significantly large. Hence, the records are removed from the dataset.

No Problems	Burst Noise
20905	609

Table 5.2: Number of records with and without electrodes problems.

5.1.3 Superclass Distribution

The superclass distribution within the dataset, comprising diagnoses ranging from Normal ECG to various cardiac conditions including Myocardial Infarction (MI), ST/T Change, Conduction Disturbance, and Hypertrophy, offers a rich and diverse perspective on cardiac health (Figure 5.6).

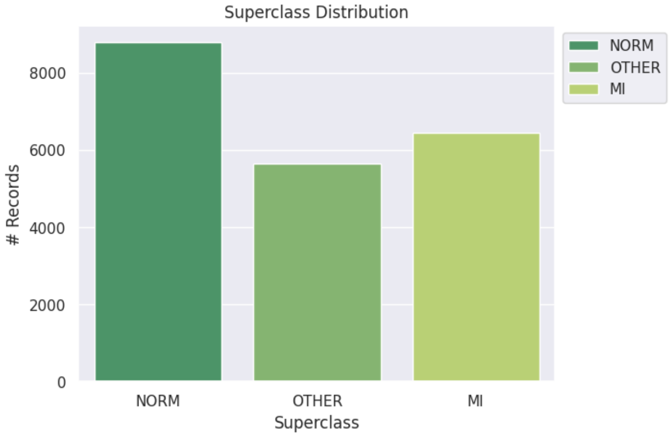

Figure 5.6: Superclass distribution in the dataset.

The aim of the project focus on Myocardial Infaction (MI). Therefore, all the records not labelled as NORM or MI will be discarded and the final dataset will contain only two labels as in Figure 5.7.

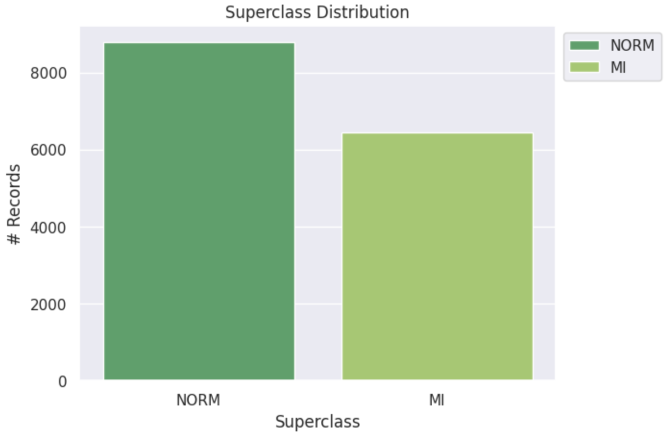

Figure 5.7: Superclass distribution of the diagnostic classes of interest.

5.2 Final Dataset

After meticulous cleaning and inspection, the final dataset emerges as a refined repository of myocardial infarction data, comprising a total of 15,222 records. Within this finalized dataset, 8,780 records represent instances of Normal ECG, while 6,442 records correspond to cases of Myocardial Infarction (MI). These subclass distributions provide valuable insights into the prevalence and distribution of cardiac conditions within the dataset. The significant number of records dedicated to both Normal ECG and MI cases underscores the dataset's representativeness and its potential to yield comprehensive insights into cardiac health and disease processes.

Chapter 6

Train-Test Split

The train-test split methodology is a common practice used to assess the performance of machine learning models. It involves dividing the available dataset into two distinct subsets: the training set and the test set [39]. The training set is utilized to train the model, allowing it to learn patterns and relationships within the data. Meanwhile, the test set serves as a proxy for unseen data, enabling the evaluation of the model's performance on data it hasn't encountered during training. By splitting the dataset into separate training and test sets, it is possible to see how well models generalize to new, unseen data. This process helps to detect overfitting, where the model learns to memorize the training data rather than capturing underlying patterns. Overfitting can lead to poor performance when the model is deployed in real-world scenarios [40].

6.1 Static Dataset Division

In order to build a robust and reliable machine learning model, it's essential to partition the dataset into three distinct subsets: the training set, the validation set, and the test set. This division allows for rigorous model development, evaluation, and fine-tuning, ensuring the model's ability to generalize well to unseen data.

6.1.1 Training Set

The training set constitutes the largest portion of the dataset and serves as the foundation for model development. It is used to train the machine learning model by exposing it to labeled examples of input data along with their corresponding target outputs. During the training process, the model learns to recognize patterns, extract features, and make predictions based on the provided input-output pairs.

6.1.2 Validation Set

The validation set plays a crucial role in the model development pipeline by serving as a mechanism for hyperparameter tuning and model selection. While the training set is used to train the model's parameters, the validation set is employed to assess the model's performance on unseen data and to fine-tune its hyperparameters. By evaluating the model's performance on the validation set, it is possible to iteratively adjust hyperparameters, such as learning rate, regularization strength or change the network architecture, to optimize performance and prevent overfitting.

6.1.3 Test Set

The test set is the final benchmark for evaluating the model's generalization performance. Unlike the training and validation sets, the test set remains untouched during the model development and hyperparameter tuning phases. It contains unseen data that the model has not seen during training or validation. By evaluating the model on the test set, it is possible to assess its ability to generalize to new, unseen data and make unbiased estimates of its performance in real-world scenarios.

6.2 Folds Distribution

The dataset has been labeled and bucketed, by the authors, into ten discrete buckets. The objective is to ensure that each bucket contains an equal distribution of records (as shown in Figure 6.1), in order to prevent biases and skewed representations in the training process. Achieving a balanced distribution across buckets is essential for model training to prevent the dominance of specific classes and ensure comprehensive learning across all categories.

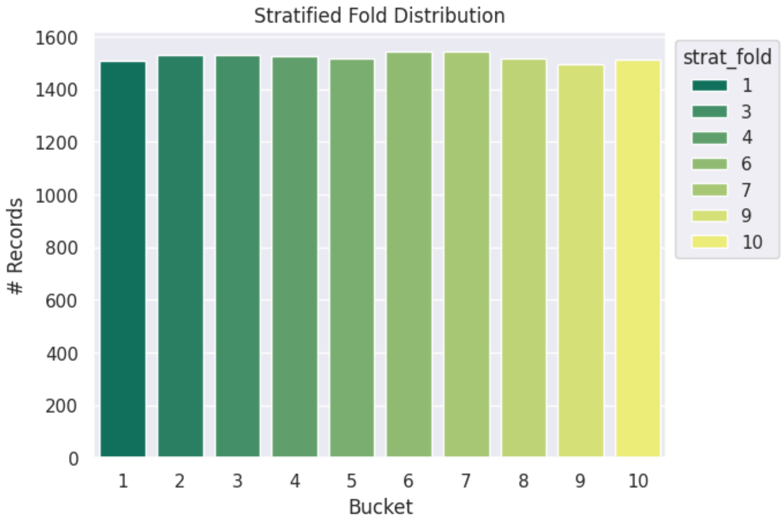

Figure 6.1: Distribution of the fold division.

It's imperative to allocate these buckets strategically to ensure effective model training and validation. Therefore, the classes representation inside each bucket is approximately the same, as represented in Figure 6.2.

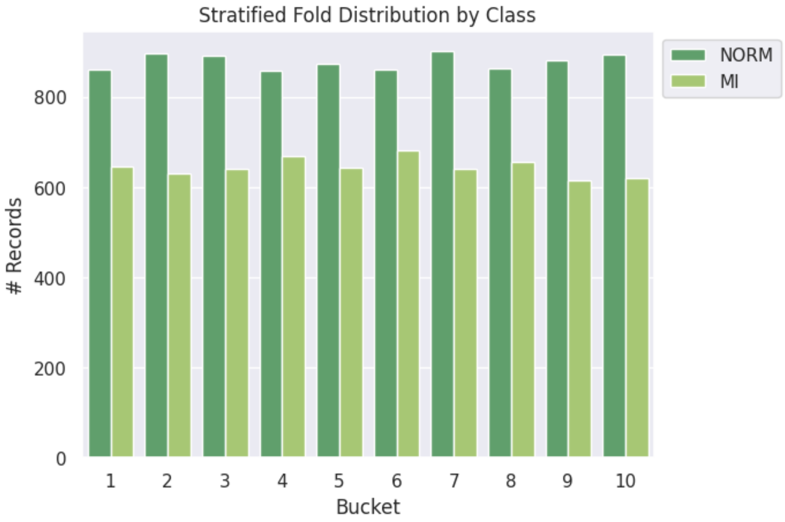

Figure 6.2: Distribution of the fold division by class.

6.2.1 Allocation Strategy

Train

The majority of the dataset, comprising eight out of the ten buckets, will be allocated for training purposes. This allocation allows the model to learn from a diverse range of examples across multiple categories.

Validation

One bucket will be exclusively reserved for validation purposes. This validation set serves as a crucial component in the model development process, enabling the assessment of model performance and the fine-tuning of hyperparameters.

Test

Finally, one bucket will be set aside for testing the trained model. This test set remains untouched during the model development and validation phases, ensuring an unbiased evaluation of the model's performance on unseen data.

6.3 Stratified K Fold

In machine learning, particularly when dealing with classification tasks, ensuring that the distribution of classes remains balanced across different folds of data is crucial. Stratified K-Fold Cross Validation (SKF-CV) [41] is a technique used to achieve this balance, especially when dealing with imbalanced datasets. Stratified K-Fold Cross Validation involves partitioning the dataset into K equal-sized folds while ensuring that each fold maintains the same class distribution as the original dataset. This technique addresses the challenge of maintaining class balance, which is essential for accurate model evaluation, especially in cases where certain classes are underrepresented [42].

6.3.1 Utilization

In the project, Stratified K-Fold Cross Validation is employed to evaluate the performance of the machine learning model across multiple folds of the data.

- Stratified Split: Before applying the Stratified K-Fold technique, the dataset is stratified, meaning that it's divided into training and test sets while preserving the original class distribution.
- Stratified K-Fold Split: Once the dataset is stratified, Stratified K-Fold Cross Validation is applied by splitting the training set into K distinct folds. Each fold contains a balanced representation of classes similar to that in the original training set.

6.3.2 Allocation Strategy

Train

The majority of the dataset, comprising eight out of the ten buckets, will be allocated for training purposes in each fold. This allocation allows the model to learn from a diverse range of examples across multiple categories, fostering robustness and adaptability to various data patterns.

Validation

One bucket will be exclusively reserved for validation purposes in each fold. This validation set serves as a crucial component in the model development process, enabling the assessment of model performance and the fine-tuning of hyperparameters.

Test

Finally, one bucket will be set aside for testing the trained model in each fold. This test set remains untouched during the model development and validation phases, ensuring an unbiased evaluation of the model's performance on unseen data.

6.4 Rationale for Allocation

6.4.1 Prevention of Overfitting

The allocation of distinct training, validation, and test sets helps mitigate the risk of overfitting, where the model learns to memorize training data rather than capturing underlying patterns. A separate validation set facilitates early detection of overfitting and gives the possibility to implement corrective measures as needed.

6.4.2 Evaluative Rigor

The division of the dataset into training, validation, and test sets ensures rigorous evaluation of the model's performance at each stage of the development process. This systematic approach gives the possibility to make informed decisions regarding model architecture, feature selection, and optimization strategies.

Chapter 7

Neural Networks

In this chapter, the architecture and functionalities of neural networks are explored comprehensively. Understanding these elements is fundamental to understand the complexities of neural network models and their applications across diverse domains. Through a systematic examination of neural network architecture and functionalities, it is possible to gain insights into the underlying principles and operational dynamics that drive the learning and predictive capabilities of these powerful computational systems.

