Towards Myocardial Infarction Diagnosis: An Investigation into Machine Learning and Deep Learning Algorithms using ECG Data

by

Husne Mubarak
20101336
Rafid Ahnaf
19201143
Fahim Ahsan
20101545
G. M. Refatul Islam
20101482
Faiza Bushra
20101554

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science

Department of Computer Science and Engineering
School of Data and Sciences
Brac University
September 2023

Declaration

It is hereby declared that

1. The thesis submitted is my/our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

Husne Mubarak
20101336

Rafid Ahnaf
19201143

Fahim Ahsan
20101545

G. M. Refatul Islam
20101482

Faiza Bushra
20101554

Approval

The thesis/project titled "Towards Myocardial Infarction Diagnosis: An Investigation into Machine Learning and Deep Learning Algorithms using ECG Data." submitted by

1. Husne Mubarak(20101336)
2. Rafid Ahnaf(19201143)
3. Fahim Ahsan(20101545)
4. G. M. Refatul Islam(20101482)
5. Faiza Bushra(20101554)

Of Summer 2023 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on September 17, 2023.

Examining Committee:

Supervisor:
(Member)

Mr. Nabuat Zaman Nahim
Lecturer
Department of Computer Science and Engineering
Brac University

Co-Supervisor:
(Member)

Mr. Rafeed Rahman
Lecturer
Department of Computer Science and Engineering
Brac University

Head of Department:
(Chair)

Sadia Hamid Kazi, PhD
Chairperson and Associate Professor
Department of Computer Science and Engineering
Brac University

Thesis Co-ordinator:
(Chair)

Dr. Md. Golam Rabiul Alam, PhD
Professor
Department of Computer Science and Engineering
Brac University

Abstract

Myocardial Infarction (MI) is a severe and potentially fatal heart condition caused by heart muscle injury due to lack of blood supply. Early detection and medical intervention can significantly reduce the mortality rate of MI. Electrocardiography (ECG) signals are commonly used to diagnose MI, but the process is susceptible to human error. This research aims to utilize machine learning and deep learning techniques to detect MI based on ECG signals and other biophysical factors. The incorporation of these factors tend to improve the predicted accuracy and resilience of the model. The performance of these machine learning and deep learning methods will depend on the carefully selected features and ECG signals. The anticipated result is expected to be a robust ML and DL-based diagnostic algorithm that can adapt to individual patient profiles, offering distinctive personalized risk assessments and magnifying the early detection of possible MI cases. Using machine learning to detect MI can enhance the accuracy and efficiency of diagnosis and identify individuals at high risk of developing MI for preventative measures. Moreover, using Deep Learning techniques that utilize image processing and visualization can help us understand why the model is making a decision. This decision taking pattern can later help us narrow down the type of MI the patient has. This study, overall, emphasizes the potential of machine learning and deep learning to improve our understanding of the relationship between medical conditions, biophysical factors and disease development.

Keywords: Myocardial Infarction; Machine Learning; ECG; Cardiovascular disease; K-Nearest Neighbour; SVM; Feature extraction; Feature selection; R-peak; QRS Complex; Lead; ECG Filtering; ECG Normalization, Visionary V2, Image, GradCAM, LIME, CNN, ResNet, Image Processing, Data Augmentation, Ensemble models, Signal Processing.

Acknowledgement

Firstly, all praise to the Great Allah for whom our thesis have been completed without any major interruption.
Secondly, to our supervisor Mr. Nabuat Zaman Nahim sir and our co-supervisor Mr. Rafeed Rahman sir for his kind support and advice in our work. They helped us whenever we needed help.

Table of Contents

List of Figures

Nomenclature

The next list describes several symbols & abbreviation that will be later used within the body of the document

AI Artificial Intelligence

AUG Area under the receiver-operating-characteristic curve

CAD Computer-aided diagnosis

$CKMB$ Creatine Kinase-Myoglobin binding

CVD Cardiovascular disease

DCT Discrete cosine Transform

DT Decision Tree

DWT Discrete Wavelet Transform

ECG Electrocardiogram

EMT Empirical Mode Transform

KNN K-nearest Neighbour

LDH Lactate Dehydrogenase

LPP Locality Preserving Projection

MI Myocardial Infarction

ML Machine Learning

OCR Optical Character Recognition

PNN Probabilistic Neural Network

SVM Support Vector Machine

SWT Stationary Wavelet Transform

Chapter 1

Introduction

Myocardial infarction, commonly called "heart attack," is a severe and potentially terminal cardiovascular disease that tends to occur when the heart muscle is not supplied sufficient blood due to impairment or blockage. The term "myocardial infarction" comes from the Greek words "myo," "cardial," and "infarction," which translates to the death of heart muscle tissue. This disease is caused by the complete or partial obstruction of the coronary arteries responsible for providing oxygenated blood to the cardiac/heart muscle. The Cardiac muscle tissue may die if the heart muscle is deprived of blood flow, leading to changes in the regular cardiac conduction system and potentially severe arrhythmias. Given the condition of MI is severe, the affected heart muscles may introduce increased mechanical stress to the heart during muscle contraction for pumping blood which may result in structural and conformational changes like left ventricular remodeling.

Unconsciousness, breathing problems, and central chest pain constitute the symptoms of myocardial infarction. But it is important to note that many victims fail to exhibit the symptoms of myocardial infarction, earning it the name "silent heart attack." Additionally, factors such as biophysical conditions, previous medical conditions, and lifestyle choices can contribute to the development of myocardial infarction. Whether the lifestyle choices that cause high risk of MI are modifiable or not, they can be divided into two segments. Modifiable choices include hypertension, stress, anxiety, diabetes mellitus, obesity, alcohol and smoking where as non-modifiable choices are age, sex and family history.

Given the high mortality rate associated with myocardial infarction, early detection is crucial for providing timely treatment and reducing fatalities. One of the most effective and cost-effective techniques for detecting myocardial infarction early is the analysis of electrocardiogram (ECG) signals, which record the heart's electrical activity. It can also be detected by the presence of bio-marker enzymes in blood like, Troponin I, high CKMB, LDH. These markers are only present in blood when there is a possible chance of cardiac muscle decay that points towards MI. Even though these reports generate proper results, it has been seen that incorrect or non-automated detection of cardiac abnormalities can bring about the death of patients suffering from myocardial infarction. Research suggests that in cases of individuals presenting with significant symptoms of cardiac disease, cancer, and infection, 10% are misdiagnosed or are already at an advanced stage of the disease. Further-

more, 53.9% of these individuals have either become disabled or passed away as a result of medical errors or incorrect diagnosis. As a result, automated detection consisting of high class accuracy and precision is a pre-requisite for assisting medical practitioners with quick and accurate diagnosis of myocardial infarction using ECG signals.

1.1 Problem Statement

Myocardial infarction (MI) is a serious medical condition which occurs when the blood supply to the heart muscle experiences a blockage. Plaque accumulation in the coronary arteries often causes this obstruction, which then results in the formation of a blood clot. This clot might prevent blood from flowing normally to the heart. After the clot has fully blocked up the artery, the oxygenated blood supply to the heart muscle is cut off, and the heart muscle tissue begins to die. The death of heart muscle cells is called an Infarction, hence the name Myocardial Infarction. The outcome of MI varies depending on the extent of the damage and the promptness of treatment. In some cases, the damage to the heart muscle can be reversed with prompt and effective treatment, but in other cases, the damage may be permanent, leading to heart failure or other serious complications. Traditional methods for detecting MI can include diagnostic tests such as electrocardiograms (ECGs) and blood tests. Some limitations of these methods include:

- Both ECGs and blood tests are dependent on the skill and experience of the technician/doctor reading them, which can lead to variations in the interpretation of results.
- Blood tests, such as troponin tests, can take several hours to return results and may not be sensitive enough to detect early stages of myocardial infarction.
- These diagnostic procedures can result in either false negatives or false positives because they cannot account for all possible risk factors

This prompts the inquiry -
"How to detect Myocardial Infarction in people of all genders and ages with more accuracy and efficiency within a short time than the traditional methods by utilizing modern machine learning and deep learning approaches?"

1.2 Research Objective

The principal objective of this research is to investigate the feasibility of using machine learning and deep learning methods in the diagnosis and timely detection of myocardial infarction (MI) in an accurate manner using ECG signals. Specifically, the following objectives will be addressed:

- To identify and analyze the various factors that can contribute to the development of MI, including biophysical conditions, previous medical conditions, and lifestyle choices.
- To investigate the prevalence of "silent" MI and its impact on mortality rates.
- To explore a simple but robust preprocessing methods that can efficiently reduce noise and enhance feature extraction.
- To develop and evaluate machine learning models that can accurately classify ECG signals as indicative of MI or not.
- To evaluate the viability of using deep learning algorithms and image classification models to identify a pattern among MI and non-MI ECG images.
- To perform a performance comparison of the developed machine and deep learning models with other existing models used for MI detection.
- To assess the potential of the proposed machine learning approach, improving accuracy and speed of MI detection, and ultimately reduce mortality rates.
- To implement eXplainable AI (XAI) techniques to understand model decisions.
- To evaluate the interpretability of the model and its generalizability on diverse population.
- To explore the use of other types of data and modalities in conjunction with ECG signals, such as biophysical signals and medical history, to improve the accuracy and efficiency of MI detection.

By addressing these objectives, this research aims to improve our understanding of the factors contributing to the development of MI and develop more effective strategies for early detection and reduction of mortality rates. The proposed machine learning approach has the potential to assist medical practitioners with quick and accurate diagnosis of MI from ECG signals, which can ultimately help to save lives.

Chapter 2

Literature Review

Electrocardiogram (ECG) is the electrical activity of the heart depicted in a graphical form. It provides an individual with thorough information about the state of a person's cardiac muscles through the presentation of electrical impulses generated from each cardiac cycle. One single ECG wave comprises of a P wave, Q wave, T wave, QRS complex and ST segment. For detecting MI using ECG, attention needs to be given to abnormality seen in the Q wave, T wave and ST segments elevation. In order to extract data from ECG reports, it needs to be digitized. Digitizing ECG reports can provide essential geomorphologic and functional details for medical and analytical research purposes. So far, numerous methods have been formed to digitize the reports. In a study conducted by Naam [6], the utilization of pixel indexing techniques was employed to examine the similarity between original and digital ECG recordings. Results indicated a high degree of similarity, ranging from 85-90%. Additionally, the use of Optical Character Recognition (OCR) was utilized to automate the extraction of demographic information available in the waveform. Separate research by Ali [3] employed image processing techniques to obtain biopotential signals from both single as well as multi-channel records. The authors of the research study [17] also proposed a new model for extracting the pixel values and their corresponding amplitude from ECG signals. Utilizing MATLAB, this model was able to read data from all twelve lead ECG signals present in medical reports. This process included examining the ECG signal report, cropping of the area of interest, binarization and pixel-to-vector conversion. The scanned image was also resized to an appropriate resolution for optimal data extraction. The correlation between original and digitized data was determined to be 98%.

ECG wave-forms of patients, being one of the easily accessible data, is mostly used for the early prognosis of Myocardial Infarction. However, due to the silent nature and seldom show of symptoms of this fatal cardiovascular disease, it is hard to detect in its early stages. This makes the whole process of using ML for the automatic detection more challenging. According to research by Rai et al.[20], it has been established that cardiologists are able to recognize cardiac abnormalities, specifically ST-segment elevation, in ECG signals associated with MI up to 82% of the time. However, it is acknowledged that manual detection methods can lead to errors and potentially result in loss of patient life. Therefore, there is a requirement for efficient, accurate and automated methods for detecting MI. In light of

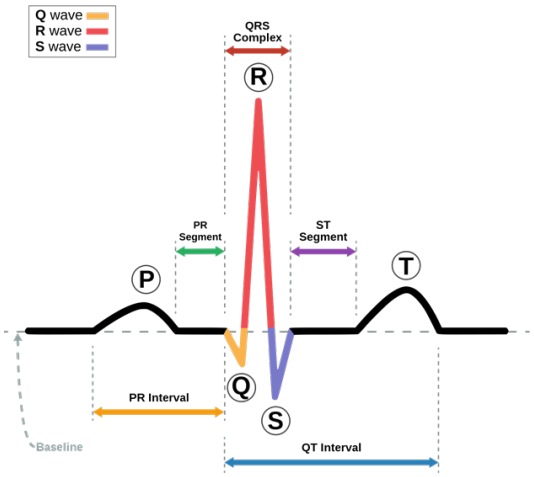


Figure 2.1: Labelled ECG Trace

this, the application of ML techniques to analyze and diagnose cardiac abnormalities has been the subject of extensive investigation. It should be noted that the performance of traditional ML-based classifiers is heavily dependent on the preprocessing of ECG signals and the extraction of various features. In an effort to enhance the performance of traditional ML techniques in the detecting myocardial infarction (MI), researchers have implemented a variety of additional features, such as wavelet transformation, principal component analysis, linear discriminant analysis, and Hilbert transformation. However, the extraction of these features using traditional ML methods often required significant time and effort. To address this issue, deep neural networks have been introduced as a means of extracting features without the need for human intervention or additional tools, thus reducing the time required. Nevertheless,a new method for categorizing MI was proposed by Sharma et al.[10] which used a filter bank referred to as Optimal Biorthogonal. An accuracy rate of 99.62% and 99.74% was observed respectively on noised and denoised ECG signals upon use of the K-Nearest Neighbor (KNN) classifier for categorization.

The general approach to using machine learning (ML) to identify myocardial infarction (MI) involves preprocessing the ECG waveforms through techniques such as filtering, segmentation, calculation of time intervals, and determination of the gradient of the time series data. This generated data is then utilized as input for the proposed algorithm.

The authors of this research [15] brought forth the idea of the inter-relation between variations in cardiac troponin I levels based on age, gender and the interval between samples in the patients suspected to have myocardial infarction. They proposed that the incorporation of said variables through machine learning can improve the diagnostic approaches for the disease. The authors selected these features because these are objective, automatically collected by electronic hospital records and are included in current diagnostic methods. Some additional characteristics were also added that included medical history and lifestyle choices such as history of coronary artery disease, diabetes, smoking, hypertension, etc. This work represents the effec-

tive demonstration of use of ML in guiding clinical decision making in patients with suspected coronary artery diseases by generating positive and negative predictive values, providing an individualized and objective assessment of MI by testing on over 11,000 patients with 3,013 records used for training and 7,998 records used for testing. The results showed that the authors' proposed algorithm (MI3) was very well calibrated and had a sensitivity of 97.8% in determining patients with low-risk or high-risk MI along with area under the receiver-operating-characteristic curve (AUG) being 0.963.

In a study conducted by Liu [19], researchers successfully developed a cutting-edge DLM for AMI(Acute Myocardial Infarction) detection, showcasing its exceptional performance in ECG analysis. A highly accurate diagnostic support tool for detecting AMI using a 12-lead ECG. The author's comprehensive and a retrospective study comprised 737/287 STEMI/NSTEMI patient ECGs totaling 1051/697 , validated by coronary angiogram (CAG), as well as 140,336 ECGs from 76,775 non-AMI patients in the emergency department. The authors meticulously trained and validated their deep learning model (DLM) on 80% and 20% of the ECGs, respectively. Their efforts achieved an impressive AUC(area under the curve) value of 0.986 for AMI detection. Notably, the DLM surpassed the performance of the conventional cardiac troponin I (cTnI) test, boasting an AUC of 0.986 compared to 0.969. However, it should be taken into account that the study did not assess the effectiveness of the deep learning algorithm in a real-world clinical setting.

The study's findings suggest that implementing the DLM as a diagnostic tool could have significant implications for enhancing AMI diagnosis, minimizing the risk of delayed or incorrect diagnoses. In addition, the authors highlighted another study that introduced an AI-enabled electrocardiogram (AI-ECG) specifically designed for screening low-risk chest pain patients in the emergency department. This AI-ECG holds potential in reducing unnecessary hospital admissions and healthcare costs, while simultaneously improving patient outcomes.

Class imbalance is a common challenge in biomedical datasets, necessitating the use of novelty detection algorithms. Synthetic Minority Oversampling Technique (SMOTE) is employed here to address the issue. SMOTE is an important technique that can not only handle class imbalance, but also prevent over-fitting by diversifying the train dataset and enhance decision boundaries which can result in classifying minority classes correctly as well. In the study demonstrating promising use cases of machine learning that is explainable in the realm of CVD prediction [16], SMOTE is utilized on high-level architecture models such as CNN, RNN and decision-tree based XGBoost model.

