Idoven Data Science Task - Diego de la Cal

It is recommended to see the report on thw web version:

https://deelak.notion.site/Idoven-Data-Science-Task-Diego-de-la-Cal-58a8c241ead74fb1b274a4ea06159263

Introduction

Cardiovascular diseases are among the leading causes of death globally and early detection is crucial for effective treatment. Electrocardiogram (ECG) recordings play a pivotal role in the diagnosis of these diseases and recent advancements in deep learning have led to a significant improvement in the accuracy of ECG signal classification. In clinical settings, cardiologists diagnose a patient based on the standard 12-channel ECG recording, but the automatic analysis of ECG recordings from a multiple-channel perspective has received limited attention. With the growing need to develop reliable methods for automatic ECG interpretation, deep convolutional neural networks (CNN) are being explored as a promising solution for classifying raw ECG recordings.

In this report, I introduce my work with the PTB-XL ECG dataset, a large dataset of clinical 12-lead ECGs from 18869 patients, each with a length of 10 seconds. The raw waveform data was annotated by up to two cardiologists, who assigned multiple ECG statements to each record. The dataset includes 71 different ECG statements, conforming to the SCP-ECG standard, and covers diagnostic, form, and rhythm statements. The dataset is further complemented by demographic metadata, infarction characteristics, likelihoods for diagnostic ECG statements, and annotated signal properties.

About the Data

Data Acquisition

The raw signal data of the PTB-XL ECG dataset was recorded and stored using devices from Schiller AG between October 1989 and June 1996. The PTB acquired the original database from Schiller AG, including full usage rights. The records were curated and converted into a structured database by the Physikalisch-Technische Bundesanstalt (PTB). The data was then streamlined for usability and accessibility for the machine learning community during the public release process in 2019.

1. The raw signal data was recorded and stored in a proprietary compressed format, providing the standard set of 12 leads with reference electrodes on the right arm.
2. General metadata, such as age, sex, weight, and height, was collected and stored in a database.
3. Each record was annotated with a report string, either by a cardiologist or by automatic interpretation using an ECG device, and was converted into a standardized set of SCP-ECG statements.
4. A significant fraction of the records was validated by a second cardiologist, and all records were validated by a technical expert.

Data Preprocessing

ECGs and patients were identified using unique identifiers (ecg_id and patient_id), and personal information in the metadata was pseudonymized. The date of birth was only recorded as age at the time of the ECG recording, with ages over 89 appearing as 300 years in compliance with HIPAA standards. Furthermore, all ECG recording dates were shifted by a random offset for each patient.

Data Description

The PTB-XL ECG dataset is organized as follows: it includes 21837 clinical 12-lead ECG records of 10 seconds length from 18885 patients, where 52% are male and 48% are female, with ages ranging from 0 to 95 years (median 62 and interquartile range of 22). The value of the dataset lies in the comprehensive collection of different co-occurring pathologies and extensive metadata.

All relevant metadata is stored in `ptbxl_database.csv` with one row per record identified by `ecg_id`. It contains 28 columns that can be categorized into:

1. **Identifiers**: Each record is identified by a unique `ecg_id`. The corresponding patient is encoded via `patient_id`. The paths to the original record (500 Hz) and a downsampled version of the record (100 Hz) are stored in `filename_hr` and `filename_lr`.
2. **General Metadata**: demographic and recording metadata such as *age*, *sex*, *height*, *weight*, *nurse*, *site*, *device* and *recording_date*
3. **ECG statements**: core components are `scp_codes` (SCP-ECG statements as a dictionary with entries of the form `statement: likelihood`, where likelihood is set to 0 if unknown) and *report* (report string). Additional fields are `heart_axis`, `infarction_stadium1`, `infarction_stadium2`, `validated_by`, `second_opinion`, `initial_autogenerated_report` and `validated_b`
4. **Signal Metadata**: signal quality such as noise (`static_noise` and `burst_noise`), baseline drifts (`baseline_drift`) and other artifacts such as `electrodes_problems`. We also provide `extra_beats` for counting extra systoles and *pacemaker* for signal patterns indicating an active pacemaker.
5. **Cross-validation Folds**: recommended 10-fold train-test splits (`strat_fold`) obtained via stratified sampling while respecting patient assignments, i.e. all records of a particular patient were assigned to the same fold. Records in fold 9 and 10 underwent at least one human evaluation and are therefore of a particularly high label quality. We therefore propose to use folds 1-8 as training set, fold 9 as validation set and fold 10 as test set.

Section	Variable	Data Type	Description
Identifiers	ecg_id	integer	unique ECG identifier
	patient_id	integer	unique patient identifier
	filename_lr	string	path to waveform data (100 Hz)
	filename_hr	string	path to waveform data (500 Hz)
General Metadata	age	integer	age at recording in years (see Fig. 3 left)
	sex	categorical	sex (male 0, female 1)
	height	integer	height in centimeters (see Fig. 3 right)
	weight	integer	weight in kilograms (see Fig. 3 middle)
	nurse	categorical	involved nurse (pseudonymized)
	site	categorical	recording site (pseudonymized)
	device	categorical	recording device
	recording_date	datetime	ECG recording date and time
ECG Statements	report	string	ECG report from diagnosing cardiologist
	scp_codes	dictionary	SCP ECG statements (see Tables 6, 7 and 8)
	heart_axis	categorical	heart's electrical axis (see Table 10)
	infarction_stadium1	categorical	infarction stadium (see Table 11)
	infarction_stadium2	categorical	second infarction stadium (see Table 11)
	validated_by	categorical	validating cardiologist (pseudonymized)
	second_opinion	boolean	flag for second (deviating) opinion
	initial_autogenerated_report	boolean	initial autogenerated report by ECG device
	validated_by_human	boolean	validated by human
Signal Metadata	baseline_drift	string	baseline drift or jump present
	static_noise	string	electric hum/static noise present
	burst_noise	string	burst noise
	electrodes_problems	string	electrodes problems
	extra_beats	string	extra beats
	pacemaker	string	pacemaker
Cross-validation Folds	strat_fold	integer	suggested stratified folds

Table 2. Columns provided in the metadata table `ptbxl_database.csv`. Each ECG is identified by a unique ID (`ecg_id`) and comes with a number of ECG statements (`scp_codes`) that can be used to train a multi-label classifier that can be evaluated based on the proposed fold assignments (`strat_fold`).