7.1 CNN

Convolutional Neural Networks (CNNs) [43] are a class of deep neural networks specifically designed for processing structured grids of data. They have proven to be highly effective in various tasks such as image classification, object detection, and even natural language processing.

7.1.1 Basic CNN

The first CNN architecture (Figure 7.1) is a basic yet effective model for tasks like sequence classification or time-series analysis.

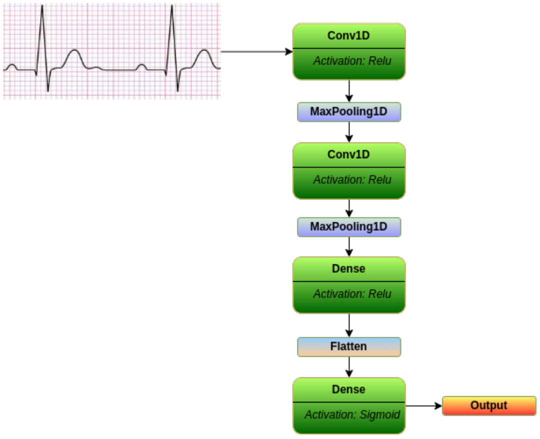

Figure 7.1: Basic CNN architecture.

Input Layer

The model begins with an input layer that receives data in the form of sequences. The input_shape parameter defines the shape of the input data that the model expects.

Convolutional Layers

The network consists of two convolutional layers, each followed by a max-pooling layer [44]. Convolutional layers are fundamental building blocks in CNNs, responsible for learning spatial hierarchies of patterns within the input data [45].

- The first convolutional layer has 64 filters with a kernel size of 3 and uses the ReLU (Rectified Linear Unit) activation function. ReLU introduces non-linearity to the network and helps in learning complex patterns.
- The second convolutional layer has 128 filters with a kernel size of 3 and also utilizes the ReLU activation function.
- Max-pooling layers are inserted after each convolutional layer to downsample the feature maps, reducing the spatial dimensions and computational complexity while retaining the most relevant information.

Flattening Layer

After the convolutional layers, a flattening layer is added to reshape the 2D feature maps into a 1D vector, which can be fed into the fully connected layers.

Fully Connected Layers

Following the flattening layer, there are two fully connected (dense) layers. These layers are responsible for learning high-level features from the representations obtained by the convolutional layers [46].

- The first dense layer has 64 units and uses the ReLU activation function.
- The final dense layer consists of a single neuron with a sigmoid activation function, which outputs a probability indicating the likelihood of a binary classification (0 or 1).

The architecture can be seen in Figure 7.2 [47].

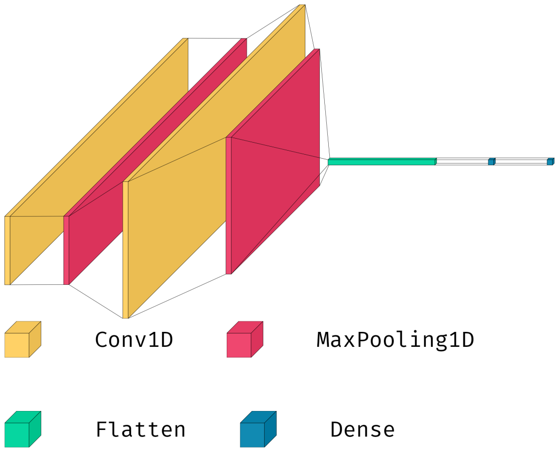

Figure 7.2: Basic CNN structure.

In conclusion, this basic CNN architecture offers a simple yet effective approach for tasks involving sequence data, such as time-series analysis or sequence classification. It leverages convolutional layers to learn hierarchical representations of input sequences and utilizes fully connected layers for high-level feature learning and classification.

7.1.2 Enhanced CNN

The Enhanced Convolutional Neural Network (CNN) architecture (Figure 7.3) is built on the basic principles of CNNs while incorporating additional layers and techniques to improve its performance and robustness.

Input Layer

Similar to the basic CNN, the model starts with an input layer that receives data in the form of sequences. The input_shape parameter defines the shape of the input data [48].

Convolutional Layer

The network includes three sets of convolutional layers, each followed by max-pooling layers.

- The first set consists of two convolutional layers with 64 filters each and a kernel size of 3. Both layers utilize the ReLU activation function.
- The second set follows the same pattern with two convolutional layers having 128 filters each and a kernel size of 3.
- The third set comprises two convolutional layers with 256 filters each and a kernel size of 3.
- Max-pooling layers are inserted after each set of convolutional layers to downsample the feature maps and reduce spatial dimensions.

Dropout Layer

Dropout layers are incorporated to mitigate overfitting, a common issue in deep neural networks. A dropout rate of 0.5 is applied after the first and second sets of convolutional layers, as well as after the first fully connected layer.

Flattening Layer

After the convolutional layers, a flattening layer reshapes the 2D feature maps into a 1D vector as input to the fully connected layers.

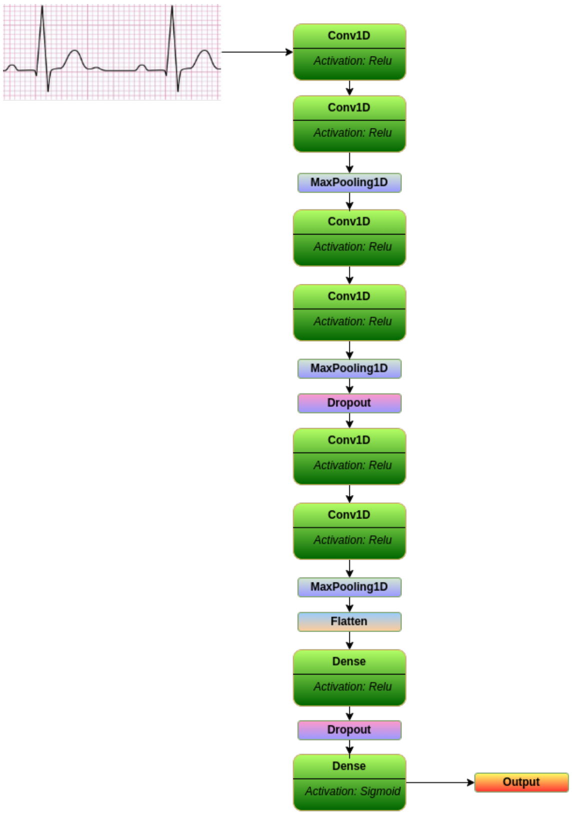

Figure 7.3: Enhanced CNN structure.

Fully Connected Layer

Two fully connected (dense) layers follow the flattening layer.

- The first dense layer has 128 units with the ReLU activation function.
- Another dropout layer with a dropout rate of 0.5 is added after the first dense layer.
- The final dense layer consists of a single neuron with a sigmoid activation function, producing a probability for binary classification tasks.

The Enhanced CNN architecture provides a more sophisticated framework for processing sequential data, offering increased model capacity and robustness compared to the basic CNN. By incorporating additional convolutional layers, dropout regularization, and deeper network structures, the enhanced model is capable of learning more intricate patterns and achieving better generalization on various tasks involving sequential data analysis. The architecture can be seen in Figure 7.4.

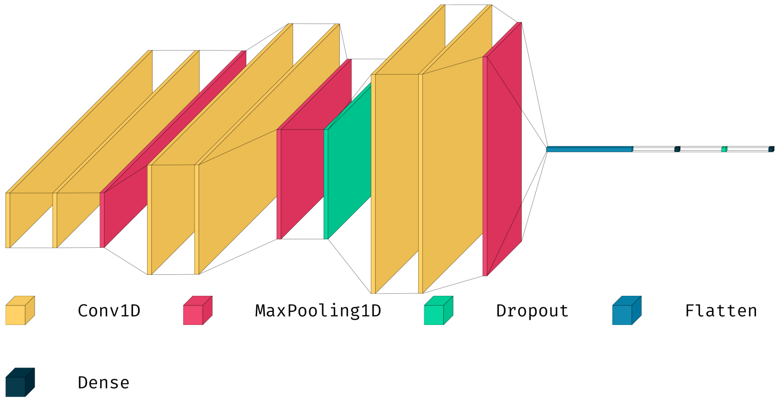

Figure 7.4: Enhanced CNN architecture.

7.2 ConvNetQuake

ConvNetQuake is a convolutional neural network (CNN) [49] architecture designed for seismic data analysis, particularly for earthquake detection and prediction (Figure 7.5). The model utilizes a series of convolutional layers to extract hierarchical features from seismic signals and provides a binary classification output indicating the presence or absence of earthquake activity. However, in this project it will be tested on ECG data analysis.

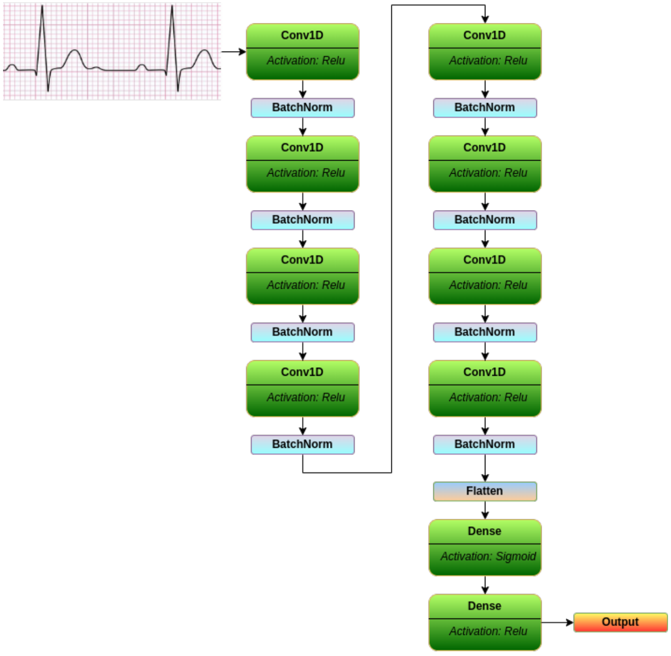

Figure 7.5: ConvNetQuake structure.

Input Layer

The model starts with an input layer, which accepts seismic data in the form of sequences. The input_shape parameter specifies the shape of the input data.

Convolutional Layers

ConvNetQuake employs a sequence of convolutional layers, each equipped with 32 filters and a kernel size of 3. Stride 2 and 'same' padding are employed to effectively reduce spatial dimensions while preserving crucial information. The Rectified Linear Unit (ReLU) activation function follows each convolutional operation, introducing non-linearity and enabling the model to capture intricate patterns within the ECG signals.

Flattening Layer

After the convolutional layers, a flattening layer is introduced to transform the 3D feature maps into a 1D vector. This transformation facilitates seamless connectivity with subsequent dense layers.

Fully Connected Layers

ConvNetQuake incorporates a single dense layer comprising 128 units, accompanied by the ReLU activation function. This layer is instrumental in capturing high-level features from the flattened representations of the ECG signals.

Output Layer

The final layer comprises a single neuron with a sigmoid activation function, producing a binary output indicative of earthquake activity. While ConvNetQuake was initially tailored for seismic data analysis, its architectural principles and deep learning techniques can be repurposed for ECG analysis. Electrocardiograms (ECGs) also involve sequential data with distinct patterns and features that signify cardiac events. By retraining ConvNetQuake on ECG datasets, the model can learn to identify abnormalities, detect arrhythmias, and predict cardiac events. Reusing ConvNetQuake architecture in ECG analysis offers a promising avenue for improving cardiovascular health monitoring and diagnosis. The architecture can be seen in Figure 7.6.

