Project Report

on

ECG Anomaly Detection using CNN Autoencoders

Submitted for Minor Project

of

B.Tech and M.Tech

in

Mathematics and Data Science

Submitted by

Arindam Phatowali

Scholar No. 214104021

Under the guidance of

Dr. C.K. Verma

Department of Mathematics, Bioinformatics and Computer Applications

Maulana Azad

National Institute of Technology Bhopal - 462003 (India)

MAULANA AZAD NATIONAL INSTITUTE OF TECHNOLOGY, BHOPAL (M.P)-462003

CERTIFICATE

This is to certify that Arindam Phatowali (Sch no- 214104021), a student of Dual Degree (B.Tech+M.Tech) of batch 2021 -2026 has completed the project titled “ECG Time Series Anomaly Detection using CNN Autoencoders” being submitted in the partial fulfilment of the requirement of the completion of Dual Degree in Mathematics and Data Science to Maulana Azad National Institute of technology Bhopal under my supervision.

Date: 24th April 2024
Place: Bhopal

Dr. C.K. Verma
(Supervisor)

INDEX

Abstract

Anomaly detection is critical in various domains, including healthcare, finance, and cybersecurity. Anomaly detection in ECG signals can help identify abnormal heartbeats, which can be indicative of various cardiac conditions. Autoencoders have shown promising results in anomaly detection tasks, as they can learn to reconstruct normal data and identify abnormal data that cannot be reconstructed accurately.

The project "ECG Time Series Anomaly Detection using CNN Autoencoder" aims to leverage the power of Convolutional Neural Networks (CNNs) and Autoencoders to develop a model capable of identifying irregularities in ECG data in real-time, ensuring high accuracy in differentiating between normal and anomalous ECG signals. This capability is crucial for enhancing cardiac health monitoring, providing a tool that can be utilized in healthcare settings to improve patient care.

Acknowledgement

I take this opportunity to express my sincere gratitude and respect to MANIT Bhopal for providing me a platform to pursue my studies and carry out 3rd Year Minor Project.

I have a great pleasure in expressing my deep sense of gratitude to Dr CK Verma for their constant support and encouragement throughout the course of this project.

I also extend my thanks to all the faculty of MDS who directly or indirectly encouraged me.

Finally, I would like to thank my friends for all their moral support they have given me during the completion of this work.

Introduction

1. Relevance of the Project

1.1 Anomaly Detection in ECG Signals:

The project is highly relevant in the context of healthcare, where the early detection of anomalies in ECG signals can significantly impact patient care. Anomalies in ECG signals can indicate various cardiac conditions, including arrhythmias, myocardial infarction, and other heart diseases. By leveraging deep learning techniques, the project aims to enhance the accuracy and efficiency of anomaly detection in ECG data, thereby facilitating timely intervention and treatment.

1.2 Real-time Monitoring:

The project emphasizes real-time anomaly detection, which is crucial for continuous monitoring of patients' cardiac health. This capability allows healthcare providers to monitor patients' ECG signals in real-time, enabling immediate response to any irregularities detected.

1.3 Transfer Learning and Feature Clustering:

The project's approach to anomaly detection involves the use of Convolutional Neural Networks (CNNs) and Autoencoders, which can be validated through features clustering and transfer learning. This methodology not only enhances the model's performance in detecting anomalies but also opens up possibilities for feature extraction and transfer learning across different datasets, thereby broadening the scope of the project's application.

2. Problem Statement and Objective:

The problem statement revolves around the detection of anomalies in ECG signals, which are irregularities that deviate from the norm. The objective is to develop a model that can accurately identify these anomalies in real-time, ensuring high accuracy in differentiating between normal and anomalous ECG signals. This is crucial for enhancing cardiac health monitoring and providing a tool that can be utilized in healthcare settings to improve patient care.

3. Scope of the Project

3.1 Technical Application: The project's scope extends to the technical aspects of developing and implementing a deep learning model for ECG anomaly detection. This includes data preprocessing, model building using CNN Autoencoders, and model evaluation techniques such as anomaly detection, features clustering, and latent space exploration.