The CNN model receives the extracted electrocardiogram (ECG) signal features, as well as auxiliary data such as age and sex, and returns a binary output indicating the risk of Acute Myocardial Infarction (AMI). The model has 12 layers, including convolutional layers, dropout layers, a global max pooling layer, and dense layers and achieved an accuracy of 89.8%, F1 score of 89.0%, and AUROC of 90.7%. The RNN model, designed for time-series data, is applied to the static ECG data in this case and gains accuracy of 84.6%, F1 score of 82.8%, and AUROC of 82.9%. The

XGBoost model, a tree-based extreme gradient boosting model, is also developed for AMI prediction. The model achieves high accuracy, F1 score, and AUROC of 97.5%, 97.1%, and 96.5%, respectively. XGBoost handles feature selection automatically, and the model performs similarly even when trained without the age and gender variables. Using game theory, Shapley values are calculated to learn more about the model's estimates and figure out which parts of the model are most important.. Shapley values provide explanations for every individual prediction and reveal the contribution of each feature. Age, ACCI, and QRS duration are identified as the most important features for the prediction of AMI. The inclusion of age and sex features significantly improves the prediction results.

In another work [21], the authors proposed the use of Computed Aided Diagnosis (CAD) systems to categorize ECG signals and prepare a cost effective system that can determine ECG abnormalities associated with MI. The features were extracted after being preprocessed and using Pan-Tompkins algorithm and the non-linear features were isolated and ranked using Student's t-test. Then, the best ranked features were input into Decision Tree (DT), k-Nearest Neighbor (KNN), Probabilistic Neural Network (PNN) and Support Vector Machine (SVM) classifiers in order to categorize MI and normal classes. The authors employed Discrete Wavelet Transform (DWT) to decompose 200 raw ECG signals and remove noise and R peak as preprocessing steps. Next, the DWT exponents were used to extract around 45 nonlinear features, which were ranked using t-distribution to find the more significant features which were then used by the classifiers such as KNN, SVM, PNN and DT which resulted in SVM having the best specificity, accuracy and sensitivity values of 93.81%, 97.96% and 98.89% respectively with a ten-fold cross-validation strategy.

Baloglu, aimed to develop a model utilizing deep convolutional neural (CNN) networks to categorize or classify various myocardial infarctions using multi-lead ECG signals in his research [12]. The paper presents a deep CNN model that can classify MI with multi-lead ECG signals. The proposed model achieved an accuracy of more than 99% in classifying MI. The study's main strength is its ability to successfully identify between 10 different MI types and regular ECG signals with more than 99% classification performance using the suggested CNN approach where the highest classification performance resulted for lead V4 with an accuracy of 99.78

Another paper on MI detection[8] uses a novel algorithm called ML-CNN based on Real time Multi lead electrocardiogram (ECG). By leveraging sub 2-D convolution and a LAP strategy, the ML-CNN proves to be more suitable for multi lead ECG analysis than traditional 2-D CNNs. The algorithm demonstrates promising results, achieving high sensitivity (95.40%), specificity (97.37%), and accuracy (96.00%) in MI detection, comparable to methods utilizing complex hand-designed features. Moreover, the algorithm exhibits brief running times on both PC and ARM embedded platforms, making it applicable in real-time systems and potentially mobile devices like wearable or portables, showcasing its computational efficiency. Future work involves parameter tuning, developing a comprehensive hardware framework, and establishing a lightweight wearable or portable system based on our algorithm.

In a research focusing on decomposition of ECG signals [5], the authors performed a comparative analysis of automated characterization of MI by decomposition of ECG signals using 3 methods which are Discrete Cosine Transform (DCT), Empirical Mode Transform (EMT) and Discrete Wavelet Transform (DWT). The obtained coefficients after decomposition are then processed through the use of Locality Preserving Projection (LPP) reduction method and subsequently ranked based on F-value. In this work, the authors mentioned that most prior works before them used two classes (normal and MI) for categorization whereas their work provides 3 classes (normal, cardiovascular dis-ease, MI) for categorization, bringing a novelty in disease detection. The ECG signals are decomposed using DWT, EMT and DCT after preprocessing them to remove noise and the features that are obtained from the decomposed coefficients are further reduced using LPP method. The authors then used Analysis of Variance (ANOVA) to rank the obtained features for feature selection, which were then used by the KNN classifier. The data used for this work featured 92,273 ECG signal beats in total consisting of 10,546 normal beats, 41,545 cardiovascular disease beats and 40,182 MI beats. After the proper features were fed into the KNN classifier with a variation value of k=5-10, the results showed that k=5 value yielded the best result with sensitivity, accuracy and specificity values of 99.7%, 98.5% and 98.5% respectively upon use of only seven features from DCT method.

Another similar study [14] explored the traditional machine learning techniques like Support Vector Machines(SVM) and Decision Trees, as well as a new technique known as Capsule Neural Networks. The author compares different machine learning models on PTB and MIT-BIH datasets, aiming to establish a baseline accuracy. Techniques such as Support Vector Machines (SVM), Convolutional Neural Networks (CNN), Artificial Neural Networks (ANN), and K-Nearest Neighbors (KNN) are applied where SVM achieves high accuracy with a polynomial kernel on the PTB dataset and similar results on MIT-BIH. During SVM implementation, the author used a polynomial kernel with C and gamma values of 2 and 4. In this way, the author generated higher accuracy on all datasets. Using the PTB ECG dataset, the author generated 0.9 accuracy, 0.88 precision and 0.89 recall. On the other hand, using MIT-BIH ECG dataset, the SVM gave an output of 0.88 accuracy, 0.85 precision and 0.87 recall. During the implementation of KNN algorithm, the author labeled the data from values starting from 10 to 50, 0.88 accuracy 0.86 precision and 0.87 recall was generated when the value was 10. ANN with ReLU activation is used and shows slightly lower accuracy on MIT-BIH due to multi-class classification. CNN, trained with varying epochs on PTB, achieves slightly lower accuracy than ANN. KNN's accuracy decreases with larger K values, and values below 10 are considered. The accuracy of KNN improves with Principal Component Analysis (PCA). CapsNet, a model capturing hierarchical relationships, is trained on both datasets but has lower accuracy compared to other models. CapsNet variations with modified parameters show no significant impact on accuracy. The addition of PCA before training CapsNet does not improve accuracy, indicating its limited impact. Overall, SVM, ANN, CNN, and KNN outperform CapsNet on ECG datasets, with CapsNet displaying relatively lower accuracy. Published research on CapsNet primarily focuses on image datasets, not numerical datasets.

The rise of wearable devices like Fitbits and Apple Watch, that are capable of recording or tracking ECG data, is expected to drive significant research advancements in this field. The increased availability of ECG datasets from these devices will provide ample data for ML models to learn from. Furthermore, the powerful hardware of these devices enables the execution of computationally expensive models like CNN and ANN. Ultimately, this much development is anticipated to enable earlier detection of myocardial infarction (MI), saving lives with early detection.

In the research paper [9], according to the authors, diagnosing IMI accurately using traditional methods such as electrocardiogram (ECG) analysis can be challenging. To address this issue, the authors propose a new approach that utilizes SWT for extraction of features from ECG signals and subsequently applies ML techniques to classify the signals as either standard or indicating IMI. They first explain the SWT process, a type of wavelet transform particularly well-suited for analyzing signals that are not changing over time. They then explain how they applied SWT to ECG signals to extract features such as the energy and entropy of the signs. After that, they are being used as input for the algorithm, and they are trained to distinguish between standard signals and those that indicate IMI. They also tested their approach on a dataset of ECG signals from patients with IMI and compared the results to a traditional ECG analysis method. They found that their process achieved higher accuracy in detecting IMI, with a sensitivity of 95.24% and a specificity of 95.65%. The authors also evaluated the algorithm's performance on a dataset of ECG signals from patients with other types of heart disease. They found that it was able to classify these signals accurately as well. Overall, the authors conclude that their approach is promising for accurately detecting IMI using ECG signals. They suggest that further research is needed to evaluate the algorithm's performance in a larger population and compare it to other machine learning-based methods.

In summary, the paper presents an approach for detecting inferior myocardial infarction (IMI) using a combination of stationary wavelet transform (SWT) and machine learning techniques. The authors applied SWT to ECG signals to extract features such as energy and entropy and then used these features as input to a machine-learning algorithm. The algorithm was trained to classify the signals as usual or indicate IMI. The authors found that their approach had higher accuracy detecting IMI than the traditional ECG analysis method. They also found that the algorithm could accurately classify signals from patients with other types of heart disease.

According to the research done by Jian et al.[18], the electrocardiogram (ECG/EKG) effectively detects myocardial infarction by recording electrical signals traveling through the heart, and can detect abnormalities in Q-wave, ST-wave and T-waves in precordial leads V1-V6. Recent advances in deep-learning and AI have made it possible to reduce the workload of ECG-screening for MI. Convolutional neural networks(CNN) uses a deep learning method that involves ECG-classification after denoising, reconstruction, annotation and labeling, data-compression and data-synthesis. A 2020-study proposed an algorithm with a 95.76% inter-patient classification accuracy, 89.2% recall and 0.92 F1-score, demonstrating the potential of

AI in accurately detecting MI. Starting with Experimental ECG-datasets, used in signal-preprocessing, then for dataset rearrangement, deriving MI locating and MI detecting datasets in the process, moving on to training models(N-Net and MSN-Net), evaluating based on performance metrics and comparing the evaluations with the state-of-the-art. Model-training gets terminated if there is no improvement after reaching continuous epoch limit keeping accuracy, sensitivity, specificity and F1-score as performance metrics which are measured and utilized as indexes.

Coming to the results, the ideal structures are determined by comparing the average accuracy. The Nine filters and single-scale features produced a 95.76% average accuracy using N-Net, whereas nine filters and four-scale features produced a 60.49% average accuracy when using MSN-Net for detecting MI. Note that at low filter numbers, the accuracy is elevated due to the number of filters. Evaluating the indexes, AUC and substantial differences as determined by performing U-test, both the MSN-Net and N-Net greatly outperformed the previous best results(state-of-the-art) in identification of MI. Though this verified accuracy is high, there may still be issues that haven't been explored but can potentially decrease the accuracy.

Chapter 3

Working Plan

3.1 Methodology

In the initial phase of this research, a comprehensive literature review has been conducted to identify the existing gaps and challenges in our concerned domain, consequently leading to the problem identification and formulation of a structured research plan. Once the problem statement is clearly stated, data collection begins. Various data sources are combined to create a solid dataset specifically designed for this research. Along with secondary numerical data sources, an image dataset is created through careful data selection and pre-processing stages that include data cleaning, normalization, and transformation. The next critical step is feature extraction and selection, which involves extracting relevant characteristics from the dataset that are statistically most suggestive of the classes being studied. These steps help to ensure that the extracted data has been optimized for computer analysis. For evaluation and enhancement of the performance of the machine and deep learning models, the data are then divided into training, validation, and test sets. The training of various ML and DL models with the motive of MI classification then makes use of these chosen features. Our study then explores different Explainable Artificial Intelligence (XAI) approaches, more notably LIME, GradCam, and Integrated Gradient, after model training in order to interpret model decisions. By improving the trained models' transparency and comprehending their decision-making justification, these XAI techniques increase the accountability of AI solutions. Once the models and their interpretations are prepared, the results are thoroughly examined using a variety of metrics to gauge the models' effectiveness, dependability, and interpretability. The data are then used to draw conclusions, which not only confirm the viability of the methodology used but also offer vital information about the implications and future directions of this research.

3.2 Work Plan

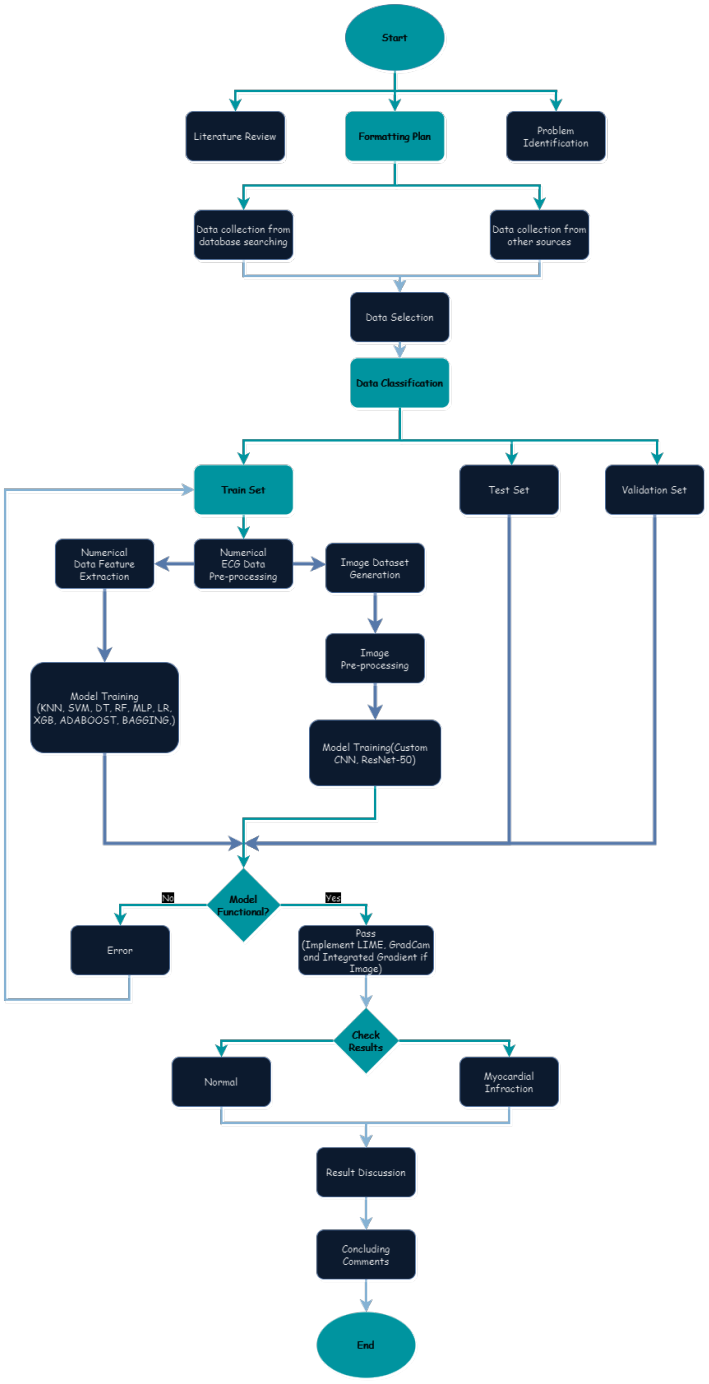


Figure 3.1: Workflow

Chapter 4

Description of the Dataset

4.1 Overview of PTBDB Dataset

For our research on the detection of Myocardial Infarction (MI) using ECG, our primary source of data was the PTBDB dataset. The PTB Diagnostic ECG database (PTBDB) is one of the most widely used datasets for evaluating ECG waveforms and algorithms. The main contributor of this dataset was Physikalisch-Technische Bundesanstalt (PTB), the National Metrology Institute of Germany. This data has been publicly available for research and benchmarking for complex physiological signals.

The PTBDB dataset is a collection of ECG signals of 549 records having both male and female subjects (between 17 and 87 years of age with an average of 61.6 years) which were collected by utilising a custom personal tape recorder not intended for commercial use. There were 16 channels in total among which 14 were for ECG, 1 for respiration and 1 for line voltage. Each ECG record includes wave signals of 15 leads; 12 conventional leads (i, ii, iii, avr, avl, avf, v1, v2, v3, v4, v5, v6) and 3 Frank leads (vx, vy, vz). The signals had a sampling rate or clock frequency of 1000 ticks per second. Meaning each second of the ECG record had 1000 data points coming from a single lead that combinedly formed the signal.

The subjects were classified into multiple groups having different heart conditions from which we only filtered out the patients with Myocardial Infarction and Healthy conditions only. Along with the ECG signal, the dataset also contained the detailed clinical summary about the patients which included, age, sex, diagnosis, medical history, medication and relevant medical conditions.

4.2 Leads: Their Significance and Explanation

One of the essential factors to consider when working with ECG signals is understanding the readings of each lead. It is the primary tool to detect MI. ECG leads provide views of the activity of the heart from various angles. Charges are then categorized into Chest Leads (V1 through V6): and Limb Leads (I, II, III, aVR, aVL, aVF). As we have said above, we currently use 15 tips from each ECG record. 12 are conventional leads, and 3 are frank leads. By examining all these leads, a doctor can quickly identify which portion of the heart is affected by MI [22]. The doctor looks for specific patterns in the ECG leads to classifying MI. The most commonly recognized pattern is ST-segment elevation, but other patterns are used to recognize MI.