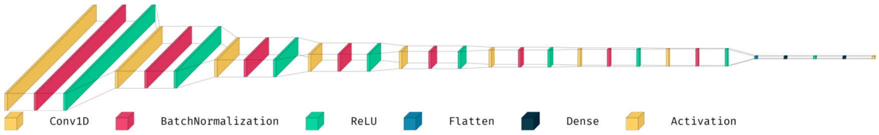

Figure 7.6: ConvNetQuake architecture.

While originally designed for seismic data analysis, ConvNetQuake's architectural principles and deep learning techniques are readily adaptable for ECG analysis tasks. By retraining ConvNetQuake on ECG datasets, the model can effectively identify abnormalities, detect arrhythmias, and predict various cardiac events based on distinctive patterns and features present in ECG signals.

7.3 LSTM

Long Short-Term Memory (LSTM) [50] [51] networks (Figure 7.7) are a type of recurrent neural network (RNN) architecture designed to capture long-term dependencies in sequential data [52]. LSTMs are particularly effective in tasks involving time-series prediction, sequence classification, and natural language processing.

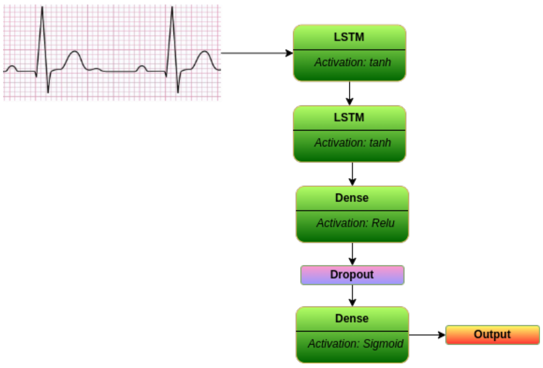

Figure 7.7: LSTM structure.

Input Layer

The model begins with an input layer that receives sequential data in the specified input_shape. In this case, the input data is expected to be in the form of sequences.

LSTM Layers

Two LSTM layers are stacked sequentially:

- The first LSTM layer has 64 units and is configured to return sequences. This allows the LSTM to output sequences instead of a single output, which is useful when stacking LSTM layers.

- The second LSTM layer also has 64 units and is configured to return only the final output of each sequence.

Dense Layers

Following the LSTM layers, there are two dense layers:

- The first dense layer consists of 64 units with a ReLU activation function. ReLU introduces non-linearity, allowing the model to learn complex patterns in the data.

- A dropout layer with a dropout rate of 0.5 is applied after the first dense layer to mitigate overfitting, a common issue in deep neural networks.

- The final dense layer consists of a single neuron with a sigmoid activation function, which produces a probability indicating the likelihood of a binary classification (0 or 1).

The architecture can be seen in Figure 7.8.

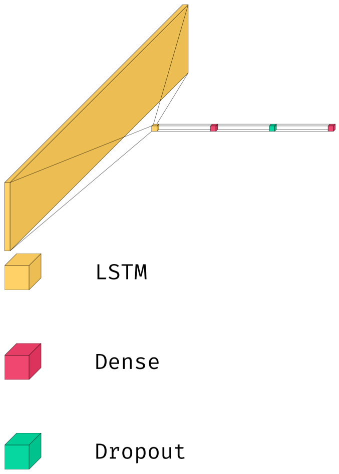

Figure 7.8: LSTM architecture.

The LSTM architecture offers a powerful framework for processing sequential data and making predictions. By leveraging the memory cell structure and gating mechanisms, LSTMs can effectively capture long-term dependencies and learn temporal patterns in the data. The model presented here demonstrates the application of LSTM networks in binary classification tasks, where it learns from sequential input data and produces binary predictions based on the learned patterns.

Chapter 8

Model Training

The training process of machine learning models plays an important role in developing accurate and reliable diagnostic tools. The training pipeline begins with the selection of appropriate deep learning models tailored to handle ECG data. These models encompass a variety of architectures ranging from Convolutional Neural Networks (CNNs) to Long Short-Term Memory networks (LSTMs), each designed to capture key aspects and patterns inherent in ECG waveforms. Furthermore, the training process involves crucial decisions regarding hyperparameters such as learning rate, batch size, and optimizer selection, which directly influence the convergence and performance of the models. The choice of loss functions, optimizers, and learning rate schedulers are meticulously configured to ensure efficient model training and convergence towards optimal solutions. Moreover, the training pipeline incorporates techniques such as data augmentation, which enriches the training dataset by generating synthetic variations of ECG signals. This augmentation process enhances the model's ability to generalize and adapt to diverse patterns present in real-world ECG recordings. Throughout the training phase, comprehensive monitoring and evaluation mechanisms are employed to assess the model's performance and convergence. Techniques such as early stopping and model checkpointing ensure that the training process is efficiently managed, preventing overfitting and ensuring the selection of the best-performing model.

8.1 Data Augmentation

Data augmentation plays a crucial role in improving the robustness and generalization capabilities of machine learning models, especially in domains like electrocardiogram (ECG) analysis. This chapter describes the significance of data augmentation techniques within the context of ECG analysis and explore various methods used to augment ECG data effectively. ECG signals are susceptible to variations caused by factors such as noise, artifacts, and baseline shifts, which can make it challenging for models to learn meaningful patterns. Data augmentation techniques offer a solution by generating diverse and realistic variations of ECG signals, thereby enabling models to learn more robust representations. In the project, various data augmentation techniques are implemented to enhance the performance of ECG analysis models. These techniques (Figure 8.1) are applied to the training data to generate augmented samples, which are then used to train the model alongside the original data.

8.1.1 Adding Noise

Random noise addition introduces variability in the signal, simulating real-world conditions where ECGs may contain noise due to environmental factors or equipment limitations.

8.1.2 Time Warp

Time warping involves shifting the temporal structure of the signal, mimicking slight variations in the duration of cardiac cycles or the pacing of heartbeats.

8.1.3 Signal Shifting

Shifting the signal along the time axis helps model to learn invariant features irrespective of the signal's temporal position, making the model more robust to temporal variations.

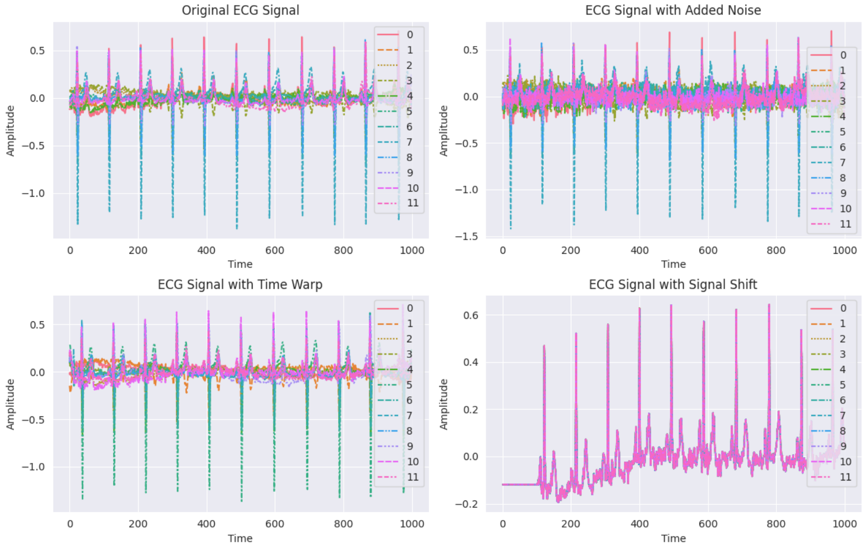

Figure 8.1: Different augmentation techniques.

8.2 Optimizers

8.2.1 Adam

Adam optimizer [53], short for Adaptive Moment Estimation, is a popular optimization algorithm widely used in training deep neural networks [54]. It combines the advantages of both AdaGrad and RMSprop, offering adaptive learning rates and momentum. The Adam optimizer computes adaptive learning rates for each parameter by maintaining two moving averages of gradients and squared gradients [55]. It calculates the first moment (mean) and the second moment (uncentered variance) of the gradients. These moments are then used to update the parameters in an adaptive manner, allowing for faster convergence and better generalization. Adam optimizer exhibits several advantages such as fast convergence, adaptive learning rates, and robustness to noisy gradients. However, it may suffer from certain limitations such as sensitivity to learning rate and momentum hyperparameters, which require careful tuning to achieve optimal performance.

8.2.2 AdamW

AdamW optimizer [56] is an extension of the Adam optimizer that introduces weight decay regularization to mitigate potential overfitting issues.

Similar to Adam, AdamW maintains two momentums for the gradients and squared gradients. However, it incorporates weight decay directly into the parameter updates, resulting in more stable optimization and improved generalization performance. AdamW addresses the weakness of Adam by effectively handling weight decay, which is crucial for preventing overfitting in deep neural networks. However, it may introduce additional computational overhead due to the incorporation of weight decay terms.

Weight Decay - Description

Weight decay is a regularization technique used to prevent overfitting by adding a penalty to the loss function based on the magnitude of the weights. This penalty term discourages the model from assigning too much importance to any single feature, leading to more generalized learning.

Weight Decay - Mechanism

Below, it is described the mechanism behind the AdamW optimizer and its benefits in detail:

1. Weight Decay Term

   - *Traditional Weight Decay Term*: In traditional weight decay, a term proportional to the weights is added to the loss function. This term is controlled by a hyperparameter, often denoted as λ. The modified loss function L becomes:

     $$L_{total} = L_{original} + \lambda \sum_i \omega_i^2$$

   - *In AdamW*: In the AdamW optimizer, weight decay is directly incorporated into the parameter updates, rather than being applied through the loss function. This direct integration helps in maintaining the separation between the optimization of the model parameters and the regularization

2. Update Rule

   - The parameter update rule in AdamW modifies the traditional Adam update by including the weight decay term:

     $$\theta_{t+1} = \theta_t - \eta(\frac{\hat{m_t}}{\sqrt{\hat{v_t}} + \epsilon} + \lambda\theta_t)$$

8.2.3 Adagrad

AdaGrad [57], short for Adaptive Gradient Algorithm, is an optimization algorithm designed to address the challenges of training neural networks with sparse data. AdaGrad adapts the learning rates of individual parameters based on the historical gradients encountered during training. It allocates larger updates to infrequent parameters and smaller updates to frequently occurring parameters, effectively mitigating the issues associated with sparse gradients. Adagrad advantages are the ability to handle sparse data and its automatic adjustment of learning rates. However, it may suffer from diminishing learning rates over time, which can lead to slow convergence or premature convergence in certain scenarios.

8.2.4 SGD

Stochastic Gradient Descent (SGD) [58] is a fundamental optimization algorithm widely used in training neural networks and machine learning models. SGD updates model parameters by computing gradients on a subset of training examples (mini-batch) and adjusting the parameters in the direction of the negative gradient. It employs a fixed or decaying learning rate to control the magnitude of parameter updates during training. The advantages of SGD are its simplicity, scalability, and ability to handle large datasets. However, it may suffer from issues such as slow convergence, sensitivity to learning rate tuning, and susceptibility to local minima.

8.2.5 RMSprop

RMSprop (Root Mean Square Propagation) [59] is an optimization algorithm designed to address the limitations of traditional stochastic gradient descent (SGD) in training deep neural networks. RMSprop adapts the learning rates for individual parameters by dividing the learning rate by the root mean square of past gradients for each parameter. This adaptive learning rate mechanism allows RMSprop to handle sparse gradients and non-stationary objectives more effectively. The advantages of RMSprop are the ability to converge fast and the robustness to noisy or sparse gradients. However, it may suffer from issues such as hyperparameter sensitivity and difficulties in tuning the decay rates [60].