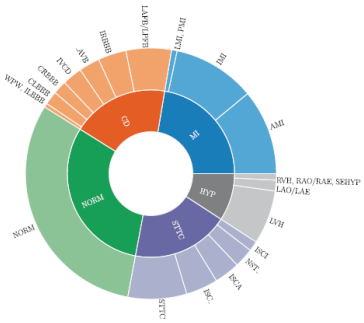


Fig. 1 Graphical summary of the *PTB-XL* dataset in terms of diagnostic superclasses and subclasses, see Table 5 for a definition of the used acronyms.

Meaning of Superclassses

- NORM: Normal ECG
- MI: Myocardial Infarction, a myocardial infarction (MI), commonly known as a heart attack, occurs when blood flow decreases or stops to a part of the heart, causing damage to the heart muscle Reference
- STTC: ST/T Change, ST and T wave changes may represent cardiac pathology or be a normal variant. Interpretation of the findings, therefore, depends on the clinical context and presence of similar findings on prior electrocardiograms.Reference
- CD: Conduction Disturbance. Your heart rhythm is the way your heart beats. Conduction is how electrical impulses travel through your heart, which causes it to beat. Some conduction disorders can cause arrhythmias, or irregular heartbeats. Reference
- HYP: Hypertrophy, Hypertrophic cardiomyopathy (HCM) is a disease in which the heart muscle becomes abnormally thick (hypertrophied). The thickened heart muscle can make it harder for the heart to pump blood. Reference

Exploratory Data Analysis

The first notebook contains the code for the Exploratory Data Analysis (EDA). The dataset is stored in a number of .csv and .wfdb files and is stored in the `data_path` directory. The first function, `load_raw_data`, loads the ECG signals from the .wfdb files based on the specified `sampling_rate` (100 or a different value). The ECG signals are returned as an array of signals.

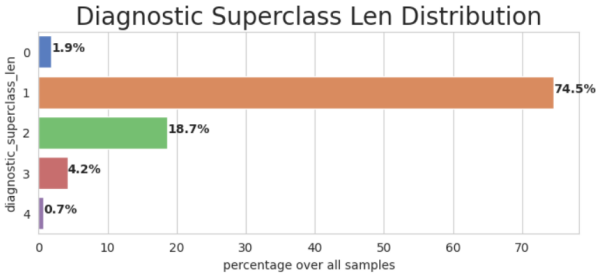
Next, the ECG annotations are loaded from the `ptbxl_database.csv` file into a Pandas dataframe `Y`. The `scp_codes` column of `Y` is a list of SCP codes, which are used to label the ECG signals. The `ast.literal_eval` function is applied to the `scp_codes` column to convert the strings to lists of SCP codes.

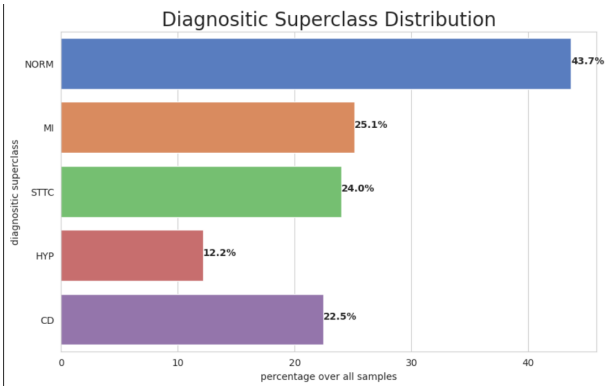
The ECG signals are then loaded into a separate dataframe `X` using the `load_raw_data` function.

A separate file `scp_statements.csv` is loaded to aggregate the SCP codes into diagnostic classes and subclasses. The diagnostic classes are stored in the `diagnostic_class` column of `agg_df`. The `aggregate_diagnostic` function is used to extract the diagnostic class for each ECG signal based on the SCP codes.

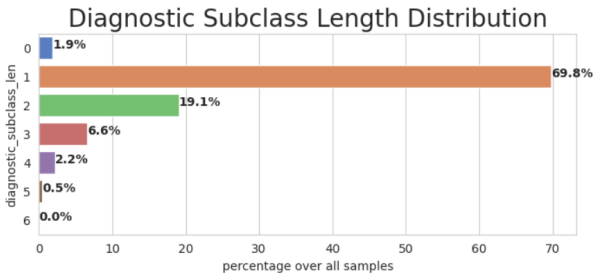
The diagnostic classes are then aggregated into a "superclass" by selecting only those classes that have a significant number of samples in the dataset. The result is stored in the `diagnostic_superclass` column of the `Y` dataframe. Then the length of this superclass is stored in the `superdiagnostic_len` column. Now inspecting these value counts, we notice that there are 405 ECGs without diagnosis. Therefore we will not take them into account for our dataset.

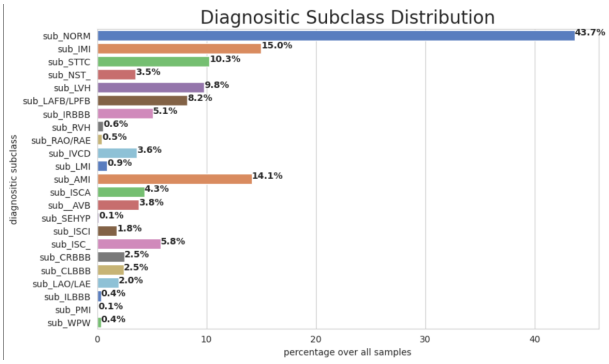
The distribution of the number of diagnostic superclasses per ECG signal is visualized using a barplot.