The most common pattern for detecting MI is the flat ECG segment between the end of the S wave and the start of the T wave, where the S wave is the second component of the QRS complex. ST-segment elevation in different leads shows different kinds of MI. For example: ST-segment in II, III, and aVF shows inferior MI(refers to the affected inferior part of the heart.), ST elevation in lead I, aVL, V5, and V6 shows Lateral MI(the heart attack that affects the lateral wall of the heart); finally, ST elevation in V3 and V4 shows Anterior MI(affecting the anterior of the front side of the heart after a heart attack).

Another pattern is the Abnormal QRS complex, which consists of 3 waves: Q wave, R wave, and S wave, and these individual waves represent the electrical impulses as it spreads through the ventricles and triggers a contraction. [4] However, these patterns are not definitive proof. There are other tests all as well, such as blood tests. Also, different noises can be added while recording due to different factors. For example, electrical interference and heart position change due to breathing, body movement, and motion artifacts. Muscle noise can also be added while recording the ECG.

ECG can vary for different patients due to various factors, such as sex, body composition, age, and other health conditions. All these varieties make it difficult for an ML model to learn general patterns that accurately reflect all the patients. If the patient has a different other heart condition, then it becomes more unclear to the ML model to learn from the training data. Despite all these, certain aspects of ECG data are standardized and consistent. For example, R-peak, the highest peak of the QRS complex, is a standard feature in all kinds of ECG reading(excluding noise). Other than R-peak, the standard deviation, mean, skew, kurtosis and average provide consistent and quantifiable mean and average.

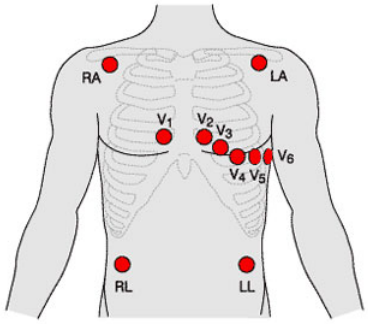


Figure 4.1: ECG Lead Positioning

4.3 Understanding ECG in the Context of PTBDB

From the PTBDB dataset, we can fetch the ECG of each of the patients using existing Python Libraries. Using the electrical data points recorded at every millisecond, we can formulate a raw ECG signal by plotting it into a graph. The raw ECG signal initially holds the information gathered by the input channels used on the patient. The time range can be adjusted according to the use case. (Figure: 4.2)

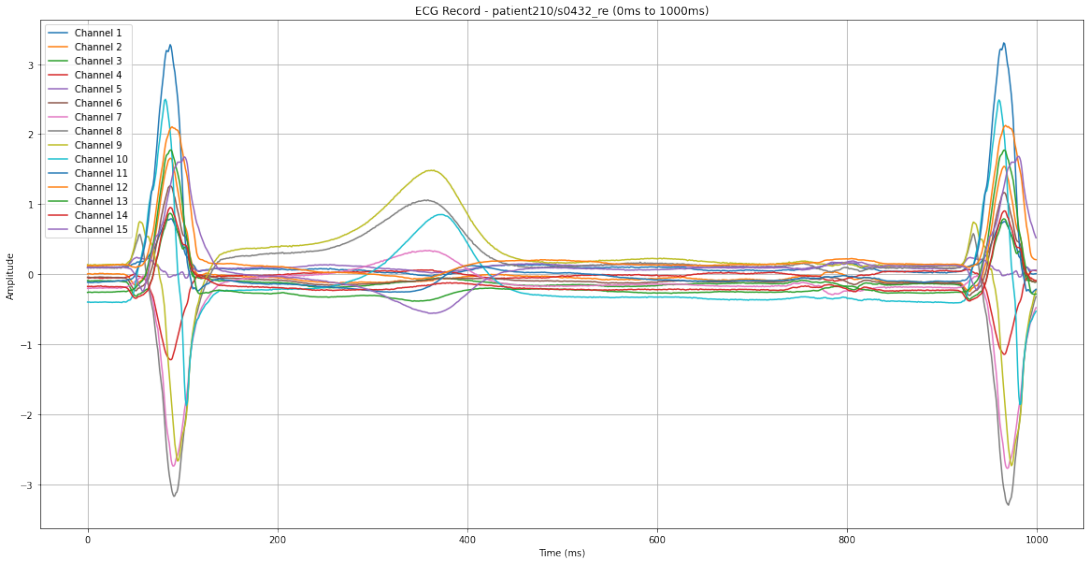


Figure 4.2: Channel Readings from ECG

Later these channels can be labeled properly to identify the conventional and Frank leads. Taking a bigger time frame of the graph gives a better view of the overall signal. (Figure 4.3)

From there to get a more clear picture, signals coming from each of the leads can be separated. (Figure 4.4). These signals coming from each of the signals are unfiltered and have various factors contributing to additional noise in them. Proper placements of the leads on the patient, patient's health condition during the recording as well as temperature and humidity of the environment affect the reading of the ECG.

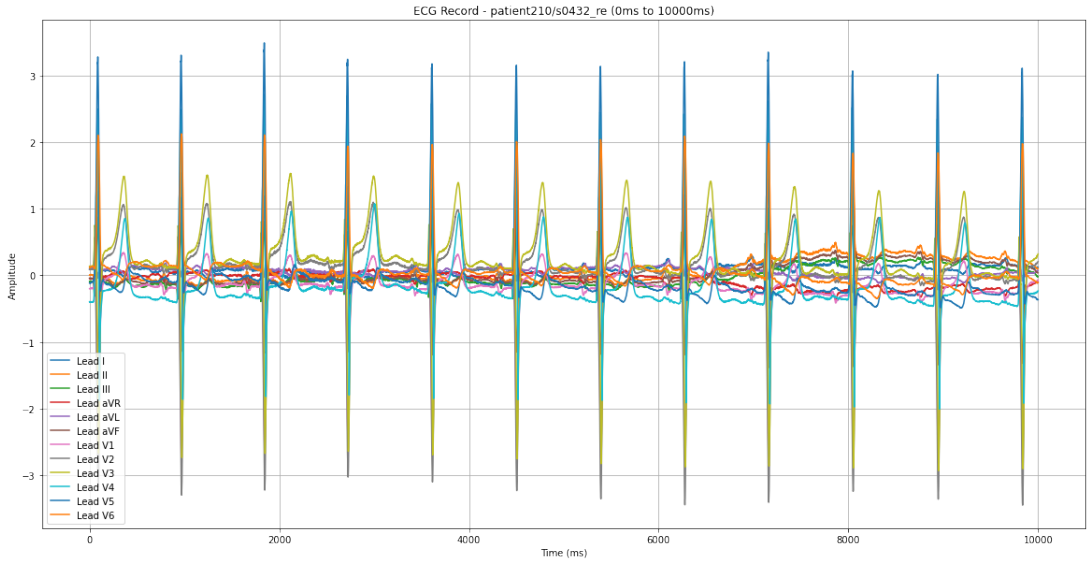


Figure 4.3: Lead Readings from ECG

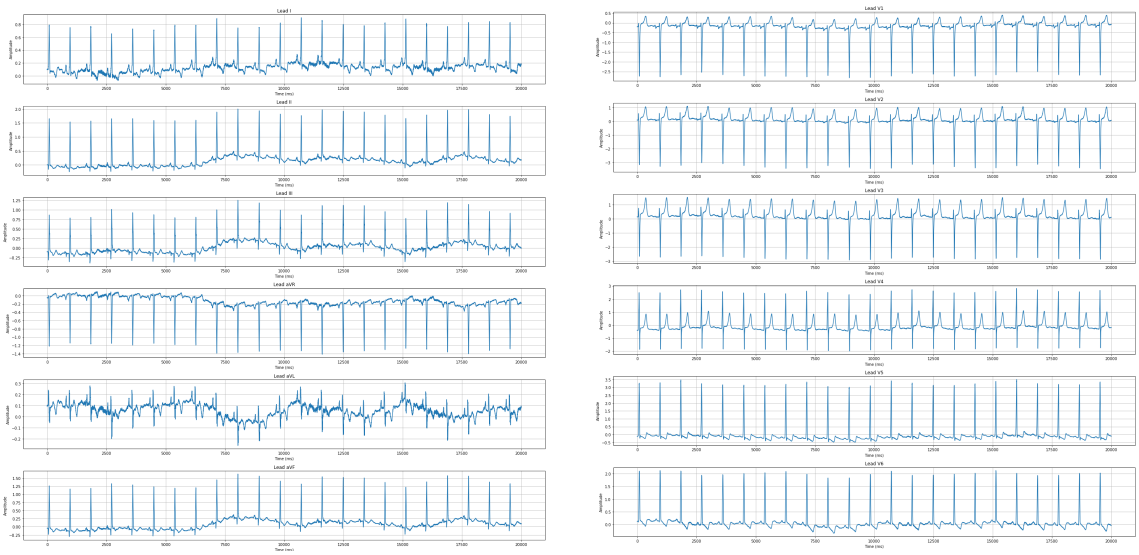


Figure 4.4: Individual Lead readings

4.4 Utilizing Existing Libraries for ECG Analysis

In order to analyze and process this signal properly, this data needs to be filtered and normalized. But the pre-processing of ECG signals is a complex process as filtering can also cause the loss of important information. For the ease of our research, an existing library which is more like a toolbox for bio-signal processing, Biosspy, has been implemented initially. Biosspy offers a lot of functionalities as well as basic filtering of ECG signals as well as the detection of R-peaks. Once the R-peaks have been identified, it is possible to calculate the R-R interval for a patient for the signals generated from each of the leads. (Figure 4.5) Even though it is hard to extract waves that are closer to the baseline of the ECG, since R-peaks are the extreme data points in the record they can be easily identified. And R-R intervals display a distinguishable pattern for both MI patients and healthy individuals [2]. Moreover, factors like the mean, standard deviation, minimum, maximum, skew and

kurtosis reading of the ECG are also some of the standard features that can assist in the detection of MI. However, the conventional markers of detecting MI cannot be extracted here due to the lack of any specific pointers in the signal.

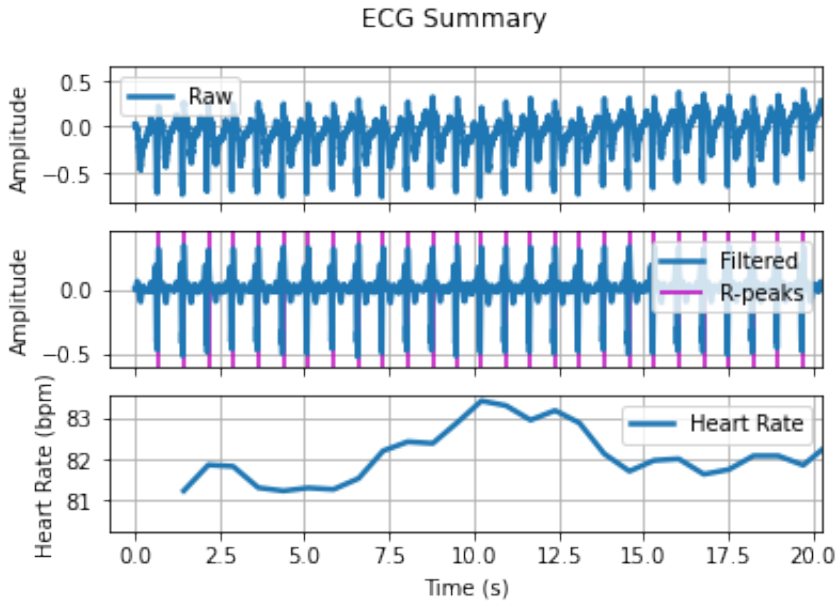


Figure 4.5: Raw and Filtered ECG with marked R-peaks

So far our data has been filtered and categorized in a way that can be fed into any ML models for predicting our disease of interest and can generate a satisfactory result.

4.5 Overview of ECG Heartbeat Categorization Dataset

For our comparison work, we have used a dataset known as "ECG Heartbeat Categorization Dataset" available widely as open source data for ECG Testing. This processed dataset is built using the PTBDB ECG Dataset from physionet and is preprocessed by the authors [7] in order to simplify the parameters to give as input in the Machine learning models. The preprocessing is done through various steps. This dataset utilized only a single lead (lead-2) whereas we selected all 15 leads. Lead 2 is later re-sampled to the sampling frequency of 125Hz. The dataset consists of 290 subjects, where 148 MI subjects are taken, 52 healthy control and the rest of the subjects are diagnosed with various diseases. The Preprocessing is done by selecting the 10s window of ECG and the amplitudes are normalized from those 10s window duration. Then the R-R time interval is calculated and necessary padding was done to equalize it to a predefined fixed signal length.

Chapter 5

Model Descriptions for Numerical Datasets

5.1 Support Vector Machine

5.1.1 Definition

Support Vector Machine or SVM is a supervised learning model which detects patterns by mapping training data points in a way such that the data points can be clearly divided into 2 different categories by a hyperplane running between them. The hyperplane acts as a means of separating the two classes leaving maximum margin. A problem of SVM is overfitting, which can be mitigated using Kernel functions to map the data in a one-to-one manner along with Regularizing parameters to avoid misclassification of the training set.

5.1.2 Working Procedure

The SVM model classifies the data into two classes based on their relative position from the hyperplane utilizing clustering of data points on either side of the hyperplane leading to an intuitive classification method. In the algorithm, the data points are first plotted, and a hyperplane is created between them to separate the maximum of these data points into different groups based on their labels. Then the hyperplane is gradually adjusted so that the distance between the extreme data points or the support vectors and the hyperplane is maximum, which is called the soft margin. The maximization of the soft margin ensures that the data will be properly classified with the least amount of error, allowing a higher success rate for the SVM model. In order to control the outliers during classification, a hyperparameter gamma is used to control the influence of each data point. [13]

5.1.3 Model Implementation

During the implementation of SVM model, a linear kernel was used to separate the 2 classes despite having a significant number of features. This also allowed for fast computation and training of the model when used along with a gamma coefficient

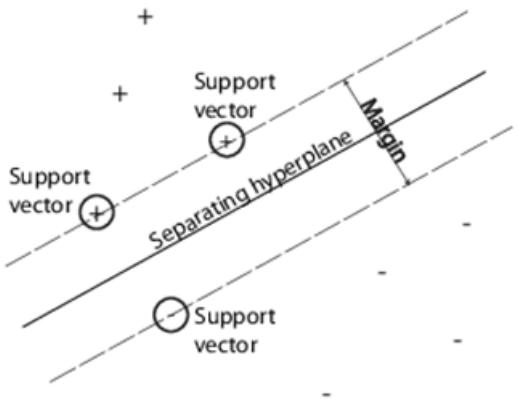


Figure 5.1: SVM Visualization

of 0.9 to control the outliers of the model, becoming less prone to overfitting. The dataset distribution percentage among train and test dataset was respectively 80% and 20%.

5.2 K-Nearest Neighbors

5.2.1 Definition

K-Nearest Neighbours (KNN) is a straightforward supervised machine learning technique with useful applications in both classification and regression. Simple but powerful, KNN classification has been successfully used to a broad variety of domains, including text classification, pattern recognition, picture and spatial classification. K-Nearest Neighbours (KNN) uses a distance measure to determine which samples in the data are most similar to a query, then takes a vote on the most frequent label (in the case of classification) or takes an average of all labels (in the case of regression) to get the final label.

5.2.2 Working Procedure

The algorithm starts by calculating the Euclidean distance between a new point data (any data value generated from the train set) and every other point around it. The new point belongs to whichever data category that has the largest population in the neighbouring boundary. Then they are sorted and the distance between the trained data cluster and the test data cluster is measured, and the test data is classified to the neighbors it belongs to. [11]

5.2.3 Model Implementation

During implementation of KNN model, we used a K value of 5. The dataset distribution percentage among train and test dataset was respectively 80% and 20%.

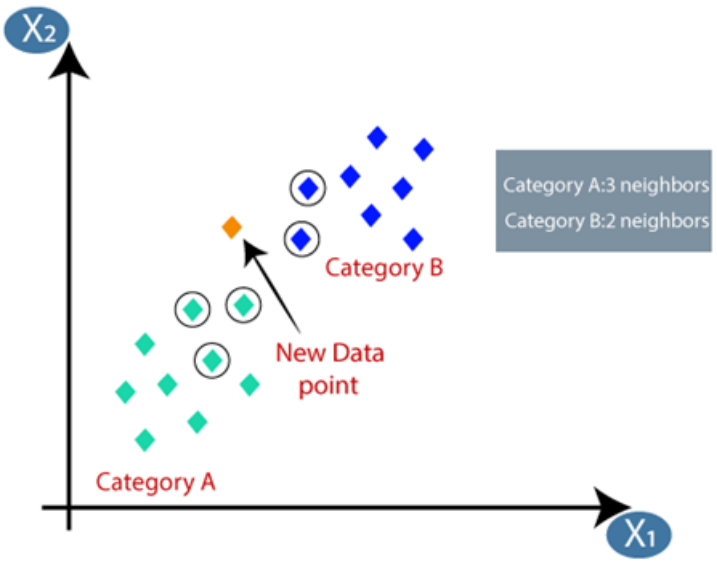


Figure 5.2: KNN Visualization

5.3 MLP

5.3.1 Definition

A multilayer perceptron (MLP) is an artificial neural network (ANN) with multiple interconnected layers. Due to its multilayer architecture, it enables effective approximation of complex nonlinear functions for generating outputs.