8.2.6 Lion

The Lion optimizer [61] is a novel optimization algorithm inspired by the hunting behavior of lions in the wild. The Lion optimizer simulates the hunting behavior of lion prides, where lions collaborate to track and capture prey. It employs a combination of exploration and exploitation strategies, including random walk and memory-based search, to navigate the search space and find optimal solutions. The advantages of Lion are the ability to escape local optima, its adaptability to different optimization landscapes, and its potential for parallelization. However, it may require extensive parameter tuning and computational resources.

8.2.7 Optimizers Comparison

The previously described algorithms has been tested using the Basic CNN model in terms of training and validation accuracy and loss over 10 epochs with a static train-test split (Figure 8.2).

- *Training Accuracy*: All optimizers improve in performance over time. ADAM and ADAMW have the highest final accuracies, while SGD has the lowest final accuracy.

- *Training Loss*: The loss decreases for all optimizers as epochs increase. However, ADAGRAD has a higher loss compared to others.

- *Validation Accuracy*: There are fluctuations in all lines indicating variance in performance during validation. ADAM and ADAMW outperform others.

- *Validation Loss*: There is an increase in loss after initial epochs for most optimizers except LION which maintains a steady decline.

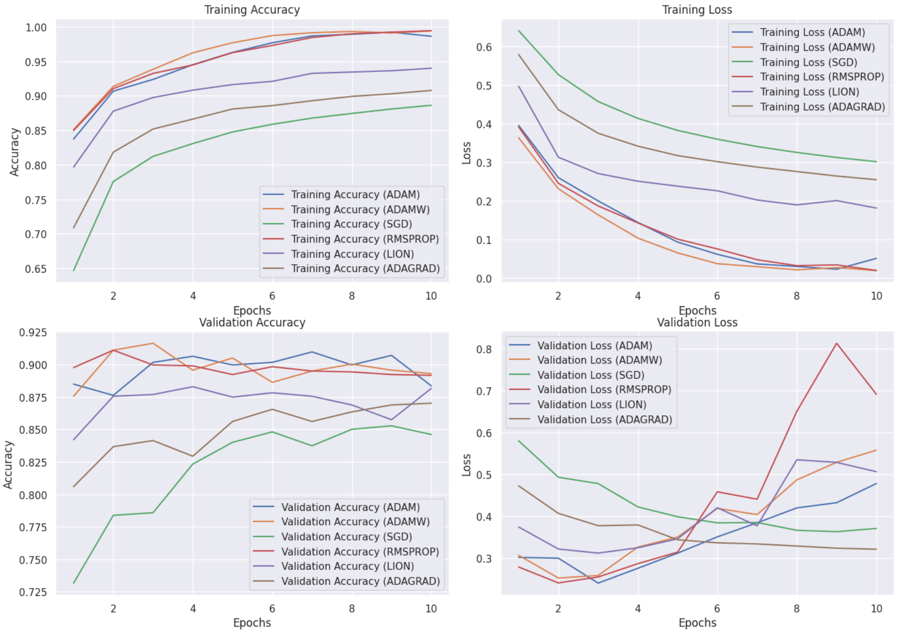

Figure 8.2: Comparison of optimizers.

In general, ADAM and ADAMW seem to perform better in terms of both accuracy and loss. However, it's important to note that the performance of the optimizer can depend on various factors. Therefore, testing all the optimizers and tuning the parameters is a gold standard in machine learning; but this comparison provides a first indication on which optimizers might perform better on the tasks.

8.3 Schedulers

8.3.1 Constant

The constant learning rate scheduler is a simple yet fundamental technique used in optimizing machine learning models. The constant learning rate scheduler maintains a fixed learning rate throughout the training process, ensuring that the model parameters are updated with consistent magnitudes at each iteration. This straightforward approach allows for stable training but may require careful selection of the learning rate hyperparameter. Advantages of the constant learning rate scheduler include simplicity, stability, and ease of implementation. However, it may suffer from limitations such as sensitivity to the initial learning rate choice and difficulties in finding an appropriate balance between convergence speed and model performance.

8.3.2 PolynomialDecay

The polynomial decay learning rate scheduler [62] is a dynamic technique used to adjust the learning rate during the training process based on a polynomial function. The polynomial decay scheduler gradually decreases the learning rate according to a polynomial function over the course of training epochs. It allows for fine-tuning the learning rate schedule to match the convergence behavior of the optimization landscape, potentially improving convergence speed and final performance. The advantages of the polynomial decay scheduler are flexibility, adaptability, and ability to handle complex optimization dynamics. However, it may introduce additional hyperparameters that require tuning and may suffer from slower convergence rates compared to more aggressive decay schedules.

8.3.3 CosineDecay

The cosine decay learning rate scheduler [63] is a cyclic technique used to modulate the learning rate based on the cosine function over the course of training epochs. The cosine decay scheduler gradually reduces the learning rate following a cosine curve, allowing for periodic adjustments that mimic the cyclical nature of optimization dynamics. It enables the exploration of diverse regions in the optimization landscape while maintaining stable convergence behavior. The advantages of the cosine decay scheduler are its ability to escape local minima, its regularization effect, and its potential for improving generalization. However, it may require careful tuning of hyperparameters such as cycle length and initial learning rate.

8.3.4 ExponentialDecay

The exponential decay learning rate scheduler [64] is a widely used technique for adjusting the learning rate during the training process based on an exponential function. The exponential decay scheduler decreases the learning rate exponentially over the course of training epochs, allowing for aggressive reductions in the learning rate magnitude. It facilitates fast convergence in the initial training stages while gradually stabilizing the learning process to prevent overshooting. The advantages of the exponential decay scheduler are its simplicity, efficiency, and effectiveness in accelerating convergence. However, it may suffer from limitations such as sensitivity to the decay rate hyperparameter and potential for premature convergence.

8.4 Loss

8.4.1 BinaryCrossEntropy

Binary Cross-Entropy loss function [65], also known as log loss, is a fundamental component in training binary classification models within the realm of deep learning. The Binary Cross-Entropy loss function measures the discrepancy between the predicted probabilities and the true labels in binary classification problems [66]. It calculates the logarithm of the predicted probability of the correct class, penalizing misclassifications with higher magnitudes for greater prediction errors. The loss function is mathematically expressed as the negative log-likelihood of the true class. The advantages of the Binary Cross-Entropy loss function are its ability to handle imbalanced datasets, its smooth gradient properties, and its effectiveness in optimizing models for binary classification tasks [67]. However, it may suffer from limitations such as sensitivity to class imbalance and difficulties in interpretation.

8.5 Hyperparameters

8.5.1 Epochs

The choice of the number of epochs plays a critical role in training deep learning models effectively. Epochs represent the number of times the entire dataset [68] is passed forward and backward through the neural network during training [69]. It determines the number of iterations the model undergoes to learn from the data and adjust its parameters accordingly. the advantages of tuning epochs, including the ability to control the duration of training and achieve convergence to an optimal solution. However, excessive epochs can lead to overfitting [70], while insufficient epochs may result in underfitting, impacting the model's performance. The advantages of tuning epochs include the ability to control the duration of training and achieve convergence to an optimal solution. However, excessive epochs can lead to overfitting, while insufficient epochs may result in underfitting, impacting the model's performance.

8.5.2 Batch Size

Batch size is a crucial hyperparameter in deep learning optimization, influencing the efficiency and effectiveness of model training. Batch size refers to the number of samples processed by the model in a single iteration during training. It affects the stability of the optimization process, the quality of parameter updates, and the utilization of hardware resources. The advantages of optimizing batch size include faster convergence, reduced memory requirements, and improved generalization. However, excessively small batch sizes may lead to noisy gradients, while excessively large batch sizes may hinder convergence and limit model capacity.

8.5.3 Learning Rate

The learning rate is a fundamental hyperparameter in training deep learning models, influencing the speed and quality of optimization. The learning rate determines the step size of parameter updates during optimization, affecting the magnitude of adjustments made to the model weights. It plays a critical role in balancing the trade-off between convergence speed and convergence quality. The advantages of fine-tuning learning rates include improved convergence, enhanced model generalization, and robustness to optimization challenges. However, selecting an inappropriate learning rate can lead to slow convergence, oscillations, or divergence during training.

8.6 Callbacks

8.6.1 EarlyStopping

The EarlyStopping callback is a powerful technique used to prevent overfitting and improve the generalization performance of deep learning models. The EarlyStopping callback monitors the validation loss or a specified metric during training and halts the training process when the performance on the validation set stops improving. It allows for early termination of training to prevent overfitting and save computational resources. The advantages of the EarlyStopping callback include its ability to prevent overfitting, improve model generalization, and facilitate efficient hyperparameter tuning. However, it may halt training prematurely if the validation metric fluctuates or plateaus without significant improvement.

8.6.2 ModelCheckPoint

The ModelCheckpoint callback is a valuable tool used to save model weights during training and facilitate model restoration and deployment. The ModelCheckpoint callback periodically saves the model weights to disk during training, allowing for model snapshots to be captured at specific intervals or based on certain criteria such as validation performance. It enables model recovery, fine-tuning, and deployment without retraining from scratch. The advantages of the ModelCheckpoint callback include its ability to preserve model progress, prevent data loss in case of interruptions, and facilitate model ensembling and transfer learning. However, it may consume additional storage space and computational resources.

Chapter 9

Model Testing

The testing phase of machine learning models is a critical step in assessing their generalization capabilities and evaluating their performance on unseen data. The testing process consists in passing the trained deep learning models to real-world electrocardiography (ECG) signals to ascertain their diagnostic accuracy and robustness. The testing pipeline begins by feeding unseen ECG signals into the trained models, which have been previously trained on a diverse dataset encompassing various cardiac conditions. These ECG signals are typically partitioned into a separate validation or test set to simulate real-world scenarios where the models encounter new and unseen data. During the testing phase, the models' predictive capabilities are evaluated based on predefined performance metrics such as accuracy, precision, recall and F1Score. These metrics provide quantitative measures of the models' ability to correctly classify ECG signals into different diagnostic categories, including normal rhythms and various cardiac abnormalities. In addition to quantitative metrics, qualitative assessment of the models' predictions is also conducted to identify potential areas of improvement and examine the models' interpretability. Visualization techniques such as confusion matrices and ROC curves provide intuitive representations of the models' performance, facilitating the identification of strengths and weaknesses. Moreover, rigorous testing procedures help mitigate the risk of overfitting and ensure that the trained models generalize well to unseen ECG data from diverse patient populations. By subjecting the models to rigorous testing protocols, healthcare practitioners can make informed decisions regarding the models' suitability for clinical deployment and diagnostic support.

9.1 Evaluation Metrics

Evaluation metrics [71] are essential tools for quantifying the performance of machine learning models. By understanding and interpreting these metrics, practitioners can make informed decisions about model selection, parameter tuning, and optimization strategies. It is crucial to choose evaluation metrics that are relevant to the specific task and objectives of the machine learning project. Additionally, considering multiple metrics provides a comprehensive view of model performance and helps identify areas for improvement.

9.1.1 Accuracy

Accuracy measures the proportion of correctly classified instances out of the total instances [72].

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Accuracy gives an overall view of how well the model performs across all classes. However, it can be misleading if the classes are imbalanced.

9.1.2 Precision

Precision measures the proportion of true positive predictions out of all positive predictions [73].

$$Precision = \frac{TP}{TP + FP}$$

Precision focuses on the accuracy of positive predictions. It is useful when the cost of false positives is high.