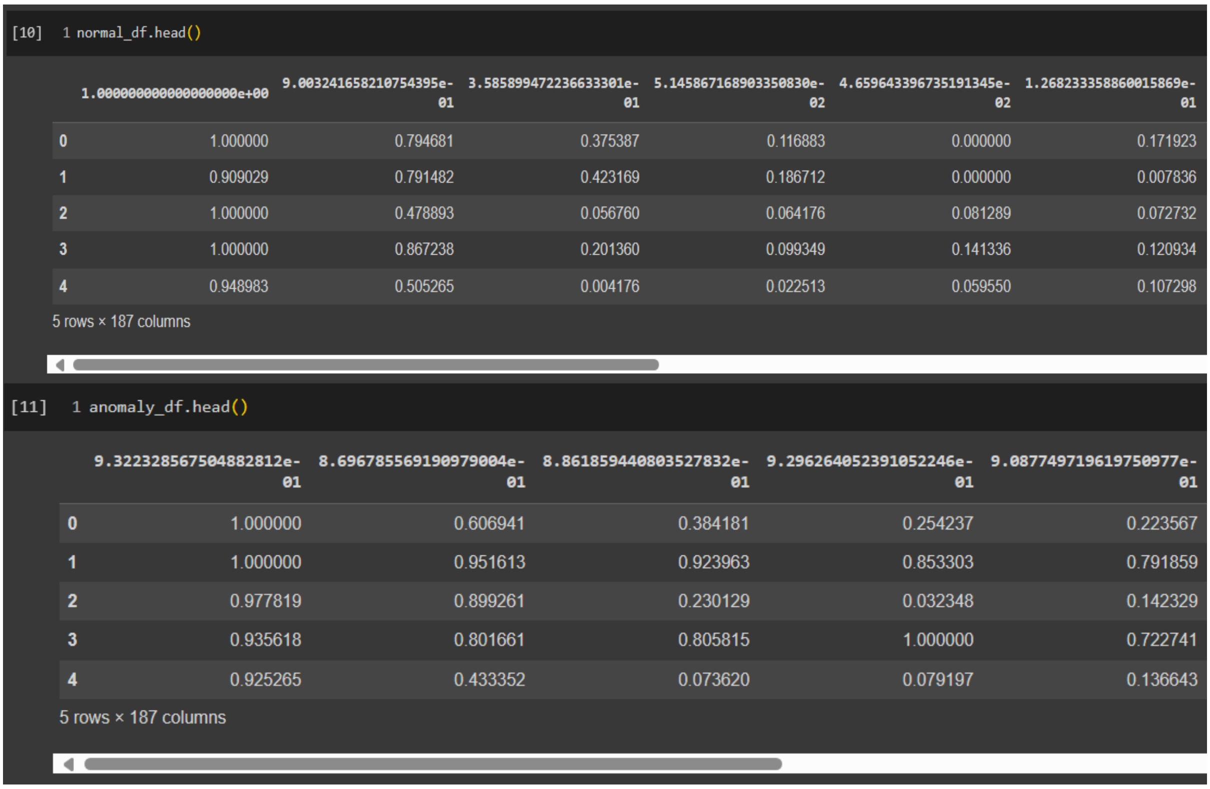
3.2 Healthcare Impact: The project's scope is not limited to the technical development of the model but also encompasses its impact on healthcare. By providing a tool for real-time anomaly detection in ECG signals, the project aims to contribute to the field of cardiac health monitoring. This can lead to improved patient outcomes by enabling early detection and treatment of cardiac conditions.

3.3 Research and Development: The project's scope also includes the exploration of new methodologies and techniques in the field of ECG anomaly detection. By investigating the use of CNN Autoencoders and other deep learning models, the project contributes to the ongoing research and development in healthcare technology, aiming to advance the field of cardiac monitoring and patient care.