5.3.2 Working Procedure

MLP works by creating a network of interconnected layers of neurons. Each neuron receives inputs from the previous layer, applies weights and biases, and passes the result through an activation function. This process repeats through one or more hidden layers until reaching the output layer. During training, the network adjusts the weights to minimize prediction errors using backpropagation.

5.3.3 Model Implementation

The architecture generates an output by multiplying inputs with weight and passing the results through the activation function. To avoid overfitting and train the model efficiently, optimizers are also used. In our model, the max iteration was 1000, optimizer was Adam, the activation function was ReLu and the initial learning rate was 0.001.

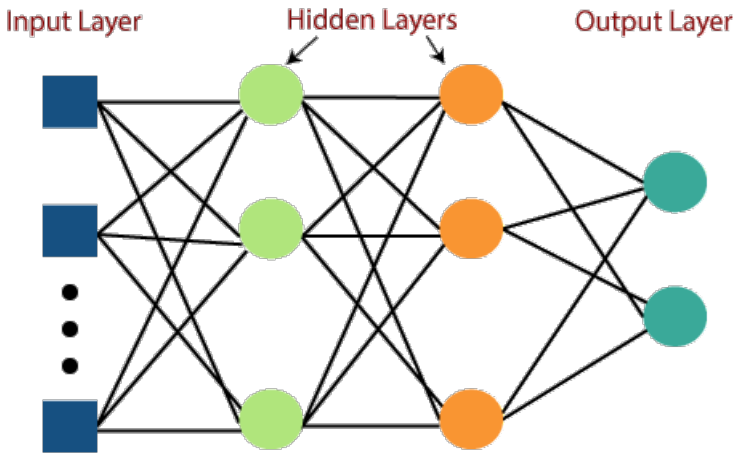


Figure 5.3: Multi Layer Perceptron Visualization

5.4 Logistic Regression

5.4.1 Definition

The primary goal of Logistic Regression is to model the probability of an event's occurrence based on one or more predictor variables using the logistic function, which maps real-valued numbers between 0 and 1.

5.4.2 Working Procedure

At its core, Logistic Regression models the probability of an event by applying the logistic (sigmoid) function to a linear combination of predictor variables. These predictor variables are features that help in determining the likelihood of the event. During training, the algorithm estimates coefficients for these input features, enabling it to predict the probability of the positive class for new data. By applying a threshold to these probabilities, we can derive binary predictions.

5.4.3 Model Implementation

Logistic Regression can handle both linearly and nonlinearly separable data to a certain degree and offers insights into feature significance, aiding in understanding the outcomes of the classification. The logistic model had 1000 iterations, LBFGS (Limited-Memory BFGS) solver as optimizer while L1 ratio was None.

5.5 Decision Tree

5.5.1 Definition

Decision trees are valuable because they can readily map decision-making processes based on input attributes and are good at managing nonlinear connections. Since it

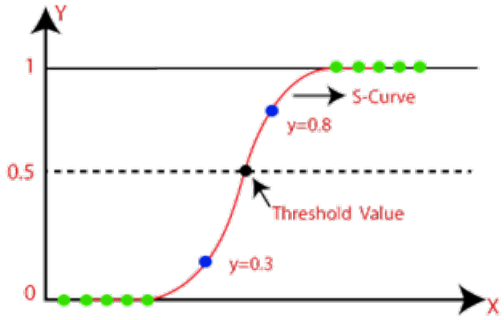


Figure 5.4: Logistic Regression Visualization

is a tree diagram, it makes complicated processes easier to understand by showing their interconnectedness.

5.5.2 Working Procedure

Decision trees are founded on maximizing the homogeneity of the target variable within each subset, which is achieved by recursively splitting input data into subsets according to the values of input features. Leaf nodes indicate projected classes or values, whereas interior nodes reflect feature splits in the final tree structure.

5.5.3 Model Implementation

The interpretability of a system is essential in many applications, and decision trees' many benefits, including their simplicity, make them a good choice. Decision trees resist outliers and missing data and can process numeric and categorical input characteristics while maintaining a simple structure. Ensemble approaches, like Bagging and Random Forests, which merge several decision trees into a more robust and accurate model, may help overfitting and their intrinsic instability.

5.6 Random Forest

5.6.1 Definition

Random Forest combines the collective insight of decision trees to make predictions. The core idea behind it is to create an ensemble of diverse decision trees, each trained on different subsets of features and data samples, ensuring diversity and robustness.

5.6.2 Working Procedure

The algorithm begins by preparing the dataset with input features and target labels. Then, through bootstrapping, multiple subsets of data are randomly sampled with replacement. At each node of a decision tree, a random subset of features is considered for splitting, introducing feature randomness. This randomization helps prevent overfitting and makes the model more robust.The construction of decision trees continues recursively until certain stopping criteria, such as maximum depth or minimum samples per leaf, are met. When making predictions for new data points,

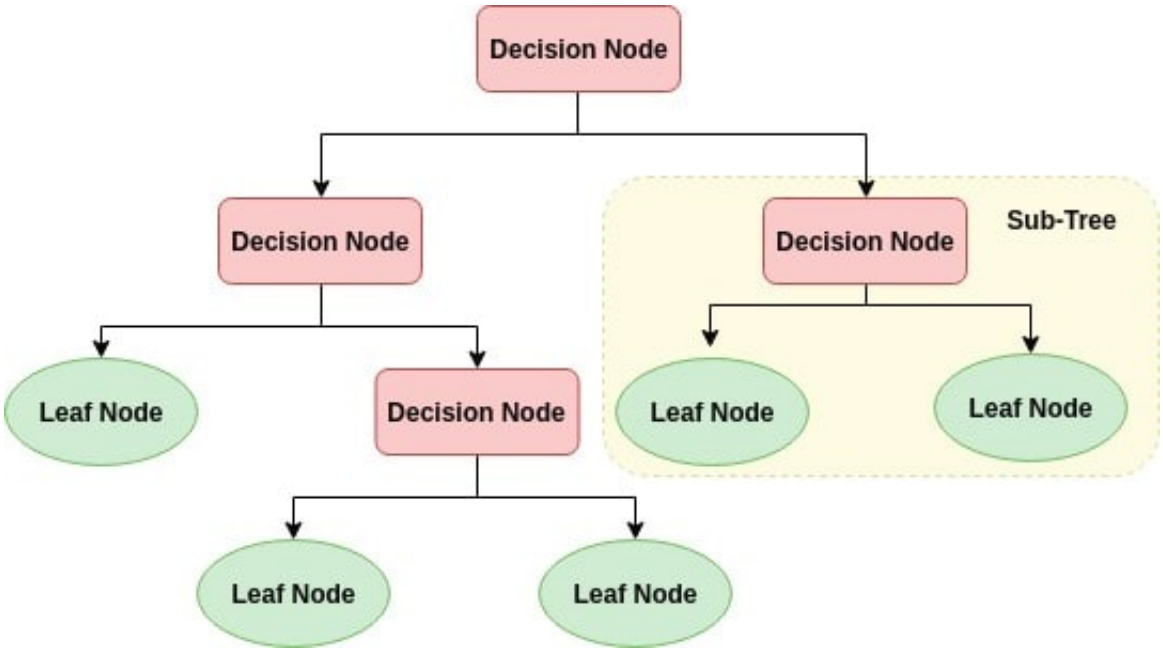


Figure 5.5: Decision Tree Visualization

the random forest combines the predictions from all the trees by majority voting for classification.

5.6.3 Model Implementation

Random Forest requires minimal hyperparameter tuning but provides high accuracy, reduced overfitting, and the ability to handle high-dimensional data . It is robust to noisy data and provides insights into feature importance. The dataset distribution percentage among train and test dataset was respectively 80% and 20%.

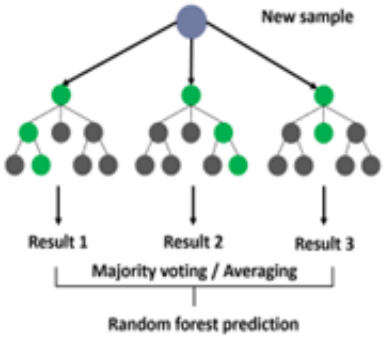


Figure 5.6: Random Forest Visualization

5.7 XGBoost

5.7.1 Definition

XGBoost, or Extreme Gradient Boosting, is a powerful machine learning algorithm that employs gradient-boosting decision trees. It constructs a series of models and combines them to create a comprehensive model with higher accuracy than any individual model in the series.

5.7.2 Working Procedure

At its core, XGBoost seeks to optimize a specific loss function while penalizing overly complex models, thus preventing overfitting. The algorithm uses gradient descent optimization to iteratively refine model parameters, ensuring high precision and adaptability across diverse datasets.

5.7.3 Model Implementation

XGBoost can handle missing values in input features and offers insights into feature importance, shedding light on the most influential predictors as well as introducing L1/L2 penalties on the weights and biases of each tree to ensuring better model generalization and performance. For our model, the booster was gbtree along with a learning rate of 0.3, n_estimator value of 100 and max depth 6.

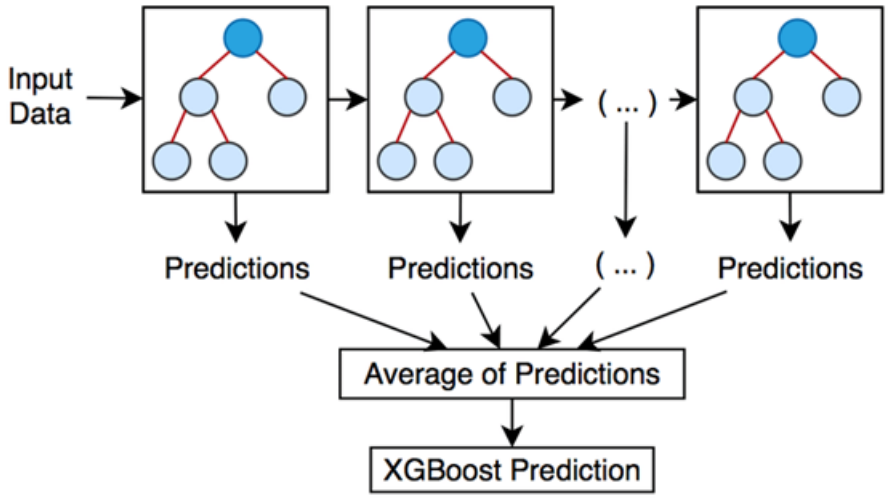


Figure 5.7: XGBoost Visualization

5.8 AdaBoost

5.8.1 Definition

AdaBoost (Adaptive Boosting) is an ensemble model where the final boosted classifier's output is a weighted sum of the outputs of many weak learners (decision

trees). It is adaptive because it adjusts for situations incorrectly labeled by earlier poor learners.

5.8.2 Working Procedure

AdaBoost's fundamental principle is to integrate the predictions of many weak learners, decision trees in our case, to create a single robust learner. The succession of weak learners corrects for the mistakes of the one before it, with the ultimate prediction resulting from a weighted majority vote. Each poor learner's performance is used to give weight to their contribution, with larger weights for those who have shown more accuracy.

5.8.3 Model Implementation

Adaboost is easy to apply to various issues since it is straightforward to develop and has few hyperparameters to tweak. In our case, n_estimator was 50 and learning rate was 1.0. Furthermore, because of its adaptive nature, AdaBoost is less likely to overfit. The dataset distribution percentage among train and test dataset was respectively 80% and 20%.

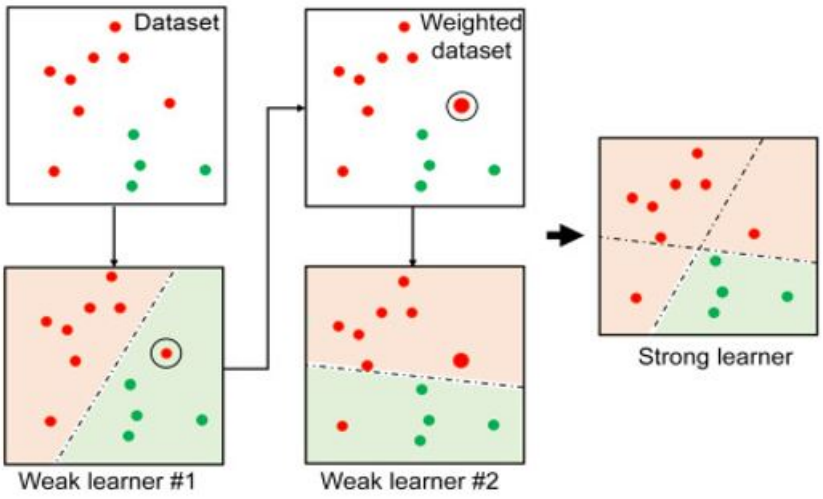


Figure 5.8: Adaboost Visualization

5.9 Bagging

5.9.1 Definition

Bagging (Bootstrap Aggregating) is an ensemble learning which works by decreasing the variance of a prediction model to combat overfitting. It works with several base learners, including decision trees, neural networks, and support vector machines, and may be used for regression and classification models.

5.9.2 Working Procedure

Bagging is based on bootstrapping, a method of sampling that includes drawing random samples with replacements to generate many varied samples from the original training dataset. A unique basis learner, in our case, decision tree is then trained using each of these samples, and the final prediction is derived by pooling the predictions of all base learners. This aggregation is often achieved by majority voting in the case of classification problems.

5.9.3 Model Implementation

Bagging is a method for improving the robustness of an ensemble of models by reducing the effect of individual model faults via using different samples and training numerous base learners. The n_estimator is 10 for our model. Since each base learner may be taught individually, Bagging is well-suited for large-scale challenges and distributed computing settings.

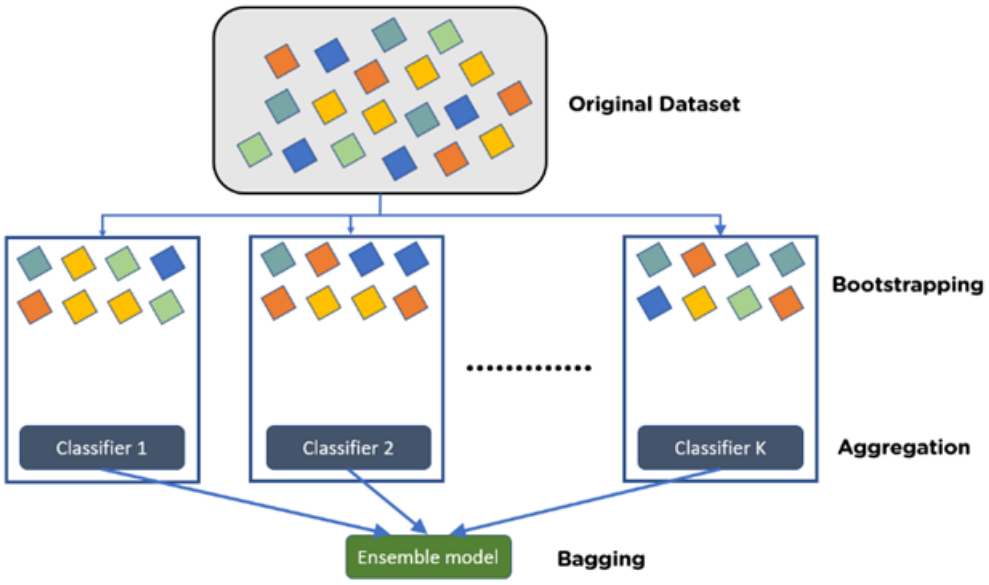


Figure 5.9: Bagging Visualization

Chapter 6

Distinctive Strategies for Signal Normalization

6.1 Numerical Dataset Implementation:

In our work, we first used the existing python library, Waveform Database (wfdb) to load the PTB Diagnostic ECG Database as our data source. Next, we isolated the data of each patient according to the presence or absence of MI and labeled the data accordingly. For comparison, we will use ECG Heartbeat Categorization Dataset which is open source for ECG Testing. After labeling, keeping the sampling rate of the ECG signal as 1000, we used our own preprocessing techniques to denoise and smoothen the signal. Then we selected our desired channels and performed computations to determine statistical features like the mean, standard deviation, skew, kurtosis, minimum and maximum R-R intervals of the signals, which were used as the features of the patient.

