Analysis of ECG Signals Using XResNet 1d

Introduction

The electrocardiogram (abbreviated as ECG or EKG) represents an electrical tracing of the heart and is recorded non-invasively from the surface of the body.**[1]** The way we monitor and diagnose cardiovascular diseases has changed dramatically as a result of the development of healthcare technologies. Of these advancements, the examination of Electrocardiogram (ECG) data is one that is essential for identifying and assessing a number of disorders associated with the heart. An important resource for the creation and verification of algorithms targeted at accurate ECG analysis is the PTB XL dataset **[2]**, a vast collection of ECG recordings.

This report explores ECG signal recognition, analysis, and exploration using the large PTB XL dataset. This dataset provides a solid basis for training and assessing Deep Learning models to identify minute patterns and abnormalities in ECG signals. It consists of annotated recordings from individuals with a range of heart diseases. It seeks to clarify the methods, strategies, and resources used to make use of this enormous database in order to comprehend cardiac health indicators, spot anomalies, and further our knowledge of cardiovascular disorders.

Problem Statement:

To implement a deep learning model for detecting and analyzing ECG signals and classify the given ECG graph into one of the five classes from PTB XL dataset. To check the flexibility while testing different leads input or data as well providing how much accuracy is achieved by the model.

Discussion:

This report presents the assigned discussion, successful integration of PTB XL dataset **[2]** and deep learning, which has enormous potential for clinical cardiology applications. Deep Learning models for automated ECG analysis have the potential to accelerate the diagnosis process, help identify cardiac issues early, and support medical professionals in giving timely therapies.

The PTB-XL ECG dataset is a sizable collection of 21799 clinical 12-lead ECGs, each lasting 10 seconds, from 18869 patients. Up to two cardiologists annotated the raw waveform data, sometimes adding many ECG comments to each record.**[2]** The SCP-ECG standard is followed by the 71 distinct ECG statements, which include diagnostic, form, and rhythm statements.

While discussing our model, the Dataset plays a dominant role to which we present 5-classes to be grouped in accordance with the Graphical structure of the ECGs. The 5-Classes were;

- **Normal ECG:**
  The heart beats between 60 and 100 beats per minute (82 bpm) in a steady sinus rhythm **[5]**, which is accounted for by Normal ECG.

  Again, Normal sinus rhythm, or NSR, is defined as the presence of a P wave with a sinus morphology preceding each QRS complex. An upright P wave in lead II and a biphasic (up and down) P wave in lead V1 are examples of sinus morphology.

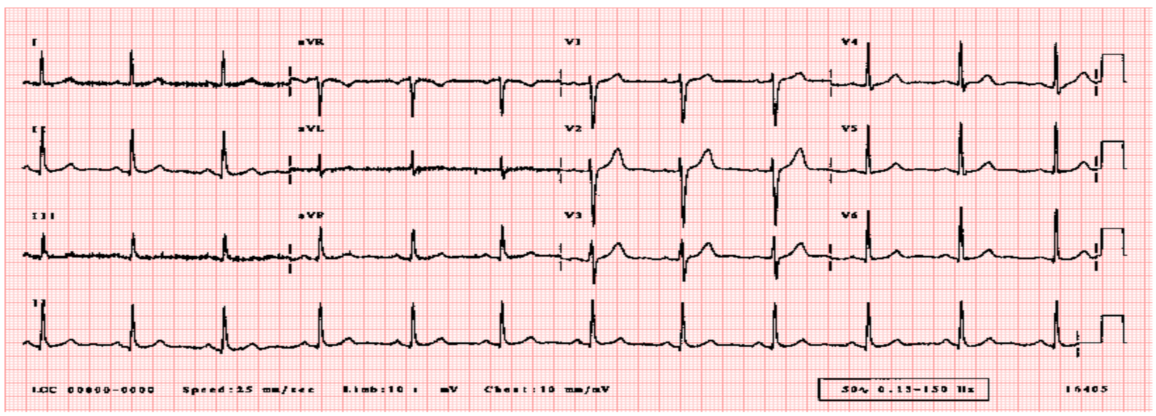

Fig-1.0: Normal adult 12-lead ECG **[6]**

- **Myocardial Infarction**:
  Reduced or stopped blood flow to a section of the myocardium causes myocardial infarction (MI), also referred to as a "heart attack." Heart attack can be "silent," going unnoticed, or it can be a devastating incident that results in hemodynamic decline and abrupt death. **[8]**