Process

1. Data Collection:

The dataset used for this project is 'The PTB Diagnostic ECG Database' which is part of 'The ECG Heartbeat Categorization Dataset' available on Kaggle originally sourced from Physionet. This Dataset has two csv files, one for normal and the other for abnormal ECG signals, the normal data contain **4045 rows** and **188 columns** and abnormal data contain **10505 rows and 188 columns**. The **188th** column in both normal and abnormal data contains a binary target which is **0** for normal and **1** for anomaly.



```
[10] normal_df.head()
```

	1.000000000000000000e+00	9.003241658210754395e-01	3.585899472236633301e-01	5.145867168903350830e-02	4.659643396735191345e-02	1.268233358860015869e-01
0	1.000000	0.794681	0.375387	0.116883	0.000000	0.171923
1	0.909029	0.791482	0.423169	0.186712	0.000000	0.007836
2	1.000000	0.478893	0.056760	0.064176	0.081289	0.072732
3	1.000000	0.867238	0.201360	0.099349	0.141336	0.120934
4	0.948983	0.505265	0.004176	0.022513	0.059550	0.107298

5 rows × 187 columns

```
[11] anomaly_df.head()
```

	9.322328567504882812e-01	8.696785569190979004e-01	8.861859440803527832e-01	9.296264052391052246e-01	9.087749719619750977e-01
0	1.000000	0.606941	0.384181	0.254237	0.223567
1	1.000000	0.951613	0.923963	0.853303	0.791859
2	0.977819	0.899261	0.230129	0.032348	0.142329
3	0.935618	0.801661	0.805815	1.000000	0.722741
4	0.925265	0.433352	0.073620	0.079197	0.136643

5 rows × 187 columns

2. Data Preprocessing:

Import the necessary libraries like NumPy, pandas, matplotlib, seaborn, sklearn, tensorflow and keras. After that load the data and take a basic understanding of data like its shape, sample, is there are any NULL values present in the dataset. Understanding the data is an important step for any machine learning or deep learning project. The collected data is data is already clean enough and there is no need of much preprocessing.

3. Exploratory Data Analysis (EDA):

Exploratory Data Analysis (EDA) is a critical step in understanding the underlying structure and characteristics of the dataset before applying any machine learning algorithms. In the context of the "ECG Anomaly Detection using CNN Autoencoder" project, EDA involves examining the normal and anomaly ECG datasets to gain insights into their distributions, patterns, and potential anomalies.

The EDA process includes two main components:

3.1 Comparing Samples:

We generate plots comparing random samples from the normal and anomaly datasets. These plots provide visual comparisons between the two classes, allowing us to observe any discernible differences in their shapes, amplitudes, or other features. By visually inspecting these plots, we can identify potential patterns or anomalies that may exist within the data.

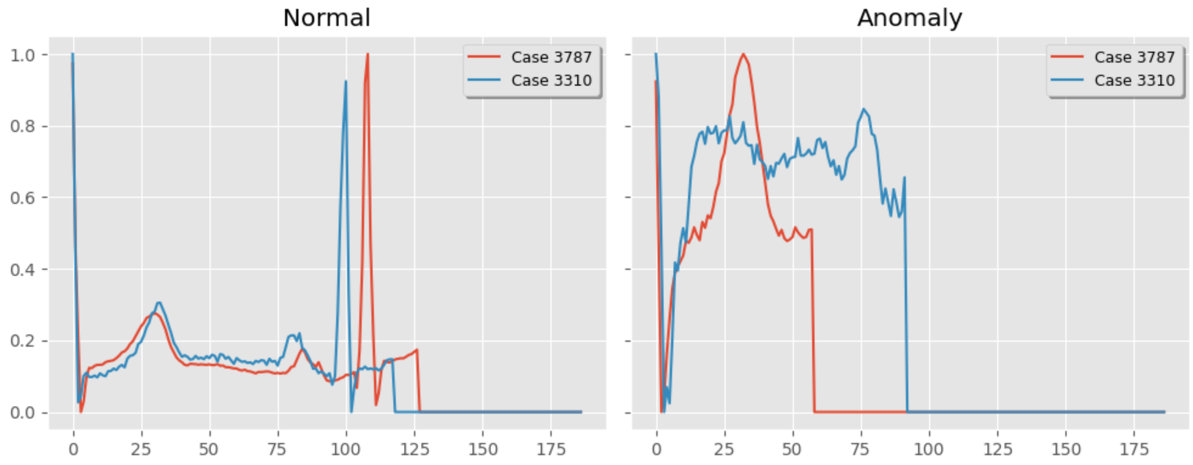


Figure 1: Comparison of two random samples from normal and anomaly datasets

3.2 Smoothed Mean and Confidence Intervals:

We plot smoothed mean and confidence intervals for both normal and anomaly classes. This involves calculating the mean of each class and smoothing it to reduce noise, along with computing confidence intervals to understand the variability within each class. These plots help us visualize the overall distribution of ECG signals within each class and identify any significant differences or outliers.