6.2 Visionary V2

Although the signal processing using biosppy filter enhanced the ECG signal quality and returned a format of the signal that can be fed into any ML model, the results generated from those were not up to the mark considering the sensitivity of medical data regarding accuracy. In the search for a better ECG filter, we came up with a multistep filtration method that performed better than other filtration methods we used so far. This multistep filter will be referred to as VisionaryV2 for the rest of our work.

Nyquist Frequency: In signal processing, the Nyquist frequency is a sampling frequency that turns a continuous function or data into a discrete sequence. It is the highest frequency that a piece of equipment can accurately measure at a given sampling rate. Most of the time, it is half of the sample rate given. [1]

ECG is a discrete time signal meaning it is a signal that is sampled at regular time intervals. There are a lot of mathematical tools that can be used to analyze and design discrete time systems that operate on discrete time signals like ECG.

For generating more accurate results, the initial step of normalization is to remove mechanical and hardware errors or noise which is usually caused due to powerline frequency. This noise introduces the maximum possible frequency into the ECG signal i.e. the powerline frequency which is rare in the ECG data of real patients. To discard those extreme frequencies, we passed our raw signals through the first layer of Visionary V2 which acts as a notch-filter. The filter first calculates a frequency ratio which is the specific frequency we are trying to remove (powerline frequency) divided by the Nyquist frequency. Since we are working with the PTBDB (German) dataset, the powerline frequency in our calculation was 50Hz (applicable for Europe and many countries in Asia) which can be updated to 60Hz depending on the geographical location while the Nyquist frequency was 500Hz in our work. The frequency ratio is then used to design the filter using the IIR(Infinite Impulse Response) algorithm that takes the frequency ratio and a quality factor which defines the bandwidth of the filter.

The mathematical foundation of IIR filters lies in their transfer function, H(s) which encapsulates the filter's behavior and is formulated as the ratio of two polynomials—namely, the numerator and the denominator. Mathematically, this is expressed as:

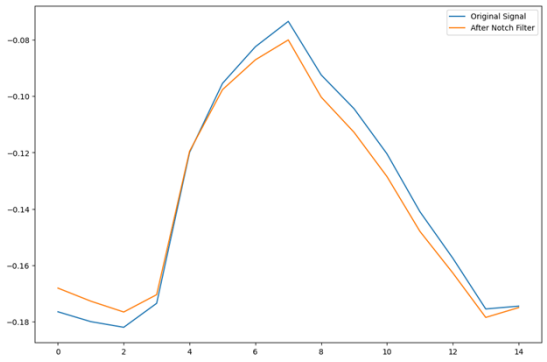
$$H(s) = \frac{b_0 + b_1 s + b_2 s^2 + \ldots + b_M s^M}{a_0 + a_1 s + a_2 s^2 + \ldots + a_N s^N}$$

Here, b_i and a_i denote the coefficients of the numerator and the denominator, respectively. The orders M and N of these polynomials define the filter's complexity and behavior. The reason why Infinite Impulse Response filter was used instead of Finite Impulse Response(FIR) filters was that theoretically IIR returns fewer coefficients than equivalent FIRs making them computationally efficient. This transfer function provides the two arrays of coefficients which helps to transform our raw signal into a desired format.

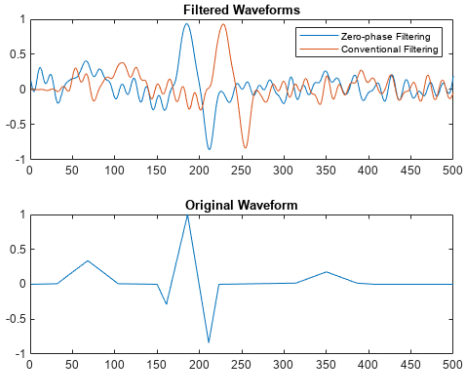
In the time domain, IIR filters are commonly implemented through a difference equation, providing a straightforward method for digital computation. The standard form of the difference equation is:

$$y[n] = b_0 x[n] + b_1 x[n-1] + \ldots + b_M x[n-M] - a_1 y[n-1] - \ldots - a_N y[n-N]$$

Here, $x[n]$ is the current input sample and $y[n]$ is the output sample where the hardware error is discarded. Later, another function is applied on the transformed signal to avoid phase-shifting. This is known as a zero-phase function which preserves the features in a filtered time waveform exactly where they occurred in the unfiltered signal and ensures that the peaks and troughs of the original and filtered signals remain aligned. Zero-phase is a critical part of biomedical signal processing especially ECG since it helps in preserving the timing information while improving the accuracy and reducing artifacts.



(a) Visionary V2 first layer



(b) Zero-Phase Visualized

After the signal is processed, it is then passed into a second layer of filtering which works as the bandpass filter. This filter is responsible for mitigating the overall noise of the signal and smoothening it. The basic function of the bandpass filter is to allow signals within a certain frequency level and discard the rest. We start by declaring lowcut and highcut parameters which specify the valid range of the filter. These parameters are defined with the help of Nyquist frequency. Through trial and error, the optimal values for lowcut and highcut parameter for our dataset were 0.001 and 0.1 respectively. Since the Laplace domain transfer function for a bandpass filter is a specific type of IIR transfer function, they have the same equation. However, the coefficients change because of the low and high cut frequencies mentioned. The returned arrays of the function are then fed into a butterworth filter. The Butterworth design approach is renowned for its maximally flat frequency response in the passband. It means within the range of frequencies that the filter allows to pass through, the gain or amplitude remains constant, exhibiting minimal ripple or variation. Mathematically, a maximally flat response aims to have as many derivatives as possible equal to zero at the point where the frequency response is evaluated, typically at the center frequency of the passband. This results in a smooth and flat curve. From Fig 6.3, it is clearly observed that sudden ups and downs in the notched signal are greatly mitigated and we get a smooth signal. The zero-phase filter is also applied here afterwards so that the timing information remains consistent. One of the key aspects of the bandpass is its tendency to pull the signal up or down towards the x-axis resulting in the creation of a filtered ECG signal that follows a general baseline. The reason why this is important is because we can overcome the limitation of unpredictable amplitude values of the ECG signal caused by conditions such as HBP, LBP, Electrolyte Level and age in certain cases.

Finally, the filtered signal is then sent to the last layer working as a highpass filter. Though applying a highpass filter might seem redundant after a bandpass filter, it plays a crucial role in some cases. Even after the bandpass filter is applied, the lower end of the frequency range might still contain unwanted elements like baseline wander caused by patient movement, respiration and surrounding conditions. A highpass filter can help fine-tune the frequency response to eliminate these issues

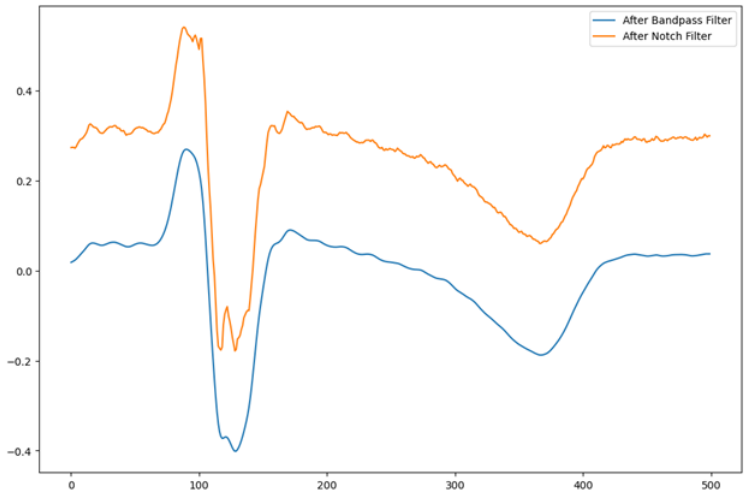


Figure 6.2: Visionary V2 second layer

more effectively. This provides adaptive filtering that adjusts the parameters that change dynamically. This layer works similarly to our earlier portion which ensures an extra layer of filtering to further denoise the original signal.

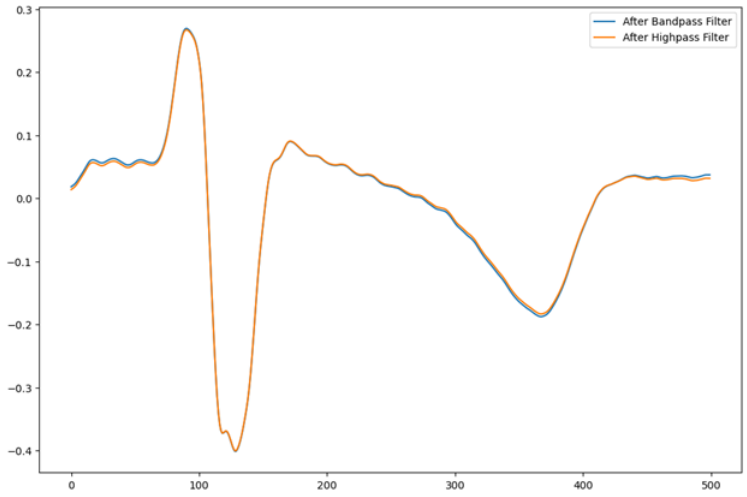


Figure 6.3: Visionary V2 third layer

Then singular normalized heartbeats are formulated taking the R-peaks, mean, standard deviation, minimum, maximum, skew and kurtosis of all heartbeats. Normalized heartbeats are then scaled proportionally so that the features required for the detection of MI can be easily identified using Machine Learning models.

Chapter 7

Model Descriptions for Generated Image Dataset

7.1 Generated Image Dataset Implementation:

Since there were no datasets readily available with ECG signals that we could use for our image recognition and processing model, we opted to generating our own dataset from the PTBDB dataset used previously. First, we selected the raw numerical data provided by the PTBDB database. This numerical data is passed through a portion of the Visionary V2 filter in order to normalize the data irrespective of age or gender of the patients while also smoothing the signal. This filter ensures that we find a baseline for the ECG signals which would allow us to determine the thresholds which are characteristic to the labels, making label prediction easier. The filtered data is then segmented with an interval of four seconds (4s) and plotted into image format to result in our initial images.

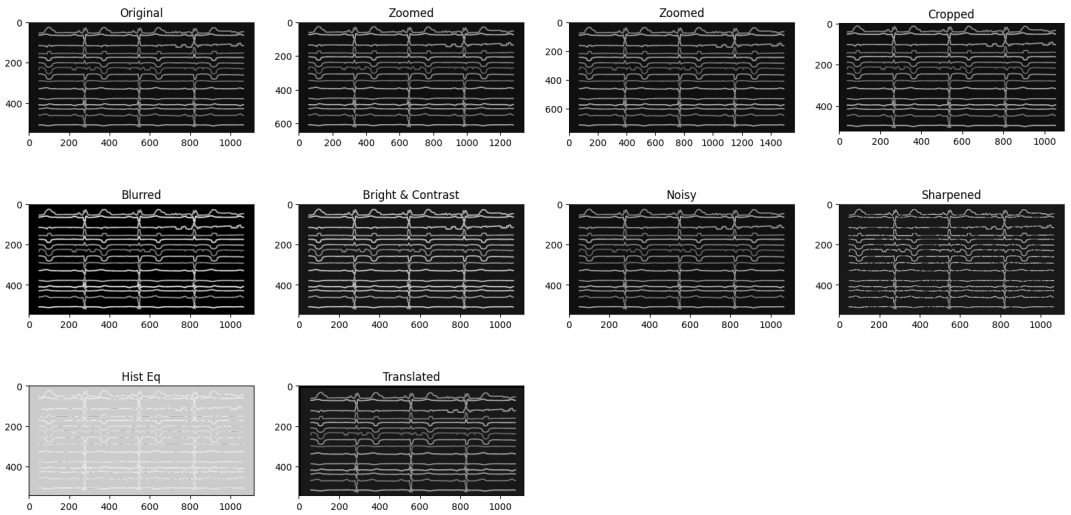


Figure 7.1: Augmentation Techniques

Following this, the images we formed are made grayscale. After this, we apply Sobel operator on each of these images for edge detection before converting the Sobel results to a 8-bit image and superimposing this resultant image on the original

image in a 90:10 ratio to enhance the original image. These enhanced images are reshaped into 256*256 size. Since, in-case of ECG images, the pixel wise detail of the signals is not of such importance. For classifying images from ECG signals, the detection of patterns is of more importance in comparison to pixel wise detail. As such, the ECG images can be downscaled more to even smaller sizes. In our case, we determined 256*256 size to provide optimal results with proper classification while also avoiding data loss. Next, we look to increase the volume of our dataset using data augmentation methods to make our model more general to different kinds of ECG signals which follow the same image generation techniques as ours.

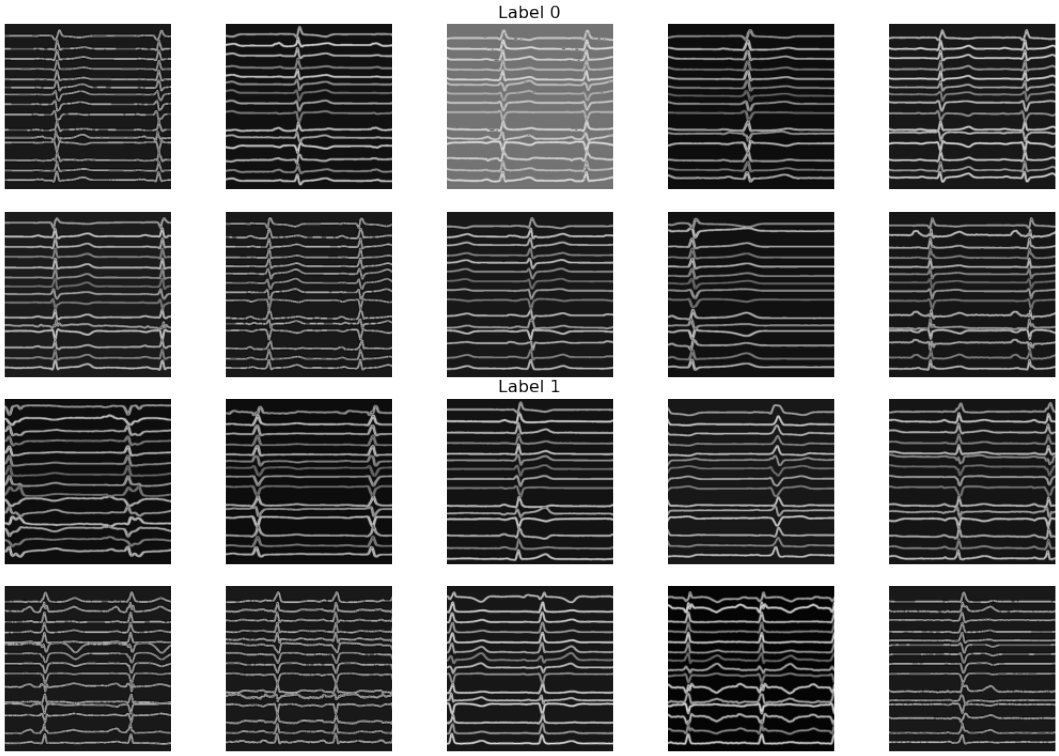


Figure 7.2: Generated Image Dataset

Since, ECG signals are sensitive medical reports which cannot be augmented randomly since it might change the proper signal orientation and pattern, we only chose the augmentation methods which leave the shape of the ECG untouched. This excluded any kind of rotation, sheering, flipping or reflecting during augmentation. In our case, we used 1.2x and 1.4x Zoom, Cropping, Gaussian Blurring, Brightness and Contrast Adjustment, Noise Addition, Sharpening, Histogram Equalization and Translation to generate our new dataset. This dataset is then split into train, test and validation sets. The validation set is created taking 15% images from each of our classes. The total number of training images for our work was 42955 belonging to both classes. Since our training set consisted of a large amount of images, we had to use ImageDataGenerator to load our training set in batches.

7.2 CNN (Convolutional Neural Networks)

A Convolutional Neural Network (CNN) is a specialized type of artificial neural network primarily geared towards the processing and analysis of visual data. It is particularly efficient in handling image and video data due to its unique architecture that includes convolutional layers, enabling automatic feature learning from the data. CNNs have the ability to recognize patterns in images by moving filters over them, and their design principles are inspired by the functioning of our visual cortex.