9.1.3 Recall

Recall measures the proportion of true positive predictions out of all actual positive instances [74].

$$Recall = \frac{TP}{TP + FN}$$

Recall is sensitive to the model's ability to correctly identify positive instances. It is useful when the cost of false negatives is high [75].

9.1.4 F1-Score

F1-score is the harmonic mean of precision and recall, providing a balance between the two metrics.

$$F1 - Score = \frac{2 \times (Precision \times Recall)}{Precision + Recall}$$

F1-score combines precision and recall into a single metric, making it useful for tasks where both false positives and false negatives are equally important.

9.2 Test Procedure

9.2.1 Hyperparameters Tuning

The strategy for tuning hyperparameters involved a systematic and iterative approach. Initially, the process began with selecting a set of baseline values for key hyperparameters such as learning rate, batch size, and the number of epochs. These values were chosen based on common practices in the field and initial experimentation (e.g., learning rate = 1e-3, batch size = 32, epochs = 100). Moreover, the parameters were associated with the different combinations of Optimizers and Schedulers. Particularly, the tuning process involved the following steps:

1. Grid Search: A coarse grid search was conducted over a predefined range of values for each hyperparameter. This helped identify regions of the hyperparameter space that yielded better performance

2. Random Search: Following the grid search, a more fine-grained random search was applied within the identified regions to pinpoint more optimal values

3. Bayesian Optimization: To further refine the hyperparameter values, Bayesian optimization was employed. This method balances exploration and exploitation, utilizing previous evaluation results to predict the performance of new sets of hyperparameters

4. Cross-Validation: Throughout the tuning process, an 8-fold cross-validation approach was used to ensure robustness and generalizability of the selected hyperparameters. This method also helped in assessing the model's performance more comprehensively by splitting the data into multiple folds

9.2.2 Betas and K-Fold

The initial values for the beta parameters in the Adam optimizer were set as $\beta_1 = 0.9$ and $\beta_2 = 0.999$. These values are standard and commonly used due to their effectiveness in various machine learning tasks. Furthermore, an 8-fold cross-validation was chosen based on a balance between computational efficiency and thoroughness in performance evaluation. This specific choice was influenced by:

- Empirical Evidence: Previous experiments and suggested that 8-fold provides a good trade-off between bias and variance. It allows for sufficient training data in each fold while ensuring a comprehensive validation process

- Dataset Size and Structure: Given the dataset's size and structure, 8-fold cross-validation was determined to maximize the utilization of available data without excessive computational cost

9.2.3 Data Augmentation

After establishing a baseline with K-Fold cross-validation, data augmentation techniques were introduced to enhance the model's ability to generalize to unseen data. Augmentation techniques, such as noise addition, time warping, and signal shifting, were applied to increase the diversity of the training dataset. Moreover, parameter tuning and iterative optimization were conducted based on observed performance metrics, such as accuracy, recall, and loss.

Chapter 10

Results

Through a meticulous examination of the obtained results, the aim is to provide a comprehensive understanding of the efficacy of the used approaches in addressing the research objectives. From the initial exploration of baseline models to the comprehensive evaluation of optimized architectures, this chapter encapsulates the essence of the work to develop robust and accurate solutions for the task at hand.

10.1 Basic CNN

Considering the myocardial infarction detection in electrocardiograms, the performance of a basic CNN model remains noteworthy. Even though some other architectures may perform better, the Basic CNN shows the ability of efficiently classify instances of myocardial infarction. While more intricate architectures exist, the basic CNN. Although achieving the highest levels of accuracy may necessitate more specialized models tailored to this specific task, the basic CNN serves as a dependable starting point.

10.1.1 No Data Augmentation

The findings presented here are derived from experiments conducted without the use of data augmentation techniques. This underscores the raw performance of the basic CNN model in MI detection, unaided by additional training data generated through transformations. Despite the absence of augmentation, the basic CNN demonstrates commendable performance across various experimental configurations. The Table 10.1 shows the best combinations of parameters and results.

Model	Optimizer	LR	Batch Size	Scheduler	StratKFold
Model_1	AdamW	1e-3	32	Polynomial Decay	True - 8
Model_2	AdamW	1e-3	64	Polynomial Decay	True - 8
Model_3	RMSprop	1e-3	64	Constant	True - 8
Model_4	Lion	1e-3	64	Polynomial Decay	True - 8

Table 10.1: Best combination of hyperparameters, Basic CNN - No Augmentation.

Particularly, the best models reach 91% of accuracy as shown in the boxplot in Figure 10.1, while precision and recall metrics exhibit variability across configurations but generally fall within acceptable ranges for medical diagnosis tasks. Table 10.2 shows the metrics score for the different models.

Model	Accuracy	Precision	Recall	F1Score
		NORM, MI	NORM, MI	NORM, MI
Model_1	0.91	0.90, 0.92	0.95, 0.85	0.92, 0.88
Model_2	0.91	0.90, 0.93	0.96, 0.84	0.93, 0.88
Model_3	0.91	0.91, 0.90	0.94, 0.87	0.92, 0.89
Model_4	0.91	0.90, 0.92	0.95, 0.84	0.92, 0.88

Table 10.2: Metrics results, Basic CNN - No Augmentation.

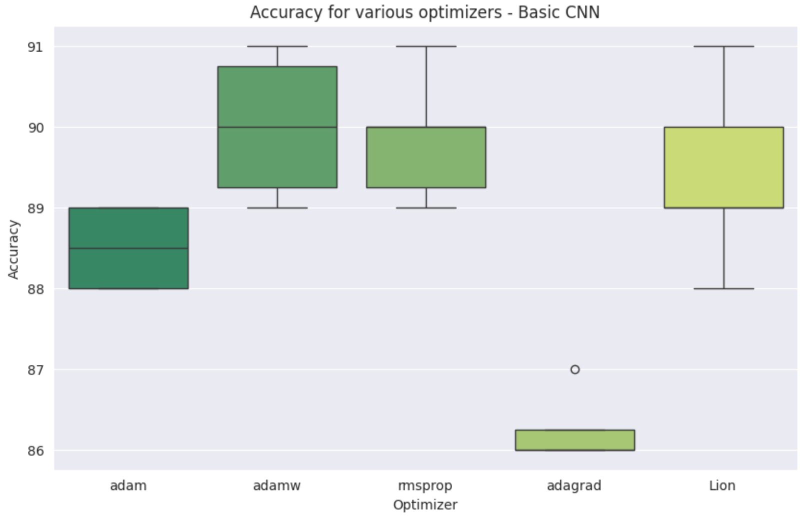

Figure 10.1: Comparison of optimizers accuracy, Basic CNN - No Augmentation.

10.1.2 Data Augmentation

The integration of data augmentation techniques into the training pipeline normally influences the performance models. In this case, the analyzed metrics remains stable. The Table 10.3 shows the best combinations of parameters with the data augmentation applied.

Model	Optimizer	LR	Batch Size	Scheduler	StratKFold
Model_1	AdamW	1e-3	32	Polynomial Decay	False
Model_2	AdamW	1e-3	32	Polynomial Decay	True - 8

Table 10.3: Best combination of hyperparameters, Basic CNN - Augmentation.

As previously said, the metrics remains stable with data augmentation techniques (Results in Table 10.4 and Figure 10.2). This behavior can be explained looking at the naïve architecture of the considered CNN. In fact, the choice of model architecture and optimization strategy can influence the extent to which data augmentation enhances performance. While CNNs are generally robust to variations in input data, the capacity of the model to learn and generalize from augmented examples may vary depending on its complexity and capacity to capture relevant features.

Model	Accuracy	Precision NORM, MI	Recall NORM, MI	F1Score NORM, MI
Model_1	0.91	0.89, 0.94	0.96, 0.83	0.92, 0.88
Model_2	0.91	0.91, 0.90	0.94, 0.86	0.92, 0.88

Table 10.4: Metrics results, Basic CNN - Augmentation.

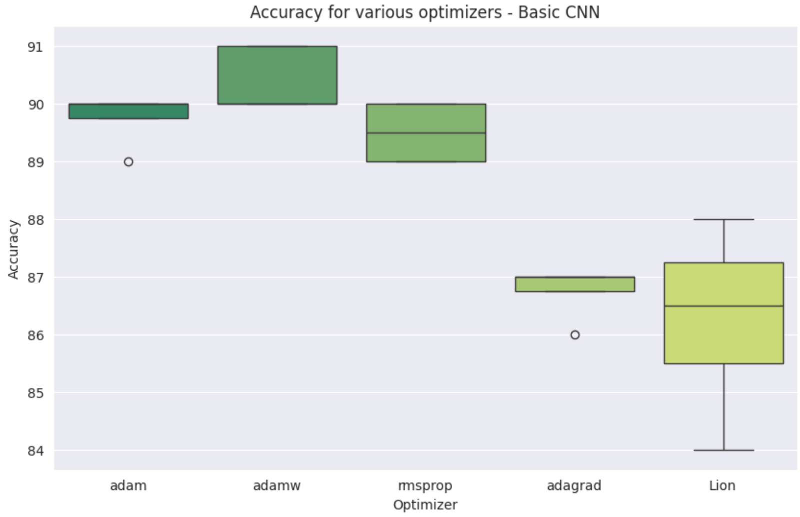

Figure 10.2: Comparison of optimizers accuracy, Basic CNN - Augmentation.

10.2 Enhanced CNN

Compared with the results of the Basic CNN, this architecture demonstrates to be more accurate and precise. This was an expected result, since the architecture is more complex and more intricate than the previous one. Furthermore, the Enhanced CNN had the ability fo efficiently and accurately detect instances of myocardial infarction without sacrificing performances.

10.2.1 No Data Augmentation

The results of this section are the ones without the use of data augmentation; therefore, this point out the raw performances of the Enhanced CNN. The best combination of parameters are reported in Table 10.5.

Model	Optimizer	LR	Batch Size	Scheduler	StratKFold
Model_1	Adam	1e-3	64	Polynomial Decay	True - 8
Model_2	AdamW	1e-3	32	Polynomial Decay	True - 8
Model_3	RMSprop	1e-3	32	Polynomial Decay	True - 8
Model_4	RMSprop	1e-3	64	Polynomial Decay	True - 8

Table 10.5: Best combination of hyperparameters, Enhanced CNN - No Augmentation.

Particularly, the best models reach 94% of accuracy, as shown in the boxplot in Figure 10.3, while (as in the Basic CNN) the other matrics shows some variabilities across the tests, but are consistent for similar performances. Results for the accuracy and for the other metrics are reported in Table 10.6.

Model	Accuracy	Precision	Recall	F1Score
		NORM, MI	NORM, MI	NORM, MI
Model_1	0.94	0.93, 0.96	0.98, 0.90	0.95, 0.93
Model_2	0.94	0.95, 0.94	0.96, 0.93	0.95, 0.93
Model_3	0.94	0.94, 0.94	0.96, 0.91	0.95, 0.92
Model_4	0.94	0.94, 0.95	0.97, 0.91	0.95, 0.93

Table 10.6: Metrics results, Enhanced CNN - No Augmentation.