Overall, EDA serves as a crucial preliminary step in understanding the structure and nuances of the ECG datasets, enabling us to make informed decisions during preprocessing, model building, and evaluation stages of the project. It helps us identify potential challenges, outliers, or anomalies within the data and guides us in selecting appropriate strategies for handling them effectively.

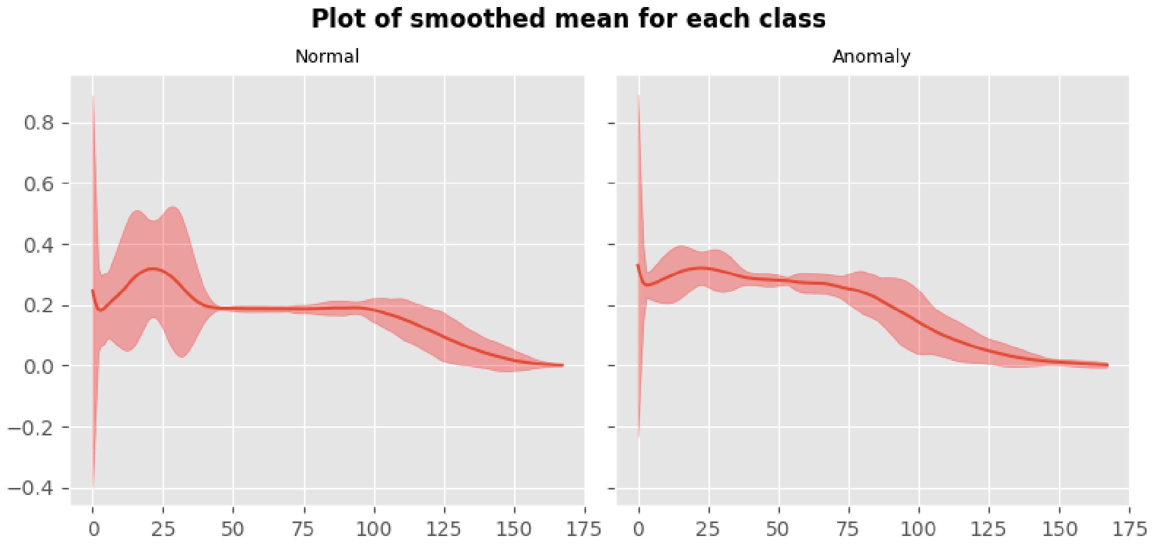


Figure 2: Plot of smoothed mean from each class

4. Model Building:

The model building phase is a critical component of the ECG Anomaly Detection project, focusing on constructing a Convolutional Neural Network (CNN) Autoencoder. This model is designed to learn the underlying patterns in the ECG signals, enabling it to distinguish between normal and anomalous signals. The CNN Autoencoder is a type of deep learning model that is particularly well-suited for this task due to its ability to capture complex, non-linear relationships in the data.