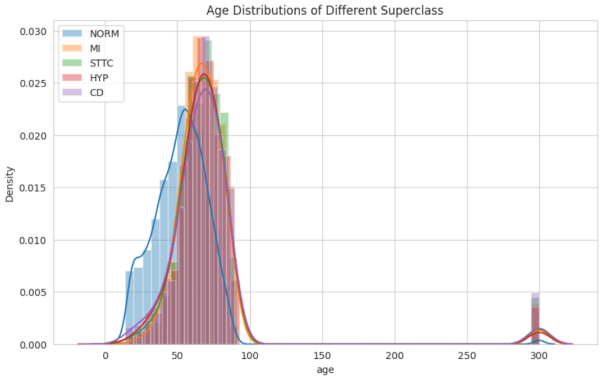
The SCP codes are aggregated into diagnostic subclasses in a similar manner as the superclasses, and the number of subclasses per ECG signal is visualized using a barplot.



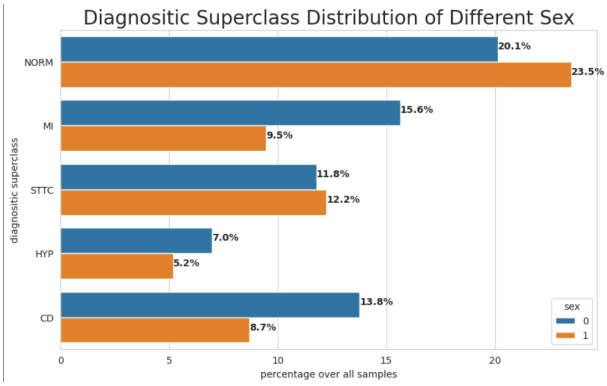


Gaining insights about how the data behaves on other variables, such as age and gender, can be useful.

- Diagnostic by Age distribution



- Diagnostic by gender



ECG signal processing

The ECG signal processing section of this project utilizes the biosppy library in Python. The following code snippet is used to extract relevant information from a given ECG signal:

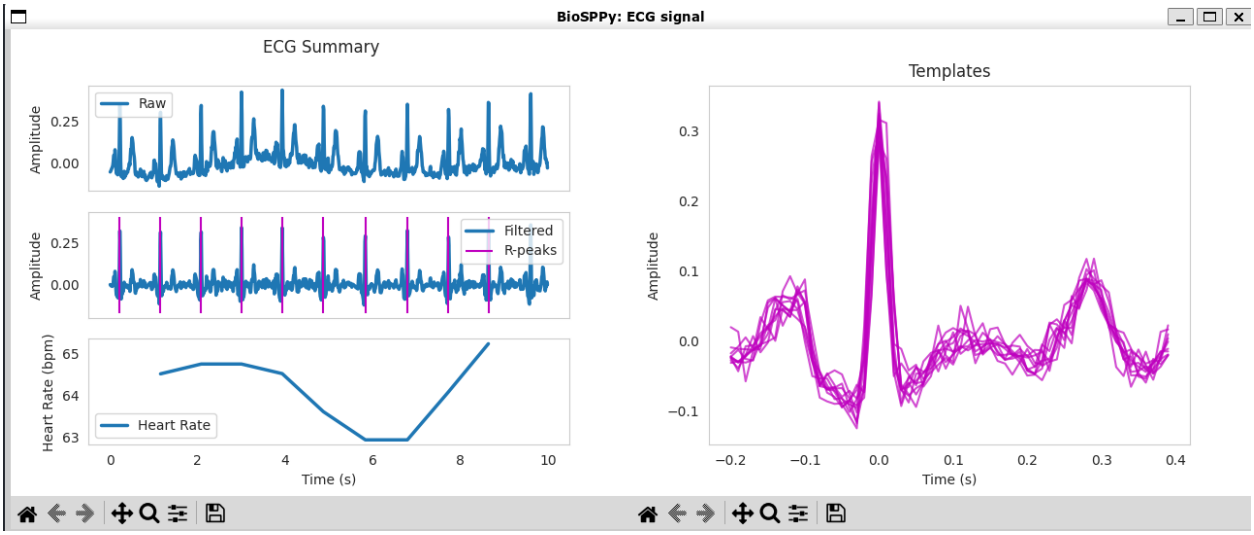
```
from biosppy.signals import ecg
[ts, fts, rpeaks, tts, thb, hrts, hr] = ecg.ecg(signal=X_data[sample_id, :, lead_id], sampling_rate=100, show=False)
```

This code returns the following information:

- `ts`: the signal time axis reference in seconds
- `filtered`: the filtered ECG signal
- `rpeaks`: the R-peak location indices
- `templates_ts`: the templates time axis reference in seconds
- `templates`: the extracted heartbeat templates
- `heart_rate_ts`: the heart rate time axis reference in seconds
- `heart_rate`: the instantaneous heart rate in beats per minute (bpm)

For this section, Lead II of the ECG signal will be analyzed. This is because, in general, lead II provides the most information about the rhythm of the heart. It is the easiest to interpret and provides a clear view of the P wave, QRS complex, and T wave. Lead II reflects the electrical activity between the right arm (RA) and left leg (LL) electrodes and provides a good representation of the overall electrical activity of the heart.

The `biosppy` library can be used to obtain a visual representation of the ECG signal, as depicted below. This library provides the original signal, the filtered signal with R-peaks, the heart rate over a period of time, and the ECG signal template.



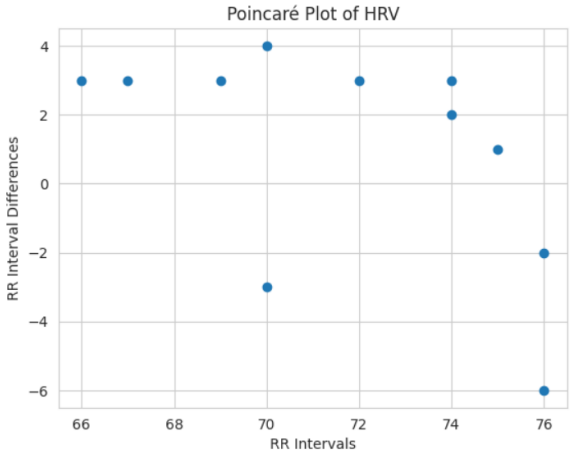
Given the data returned from the ECG processing, there are several additional analyses that can be performed:

1. Heart rate variability (HRV) analysis: HRV can provide insights into the variability of the time interval between heart beats, which is an indicator of autonomic nervous system function and cardiovascular health. RMSSD is a useful tool for analyzing HRV, as it reflects the changes in the autonomic nervous system. A higher RMSSD value indicates a higher HRV, suggesting a more stable and adaptable cardiovascular system. Additionally, a low HRV has been linked to an increased risk of cardiovascular events and death, making RMSSD a commonly used marker of cardiovascular health and disease.
2. Rhythm analysis: By analyzing the beat-to-beat variability in the R-R intervals, you can identify arrhythmias and other cardiac abnormalities.
3. Morphology analysis: By analyzing the shape of the extracted heartbeat templates, you can gain insight into the underlying electrical activity of the heart and identify certain cardiac conditions.
4. Signal quality analysis: You can also use the filtered ECG signal and other output data to assess the quality of the ECG signal, such as detecting noise or artifacts, and identify situations where the ECG signal may be unreliable.
5. Time-frequency analysis: By applying time-frequency analysis techniques like the Fast Fourier Transform (FFT) or wavelet transform, you can identify frequency-domain patterns in the ECG signal that may be indicative of certain cardiac conditions.

In the notebook, there are multiple examples of analysis conducted. For example, Heart Rate Variability (HRV) can be represented in various methods, including:

1. Time Domain Analysis: This method uses statistical measures such as mean, standard deviation, and variance to represent HRV.
2. Frequency Domain Analysis: This method involves transforming the HRV signal into the frequency domain using techniques such as Fast Fourier Transform (FFT), and then calculating power spectral density (PSD) to represent HRV.
3. Non-Linear Analysis: This method involves analyzing the complexity of the HRV signal using techniques such as Poincaré Plot and Sample Entropy.

For example, a Poincaré plot of a 10-second ECG sample from our dataset is shown below.

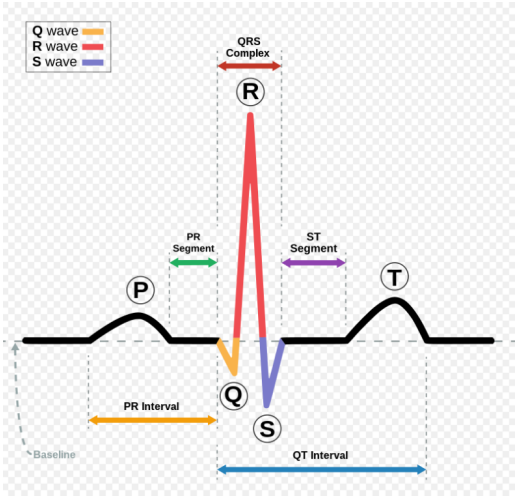


ECG QRS

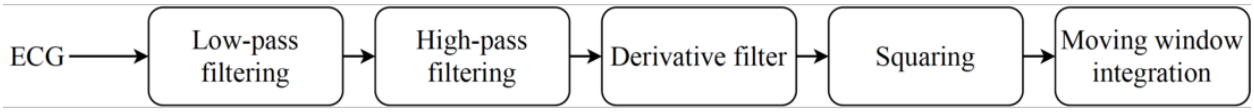
The Pan–Tompkins algorithm is a widely used algorithm for detecting QRS complexes in ECG signals. ECG signals provide important information about the electrical activity of the heart, including the heart rate, rhythm, and the presence of various types of heart disease. The detection of QRS complexes is an important step in ECG analysis, as the QRS complex is the most prominent feature in the ECG signal, and provides important information about the heart's electrical activity.

This algorithm is a computational method that uses signal processing techniques to detect the QRS complex in ECG signals. It consists of five main stages, including:

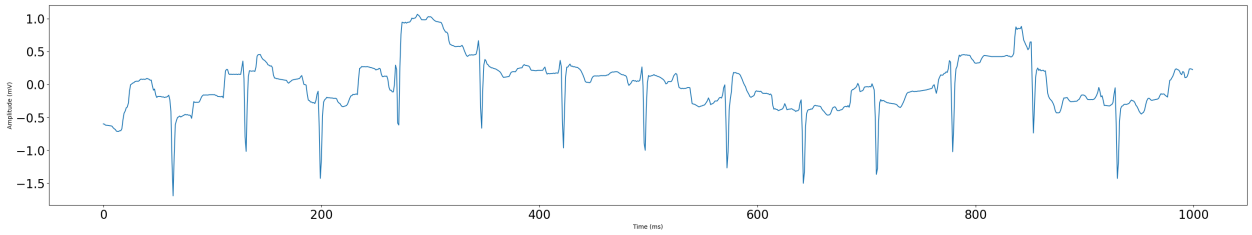
1. Preprocessing: This stage includes the filtering of the ECG signal to remove noise and enhance the QRS complex.
2. Differentiation: This stage calculates the first derivative of the ECG signal to emphasize the QRS complex and eliminate other signals.
3. Squaring: This stage takes the squared value of the differentiated signal to further enhance the QRS complex.
4. Moving Average: This stage uses a moving average filter to smooth the squared signal and reduce noise.
5. Thresholding: This stage sets a threshold value to detect the QRS complex. If the smoothed signal exceeds the threshold value, it is considered to be part of the QRS complex.



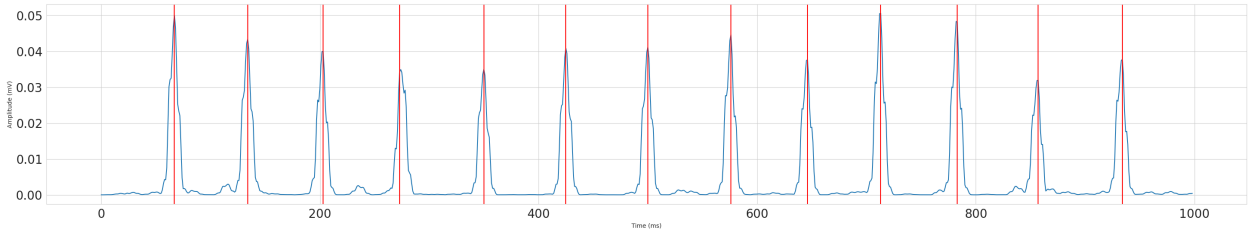
The Pan–Tompkins algorithm applies a series of filters to highlight the frequency content of this rapid heart depolarization and removes the background noise. The paper suggests following pre-processing steps:



In this case study, I applied the Pan–Tompkins algorithm to detect QRS complexes in Lead II of our ECG dataset. This improved the accuracy of the results, as seen in the comparison below. The first image shows a raw ECG signal from the dataset, and the second image shows the same signal after applying the algorithm and QRS complex detector.