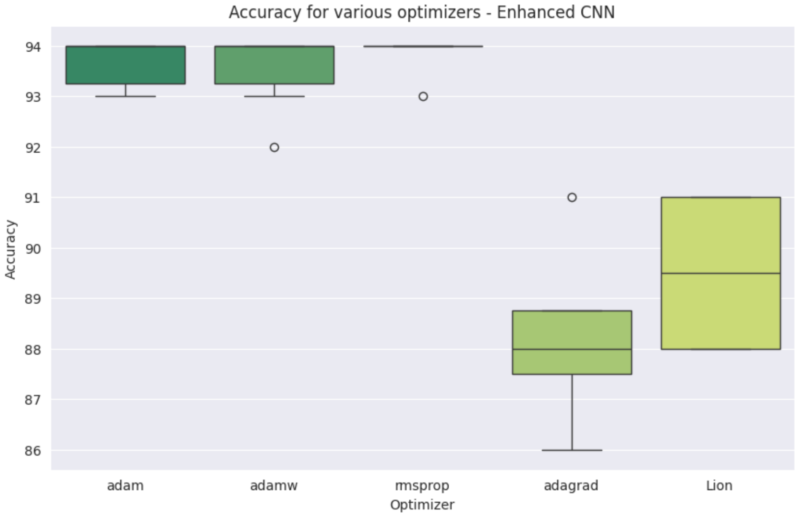

Figure 10.3: Comparison of optimizers accuracy, Enhanced CNN - No Augmentation.

10.2.2 Data Augmentation

In this case the integration of data augmentation techniques bring a noticeable result for all the metrics considered. Table 10.7 report the best hyperparameters combination.

Model	Optimizer	LR	Batch Size	Scheduler	StratKFold
Model_1	AdamW	1e-3	32	Constant	False
Model_2	RMSprop	1e-3	64	Polynomial Decay	False
Model_3	RMSprop	1e-3	64	Polynomial Decay	False
Model_4	AdamW	1e-3	64	Polynomial Decay	True - 8

Table 10.7: Best combination of hyperparameters, Enhanced CNN - Augmentation.

The Table 10.8 and Figure 10.4 show the results obtained after the augmentation of the data, comfirming the utility of applying augmentation techniques to the data.

Model	Accuracy	Precision NORM, MI	Recall NORM, MI	F1Score NORM, MI
Model_1	0.95	0.94, 0.96	0.98, 0.90	0.95, 0.93
Model_2	0.95	0.94, 0.96	0.97, 0.91	0.95, 0.93
Model_3	0.95	0.93, 0.95	0.96, 0.90	0.95, 0.93
Model_4	0.95	0.95, 0.95	0.97, 0.92	0.96, 0.94

Table 10.8: Metrics results, Enhanced CNN - Augmentation.

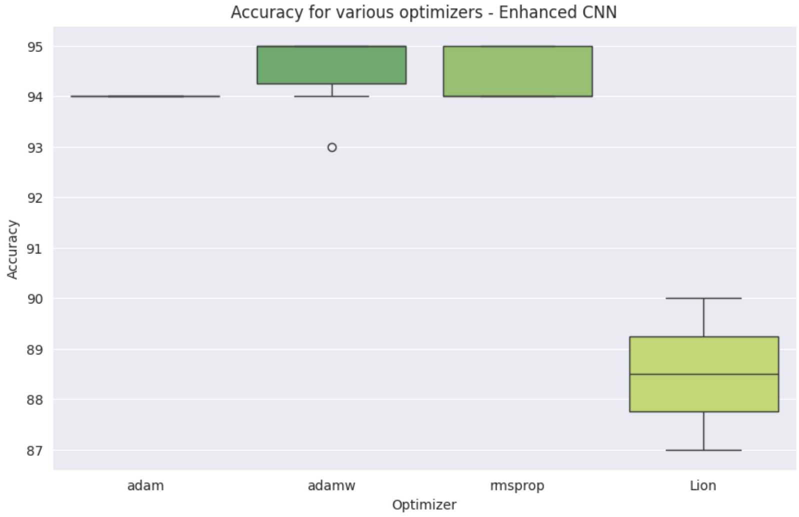

Figure 10.4: Comparison of optimizers accuracy, Enhanced CNN - Augmentation.

10.3 ConvNetQuake

This Neural Network provides results that are very similar to the ones of the Enhanced CNN in terms of metrics. On the other hand, it shows better performances in terms of training time without sacrificing the accuracy. Even though this architecture was initially design to detect earthquakes, it shows to be suitable for ECG analysis.

10.3.1 No Data Augmentation

Raw performances of ConvNetQuake (without Data Augmentation) are reported in Table 10.9 along with the best combination of hyperparameters.

Model	Optimizer	LR	Batch Size	Scheduler	StratKFold
Model_1	Lion	1e-3	32	Polynomial Decay	False
Model_2	Lion	1e-3	64	Polynomial Decay	False
Model_3	Lion	1e-3	32	Polynomial Decay	True - 8
Model_4	Lion	1e-3	64	Polynomial Decay	True - 8

Table 10.9: Best combination of hyperparameters, ConvNetQuake - No Augmentation.

In particular, performances in this case reach 95% of accuracy for the best model, showing a small improvement compared to the Enhanced CNN. The results are shown in the boxplot in Figure 10.5 and in Table 10.10.

Model	Accuracy	Precision	Recall	F1Score
		NORM, MI	NORM, MI	NORM, MI
Model_1	0.94	0.95, 0.92	0.95, 0.92	0.95, 0.92
Model_2	0.94	0.94, 0.94	0.96, 0.91	0.95, 0.93
Model_3	0.95	0.93, 0.96	0.98, 0.90	0.95, 0.93
Model_4	0.94	0.93, 0.86	0.97, 0.89	0.95, 0.92

Table 10.10: Metrics results, ConvNetQuake - No Augmentation.

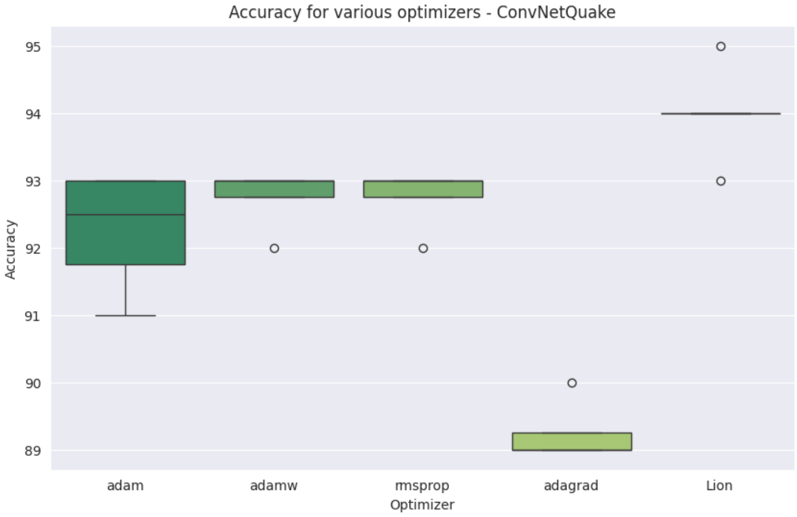

Figure 10.5: Comparison of optimizers accuracy, ConvNetQuake - No Augmentation.

10.3.2 Data Augmentation

Data augmentation techniques in this case demonstrates to be effective since the overall performances increased; however, the top performances remain stable. Best combination of hyperparameters are reported in Table 10.11.

Model	Optimizer	LR	Batch Size	Scheduler	StratKFold
Model_1	Adam	1e-3	32	Polynomial Decay	False
Model_2	Lion	1e-3	32	Polynomial Decay	False
Model_3	Lion	1e-3	64	Polynomial Decay	False
Model_4	AdamW	1e-3	32	Polynomial Decay	True - 8

Table 10.11: Best combination of hyperparameters, ConvNetQuake - Augmentation.

Performances remains arounf 95% of accuracy, with other metrics consistent across the various hyperparameters combination. Results are shown both in Figure 10.6 and Table 10.12.

Model	Accuracy	Precision	Recall	F1Score
		NORM, MI	NORM, MI	NORM, MI
Model_1	0.95	0.93, 0.95	0.97, 0.90	0.95, 0.93
Model_2	0.95	0.97, 0.90	0.93, 0.95	0.95, 0.92
Model_3	0.94	0.98, 0.89	0.93, 0.97	0.94, 0.92
Model_4	0.95	0.97, 0.90	0.94, 0.95	0.95, 0.93

Table 10.12: Metrics results, ConvNetQuake - Augmentation.

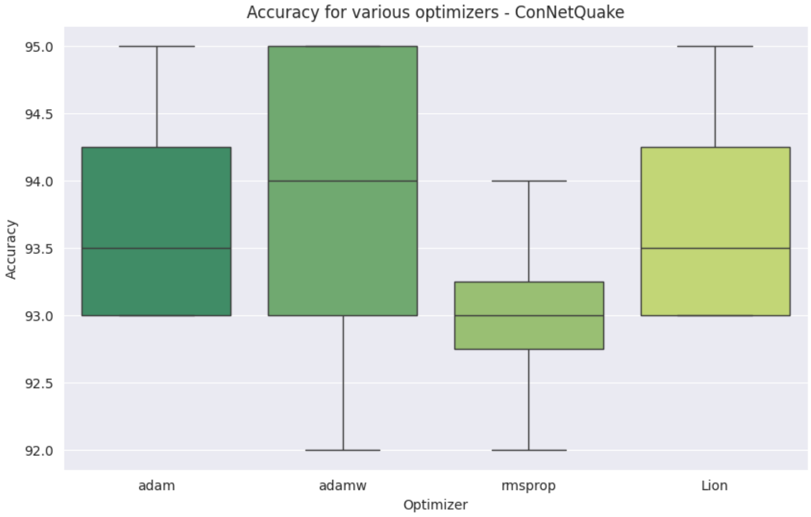

Figure 10.6: Comparison of optimizers accuracy, ConvNetQuake - Augmentation.

10.4 LSTM

The results of this neural network are the most promising. Particularly, the results are slightly better to the ones obtained with the previous architecture, demonstrating the efficiency of LSTM in ECG analysis and abnormalities detection.

10.4.1 No Data Augmentation

Raw performances along with the best combination of hyperparameters are reported in Table 10.13.

Model	Optimizer	LR	Batch Size	Scheduler	StratKFold
Model_1	Lion	1e-3	32	Polynomial Decay	False
Model_2	Lion	1e-3	32	Constant	True - 8
Model_3	Lion	1e-3	32	Polynomial Decay	True - 8
Model_4	Lion	1e-3	64	Polynomial Decay	True - 8

Table 10.13: Best combination of hyperparameters, LSTM - No Augmentation.

Particularly, the results for LSTM are around 94% for the best combinations of hyperparameters. Results can be seen in Figure 10.7 and Table 10.14.

Model	Accuracy	Precision	Recall	F1Score
		NORM, MI	NORM, MI	NORM, MI
Model_1	0.94	0.94, 0.95	0.97, 0.91	0.95, 0.93
Model_2	0.94	0.93, 0.96	0.98, 0.89	0.95, 0.93
Model_3	0.94	0.93, 0.95	0.97, 0.89	0.95, 0.92
Model_4	0.94	0.94, 0.95	0.97, 0.90	0.97, 0.90

Table 10.14: Metrics results, LSTM - No Augmentation.

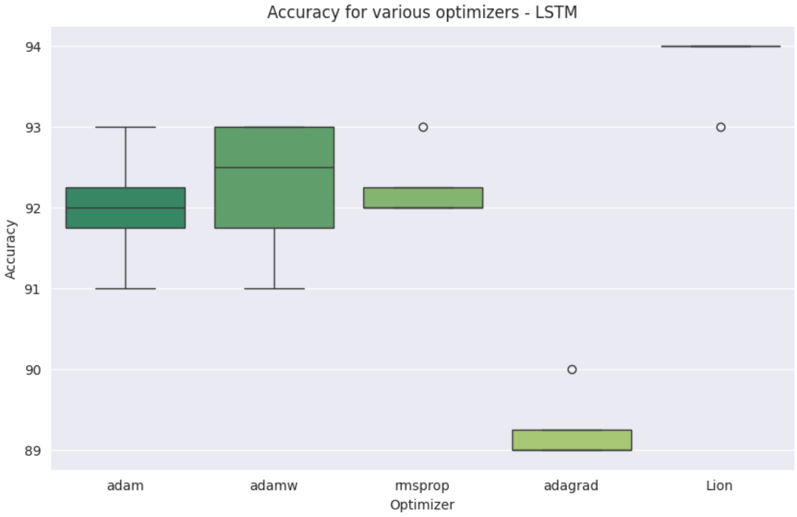

Figure 10.7: Comparison of optimizers accuracy, LSTM - No Augmentation.