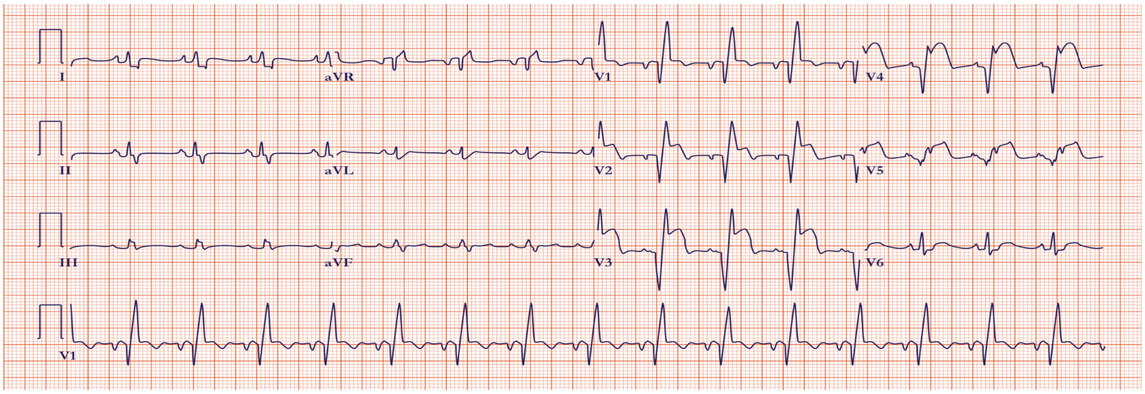

Fig-1.1: ECGs in Acute Myocardial Infarction **[7]**

- **ST/T Change**:
  Changes in the ST and T waves could be a sign of heart disease or a typical variation. As a result, the clinical setting and the existence of comparable results on earlier ECGs must be taken into consideration when interpreting the results.

  Among hypertension patients, nonspecific alterations in the ST-segment and T-wave (ST-T) are among the most common ECG abnormalities.**[9]**

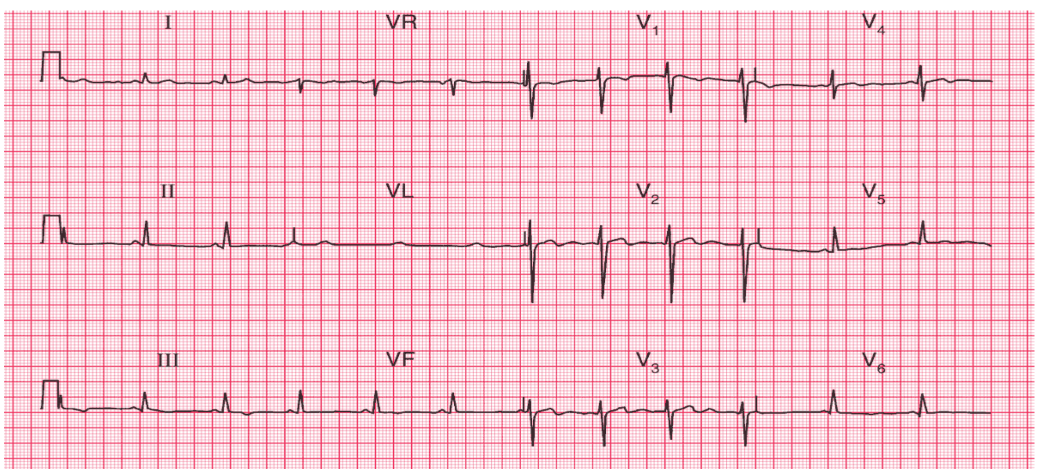

Fig-1.2: Nonspecific ST segment and T wave changes **[10]**

It is considered, changes in the ST and T waves could be a sign of heart disease or a typical variation.

- **Conduction Disturbance:**
  A conduction disturbance affects the electrical system that regulates the rhythm and pace of your heart. It is sometimes referred to as heart block. The cardiac conduction system is the name given to this mechanism.**[12]**

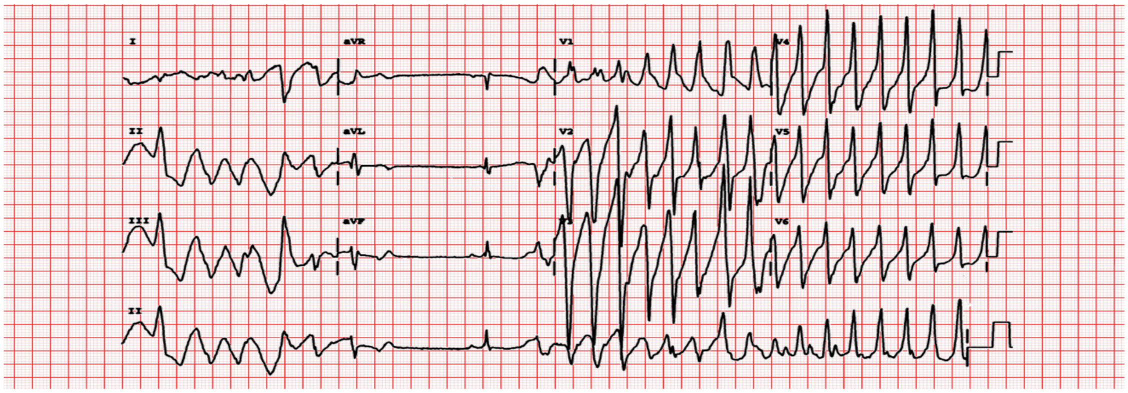

Fig-1.3: One sort of Conduction abnormalities **[11]**

- **Hypertrophy:**
  A thicker and sometimes inefficiently pumping left pumping chamber in the heart is known as left ventricular hypertrophy, or LVH. The heart muscle is occasionally overworked by conditions like aortic stenosis or high blood pressure. The inner walls of the heart may thicken as a reaction to this pressure overload. The left ventricle may become weaker, stiffer, and less elastic as a result of these thicker walls, which could inhibit normal blood flow.**[14]**