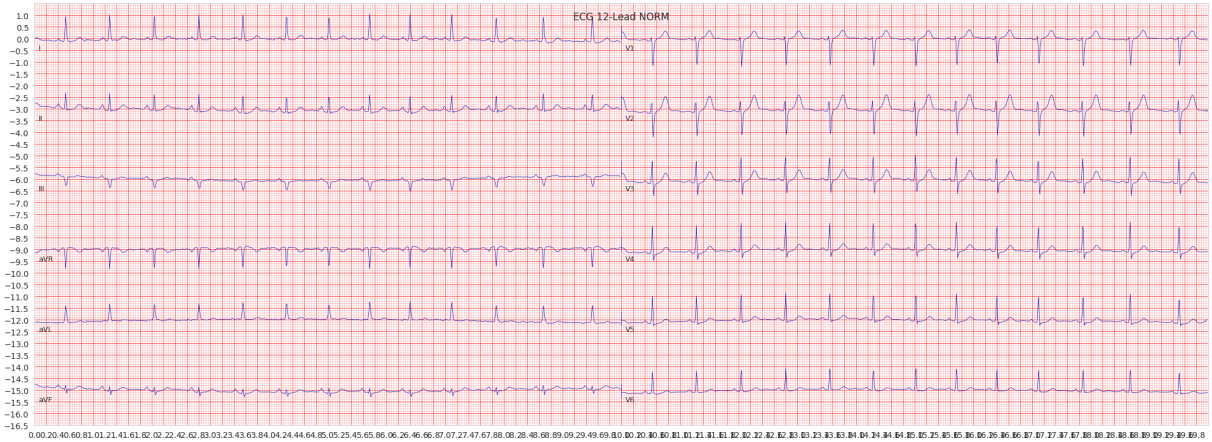


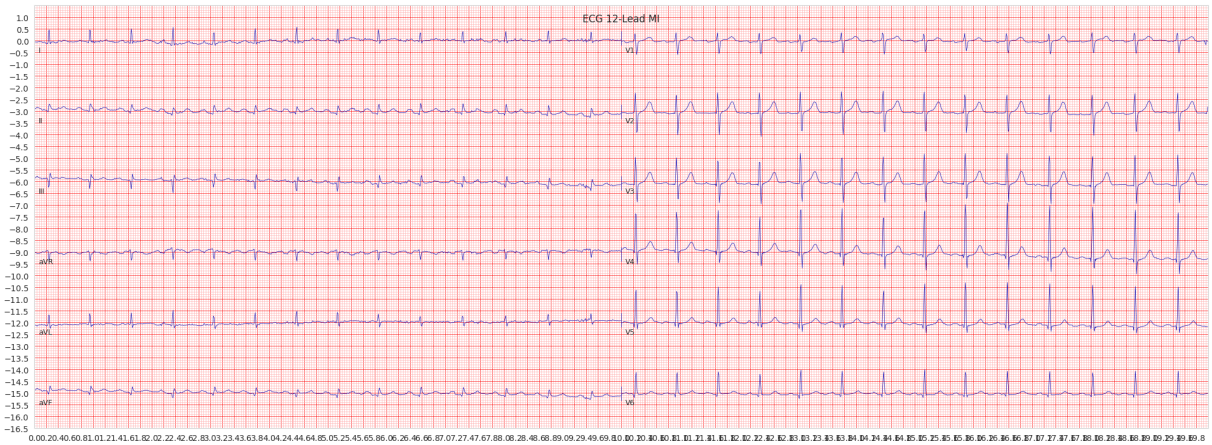
Raw ECG signal

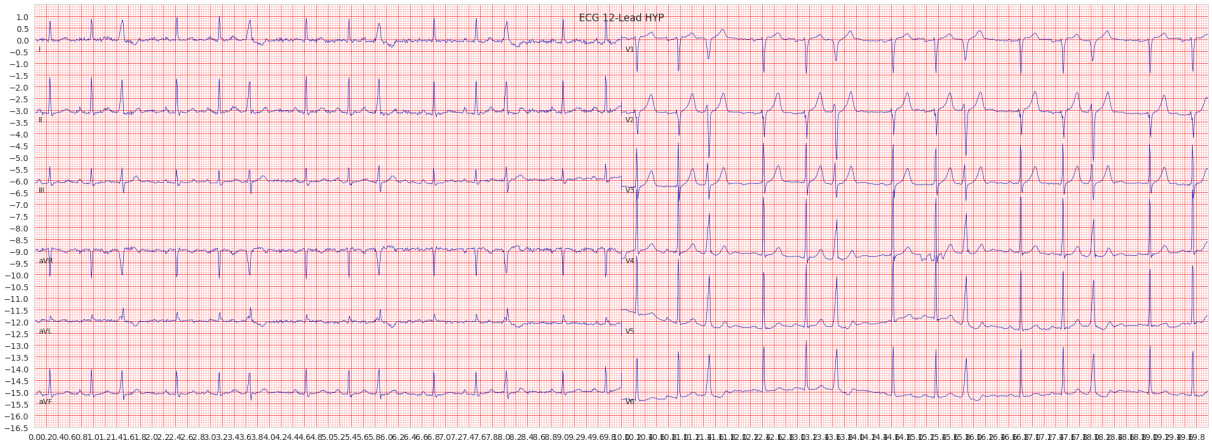


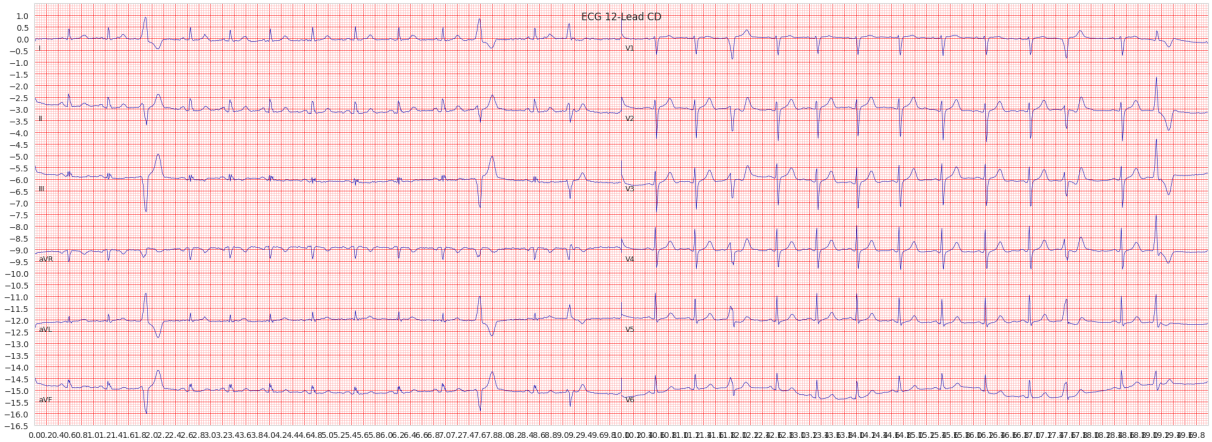
Processed ECG Signal after Pan–Tompkins algorithm and QRS detector

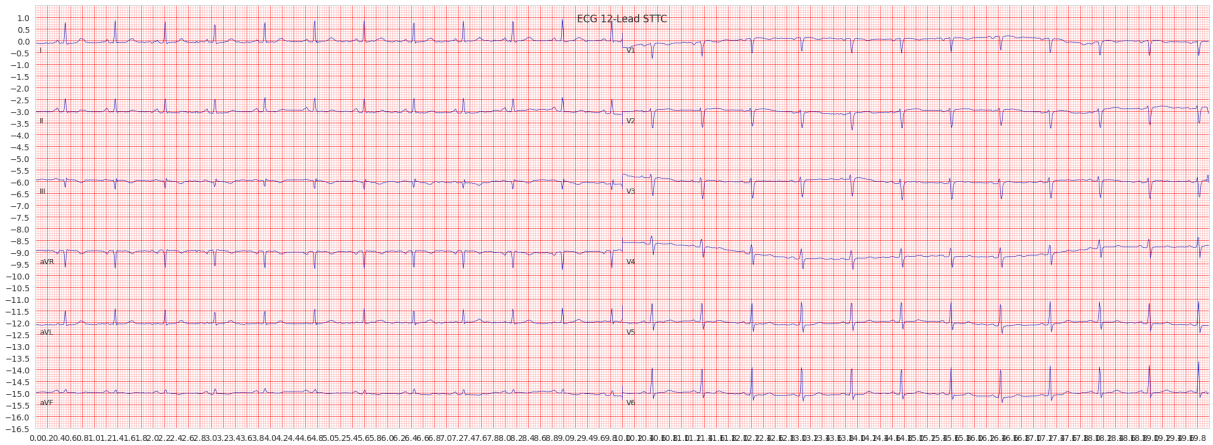
▼ 12-Lead ECG visualization - Click on Toggle to display

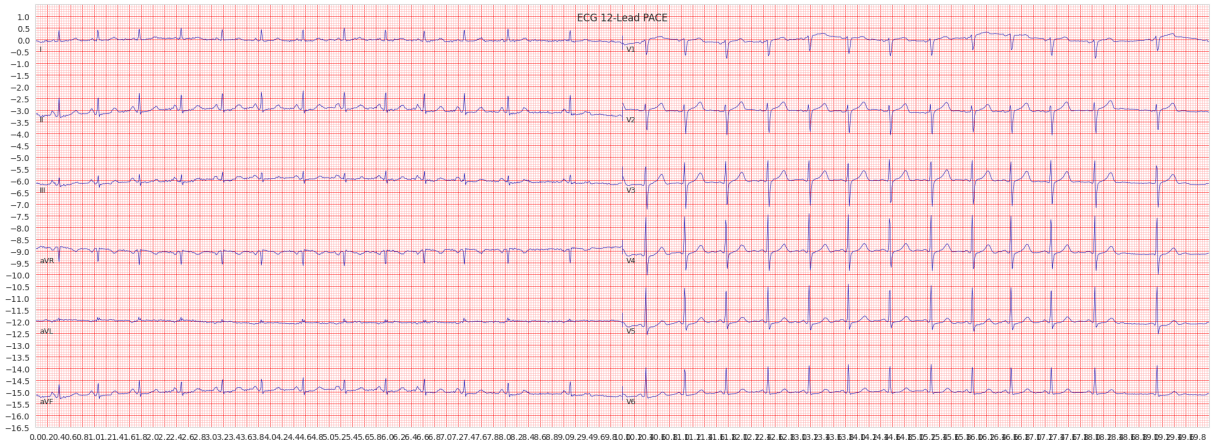



ECG 12-Lead MI
I
II
III
aVR
aVL
aVF
V1
V2
V3
V4
V5
V6


ECG 12-Lead HYP
I
II
III
aVR
aVL
aVF
V1
V2
V3
V4
V5
V6


ECG 12-Lead CD
I
II
III
aVR
aVL
aVF
V1
V2
V3
V4
V5
V6





Feature Engineering in ECG data

The feature engineering stage is a crucial step in the ECG signal analysis process, as it involves transforming raw data into a set of relevant and informative features that can be used to understand the underlying properties and patterns in the signal. In this section, we will be extracting a wide range of features from our ECG signal, including time and amplitude-based features, heart rate variability (HRV) metrics, and heart rate statistics.

The time and amplitude-based features include the `PR interval` (the time interval between the P wave and the R wave), `P max` (the maximum amplitude of the P wave), `P to R` (the ratio of the amplitude of the P wave to the R wave), and `ST interval` (the time interval between the S wave and the T wave). Other features include the `T max` (the maximum amplitude of the T wave), `P to Q` (the difference in amplitude between the P wave and the Q wave), and `T to R` (the ratio of the amplitude of the T wave to the R wave).