7.2.1 Definition

CNNs operate on the notion that each neuron in the network perceives only a limited portion of the input data, distinguishing it from traditional neural networks where every neuron processes the entire input. This principle of local receptive fields enables CNNs to learn features that remain consistent even when the input data's location varies. A CNN employs a series of layers, starting with the convolutional layer, which applies learnable filters to the input, identifying local features like textures or edges. As the data progresses through subsequent convolutional layers, more intricate features are discerned by synthesizing information from preceding layers. The concept of weight sharing minimizes the input parameters, permitting pattern recognition across the entire input spectrum. Pooling layers further bolster the network by reducing the spatial dimensions of the feature maps.

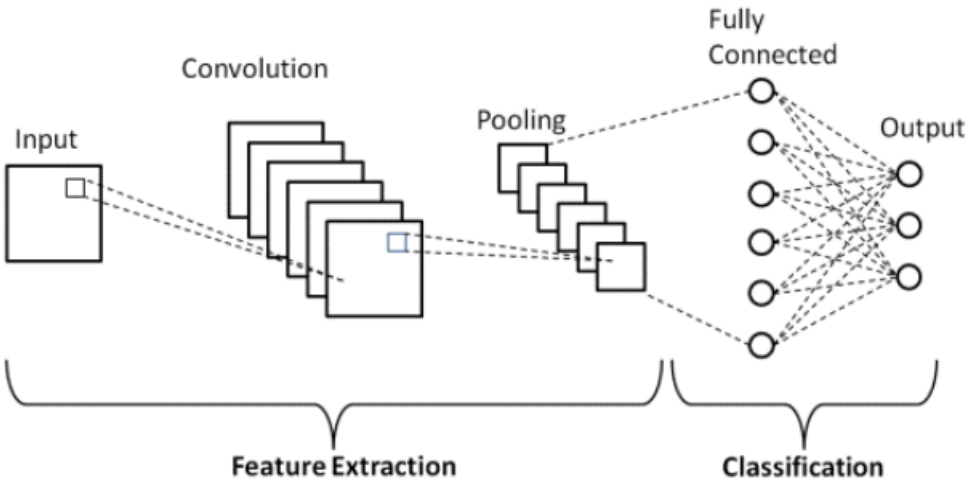


Figure 7.3: Convolutional Neural Networks

7.2.2 Advantages and Applications of CNNs

CNNs offer a plethora of advantages and have found widespread adoption across numerous domains. Their forte lies in processing visual data, making them exemplary for tasks such as image recognition, object detection, and image segmentation. Their hierarchical structure facilitates the extraction of intricate features directly from raw pixel data, obviating manual feature engineering and thus diminishing human intervention in feature extraction. Their scalability and deep architectures often result

in enhanced performance, capturing intricate patterns with precision. The ability to harness pre-trained models for transfer learning expedites the inception of novel applications. CNNs have made significant inroads in diverse areas like healthcare, diagnostics, autonomous driving, natural language processing, and semantic segmentation, among others.

In essence, CNNs have redefined the landscape of computer vision and related applications, owing to their proficiency in discerning complex patterns and structures in visual data.

7.2.3 Custom CNN Model Architecture

In our work, we created a custom sequential model following the structure and architecture of a CNN model. The model takes in a 256*256 grayscale image as input in batches of 32 images and applies 32 filters on the input in the first convolutional block which keeps the same padding and uses the ReLu activation function. Following this, batch normalization is performed to stabilize the activations and accelerate the training process. Next, max pooling is applied, reducing the image to half its original size. There are 3 total convolutional blocks in our model ultimately resulting in a 32*32 image which is flattened into a 1D array before being input into the initial dense layer of 256 neurons with ReLu activation, followed by a dropout of 0.5. Next is another ReLu activated dense layer of 128 neurons and 0.5 dropout until finally we reach the output layer with sigmoid activation and 1 neuron, which provides us with the prediction of the model. Our model uses the Adam optimizer with an initial learning rate of 0.001 and the binary crossentropy loss function since we only have two classes for prediction. We also used ReduceLROnPlateau to update the learning rate on validation accuracy loss while Model Checkpoint was also used to retrieve our best results from the CNN model.

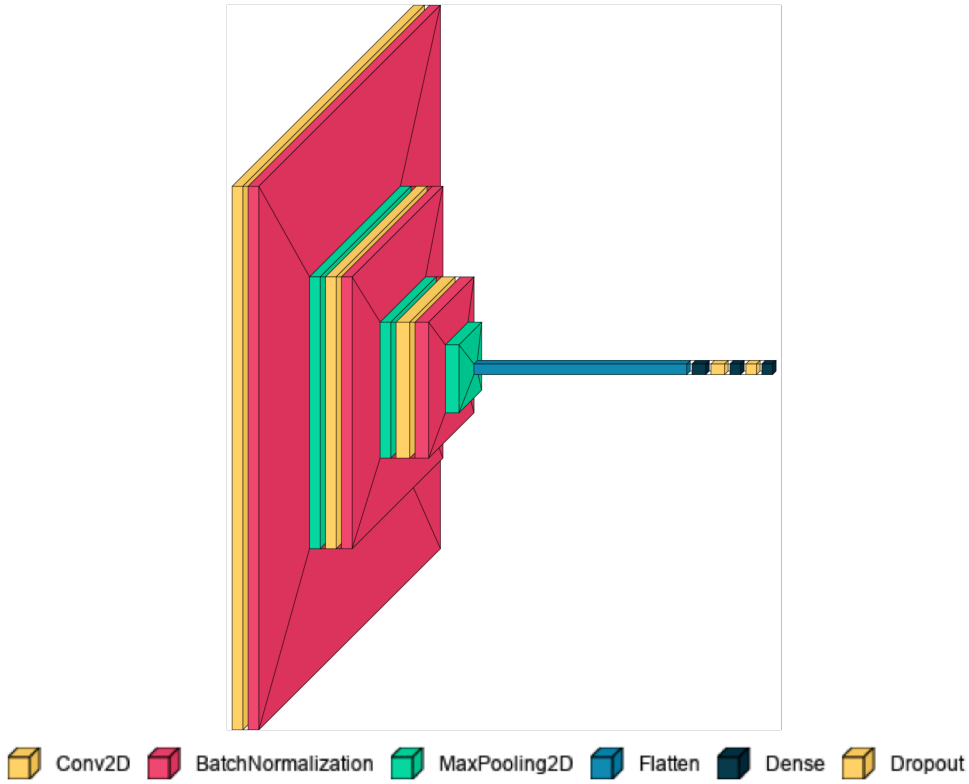


Figure 7.4: Custom CNN Architecture Visualization

7.3 ResNet-50

ResNet, an abbreviation for Residual Network, emerged as a distinct convolutional neural network (CNN) in 2015. Characterized by its 50-layer depth – encompassing 48 convolutional layers, a singular MaxPool layer, and an average pool layer – this architecture has revolutionized the landscape of deep learning. Residual networks architecturally stand out due to their stacking of residual blocks, and are a subset of artificial neural networks (ANN).

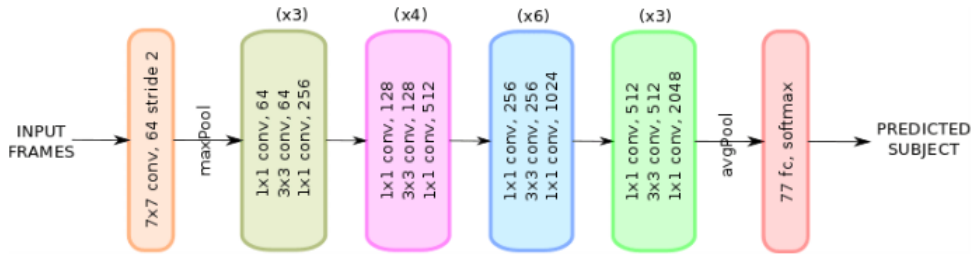


Figure 7.5: ResNet-50

7.3.1 Fundamental Concept

At the crux of ResNet-50 lies the aspiration to proficiently train profoundly deep neural networks. While these deep networks bear the potential to discern intricate

patterns and data representations, their depth introduces the notorious vanishing gradient problem. This challenge hampers the learning process in the initial layers, invariably leading to stunted performance enhancement as depth increases. ResNet-50's innovative residual learning technique is the antidote to this dilemma, enabling the network to delve deeper into the nuances of input data, thereby enhancing its accuracy metrics in comparison to conventional CNNs.

7.3.2 Merits and Applications

The core idea behind ResNet-50 is to address the challenge of training very deep neural networks effectively. Residual learning allows ResNet-50 to learn more complex features from the input data, which makes it more accurate than other CNNs. The primary benefit is that it allows for the training of extremely deep neural networks, allowing for the capture of complex and hierarchical features in the data. Skip connections help to overcome the vanishing gradient problem and make the training process more tractable. ResNet-50 consistently provides high accuracy, and has become a preferred choice for precision-oriented tasks. Pre-trained, finely-tuned, or feature extractor models significantly speed up transfer learning, eliminating the need for large amounts of data and computing power. In addition, the architecture generalizes well across a range of computer vision tasks such as image classification and object detection, as well as semantic segmentation. In summation, ResNet-50 stands as a testament to the advancements in deep learning, pushing the boundaries of what convolutional neural networks can achieve.

In summation, ResNet-50 stands as a testament to the advancements in deep learning, pushing the boundaries of what convolutional neural networks can achieve.

7.3.3 Custom ResNet-50 Architecture

For our custom ResNet-50 model, we used the ResNet building blocks i.e. the residual layers, convolution layers, batch normalization and pooling layers. The input layer first takes in an image of 256*256 size and passes the input to the initial convolution layer that applies 32 filters of 3*3 kernel size on the image before performing batch normalization on the resulting image which improves convergence speed of the ResNet model and subsequently ReLU activation function is used which helps counter the vanishing gradient problem. Next, we make use of the residual blocks to help the model learn the residual mapping between the input and the output, while also providing a shortcut connection incase layer skipping is necessary. This layer skipping allows the model to avoid being computationally expensive even though it has numerous layers. The residual blocks consist of an initial convolution layer, batch normalization, a ReLU activation function which is then followed by a shortcut connection. After the shortcut connection, there exists another convolution layer, a batch normalization with shortcut connection and a final batch normalization to sum up the residual block. After 16 residual blocks of similar design, the Global Average Pooling Layer computes the global average of the spatial features bringing the dimensions down to 32. This finally leads to our output layer consisting of a single neuron with the sigmoid activation function to provide us with the prediction from

our model. The optimizer used for the model is Adam and the loss function is the binary crossentropy. We also used model checkpoint to retain the best performance of the model based on accuracy.

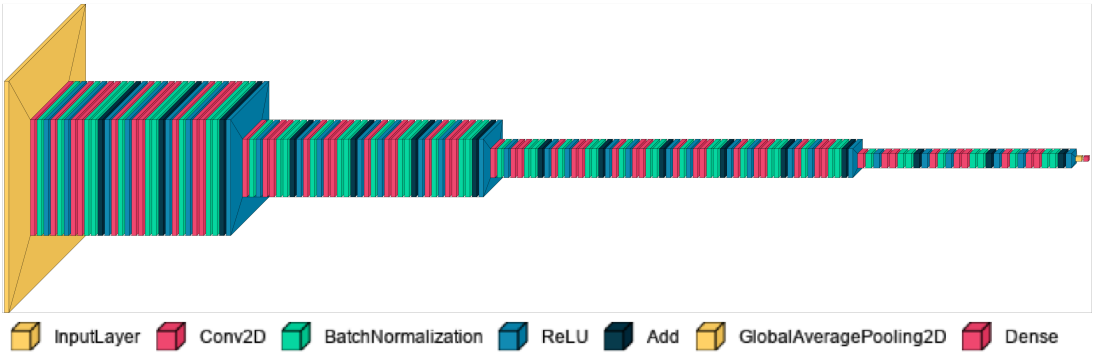


Figure 7.6: Custom ResNet-50 Architecture Visualization

7.4 LIME (Local Interpretable Model-agnostic Explanations)

LIME, which stands for "local interpretable model-agnostic explanations," is a framework for model interpretation and explainability used in Machine Learning and Artificial Intelligence (ML&AI). The primary purpose of LIME is to aid users in understanding and interpreting predictions made by complex machine learning models. Such models are frequently dubbed as "black-box" models due to the inherent difficulty in understanding their internal mechanics.

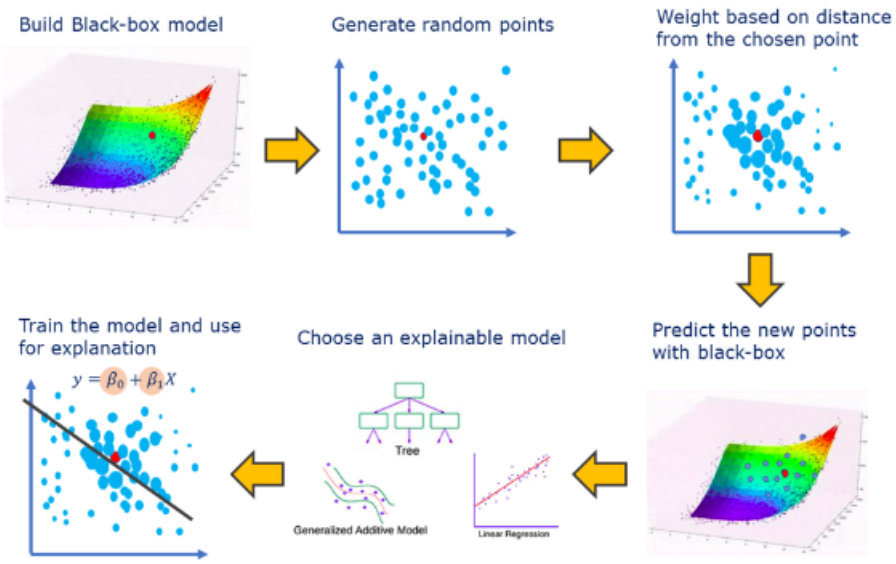


Figure 7.7: Local Interpretable Model-agnostic Explanations

7.4.1 Working Mechanism

LIME works by creating a simplified model(called an "explanation model") that explains the prediction of the original model. The process of developing an explanation model begins by introducing variations into the data and seeing how the original model's predictions change. These changes are usually small for easy interpretation. Then, training the explanation model to predict how the original model will change based on the changes in the data. Usually, this model is similar to linear regression. After getting the explanation model trained,it can be used to explain the original model's prediction. Basically,it is asked to predict how a change in the original model will affect a specific input data point, and then use that change to explain why it made the prediction it did.

7.4.2 Significance and Limitations

LIME is a great way to explain machine learning predictions. It's easy to use, and it can explain any model, no matter how complex it is or how it's trained. Plus, it's understandable by humans, so you can get a better understanding of how and

why machine learning models make the predictions they do. There are lots of open source versions of LIME that work with different machine learning frameworks, but there are some drawbacks. It's expensive to create explanations for big data sets, since LIME has to create a model for each data point. Plus, the explanations it comes up with can be wrong, especially if the data is changed.

In essence, LIME has democratized model interpretability, bridging the gap between intricate machine learning models and human comprehension, albeit with a few caveats.

7.5 Grad-CAM: Visual Explanations from Deep Networks

Grad-CAM, an acronym for Gradient-weighted Class Activation Mapping, is a technique tailored for producing "visual explanations" from models in deep learning, specifically those used in vision tasks. Born out of the necessity to demystify the decision-making processes of deep learning models, Grad-CAM offers visual cues on regions of input data (like images) that play pivotal roles in neural model predictions.

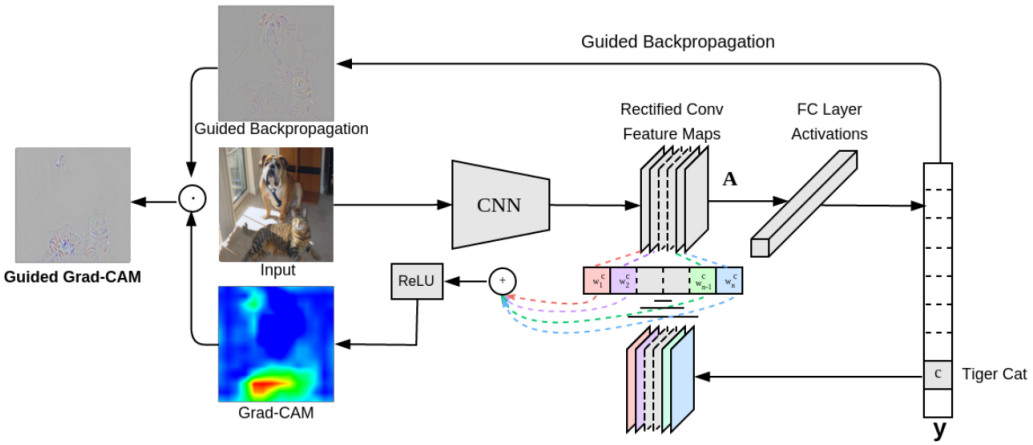


Figure 7.8: Visualization using Grad-CAM

7.5.1 Core Mechanism

The essence of Grad-CAM revolves around its ability to produce a coarse localization map, highlighting regions in the input that influenced the model's decision. The system uses the gradients of a given target idea, channeled to the last convolutional layer, to generate a rough map that emphasizes the key areas in the image necessary for identifying that idea. By examining the flow of these gradients, Grad-CAM effectively discerns the significance of particular neurons in decision-making and generates heatmaps for visualization.