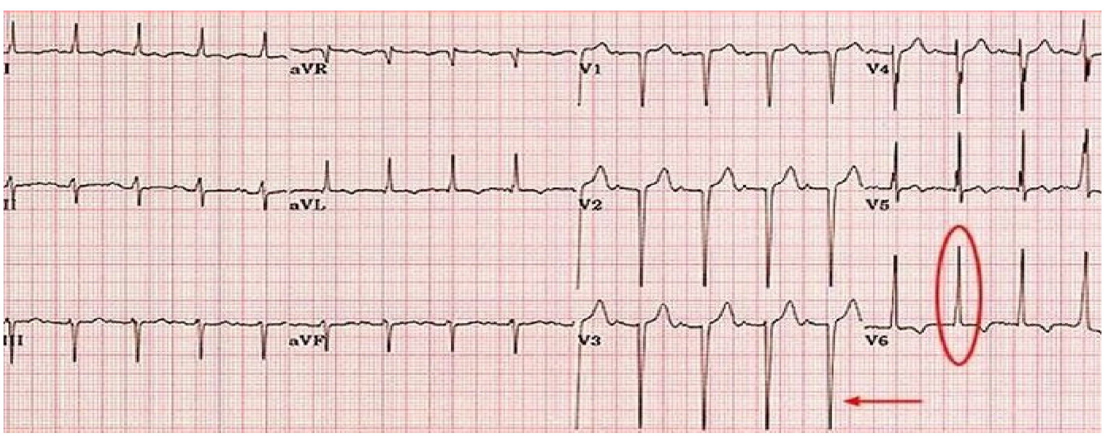

Fig-1.4(a): Left Ventricular Hypertrophy **[15]**

Right ventricular hypertrophy is the result of the thickening of the right ventricular wall as a result of prolonged pressure overload. When lead V1 exhibits an R/S ratio larger than 1 and no other contributing factors are present, or when the lead V1 R wave measures more than 7 millimeters in height, RVH is diagnosed. The strain pattern appears when there is significant pressure and a relatively thick right ventricular wall.**[13]**

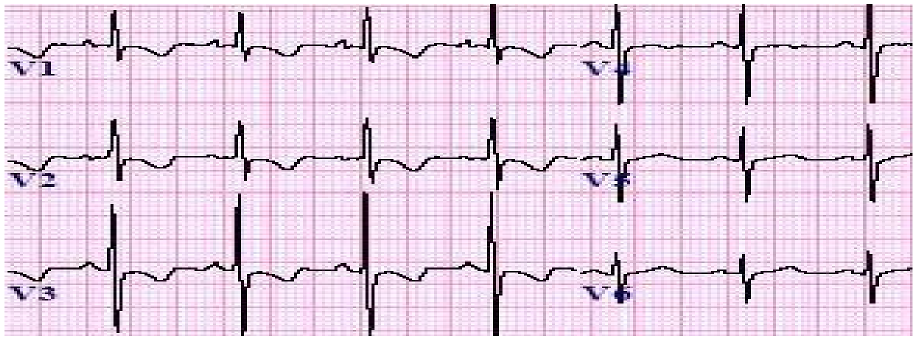

Fig-1.4(b): Right Ventricular Hypertrophy (RVH) ECG **[16]**

With each class having an immense role, the ECG graphs were analyzed with careful training and validation. The model is to establish a certain aspect of difference in the classification of these classes where it aims to differentiate the Normal ECGs with the irregular and most common potential fibrillations.

Design and Implementation

The design was implied based on the resnet1d model with several layers(explained in detail in steps for implementation). The entire code is divided into 5 parts; namely utils, stratify, basic_conv1d, xresnet1d, fastai_model_xresnet1d101. For a successful and hasslefree implementation, a certain criteria is chosen and given below:

- Data Preprocessing
- Feature Extraction
- Deep Learning Model Selection
- Training and Validation (which is included while discussing "Model Selection")
- Model Evaluation Metrics (which is included while discussing "Model Selection")
- Interpretability and Visualization (showcased in the "Result" section)

All the above criteria were optimally aimed and worked for implementation.

Platform: Visual Studio Code
Mode of use and support: Python, OpenCV concepts, Numpy, Matplot, Torch, Fastai

Now, as mentioned earlier, the code is divided into 5 parts in place of robust implementation.The steps are discussed in an intertwined way with each step having a detailed step-by-step procedure.