For HRV analysis, we will be computing several key metrics, including `RMSSD, NN50, pNN50, NN20, pNN20, SDNN, mRRi, stdRRi, mHR`, and others, as explained below:

- RMSSD (Root mean square of successive differences): measures the square root of the average of the squared differences between successive RR intervals. This metric provides information about the difference between consecutive heartbeats, and is widely used to assess the HRV.
- NN50 (Number of pairs of successive NN intervals that differ by more than 50ms): counts the number of pairs of successive NN intervals that differ by more than 50ms.

- pNN50 (Proportion of NN50 divided by total number of NN intervals): calculates the proportion of NN50 divided by the total number of NN intervals.
- NN20 (Number of pairs of successive NN intervals that differ by more than 20ms): counts the number of pairs of successive NN intervals that differ by more than 20ms.
- pNN20 (Proportion of NN20 divided by total number of NN intervals): calculates the proportion of NN20 divided by the total number of NN intervals.
- SDNN (Standard deviation of NN intervals): calculates the standard deviation of the NN intervals. This metric provides information about the variability of the heart rate.
- mRRi (mean length of RR interval): calculates the mean length of the RR intervals.
- stdRRi (standard deviation of RR intervals): calculates the standard deviation of the RR intervals.
- mHR (mean heart rate): calculates the mean heart rate, in ms.

Additionally, we will calculate various `heart rate statistics`, including the `mean`, `median`, `mode`, `standard deviation`, and `maximum` and `minimum` values.

Finally, we will also compute several R-peak related features, including `r_mean` (the mean value of the R-peak values in the signal), `r_std` (the standard deviation of the R-peak values in the signal), `beats_to_length` (the ratio of the number of R-peaks in the signal to the length of the signal), `filtered_r` (the number of R-peaks that have an inter-beat time that is less than 0.5 times the mean inter-beat time), and `rel_filtered_r` (the ratio of the number of filtered R-peaks to the total number of R-peaks in the signal).

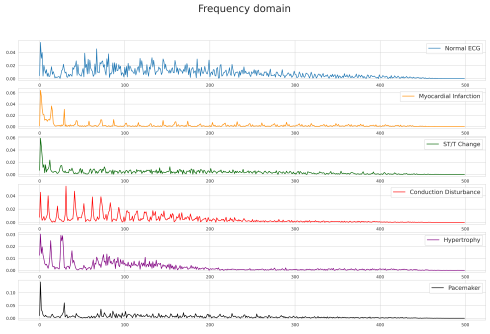
Some of these features are extracted using the `feature_extractor` function and the full features can be seen on the notebook. With these features in hand, we will be able to gain a much deeper and more comprehensive understanding of our ECG signal and its underlying patterns and properties.

Comparison in Time Domain as in Frequency Domain

The comparison of ECG signals from different subsets of diagnoses was performed both in the Time Domain and the Frequency Domain. In the Time Domain, ECG signals were plotted on a graph to show their shape, amplitude, and duration. The comparison allowed us to observe any variations in the morphological features of ECG signals, such as the height, width, and shape of P, Q, R, S, and T waves.

In the Frequency Domain, the ECG signals were transformed using the Fast Fourier Transform (FFT) algorithm, which converts a time-domain signal into its frequency components. The FFT plot shows the magnitude of each frequency component of the ECG signal, which provides information about the rhythm and regularity of the heartbeats. In this analysis, we also plotted the ECG signal from a pacemaker for reference to help us better understand the results of the comparison.

The results of the comparison in both domains provided a comprehensive understanding of the differences between ECG signals from different diagnoses. By observing the morphological features of the ECG signals in the Time Domain and the frequency components in the Frequency Domain, we can draw conclusions about the diagnosis and the health status of the patients. This analysis helps us to develop a more accurate and effective ECG analysis tool that can be used in medical practice.



Deep Learning models

Deep learning models have recently shown promising results for electrocardiogram signal classification tasks. Compared to traditional methods such as feature engineering and rule-based approaches, deep learning models are better equipped to capture complex patterns in the ECG signal, which can help to improve the accuracy of diagnosis and prognosis. Research states that the results of the models showed that convolutional neural networks, in particular resnet- and inception-based architectures, had the strongest performance across all tasks outperforming feature-based algorithms by a large margin.

I followed the trend of training 1d convolutional neural networks (CNN) on raw ECG signals for heart arrhythmia classifcation. I adapted the ECGNet and ResNet101 models to the code used to analyze the ECG signals. The ECGNet model is a Convolutional Neural Network with multiple kernel sizes, the second network is a Long Short Term Memory (LSTM) network, and the third network is a combination of CNN and LSTM, known as the CLSTM network. The ResNet101 model is a deep residual learning framework to facilitate the learning of deeper networks.

Training

According to the source of the dataset, it is recommended to use stratified 10-folds in order to ensure that the predictive model is accurately trained on the entire dataset. The same patients should be kept within the same folds; this practice helps to minimize bias and ensure that the dataset's integrity is maintained. The 9th and 10th folds are of higher quality and should be used as the validation and test sets, respectively. This ensures that the model is trained on diverse data, while the validation and test sets are used to accurately evaluate the performance of the model. By using this approach, the model can be tuned to ensure that it generalizes well to unseen data.

> Cross-validation Folds: recommended 10-fold train-test splits (strat_fold) obtained via stratified sampling while respecting patient assignments, i.e. all records of a particular patient were assigned to the same fold. Records in fold 9 and 10 underwent at least one human evaluation and are therefore of a particularly high label quality. We therefore propose to use folds 1-8 as training set, fold 9 as validation set and fold 10 as test set.

Following the recommendations, the ECG signals were divided into three subsets: training, validation, and testing data. The training data, X_train_scale, consists of 17,090 ECG signals with a shape of (17090, 1000, 12), meaning that each signal is a 1000-sample, 12-lead ECG signal. The corresponding diagnostic labels for these signals are stored in y_train, which has a shape of (17090, 5). The validation data, X_val_scale, contains 2147 ECG signals with a shape of (2147, 1000, 12) and the diagnostic labels for these

signals are stored in y_val, which has a shape of (2147, 5). The testing data, X_test_scale, consists of 2159 ECG signals with a shape of (2159, 1000, 12) and the diagnostic labels for these signals are stored in y_test, which has a shape of (2159, 5).