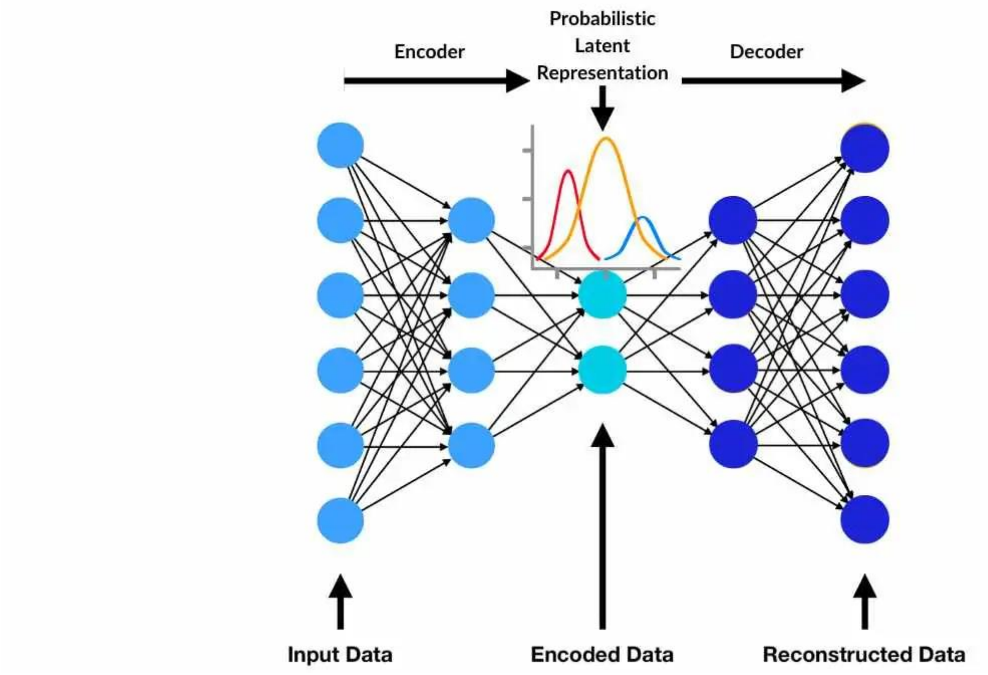


Figure 3: Structure of Autoencoder

The architecture of the CNN Autoencoder consists of two main parts: the encoder and the decoder. The encoder is responsible for compressing the input ECG signals into a lower-dimensional representation, capturing the essential features of the data. This is achieved through a series of convolutional layers, each followed by a batch normalization layer to improve the model's performance. The encoder's output is then passed through a bottleneck layer, which further compresses the data.

The decoder, on the other hand, is tasked with reconstructing the original ECG signals from the compressed representation. It does this by reversing the operations performed by the encoder, starting with the bottleneck layer and progressively expanding the data through a series of deconvolutional layers. Each deconvolutional layer is followed by a batch normalization layer to ensure the model's stability and performance.

The model is trained using a dataset of normal ECG signals, with the goal of learning to reconstruct these signals accurately. The training process involves feeding the ECG signals through the encoder and decoder, adjusting the model's weights to minimize the difference between the reconstructed signals and the original inputs. This is achieved through a process known as backpropagation, where the model learns to adjust its weights in response to the error between the predicted and actual outputs.

The training is conducted over 100 epochs, with each epoch representing one complete pass through the entire dataset. During training, the model's performance is monitored using a validation set, which is a subset of the training data not used in the training process. This allows for the early detection of overfitting, where the model performs well on the training data but poorly on unseen data.

To further enhance the model's performance and robustness, early stopping is employed. Early stopping halts the training process when the model's performance on the validation set starts to degrade, preventing the model from overfitting to the training data. This is achieved by monitoring the validation loss and stopping the training when the loss begins to increase, indicating that the model is starting to memorize the training data rather than learning to generalize from it.

The model is trained with a batch size of 128, which is a common choice for deep learning models. This batch size balances the trade-off between computational efficiency and model performance, allowing the model to learn effectively without requiring excessive computational resources.

After training, the model's performance is evaluated by plotting the training and validation loss against the number of epochs. Additionally, the Training dataset error, Testing dataset error, and Anomaly dataset error are calculated to assess the model's reconstruction accuracy. A threshold is set to distinguish normal data from anomalies, typically defined as the mean of the training loss plus one standard deviation. This threshold guides the model in identifying anomalies based on the reconstruction error.

In summary, model building in this project involves training a CNN autoencoder on normal ECG signals to learn their patterns and reconstruct them accurately. By optimizing the model's architecture, training process, and evaluation metrics, the goal is to develop a robust anomaly detection system for ECG signals that can effectively differentiate between normal and abnormal patterns.