Data Preprocessing(stratify):
Firstly, the raw data from PTB XL dataset **[2]** needs to be preprocessed. This processing is done based on stratifying technique presented by the institution that provided the dataset. Overall strategy is as follows:

- Stratify based on patient IDs to ensure patients are not spread across folds
- Balance labels including SCP codes, gender, and age buckets across folds
- Optionally reserve some folds for higher quality data

Implementation details:

1. Compute a "quality" score for each ECG based on human validation
2. Extract all unique patient IDs
3. For each patient ID:
    a. Aggregate all their labels (SCP codes, gender, age bucket)

    b. Track number of ECGs per patient
4. Define target ratios for folds (usually uniform 1/n_folds)
5. Stratification main logic:
    a. Iterate through each patient ID
    b. Identify which fold has most space for this patient's labels
    c. Assign patient's ECGs to that fold
    d. Update counts and available space per fold and per label
6. Return:
    a. List of folds, each containing indexes of stratified patient IDs
    b. List of folds, each containing stratified patient data

The stratification loop, which iteratively selects patients and places them in the fold with the greatest amount of space left for their set of labels, contains the essential logic. This is an application of k-folds for stratified groups. Distributions can be balanced by counting the number of ECGs per patient and the tracking space per label. Some folds may provide cleaner data with quality scores.

Feature Extraction(utils):

All the feature extraction and other important multi-label classification on ECG signals is done in the "utils" part of the code. Loads the PTB-XL dataset's label data (Y) and ECG signal data (X). extracts CSV files with SCP codes for every signal as well as the raw ECG signals. Below given are the detailed explanation of implemented key functions:

1. load_dataset()
    a. Inputs:
        i. path: Base folder for PTB-XL dataset
        ii. sampling_rate: 100 Hz or 500 Hz ECG data
    b. Reads PTB XL database csv file
        i. Contains metadata like filename, age, sex, scp_codes
    c. Parses scp_codes string into dictionary using ast.literal_eval
    d. Loads raw ECG signal data for each file using WFDB(Waveform Database Software Package)
        i. Loops through filenames and reads signal + metadata
    e. Caches raw signals to speed up next run
    f. Returns:
        i. X: numpy array of raw signal data for each ECG recording
        ii. Y: dataframe with metadata for each recording

2. Combines the specific SCP codes to provide higher-level labels for rhythm, form, and other criteria. adds new label columns to the dataframe.

compute_label_aggregations()

a. Goal is to create labels at different semantic levels
b. Inputs:
    i. df: dataframe with scp_codes for each recording
    ii. ctype: type of labels to create, e.g. "diagnostic", "rhythm"
c. Loads aggregation mapping file
d. Applies functions to aggregate scp codes into particular groups
e. Creates new columns in df for each label type
f. Returns updated df

3. Filters the dataset based on a chosen label type. Also bins the multilabel labels into multi-hot encoded labels. Saves the encoder.

   select_data()

   a. Filters signals and labels by chosen label type
   b. Only keeps classes with sufficient samples
   c. Fits MultiLabelBinarizer() to hot-encode labels
   d. Splits multilabel labels into multi-hot
   e. Saves the binarizer for future inference
   f. Returns:
       i. X: selected signals
       ii. Y: selected metadata
       iii. y: multi-hot encoded labels
       iv. mlb: fitted encoding model

4. Standardizes the ECG signals to have 0 mean and unit variance. Fits scaler on train data and applies it to all splits.

   preprocess_signals()

   a. Standardizes signals to have 0 mean, unit variance
   b. Fits scaler on training data
   c. Applies same transformation to all splits
   d. Saves scaler for future inference
   e. Returns standardized signal data for each split

5. Finds the decision threshold for model probabilities that optimizes a metric, such as sensitivity-specificity, using the find_optimal_cutoff_threshold() function.

6. Aggregates saved findings from various tests into a table to provide a summary of the model's performance using “generate_ptbxl_summary_table()”.

Model Selection(basic_conv1d, xresnet1d, fastai_model):

1. **Basic Conv1d:**
   The primary model class that describes the architecture of a 1D convolutional neural network. Enables the configuration of a classification head, batch norm, squeeze-excite, dropout, number and size of convolution layers, activation functions, and pooling.

   - **Layers:**
     a. _conv1d: A 1D convolution layer with batch norm, activations, and dropout
     b. _fc: A linear fully connected layer with batch norm and activations
     c. AdaptiveConcatPool1d: concatenates the output after performing adaptive max and average pooling. utilized to create the categorization head.
     d. SqueezeExcite1d: Increases model accuracy by combining excitation and squeeze.
     e. create_head1d: After the conv layers, this function generates a classification head with completely connected layers.

   - **Functions:**
     a. weight_init: Sets the layers' initial weights according to their kind.
     b. Conv and linear layers are returned independently by get_layer_groups. beneficial for varying rates of learning.
     c. The final linear classification layer is returned by get_output_layer.
     d. The last linear classification layer is configured using set_output_layer. Helpful for fine-tuning the parameters.

2. **XResNet1D:**
   The Residual Neural Network (ResNet) architecture's XResNet1D model is a variation created especially for one-dimensional data, like time series signals like electrocardiograms (ECGs) or other sequential data. Its main objective is to efficiently learn features from sequential data for tasks such as anomaly detection, signal classification, and prediction by utilizing deep layer topologies and residual connections.

   - **ResBlock Class:** Describes a type of residual block that is utilized on a network. This block includes several layers of convolution, layers of normalization, and optional parts like Self-Attention (SA) mechanisms and Squeeze-and-Excitation (SE) Modules.