10.4.2 Data Augmentation

For this Neural Network the data augmentation shows an important improvement, that represents the best result obtained in this project. Table 10.15 shows the best combination of hyperparameters.

Model	Optimizer	LR	Batch Size	Scheduler	StratKFold
Model_1	AdamW	1e-3	64	Constant	False
Model_2	AdamW	1e-3	32	Polynomial Decay	False
Model_3	RMSprop	1e-3	32	Constant	False
Model_4	RMSprop	1e-3	64	Polynomial Decay	True - 8

Table 10.15: Best combination of hyperparameters, LSTM - Augmentation.

In particular, the accuracy is 96% for the best combination of hyperparameters. The other metrics show better results compared to the other architectures, enhancing the effectivness of this Neural Network for time-series analysis. The results are shown in Figure 10.8 and in Table 10.16.

Model	Accuracy	Precision NORM, MI	Recall NORM, MI	F1Score NORM, MI
Model_1	0.96	0.95, 0.95	0.97, 0.92	0.96, 0.93
Model_2	0.96	0.95, 0.96	0.97, 0.93	0.97, 0.93
Model_3	0.95	0.95, 0.92	0.95, 0.93	0.95, 0.93
Model_4	0.95	0.95, 0.94	0.96, 0.92	0.95, 0.93

Table 10.16: Metrics results, LSTM - Augmentation.

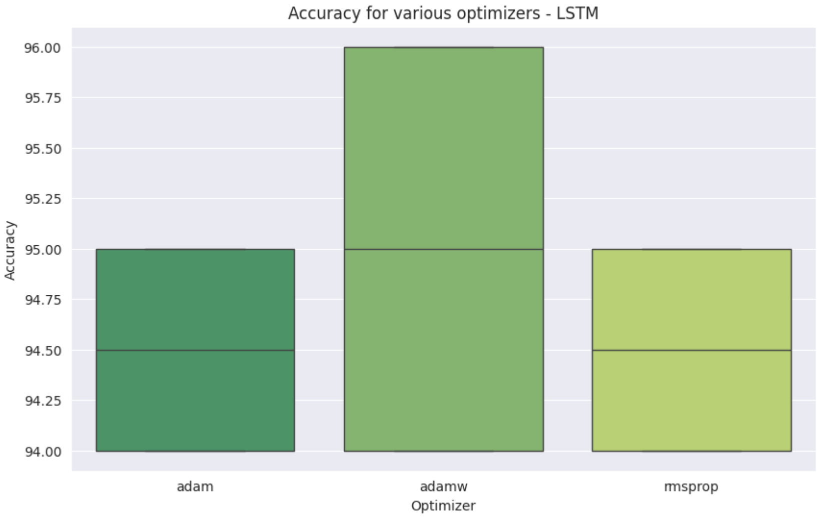

Figure 10.8: Comparison of optimizers accuracy, LSTM - Augmentation.

10.5 Discussion

10.5.1 Hyperparameters and Optimizers

The selection and tuning of hyperparameters and optimizers are critical in deep learning to ensure optimal model performance. In this study, several hyperparameters were fine-tuned using a combination of grid search, random search, and Bayesian optimization techniques.

Hyperparameters

Considering the used hyperparameters:

- *Learning Rate*: The learning rate was adjusted within the range of 1e-5 to 1e-3. Lower learning rates generally provided more stable convergence, while higher rates accelerated training but risked overshooting minima.

- *Batch Size*: Various batch sizes (16, 32, 64, 128) were evaluated. Smaller batch sizes led to noisier updates and potentially better generalization, while larger batch sizes offered more stable and faster training
- *Number of Epochs*: The models were trained for up to 100 epochs, with early stopping based on validation loss to prevent overfitting
- *Optimizer Choice*: Different optimizers, including Adam, AdamW, and SGD, were tested. Adam and AdamW typically provided better convergence rates and final performance due to their adaptive learning rate mechanisms

Optimizers

Considering the used optimizers:

- *Adam*: Adam optimizer was selected for most models due to its ability to handle sparse gradients and adaptive learning rate, leading to faster convergence
- *SGD*: Stochastic Gradient Descent (SGD) was also tested but generally required careful learning rate scheduling to achieve competitive performance
- *RMSprop and AdamW*: RMSprop and AdamW were used in specific models where their benefits in handling non-stationary objectives and weight decay, respectively, were advantageous

10.5.2 Impact of Data Augmentation

Data augmentation played a crucial role in enhancing the model's ability to generalize to unseen data. By artificially expanding the training dataset, the models learned to recognize patterns invariant to common transformations and noise. Considering the used techniques:

- *Adding Noise*: Random noise was added to ECG signals to simulate real-world variability and improve robustness
- *Time Warp*: Temporal distortions were applied to the signals to help the model become invariant to timing variations in the ECG waveforms

- *Signal Shifting*: Signals were shifted along the time axis to teach the model to handle slight misalignments in the data

10.5.3 Models Comparison

Analyzing the performance of different neural network architectures revealed valuable insights. Particularly, the Enhanced CNN showed superior performance compared to the Basic CNN, particularly when data augmentation was applied. The additional layers and complexity of the Enhanced CNN allowed it to capture more intricate patterns in the ECG signals, resulting in higher accuracy and better generalization. While ConvNetQuake, originally designed for seismic data, adapted well to ECG analysis, LSTM networks were particularly effective in capturing temporal dependencies within the ECG data. However, LSTMs required more computational resources and longer training times.

10.5.4 Error Analysis

A detailed error analysis was conducted to understand the misclassifications made by the models: By examining the instances where the models incorrectly predicted the presence or absence of myocardial infarction, patterns and common features among these errors were identified. This analysis highlighted areas where the models could be improved, such as better handling of noisy or ambiguous signals. For instance, many false positives were observed in cases where the ECG signals had significant noise or artifacts that mimicked the features of myocardial infarction. Similarly, false negatives often occurred in cases where the infarction signs were subtle or atypical. Addressing these issues could involve enhancing the preprocessing steps to better clean the data and developing more sophisticated feature extraction methods to capture subtle patterns more effectively.

Chapter 11

Conclusions

In conclusion, this project aimed to develop and evaluate deep learning models for myocardial infarction (MI) detection using electrocardiogram (ECG) data. Through the implementation of various models, training procedures, and evaluation techniques, several key findings and insights have been obtained.

11.1 Key Findings

The results revealed performance differences across the evaluated architectures. While Basic CNN provided a solid foundation, Enhanced CNN demonstrated superior performance, leveraging deeper architectures and enhanced feature extraction capabilities. ConvNetQuake, originally designed for earthquake detection, surprisingly exhibited adaptability to ECG analysis, rivaling the performance of the Enhanced CNN. Notably, LSTM networks emerged as standout performers, particularly adept at capturing temporal dependencies inherent in ECG signals. Although there is room for improvement, the evaluated metrics showed promising results, defining a strong baseline for future development.

11.2 Future Directions

Moving forward, there are a multitute of different paths that can be taken to improve the project results and findings. Particularly, one possible development might be the possibility to use a pretrained model as a feature extractor or for fine tuning. Although this is not a complicated task, it is not straightforward to find a model that is suitable for the ECG classification. Another future development is represented by the possibility of using images for ECG classification. However, this might have some problems related to the quality, format and quantity of the images. Furthermore, the possibility to acquire a great quantity of data related to ECGs to integrate into the PTB-XL database is another option to follow in order to create a stronger dataset to work with. In conclusion, the study will serve as base for future developments and improvements, having always as key objective the possibility to save more human lives.

Bibliography

[1] Elizabeth Wilkins et al. *European Cardiovascular Disease Statistics 2017*. English. Belgium: European Heart Network, Feb. 2017.

[2] Gilles R Dagenais et al. "Variations in common diseases, hospital admissions, and deaths in middle-aged adults in 21 countries from five continents (PURE): a prospective cohort study". In: *The Lancet* 395.10226 (2020), pp. 785–794.

[3] Kristian Thygesen et al. "Fourth Universal Definition of Myocardial Infarction (2018)". In: *Journal of the American College of Cardiology* 72.18 (2018), pp. 2231–2264. ISSN: 0735-1097. DOI: `https://doi.org/10.1016/j.jacc.2018.08.1038`. URL: `https://www.sciencedirect.com/science/article/pii/S0735109718369419`.

[4] Zachi I Attia et al. "Novel bloodless potassium determination using a signal-processed single-lead ECG". In: *Journal of the American heart Association* 5.1 (2016), e002746.

[5] Antônio H Ribeiro et al. "Automatic diagnosis of the 12-lead ECG using a deep neural network". In: *Nature communications* 11.1 (2020), p. 1760.

[6] Junaid Abdul Wahid et al. "A hybrid ResNet-ViT approach to bridge the global and local features for myocardial infarction detection". In: *Scientific Reports* 14.1 (Feb. 2024). ISSN: 2045-2322. DOI: `10.1038/s41598-024-54846-8`. URL: `http://dx.doi.org/10.1038/s41598-024-54846-8`.

[7] David M Mirvis and Ary L Goldberger. "Electrocardiography". In: *Heart disease* 1 (2001), pp. 82–128.

[8] *Chambers of the Heart*. `https://www.elitecardiovascular.com/cardio\-vascular-conditions/cardiac-conditions/basics-of-the-heart/`.

[9] Francisco Torrent-Guasp et al. “Towards new understanding of the heart structure and function”. In: *European journal of cardio-thoracic surgery* 27.2 (2005), pp. 191–201.

[10] Arnold M Katz. *Physiology of the Heart*. Lippincott Williams & Wilkins, 2010.

[11] *Heart Complete Structure*. `https://savecatchingfire.blogspot.com/2018/05/heart-structure-anatomy.html`.

[12] Yasar Sattar and Lovely Chhabra. “Electrocardiogram”. In: *StatPearls [Internet]*. StatPearls Publishing, 2023.

[13] Charles E Kossmann. “The normal electrocardiogram”. In: *Circulation* 8.6 (1953), pp. 920–936.

[14] *Standard ECG Wave*. `https://www.firstaidforfree.com/a-basic-guide-to-ecgekg-interpretation/`.

[15] Phil Jevon. “Procedure for recording a standard 12-lead electrocardiogram”. In: *British Journal of Nursing* 19.10 (2010), pp. 649–651.

[16] *Arms placement*. `https://www.ausmed.co.uk/learn/articles/ecg-lead-placement`.

[17] *Precordial placement*. `https://www.urmc.rochester.edu/pediat\rics/cardiology-fellowship/ecg-placement.aspx`.

[18] Jarosław Wasilewski and Lech Poloński. “An introduction to ECG interpretation”. In: *ECG signal processing, classification and interpretation: a comprehensive framework of computational intelligence*. Springer, 2011, pp. 1–20.

[19] *AED USA*. `https://www.aedusa.com/knowledge/which-of-the-following-rhythms-can-be-treated-with-an-automated-external-defibrillator-aed/`.

[20] *Polarization and Depolarization*. `https://melanie-blogfisher.blogspot.com/2022/04/describe-actions-of-depolarization-and.html`.