7.5.2 Significance and Applications

The introduction of Grad-CAM brought forth a fresh perspective on model interpretability in the domain of deep learning. As deep learning models, especially CNNs, started gaining traction in various sectors due to their superior performance, there arose a pressing need for understanding their inner workings. This is where Grad-CAM bridges the gap, offering insights not just for researchers but also end-users who can gain confidence in model predictions by "seeing" the rationale behind decisions.

In essence, Grad-CAM has emerged as a beacon in the vast realm of deep learning, providing much-needed transparency and paving the way for more trustworthy AI systems.

7.6 Integrated Gradients: Interpreting Deep Learning Decisions

Integrated Gradients, a pioneering method developed to demystify the inner workings of deep learning systems, aims primarily to trace the origins of a deep neural network's forecast back to the traits associated with its inputs. This technique sets itself apart in an extensive assortment of interpretability tools.

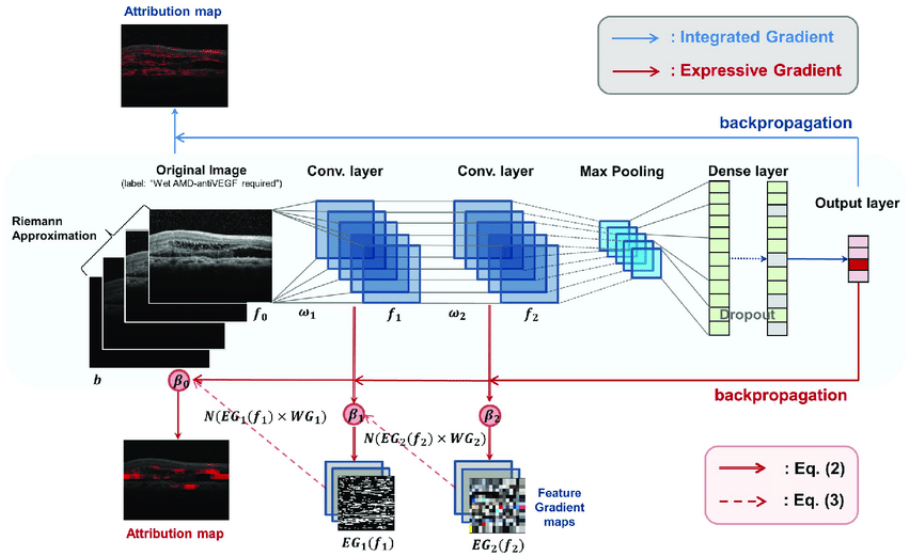


Figure 7.9: Visualization using Integrated Gradients

7.6.1 Operational Mechanism

Central to Integrated Gradients is its straightforward yet mathematically sound premise. Given both an input and its accompanying baseline (often representing the absence of a signal), the approach calculates the integral of the gradients pertaining to the model's output and its relationship with the input along a straight path starting from baseline to said input. By decomposing predictive discrepancies

between input and baseline, Integrated Gradients assigns these differences appropriately across respective feature inputs.
The real genius behind this method surfaces when considering its granularity – providing attribution scores for each feature. These scores signify how much a given trait contributes to specific predictions.

7.6.2 Significance and Utility

In this epoch of "opaque" deep-learning algorithms, the advent of Integrated Gradients marked a significant leap towards transparent AI.
Integrated Gradients has various applications in numerous domains. For example, in healthcare, it can identify precise regions in medical imagery that are vital for diagnosis. In finance, it can spotlight the most influential factors steering a financial forecast thereby assisting in more prudent decision making.
Additionally, Integrated Gradients boasts several enticing attributes. It adheres to two critical axioms for any attribution approach: Sensitivity and Implementation Invariance which ensure that it provides meaningful and consistent attributions across diverse scenarios.

In essence, Integrated Gradients serves as a flag-bearer within the realm of model interpretability fostering assurance, enhancing transparency and ensuring responsible deployment of AI solutions across industries.

Chapter 8

Feature Engineering

One of the most fundamental steps in extraction or processing the data into suitable format for implementing various models and algorithms is Feature Engineering. Feature Engineering is required in analysis and signal processing, especially in processing data like the Electrocardiogram (ECG).

In our work, there is significant use of powerful toolkits, algorithms and python libraries to implement these sorts of ECG signals. The Biosppy Library provides a wide variety of tools, algorithms and toolkits that are used for extracting relevant features from biophysical signals to fit into the required models. The PhysioNet's WFDB (Waveform Database) library grants simplified access to our working database, PTBDB. This python library module loads the ECG data directly into the working directory. The most powerful toolkit and python module of Biosppy is 'biosppy.ecg'. The most prominent use of this module is the wide range of functionalities of preprocessing the ECG Signals, collected using the Wfdb modules, detection of R-Peaks and extracting various features and important relevant information to those ECG Signals. Biosppy.ecg provides methods for preprocessing the signal, including filtering, resampling, and baseline removal.

Strong algorithms established the groundwork for our study for the multistep Visionary V2. The capacity to change and pre-process the raw ECG signal in a totally new method has been made possible by concepts like Infinite Impulse Response, Transfer Function, Zero-Phase functions, Butterworth approach, along with a few more filters. These complex features are instrumental in detecting subtle patterns in the ECG waveforms that are often overlooked by traditional methods. We can see from the result section that this preprocessing method is easier for machine learning algorithms to interpret.

The Visionary V2 processing handled the initial data generation in the Image Processing section. The generation of the data for the classification models was greatly aided by algorithms for edge detection, image channel manipulation, image dimension manipulation, and data augmentation. Custom model architectures enabled us to manipulate the images throughout the entire process and train the machine on the features which truly differentiate the malignant and benign images. because it enables us to comprehend why the model is making a decision and which aspect of the image is supporting it. This can further be used for narrowing down the specific

type of MI the patient is afflicted with as XAI provides human readable results from the model.

Summing up, feature engineering is crucial for the successful implementation of machine learning and deep learning algorithms for ECG signal analysis. Our methodology is successful in producing a feature set that is both extensive and computationally efficient by combining time-domain, laplacian domain and statistical features, as well as by ensuring their proper scaling and dimensionality reduction. These developments serve as the backbone of our Visionary V2 processing pipeline, laying the groundwork for precise and comprehensible classification model development.

Chapter 9

Analysis

9.1 Overview of result

9.1.1 Numerical Datasets

As per our research approach, we have applied eight machine learning models: Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Random Forest, Multi-Layer Perceptron(MLP), Logistics Regression, XGBoost, AdaBoost, Bagging Classifier and Decision Tree. The machine learning models were trained in the data processed through different filters and algorithms which we have divided into three categories. These are the data processes through the popular Biosppy library, preprocessed open source ECG Heartbeat Categorization dataset and our unique approach where data is preprocessed using the multistep filtration process named Visionary V2. The outcomes of these models on the mentioned processing technique and dataset varied significantly in most of the cases. A comparative table has been given to highlight the outcomes:

9.1.2 Numerical Data Results

Table 9.1: Base ML Models Performance Metrics

Model Name	*Type*	*Biosppy*	*EHC*	*VisionaryV2*
SVM	Accuracy	0.811	0.751	0.811
	Precision	0.811	0.819	0.811
	Recall	0.811	0.751	0.811
	F1 Score	0.726	0.763	0.726
KNN	Accuracy	0.811	0.942	0.855
	Precision	0.764	0.943	0.847
	Recall	0.811	0.942	0.855
	F1 Score	0.760	0.942	0.850
MLP	Accuracy	0.833	0.865	0.788
	Precision	0.820	0.862	0.654
	Recall	0.833	0.865	0.788
	F1 Score	0.788	0.861	0.715
Logistics Regression	Accuracy	0.822	0.778	0.888
	Precision	0.852	0.858	0.957
	Recall	0.855	0.860	0.955
	F1 Score	0.826	0.858	0.953
Decision Tree	Accuracy	0.711	0.918	0.933
	Precision	0.723	0.920	0.937
	Recall	0.711	0.918	0.933
	F1 Score	0.717	0.919	0.934

The SVM model achieved an accuracy of 81.1%, a precision of 81.1%, a recall of 81.1%, and an F1 score of 72.6% for Biosppy library and Visionary V2. This outcome shows that both of the processing techniques were capable of producing the same result.

The KNN model achieved an accuracy of 85.5%, a precision of 84.7%, a recall of 85.5%, and an F1 score of 85% for Visionary V2. On the contrary, Biosppy library achieved an accuracy of 81.5%, a precision of 76.4%, a recall of 81.1%, and an F1 score of 76%. In this case, KNN model performed much better on Visionary V2.

The Multi-Layer Perceptron (MLP) model achieved an accuracy of 83.3%, a precision of 82%, a recall of 83.3%, and an F1 score of 78.8% for Biosppy library. On the other hand, Visionary V2 achieved an accuracy of 78.8%, a precision of 65.4%, a recall of 78.8%, and an F1 score of 71.5%. It shows that MLP model was not as suitable as the previously mentioned models.

The Logistics Regression model achieved an accuracy of 88.8%, a precision of 95.7%, a recall of 95.5%, and an F1 score of 95.3% for Visionary V2. On the other hand, Biosppy library achieved an accuracy of 82.2%, a precision of 85.2%, a recall of 85.5%, and an F1 score of 82.6%. Even though the accuracy metric was not favorable

in Visionary V2, but the other metrics gave an indication of better performance for this model.

The Decision model achieved an accuracy of 71.1%, a precision of 72.3%, a recall of 71.1%, and an F1 score of 71.7% for Biosppy library. On the other hand, Visionary V2 achieved 93.3%, a precision of 93.7%, a recall of 93.3%, and an F1 score of 93.4%. Even though the performance outcome for Biosppy library is not so optimal, the performance outcome of Visionary V2 is optimal and one of the highest performing.

Upon evaluating the base models, Decision Tree was identified as the most effective for our binary classification tasks. Consequently, we extended our study to include ensemble models built on decision trees.

Table 9.2: Ensemble ML Models Performance Metrics

Model Name	***Type***	***Biosppy***	***EHC***	***VisionaryV2***
Random Forest	Accuracy	0.877	0.971	0.955
	Precision	0.877	0.971	0.948
	Recall	0.877	0.971	0.955
	F1 Score	0.859	0.971	0.953
XGBoost	Accuracy	0.855	0.985	0.955
	Precision	0.820	0.985	0.654
	Recall	0.833	0.985	0.788
	F1 Score	0.788	0.985	0.715
AdaBoost	Accuracy	0.888	0.860	0.944
	Precision	0.884	0.858	0.943
	Recall	0.888	0.860	0.944
	F1 Score	0.886	0.858	0.942
Bagging	Accuracy	0.877	0.959	0.966
	Precision	0.870	0.959	0.967
	Recall	0.877	0.959	0.966
	F1 Score	0.864	0.959	0.965

The Random Forest model achieved an accuracy of 95.5%, a precision of 94.8%, a recall of 95.5%, and an F1 score of 95.3% for Visionary V2. On the other hand, Biosppy library achieved an accuracy of 87.7%, a precision of 87.7%, a recall of 87.7%, and an F1 score of 85.9%. The Random Forest model is one of the best performing model for Visionary V2 in comparison of the aforementioned other processing technique.

The XGBoost model achieved an accuracy of 88.8%, a precision of 95.7%, a recall of 95.5%, and an F1 score of 95.3% for Visionary V2. On the other hand, Biosppy library achieved an accuracy of 82.2%, a precision of 85.2%, a recall of 85.5%, and an F1 score of 82.6%. Initially, the accuracy metric was not favourable in Visionary V2, but the other metrics gave an indication of better performance for this model.

The AdaBoost model achieved an accuracy of 88.8%, a precision of 88.4%, a recall of 88.8%, and an F1 score of 88.6% for Biosppy library. On the other hand, Visionary V2 achieved an accuracy of 94.4%, a precision of 94.3%, a recall of 94.4%, and an F1 score of 94.2%. The AdaBoost model is also one of the best performing model for Visionary V2.

The Bagging model achieved an accuracy of 87.7%, a precision of 87%, a recall of 87.7%, and an F1 score of 86.4% for Biosppy library. On the other hand, Visionary V2 achieved 96.6%, a precision of 96.7%, a recall of 96.6%, and an F1 score of 96.5%. The Bagging model shows the best performance in case of Visionary V2.

9.1.3 Image Dataset

For Generated Image Dataset, we have built two models: Custom CNN and Custom ResNet-50. The outcomes of these models on the mentioned dataset varied significantly in most of the cases. A comparative table has been given to highlight the outcome of these models on the mentioned dataset are:

Table 9.3: Image Performance Metrics

Model Name	Result Type	Class 0	Class 1
Custom CNN	Precision	0.987	0.986
	Recall	0.986	0.987
	F1 Score	0.986	0.987
	Sensitivity	0.986	0.987
	Specificity	0.987	0.987
Custom ResNet-50	Precision	1.000	0.999
	Recall	0.999	1.000
	F1 Score	0.999	0.999
	Sensitivity	0.999	1.000
	Specificity	1.000	0.999

Looking at the training curves of the Custom CNN model, it is visible that the Model had a steady accuracy and loss scaling throughout the traversal of Epochs during training. Firstly looking at the model accuracy, it is seen that the growth of the train curve had a steady growth, nearing almost to the maximum accuracy. But throughout the process the test data had a few drops which is acceptable as it did not overfit at any epoch. The loss curve changes accordingly with the accuracy curve. Though it had a drastic fall at the start of epochs, after a certain amount of traversal, the train curve loss of the model starts to become stable, and thus at the end of the training, it is seen that the train curve was near to lowest point of loss.

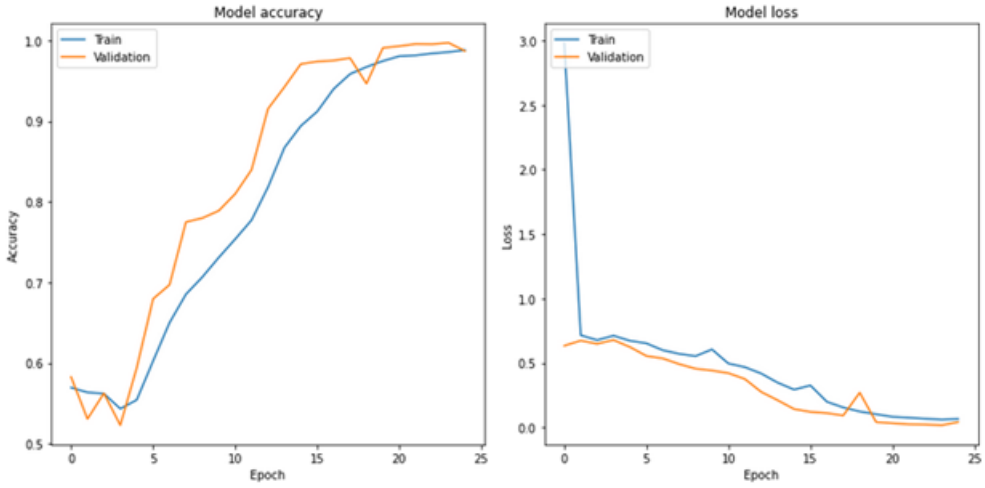


Figure 9.1: Custom CNN Training Curve

Unlike The Custom CNN Model Accuracy curve, the Custom ResNet-50 model accuracy curve had a steady accuracy and loss scaling throughout the traversal of Epochs during model training. Firstly looking at the model accuracy, it is seen that the growth of the train curve was steady, nearing almost to the maximum accuracy. The validation curve initially had a drastic fall but soon it was similar to the train curve after a certain amount of epoch. The Custom ResNet-50 Model loss curve had a drastic rise at the start of epoch. The loss train curve of the model maintained a steady flow throughout the whole epoch. The validation curve initially had a drastic loss rise but soon after a certain amount of epoch, it was similar to the train curve. It is important to note that the model actually performed well on the test data and did not overfit at any stage which makes it more reliable.