- **XResNet1D Class:** Outlines the primary architecture and puts in place a series of layers for classification and regression tasks, such as stem layers, ResBlocks, pooling layers, and a fully linked head.
- **ConvClass:** Outlines a convolutional layer that has choices for various kernel sizes, stride, activation functions, and normalization strategies.
- **AdaptiveAvgPool:** Builds a layer of adaptive average pooling for varying input sizes.
- **MaxPool and AvgPool:** Functions that produce the layers for average pooling and maximum pooling, respectively.
- **ResBlock:** Builds a residual block with identity pathways, optional SA methods, and ConvLayers.

The code provides flexibility in customizing various components, activation functions, normalization techniques, and the overall network architecture designed for processing one-dimensional sequential data such as ECG signal analysis. It also serves to define the structure and operations of the XResNet1d model.

3. **Fastai_Model:**
   Using the Fastai library, the fastai_model class can be used as a model to construct a machine learning model for time series classification or regression applications. Offering an adaptable framework for building and honing machine learning models with Fastai is the aim of this project.

- **Importing Libraries and Dependencies:**
  The code starts by importing a number of libraries, including PyTorch, Fastai, and other utility modules like timeseries_utils. Additionally, it imports particular methods and classes needed for visualizing, evaluating, and training models.
- **Class Definition - ClassificationModel:** The fastai_model particular model is based on this class .For the purpose of fitting the model and generating predictions, subclasses are anticipated to override its fit() and predict() functions.
- **Initialization and Configuration:**
  A number of attributes, such as the model name, number of classes, input size, channels, loss function, learning rate, etc., are initialized by the constructor __init__. Time series data categorization and regression tasks are the focus of this class.
- **‘fit()’ Method:**
  This function first prepares the data, then uses TimeseriesDatasetCrops to construct datasets and train the model. Based on the supplied configurations, it defines the metrics and loss functions and builds the model architecture.

- **'predict()' Method:**
  Loads the trained model, prepares the data, and provides predictions based on the supplied data. The predictions that have been combined over several input data segments are returned.
- **Model Selection and Training:**
  a. Fastai's Learner class is used to initialize the model architecture, which is chosen based on the resnet1d101.
  b. The Learner object contains configurations for metrics, loss functions, and other features.
  c. To discover an appropriate learning rate, the model is trained using the fit_one_cycle() method and the learning rate finder (lr_find_plot()).

These models contain training and validation sections that basically splits the dataset into training, validation and testing data. The folder named "data" consists of all the necessary data utilized in building this model. Then forth, the training process is done by selecting a varying learning rate to achieve better accuracy.

Results and Performance Evaluation

The model performed the classification with a training accuracy of **92.5%**, all the data predicted and the results given by the model in '.csv' format are present in the results folder.

The plotted graph of varying learning rate is shown below in Fig 2.0, which focuses on the less rate achieved in case of loss and then initializes training the model.