The diagnostic labels for each ECG signal have been processed using MultiLabelBinarizer() and are stored as arrays of 5 elements. The use of MultiLabelBinarizer() allows for multiple diagnoses to be assigned to each ECG signal, allowing for a more comprehensive analysis of the diagnostic information contained within the ECG signals.

Optimization

In deep learning, training models can consume a large amount of GPU memory, especially for large and complex models. To optimize GPU memory usage during model training, I had to consider several key factors:

1. Batch size: The larger the batch size, the more memory is required to store intermediate values. Reducing the batch size can significantly reduce GPU memory usage.
2. Model architecture: Different model architectures have different memory requirements. For example, some models such as Convolutional Neural Networks (CNNs) can consume more memory than Recurrent Neural Networks (RNNs).
3. Data type: PyTorch and other deep learning frameworks allow for different data types to be used such as float32 or float16. Reducing the data type from float32 to float16 can significantly reduce memory usage, but can also reduce accuracy.
4. GPU hardware: GPU hardware with more memory and faster processing speeds can handle larger models and batch sizes, resulting in more efficient training.
5. Regularization techniques: Regularization techniques such as dropout can help reduce overfitting, but can also increase memory usage. Using early stopping and weight decay can help reduce memory usage while still achieving good results.

This optimization is essential for efficient and effective training. By considering the factors outlined above, one can find the right balance between memory usage and model performance, leading to more successful model training.

Results

For a reason, my wandb environment was not set up properly so I couldn't upload the logs to the online platform to share it publicly. However, below are shown the results regarding the performance of both of our Deep Learning models.

ResNet101

The training process resulted in a loss of 0.02 and a mean accuracy of 99.28%. The model achieved a high ROC score of 99.96% during training. On the other hand, during testing, the model presented a loss of 2.77, a mean accuracy of 84.58% and a ROC score of 85.76%.

When evaluating the model on a separate test set, it achieved a ROC score of 84.28%, a mean accuracy of 84.68%, and an F1 score (max) of 72.43%. The accuracy for each diagnostic subclass ranged from 82.98% to 87.92%. The precision, recall and F1 scores for each subclass were also calculated and varied from 70.54% to 84.29%.

In conclusion, the ResNet101 model demonstrated good performance on the ECG signals and achieved promising results in terms of accuracy, ROC score and F1 score.

ECGNet

The ECGNet model was evaluated using various metrics and the results are as follows:

For the training phase, the model achieved a train loss of 0.023 and a train mean accuracy of 99.1%, with a train ROC score of 99.9%. Meanwhile, in the testing phase, the model's test mean accuracy was 82.5%, with a test ROC score of 82.9%.

In the testing phase, the overall ROC score was 82.6%, the mean accuracy was 82.7%, and the accuracy ranged from 79.57% to 87.36%. The F1 score (Max) was 68.2%, and the class AUC ranged from 78.6% to 86%. The class precision recall F1 score ranged from 63.8% to 71.9%.

It's worth noting that while the model performed well in the training phase, there is room for improvement in the testing phase, particularly in terms of reducing the test loss and increasing the overall accuracy and F1 score.

Tools

- https://github.com/PIA-Group/BioSPPy
- https://pypi.org/project/hrv-analysis/
- https://github.com/Pramod07Ch/Pan-Tompkins-algorithm-python
- https://github.com/dy1901/ecg_plot
-

References

- Pan, Jiapu, and Willis J. Tompkins. "A real-time QRS detection algorithm." *IEEE transactions on biomedical engineering* 3 (1985): 230-236.
- Reddy, Likith, et al. "Imle-net: An interpretable multi-level multi-channel model for ecg classification." *2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2021.
- Clifford, Gari D., et al. "AF classification from a short single lead ECG recording: The PhysioNet/computing in cardiology challenge 2017." *2017 Computing in Cardiology (CinC)*. IEEE, 2017.
- B. Murugesan et al., "ECGNet: Deep Network for Arrhythmia Classification," 2018 IEEE International Symposium on Medical Measurements and Applications (MeMeA), Rome, Italy, 2018, pp. 1-6, doi: 10.1109/MeMeA.2018.8438739.
- He, Kaiming, et al. "Deep residual learning for image recognition." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.
- Wagner, Patrick, et al. "PTB-XL, a large publicly available electrocardiography dataset." *Scientific data* 7.1 (2020): 154.
- Hoshi, Rosangela Akemi, et al. "Poincaré plot indexes of heart rate variability: relationships with other nonlinear variables." *Autonomic Neuroscience* 177.2 (2013): 271-274.

Further Approach: Embedding Space from ECG signals

In addition to the main approach used in this study, I also wanted to explore another method based on embeddings inspired by recent papers from the field of neuroscience. The papers I referred to are:

- "Decoding speech from non-invasive brain recordings" (**https://arxiv.org/pdf/2208.12266.pdf**)
- "Attention modulates neural representation to render reconstructions according to subjective appearance" (**https://www.nature.com/articles/s42003-021-02975-5.pdf**)

Both of these papers use EEG signals to decode speech and language processing in real-time from non-invasive recordings of brain activity. Despite the fact that they use EEG signals and not ECG signals, I believe that the concept of creating embeddings from raw signals can be applied to ECG signals as well.

My vision was to create an Autoencoder based on the transformer architecture to create embeddings from 12-Lead ECG signals. This would allow the ECG signals that represent a specific disease to be closer in the latent space, thereby making it easier to classify them properly. By creating an embedding for a new ECG signal, it would be possible to calculate its nearest cluster, which would help us to classify the signal.

Unfortunately, I did not have enough time to implement this approach in this study. However, I believe that it holds great potential for future research in ECG signal classification and analysis. The use of embeddings and the transformer architecture could be beneficial for this task as they have been shown to be effective in capturing complex relationships within data and improving performance in various machine learning tasks.