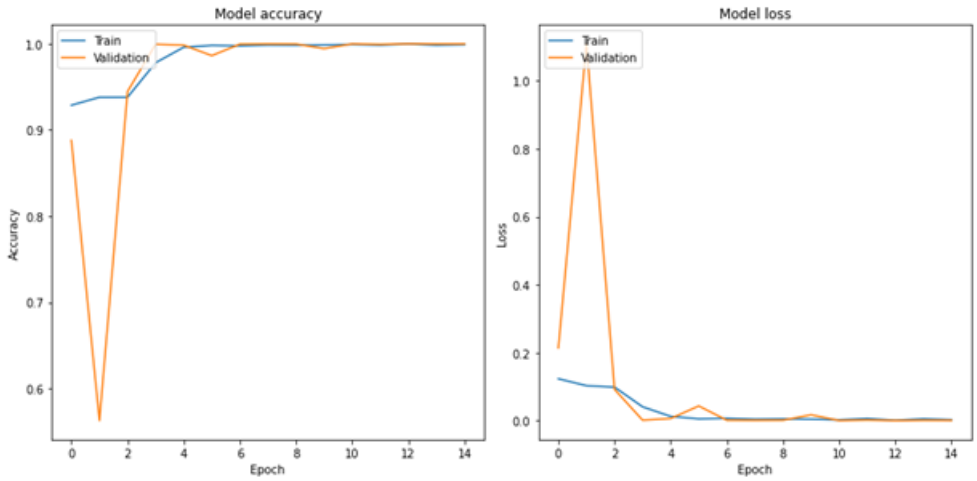


Figure 9.2: Custom ResNet-50 Training Curve

Now let us examine the confusion matrix for both of the models of Figure 8.1. During data validation, the Custom CNN model architecture has correctly predicted 3453 images as true Class 0 whereas it was falsely labeled the rest of the 47 images as Class 1. On the other hand, it could predict 3456 images as Class 1 for true Class 1 labels whereas it could not label the rest of the 44 images correctly. It is a decent result considering that the model is very small and lightweight in nature and it is possible for the model to generate a prediction quickly. However, since we are working with sensitive medical data, having such false predictions is a huge drawback and can risk human lives.

To mitigate the failings of the previous model, the deeper and complex custom ResNet-50 had a better prediction performance on the validation data according to the confusion matrix. From Figure 8.1, it is seen that for Class 0, Custom ResNet-50 could correctly label 3499 images as Class 0 and only one image was wrongly predicted. Lastly, it is seen that for Class 1, it could correctly predict all of the 3500 images as Class 1 and not a single image was mistakenly predicted. It is important

to state that the model has not overfitted as mentioned earlier and the result we are getting is actually good compared to the CNN model. Since it has been trained in very specifically modified and augmented images, this model can predict benign and malignant images with high accuracy given an unknown ECG is processed in the same manner. Due to the model's high performance, it is very suitable for medical usage.

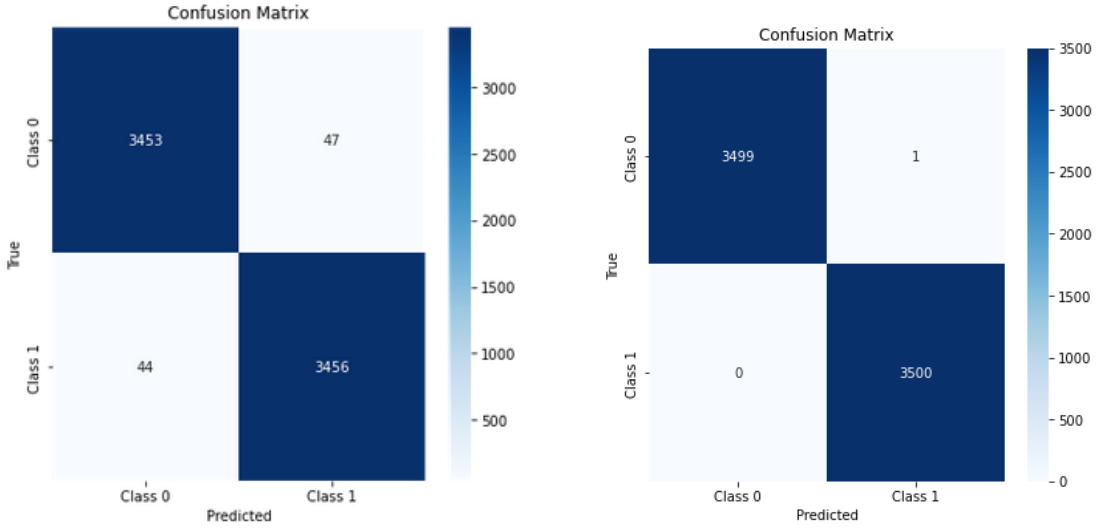


Figure 9.3: Custom CNN and Custom ResNet-50

9.2 Dataset Implementation

In the first part of our research approach, we have used different processing techniques to produce a satisfactory outcome for our numerical dataset implementation. We have applied a total of eight machine-learning models for our numerical dataset. If we evaluate the performance metrics of those models, the outcome varied significantly from model to model.

In our generated image dataset, we have approached our research outcome by using two custom model architectures: Custom CNN and Custom ResNet-50. As previously mentioned, CNN had a significant performance, over 98% F1 score for both classes during model training. On the other hand, ResNet-50 had a better performance in comparison to CNN model and by far better performance if it is compared to the machine-learning models implemented in numerical datasets.

To extend our research, after a successful positive prediction, we have further implemented Explainable AI models such as Grad-CAM, Integrated Gradient, and LIME. Grad-Cam highlights a pattern or a portion of that image that had a significant impact during the prediction. Moreover, the implementation of LIME shows or marks a specific boundary or perimeter which had a significant impact during prediction. The implementation of Grad-Cam and LIME signifies the impacting factor during the model prediction. Moreover, it helps to understand the true performance of the custom models as it marks down or highlights those impacting regions or factors and explains to us why it made the decision that it made. The implementation of these Explainable AI modules can help us take our research further in the future and can have a very significant impact on the overall process of MI identification.

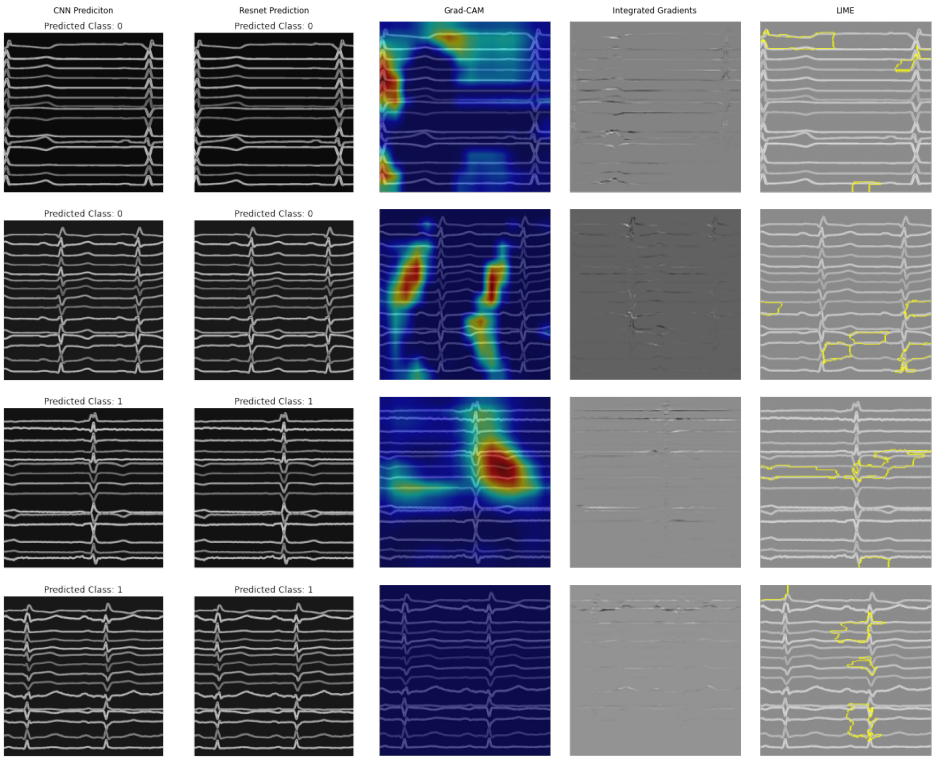


Figure 9.4: Grad-Cam, Integrated Gradient and LIME visualization

9.3 Future Work

The results generated from our study provide a promising foundation for the use of machine learning and deep learning models in the identification of cardiovascular diseases such as MI. But there are a few possible directions for future study that could help us improve and build on our work.

Firstly, we will try exploring other machine learning and deep learning models that might provide improved performance or offer other advantages. Since we have implemented most of the popular models, we can try building custom models that will explicitly specialize in learning ECG features that are better for classification.

Besides, the addition of other factors that may enhance prediction performance should also be explored. The prediction power of these models might be improved, for example, by including patient demographics, medical history, or other features of physiological signals like QRS complex, ST segment elevation, P wave and so on.

Moreover, we can try generalizing the results of the explainable AI models furthermore for human interpretation. Since, different types of myocardial infarction specifically depend on the abnormal output of certain leads, these variations are easily identifiable in the images. By generalizing the model predictions and the visual features they are considering during prediction, we can easily narrow down the type of MI a patient has. Though the scarcity of abundant data can be a very concerning limitation in this regard.

Finally, a major next step will be to incorporate these machine learning models into a functioning clinical decision support system. This requires thinking about things like user interface design, system integration, and rules and regulations. It would be crucial to assess the system's effectiveness and practicality in a clinical environment.

With these prospective endeavors, we expect to continue advancing the field of machine and deep learning in cardiology, with the ultimate objective of enhancing MI detection and patient outcomes.

Chapter 10

Conclusion

In conclusion, this study explores the potential of using machine learning and deep learning techniques for the diagnosis of myocardial infarction (MI) based on analysis of ECG signals. The research involved collecting ECG data from the PTBDB dataset, preprocessing and extracting features from the signals, and developing and evaluating multiple machine learning models on this data. Models such as SVM, KNN, Random Forest, and XGBoost were implemented and their performance was compared. The results indicate that certain models like Random Forest and XGBoost achieve high accuracy in detecting MI from the ECG signals.

The research was extended by generating an image dataset from the ECG signals and building convolutional neural network models like Custom CNN and ResNet-50 for image classification. The deep learning models demonstrated excellent performance, with over 99% accuracy in detecting MI. Techniques like Grad-CAM and LIME were also implemented to visualize the predictions of the deep learning models.

Overall, our study highlights the potential of machine learning and deep learning for the automated detection of MI based on analysis of ECG waveforms and other physiological signals. With further research to validate and improve the models on larger diverse datasets, as well as integration into clinical decision support systems, these AI techniques could enable earlier diagnosis of MI and help reduce associated mortality rates. The models' ability to provide personalized risk assessments could also assist in preventative screening and treatment. In conclusion, this thesis serves as a promising step towards developing AI systems that can augment clinicians' capabilities in accurately diagnosing cardiovascular diseases.

Bibliography

[1] E. Robinson and D. Clark, "Sampling and the nyquist frequency," *The Leading Edge*, vol. 10, no. 3, pp. 51–53, 1991. DOI: https://doi.org/10.1190/1.1436812.

[2] H. V. Huikuri, T. H. Mäkikallio, C.-K. Peng, A. L. Goldberger, U. Hintze, and M. Møller, "Fractal correlation properties of r-r interval dynamics and mortality in patients with depressed left ventricular function after an acute myocardial infarction," vol. 101, pp. 47–53, 2000. DOI: https://doi.org/10.1161/01.CIR.101.1.47.

[3] S. Ali and Al-Mejrad, "Bio-potential signal extraction from multi channel paper recorded charts," vol. 8, no. 6, pp. 520–524, 2011.

[4] S. Lee, J.-S. Kim, and K.-H. Park, "Pvc detection based on the distortion of qrs complex on ecg signal," *The Journal of Korean Institute of Communications and Information Sciences*, vol. 40, pp. 731–739, 2015.

[5] U. R. Acharya, H. Fujita, M. Adam, *et al.*, "Automated characterization and classification of coronary artery disease and myocardial infarction by decomposition of ecg signals: A comparative study," *Information Sciences*, vol. 377, Oct. 2016. DOI: 10.1016/j.ins.2016.10.013.

[6] J. Naam, C. Suharinto, and S. Sumijan, "Digitalisasi grafik elektrokardiogram dengan teknik pixel indexing," vol. 1, no. 1, pp. 172–176, 2017.

[7] M. Kachuee, S. Fazeli, and M. Sarrafzadeh, "ECG heartbeat classification: A deep transferable representation," *CoRR*, vol. abs/1805.00794, 2018. [Online]. Available: http://arxiv.org/abs/1805.00794.

[8] W. Liu, M. Zhang, Y. Zhang, *et al.*, "Real-time multilead convolutional neural network for myocardial infarction detection," *IEEE Journal of Biomedical and Health Informatics*, vol. 22, no. 5, pp. 1434–1444, 2018. DOI: 10.1109/jbhi.2017.2771768.

[9] L. Sharma and R. Sunkaria, "Inferior myocardial infarction detection using stationary wavelet transform and machine learning approach," *Signal, Image and Video Processing*, vol. 12, Feb. 2018. DOI: 10.1007/s11760-017-1146-z.

[10] M. Sharma, R. S. Tan, and U. R. Acharya, "A novel automated diagnostic system for classification of myocardial infarction ecg signals using an optimal biorthogonal filter bank," pp. 341–356, 2018. [Online]. Available: https://doi.org/10.1016/j.compbiomed.2018.07.005.

[11] J. Sun, W. Du, and N. Shi, "A survey of knn algorithm," *Information Engineering and Applied Computing*, vol. 1, May 2018. DOI: 10.18063/ieac.v1i1.770.

[12] U. B. Baloglu, M. Talo, O. Yildirim, R. S. Tan, and U. R. Acharya, "Classification of myocardial infarction with multi-lead ecg signals and deep cnn," *Pattern Recognition Letters*, vol. 122, pp. 23–30, 2019. DOI: 10.1016/j.patrec.2019.02.016.

[13] S. Ghosh, A. Dasgupta, and A. Swetapadma, "A study on support vector machine based linear and non-linear pattern classification," pp. 24–28, Feb. 2019. DOI: 10.1109/ISS1.2019.8908018.

[14] A. Mathur, "Detecting myocardial infarctions using machine learning methods," 2019. [Online]. Available: https://doi.org/10.31979/etd.xq6s-v4em.

[15] M. Than, J. Pickering, Y. Sandoval, *et al.*, "Machine learning to predict the likelihood of acute myocardial infarction," *Circulation*, vol. 140, Aug. 2019. DOI: 10.1161/CIRCULATIONAHA.119.041980.

[16] L. Ibrahim, M. Mesinovic, K.-W. Yang, and M. A. Eid, "Explainable prediction of acute myocardial infarction using machine learning and shapley values," *IEEE Access*, vol. 8, pp. 210 410–210 417, 2020. DOI: 10.1109/ACCESS.2020.3040166.

[17] T. Tabassum and M. Ahmad, "Numerical data extraction from ecg paper recording using image processing technique," pp. 355–358, 2020. DOI: 10.1109/ICECE51571.2020.9393068.

[18] J.-Z. Jian, T.-R. Ger, H.-H. Lai, *et al.*, "Detection of myocardial infarction using ecg and multi-scale feature concatenate," *Sensors*, vol. 21, p. 1906, Mar. 2021. DOI: 10.3390/s21051906.

[19] W.-C. Liu, C.-S. Lin, C.-S. Tsai, *et al.*, "A deep learning algorithm for detecting acute myocardial infarction," *EuroIntervention*, vol. 17, no. 9, pp. 765–773, 2021. DOI: https://doi.org/10.4244/eij-d-20-01155.

[20] H. M. Rai and K. Chatterjee, "Hybrid CNN-LSTM deep learning model and ensemble technique for automatic detection of myocardial infarction using big ecg data," 2021. DOI: 10.1007/s10489-021-02696-6.

[21] C. Sridhar, S. L. Oh, J. Vicnesh, *et al.*, "Accurate detection of myocardial infarction using non linear features with ecg signals," *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, Mar. 2021. DOI: 10.1007/s12652-020-02536-4.

[22] M. Zhong, F. Li, and W. Chen, "Automatic arrhythmia detection with multi-lead ecg signals based on heterogeneous graph attention networks," *Mathematical Biosciences and Engineering : MBE*, vol. 19, no. 12, pp. 12 448–12 471, 2022. DOI: https://doi.org/10.3934/mbe.2022581.