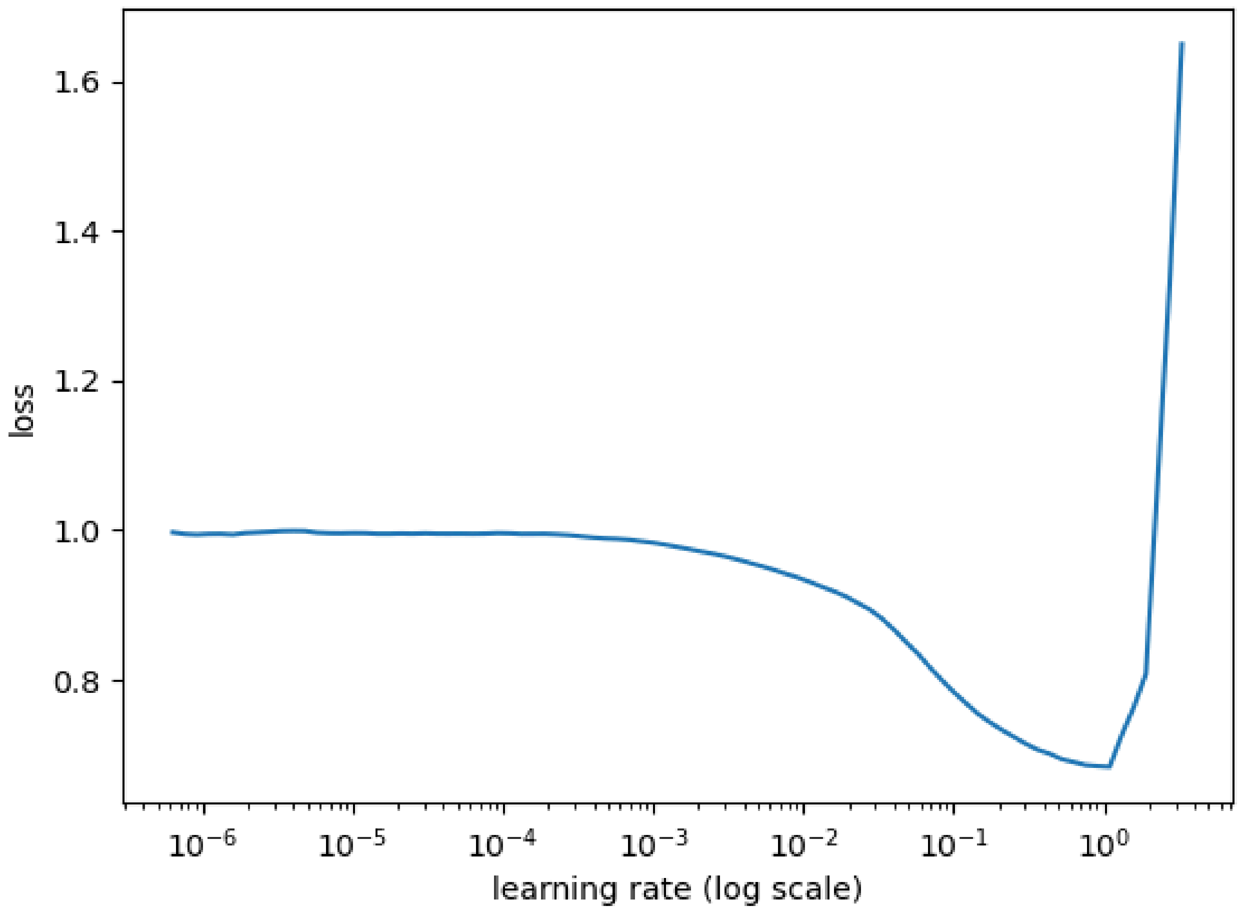

Fig-2.0: Finding learning rate

Again, the plot of losses corresponding to the number of batches is shown in Fig 2.1. This tells us about the process and loss factors putting out the efficiency of it.

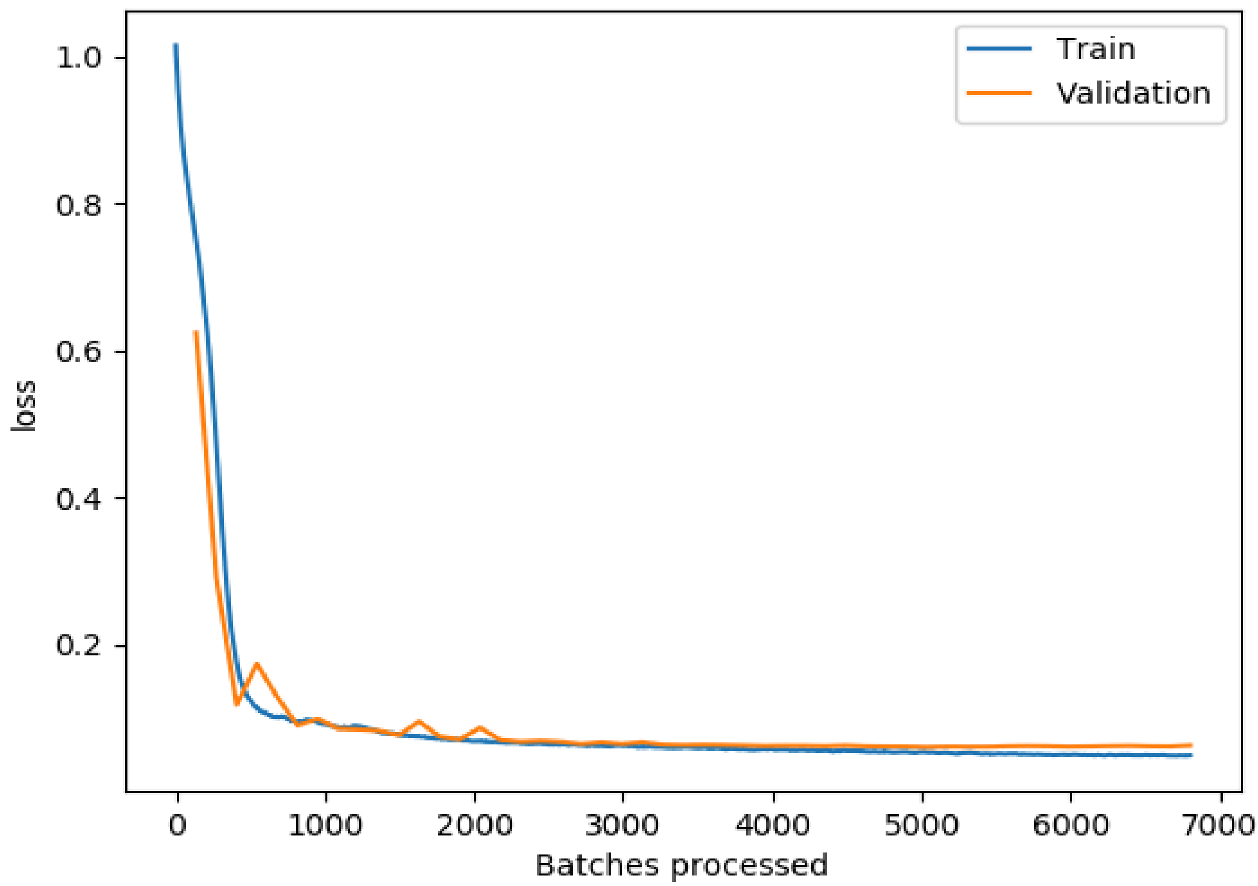

Fig 2.1: Losses corresponding to batches processed on training and validation data

The accuracy of both the training and validation data corresponding to the number of epochs is shown in Fig 2.2. It can be observed from the accuracy graph, the accuracy of validation is decreased by a small amount when compared to training.