[21] Frank G Yanowitz. “Introduction to ECG interpretation”. In: *LDS Hospital and Intermountain Medical Center* (2012).

[22] Grant W Reed, Jeffrey E Rossi, and Christopher P Cannon. “Acute myocardial infarction”. In: *The Lancet* 389.10065 (2017), pp. 197–210.

[23] JG Kingma et al. “Myocardial infarction: An overview of STEMI and NSTEMI physiopathology and treatment”. In: *World Journal of Cardiovascular Diseases* 8.11 (2018), p. 498.

[24] Birgit Vogel et al. "ST-segment elevation myocardial infarction". In: *Nature reviews Disease primers* 5.1 (2019), p. 39.

[25] *MI common symptoms.* https://www.cbp.gov/employee-resour\ces/health-wellness/healthiercbp/reduce-your-risk-heart-disease.

[26] *Artery Occlusion.* https://www.verywellhealth.com/non-st-segment-elevation-myocardial-infarction-nstemi-1746017.

[27] Joshua Chadwick Jayaraj et al. "Epidemiology of myocardial infarction". In: *Myocardial Infarction* 10 (2019).

[28] Miha Tibaut, Dusan Mekis, and Daniel Petrovic. "Pathophysiology of myocardial infarction and acute management strategies". In: *Cardiovascular & Hematological Agents in Medicinal Chemistry (Formerly Current Medicinal Chemistry-Cardiovascular & Hematological Agents)* 14.3 (2016), pp. 150–159.

[29] Calvin Hwang and Joel T Levis. "ECG diagnosis: ST-elevation myocardial infarction". In: *The Permanente Journal* 18.2 (2014), e133.

[30] *ST-Elevation ECG.* https://www.stepwards.com/?glossary=st-elevat\-ions.

[31] *STEMI phases.* https://epomedicine.com/emergency-medicine/acute-stemi-management-mnemonic-based-approach/.

[32] Karl Heinrich Scholz et al. "Impact of treatment delay on mortality in ST-segment elevation myocardial infarction (STEMI) patients presenting with and without haemodynamic instability: results from the German prospective, multicentre FITT-STEMI trial". In: *European heart journal* 39.13 (2018), pp. 1065–1074.

[33] Jonghanne Park et al. "Prognostic Implications of Door-to-Balloon Time and Onset-to-Door Time on Mortality in Patients With ST-Segment
Elevation Myocardial Infarction Treated With Primary Percutaneous Coronary Intervention". In: *Journal of the American Heart Association* 8.9 (May 2019). ISSN: 2047-9980. DOI: 10.1161/jaha.119.012188. URL: http://dx.doi.org/10.1161/JAHA.119.012188.

[34] Patrick Wagner et al. *PTB-XL, a large publicly available electrocardiography dataset.* 2022. DOI: 10.13026/KFZX-AW45. URL: https://physionet.org/content/ptb-xl/1.0.3/.

[35] Ary L. Goldberger et al. "PhysioBank, PhysioToolkit, and PhysioNet: Components of a New Research Resource for Complex Physiologic Signals". In: *Circulation* 101.23 (June 2000). ISSN: 1524-4539. DOI: 10.1161/01.cir.101.23.e215. URL: http://dx.doi.org/10.1161/01.cir.101.23.e215.

[36] Patrick Wagner et al. "PTB-XL, a large publicly available electrocardiography dataset". In: *Scientific Data* 7.1 (May 2020). ISSN: 2052-4463. DOI: 10.1038/s41597-020-0495-6. URL: http://dx.doi.org/10.1038/s41597-020-0495-6.

[37] Deborah R Zucker et al. "Presentations of acute myocardial infarction in men and women". In: *Journal of general internal medicine* 12 (1997), pp. 79–87.

[38] Viktor Čulić et al. "Symptom presentation of acute myocardial infarction: influence of sex, age, and risk factors". In: *American heart journal* 144.6 (2002), pp. 1012–1017.

[39] *Model Evaluation.* https://databasecamp.de/en/ml/model-evalua\\tion-en.

[40] *Overfitting and underfitting.* https://www.scribbledata.io/glossary/overfitting-and-underfitting/.

[41] Swarnalatha Purushotham and BK Tripathy. "Evaluation of classifier-models using stratified tenfold cross validation techniques". In: *International conference on computing and communication systems.* Springer. 2011.

[42] Wilfred Lin. "Restorable Shortest Path Tiebreaking in fault-tolerant network design". In: (Nov. 2023).

[43] Jianxin Wu. "Introduction to convolutional neural networks". In: *National Key Lab for Novel Software Technology. Nanjing University. China* 5.23 (2017), p. 495.

[44] Daniel Jakubovitz and Raja Giryes. "Improving DNN Robustness to Adversarial Attacks Using Jacobian Regularization". In: *Lecture Notes in Computer Science.* Springer International Publishing, 2018, pp. 525–541. ISBN: 9783030012588. DOI: 10.1007/978-3-030-01258-8_32. URL: http://dx.doi.org/10.1007/978-3-030-01258-8_32.

[45] Han V. Nguyen, Magnus O. Ulfarsson, and Johannes R. Sveinsson. "Sure Based Convolutional Neural Networks for Hyperspectral Image Denoising". In: *IGARSS 2020 - 2020 IEEE International Geoscience and Remote Sensing Symposium*. IEEE, Sept. 2020. DOI: 10.1109/igarss39084.2020.9324734. URL: http://dx.doi.org/10.1109/IGARSS39084.2020.9324734.

[46] *Rpubs research*. https://rpubs.com/danhoang/1039760.

[47] Paul Gavrikov. *visualkeras*. https://github.com/paulgavriko/vis\-ualkeras. 2020.

[48] Ajay Devda et al. *Enhanced Precision in Rainfall Forecasting for Mumbai: Utilizing Physics Informed ConvLSTM2D Models for Finer Spatial and Temporal Resolution*. 2024. DOI: 10.48550/ARXIV.2404.01122. URL: https://arxiv.org/abs/2404.01122.

[49] Thibaut Perol, Michaël Gharbi, and Marine Denolle. *Convolutional Neural Network for Earthquake Detection and Location*. 2017. DOI: 10.48550/ARXIV.1702.02073. URL: https://arxiv.org/abs/1702.02073.

[50] Yong Yu et al. "A review of recurrent neural networks: LSTM cells and network architectures". In: *Neural computation* 31.7 (2019).

[51] *A Gentle Introduction to LSTM*. https://machinelearningmastery.com/gentle-introduction-long-short-term-memory-networks-experts/.

[52] *LSTM*. https://github.com/Naikvarad/Long-Short-Term-Memory-LSTM-/.

[53] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG].

[54] *Optimization techniques in machine learning*. https://medium.com/@grem-\ontmatheo22/understanding-optimization-techniques-in-machine-learning-2db67d800062.

[55] K. R. Prilianti et al. "Performance comparison of the convolutional neural network optimizer for photosynthetic pigments prediction on plant digital image". In: *AIP Conference Proceedings*. Author(s), 2019. DOI: 10.1063/1.5094284. URL: http://dx.doi.org/10.1063/1.5094284.

[56] Ilya Loshchilov and Frank Hutter. *Decoupled Weight Decay Regularization*. 2019. arXiv: 1711.05101 [cs.LG].

[57] Kushal Chakrabarti and Nikhil Chopra. *Generalized AdaGrad (G-AdaGrad) and Adam: A State-Space Perspective.* 2021. arXiv: `2106.00092 [cs.LG]`.

[58] Sebastian Ruder. *An overview of gradient descent optimization algorithms.* 2017. arXiv: `1609.04747 [cs.LG]`.

[59] D Vijendra Babu, C Karthikeyan, Abhishek Kumar, et al. "Performance analysis of cost and accuracy for whale swarm and RMSprop optimizer". In: *IOP Conference Series: Materials Science and Engineering*. Vol. 993. 1. IOP Publishing. 2020, p. 012080.

[60] Zeqing Yang et al. "A New Performance Optimization Method for Linear Motor Feeding System". In: *Actuators* 12.6 (June 2023), p. 233. ISSN: 2076-0825. DOI: `10.3390/act12060233`. URL: `http://dx.doi.org/10.3390/act12060233`.

[61] Xiangning Chen et al. *Symbolic Discovery of Optimization Algorithms.* 2023. arXiv: `2302.06675 [cs.LG]`.

[62] Purnendu Mishra and Kishor Sarawadekar. "Polynomial learning rate policy with warm restart for deep neural network". In: *TENCON 2019-2019 IEEE Region 10 Conference (TENCON)*. IEEE. 2019, pp. 2087–2092.

[63] Zeke Xie, Issei Sato, and Masashi Sugiyama. "Understanding and scheduling weight decay". In: *arXiv preprint arXiv:2011.11152* (2020).

[64] Zhiyuan Li and Sanjeev Arora. "An exponential learning rate schedule for deep learning". In: *arXiv preprint arXiv:1910.07454* (2019).

[65] Usha Ruby and Vamsidhar Yendapalli. "Binary cross entropy with deep learning technique for image classification". In: *Int. J. Adv. Trends Comput. Sci. Eng* 9.10 (2020).

[66] *Binary Cross Entropy Loss.* `https://www.askpython.com/python/exampl-\es/binary-cross-entropy-loss`.

[67] Hongfang Yuan et al. "ResD-Unet Research and Application for Pulmonary Artery Segmentation". In: *IEEE Access* 9 (2021), pp. 67504–67511. DOI: `10.1109/ACCESS.2021.3073051`.

[68] Yage Zhao et al. "Compact Modeling of Advanced Gate-All-Around Nanosheet FETs Using Artificial Neural Network". In: *Micromachines* 15.2 (Jan. 2024), p. 218. ISSN: 2072-666X. DOI: `10.3390/mi15020218`. URL: `http://dx.doi.org/10.3390/mi15020218`.

[69] Miriam Nnadili et al. "Surfactant-Specific AI-Driven Molecular Design: Integrating Generative Models, Predictive Modeling, and Reinforcement Learning for Tailored Surfactant Synthesis". In: *Industrial &amp; Engineering Chemistry Research* 63.14 (Apr. 2024), pp. 6313–6324. ISSN: 1520-5045. DOI: 10.1021/acs.iecr.4c00401. URL: http://dx.doi.org/10.1021/acs.iecr.4c00401.

[70] *Optimizing large language models harnessing hyperparameters.* https://www.linkedin.com/pulse/optimizing-large-language-models-harness-\ing-hyperparameters/.

[71] Miriam Steurer, Robert J Hill, and Norbert Pfeifer. "Metrics for evaluating the performance of machine learning based automated valuation models". In: *Journal of Property Research* 38.2 (2021), pp. 99–129.

[72] *Learning curves in machine learning.* https://wandb.ai/mostafai\brahim17/ml-articles/reports/A-Deep-Dive-Into-Learning-Curves-in-Machine-Learning--Vmlldzo0NjA1ODY0.

[73] *Validation metrics.* https://medium.com/@h.jurza/mastering-valid\\ation-metrics-evaluate-your-models-like-a-pro-4fda25abe3cb.

[74] *Understanding binary classification metrics.* https://www.linkedin.com/pulse/understanding-binary-multiclass-classification-metrics-anjil-adhikari-44quf/.

[75] Pawan K. Jha, Utham K. Valekunja, and Akhilesh B. Reddy. "SlumberNet: deep learning classification of sleep stages using residual neural networks". In: *Scientific Reports* 14.1 (Feb. 2024). ISSN: 2045-2322. DOI: 10.1038/s41598-024-54727-0. URL: http://dx.doi.org/10.1038/s41598-024-54727-0.