We thought of two probable reasons for this case. Firstly, I think the model also learnt the noise and random fluctuations that are specific to the training dataset. As a result, the accuracy in the validation dataset might be comparatively less higher than what was anticipated in the model. Secondly, the regularization methods applied for the training dataset might not be as particularly active during validation.

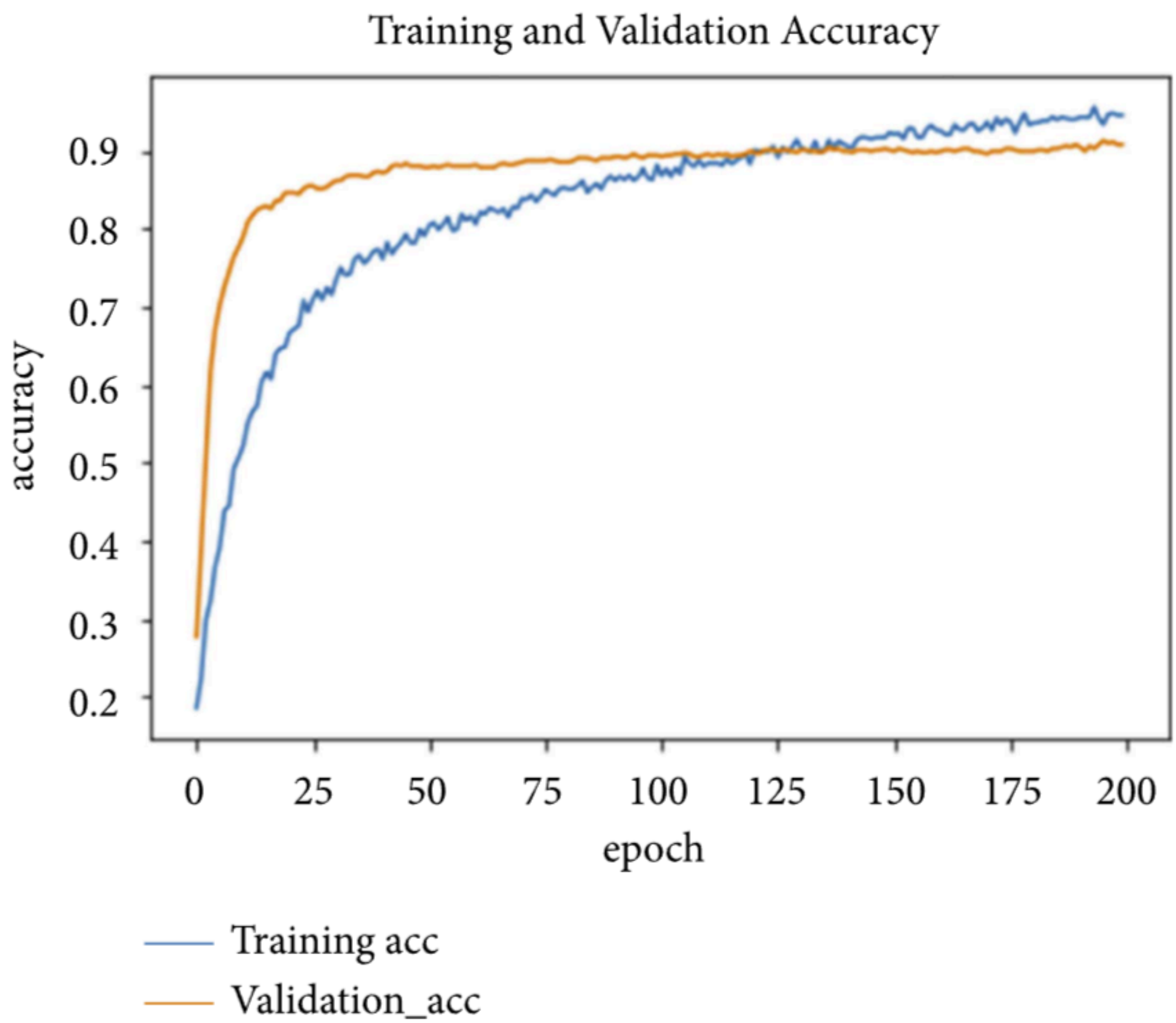

Fig 2.2: Accuracy in training and validation dataset

All the predictions of all the five classes are saved in '.npy' format. Also the macro_auc of each data(train, validation, test) are saved in '.csv' format. Macro_auc is basically the average Area Under Curve(AUC) calculated across different classes.

From the results observed, the model could successfully classify the 5 following classes: Normal ECG, Myocardial Infarction, ST/T Change, Conduction Disturbance, Hypertrophy. We are still hoping to do more, but considering what we expected the results were pretty well derived and outperformed what was thought of the model's outcome.

Conclusion

An ECG can be used to diagnose coronary heart disease, which includes angina and heart attacks, as well as irregular cardiac rhythms**[17]**. While considering the above cases, a detailed study was done to the pre-existing models. From there data classification and the ECG concept were illustrated providing well acknowledged results previewed from the existing literatures**[3][4]**. Those were our motivation to come up with the provided model and hence training a dataset with ECGs.

Moving forward to our work, we focused to illustrate as much clarifications of how the data is classified. The training is done with mentioned classes (5 classes) and based on the comparisons of ECG graphs the results were analyzed on how close it was to the classified dataset. The results helped understand how each classification has worked, based on matching the criteria specified to the data and this gives out a proper outlook to a good precision. Even if the concept led to a different Lead group, such as our very own readings (smart watch, apple watch, etc.) will compare and classify to which class our ECG falls upon.

Nevertheless, after classification we tend to research more onto an active form of work in analyzing more classes to our model. We are keeping in mind as to what has to be focused in future works to better our built model.

Reference:

1. Y. Sattar and L. Chhabra, "Electrocardiogram," 2023.
Available: Electrocardiogram [Accessed: 10-Dec-2023]

2. P. Wagner, N. Strodthoff, R.-D. Bousseljot, W. Samek, and T. Schaeffter, "PTB-XL, a large publicly available electrocardiography dataset." PhysioNet, 09-Nov-2022.
Available: PTB-XL, a large publicly available electrocardiography dataset v1.0.3 [Accessed: 6-Dec-2023]

3. S. Aziz, S. Ahmed, and M.-S. Alouini, "ECG-based machine-learning algorithms for heartbeat classification," Sci. Rep., vol. 11, no. 1, pp. 1–14, 2021.
Available: ECG-based machine-learning algorithms for heartbeat classification | Scientific Reports [Accessed: 7-Dec-2023]

4. N. D. Gai, "ECG beat classification using machine learning and pre-trained convolutional neural networks," arXiv [eess.SP], 2022.
Available: [2207.06408] ECG beat classification using machine learning and pre-trained convolutional neural networks [Accessed: 7-Dec-2023]

5. "Normal ECG," Queensu.ca. [Online].
Available:Normal ECG. [Accessed: 10-Dec-2023].

6. D. Jenkins and S. Gerred, "Ecglibrary.com: Normal adult 12-lead ECG," Ecglibrary.com. [Online]. Available: https://ecglibrary.com/norm.php. [Accessed: 11-Dec-2023].

7. "ECGs in acute myocardial infarction," ACLS Medical Training, 28-Oct-2019. [Online]. Available:ECGs in Acute Myocardial Infarction - ACLS Medical Training.
[Accessed: 10-Dec-2023].

8. N. Ojha and A. S. Dhamoon, Myocardial Infarction. StatPearls Publishing, 2023.
Available: Myocardial Infarction - StatPearls - NCBI Bookshelf.
[Accessed: 9-Dec-2023].

9. H. Bao et al., "Nonspecific ST-T changes associated with unsatisfactory blood pressure control among adults with hypertension in China: Evidence from the CSPPT study," Medicine (Baltimore), vol. 96, no. 13, p. e6423, 2017.
Available: Nonspecific ST-T changes associated with unsatisfactory blood pressure control among adults with hypertension in China - PMC. [Accessed: 9-Dec-2023].

10. Shade, “Nonspecific ST segment and T wave changes,” Manual of Medicine, 18-Apr-2022. [Online]. Available:Nonspecific ST segment and T wave changes - Manual of Medicine. [Accessed: 11-Dec-2023].

11. “Conduction abnormalities,” Default. [Online]. Available:Conduction Abnormalities | CDEM. [Accessed: 11-Dec-2023].

12. “Conduction disorders,” NHLBI, NIH. [Online]. Available: Arrhythmias - Conduction Disorders | NHLBI, NIH. [Accessed: 12-Dec-2023].

13. L. Steven, “Right ventricular hypertrophy (RVH) ECG review,” Healio, 29-Apr-2015. [Online]. Available:Right Ventricular Hypertrophy (RVH) ECG Review | Learn the Heart. [Accessed: 12-Dec-2023].

14. “Left ventricular hypertrophy,” Cleveland Clinic. [Online]. Available:Left Ventricular Hypertrophy (LVH): Causes, Symptoms and Treatment. [Accessed: 12-Dec-2023].

15. “Left ventricular hypertrophy,” Queensu.ca. [Online]. Available:Left ventricular hypertrophy. [Accessed: 10-Dec-2023].

16. L. Steven, “Right ventricular hypertrophy (RVH) ECG review,” Healio, 29-Apr-2015. [Online]. Available:Right Ventricular Hypertrophy (RVH) ECG Review | Learn the Heart. [Accessed: 10-Dec-2023].

17. “ECG test,” Gov.au. [Online]. Available: ECG test - Better Health Channel. [Accessed: 12-Dec-2023].