5. Model Evaluation:

Model evaluation is a critical step in the "ECG Anomaly Detection using CNN Autoencoder" project, where we assess the performance of the trained CNN Autoencoder model in detecting anomalies in ECG signals accurately.

To begin the evaluation process, we employ various metrics and visualization techniques to analyze the model's performance across different datasets. One of the primary evaluation metrics used is accuracy, which measures the proportion of correctly classified instances out of the total number of instances. Additionally, we compute other performance metrics such as precision, recall, and F1-score, which provide insights into the model's ability to correctly identify anomalies while minimizing false positives and false negatives.

The model evaluation process involves several key components:

5.1 Evaluate_model Function:

The project includes an evaluate_model function that takes a model and a dataset as input and evaluates the model's performance on that dataset. If the provided dataset is the anomaly dataset, the function calculates the accuracy based on how many loss values are greater than the predefined threshold, indicating anomalies. For non-anomaly datasets, the accuracy is calculated based on how many loss values are less than or equal to the threshold, indicating normal data. The function returns the accuracy as a formatted string.

A code then calls the evaluate_model function for three different datasets: the training data (X_train), the testing data (X_test), and the anomaly data (anomaly). This provides a comprehensive evaluation of the model's performance on both normal and abnormal ECG signals.

5.2 Prepare_labels Function:

To compute performance metrics such as accuracy, precision, recall, and F1-score, the project includes a prepare_labels function. This function concatenates the true labels (ytrue) for the training, testing, and anomaly datasets, and then predicts the output and calculates the loss for each dataset. It converts the loss values to binary predictions based on whether they are above or below the predefined threshold and concatenates the predicted labels (ypred) for the training, testing, and anomaly datasets.

5.3 Plot_confusion_matrix Function:

The plot_confusion_matrix function is used to visualize the confusion matrix based on the true and predicted labels obtained from the prepare_labels function. It calculates various performance metrics, such as accuracy, precision, recall, and F1-score, and prints them out. It then constructs the confusion matrix using sns.heatmap and sets the x-axis and y-axis labels and title for the plot.

5.4 Model Evaluation and Visualization:

Finally, the code calls the plot_confusion_matrix function to visualize the confusion matrix for the model's performance on the training, testing, and anomaly datasets. It also prints out a classification report using classification_report from scikit-learn, which includes precision, recall, F1-score, and support for each class.

Overall, the model evaluation phase provides a comprehensive assessment of the CNN Autoencoder model's performance in detecting anomalies in ECG signals. Through a combination of quantitative metrics and visualizations, we gain valuable insights into the model's effectiveness and identify areas for improvement, ultimately guiding further iterations and refinements to enhance anomaly detection accuracy.

Result:

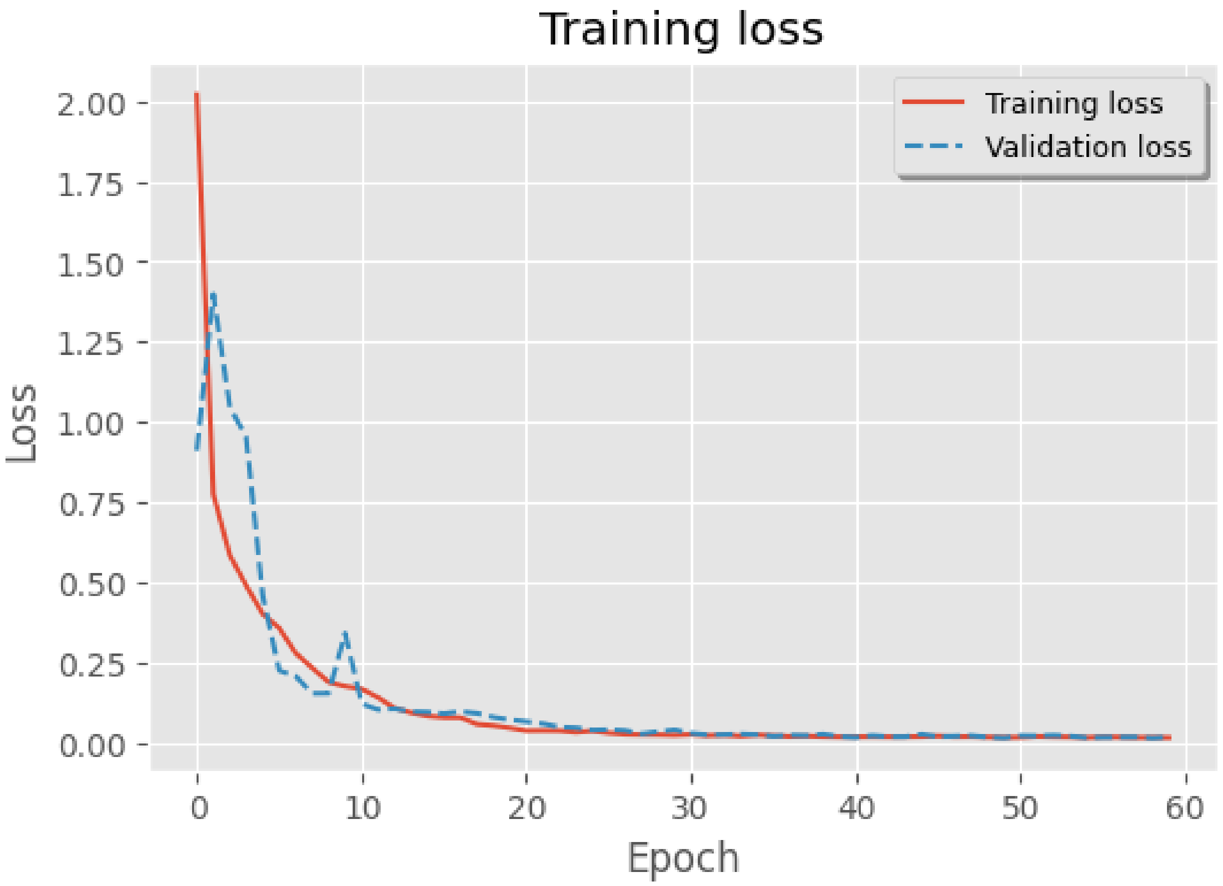


Figure 4: Plot of training loss against number of epochs

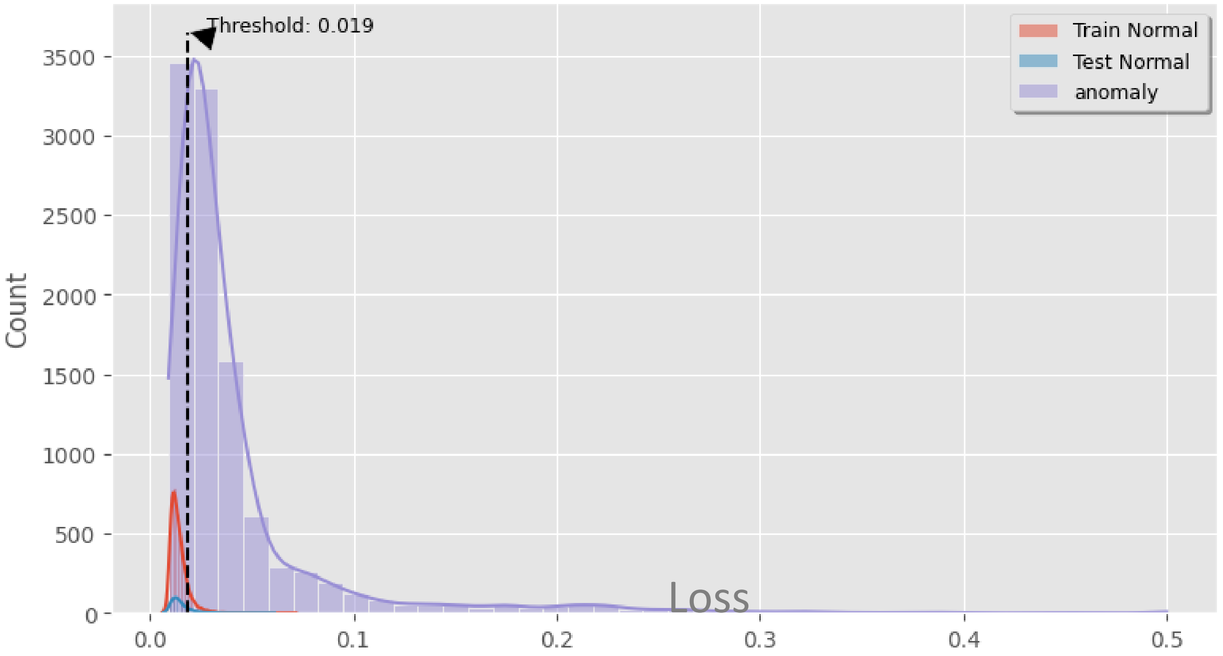
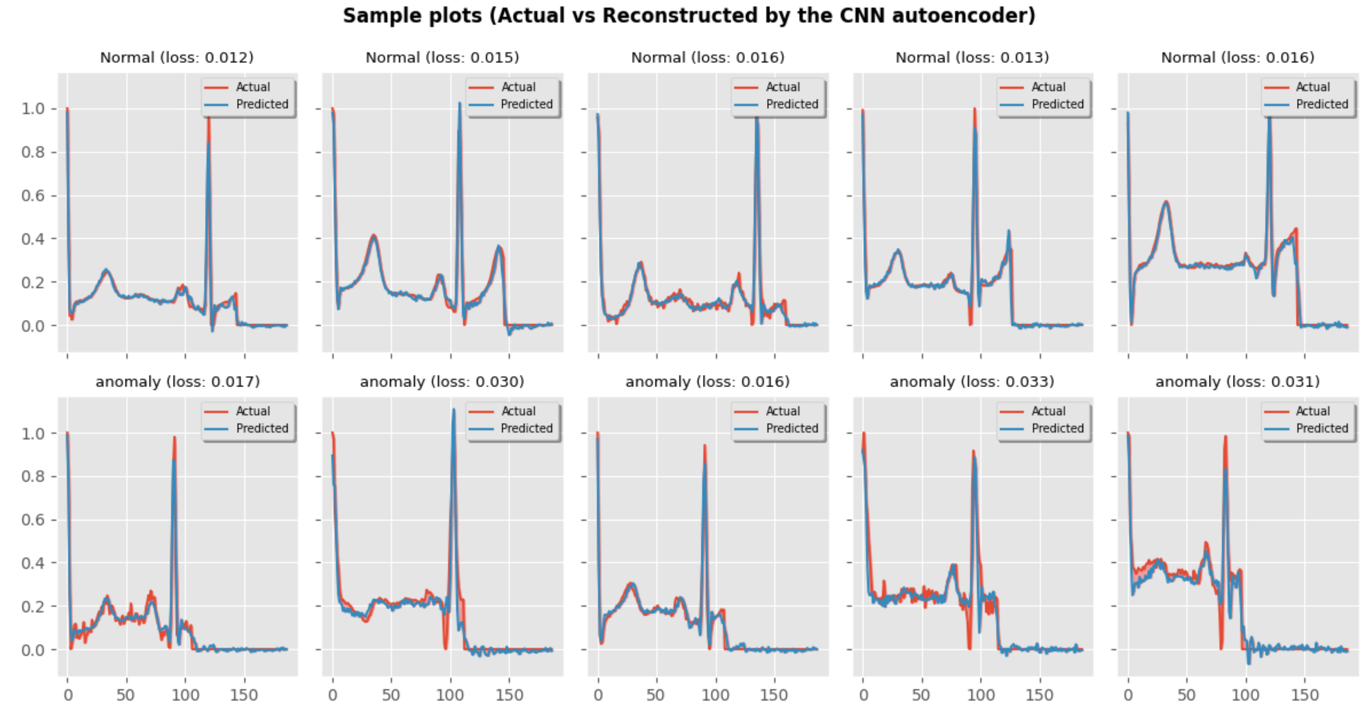


Figure 5: Loss Distribution and Threshold Visualization



Sample plots (Actual vs Reconstructed by the CNN autoencoder)

Normal (loss: 0.012)

Normal (loss: 0.015)

Normal (loss: 0.016)

Normal (loss: 0.013)

Normal (loss: 0.016)

anomaly (loss: 0.017)

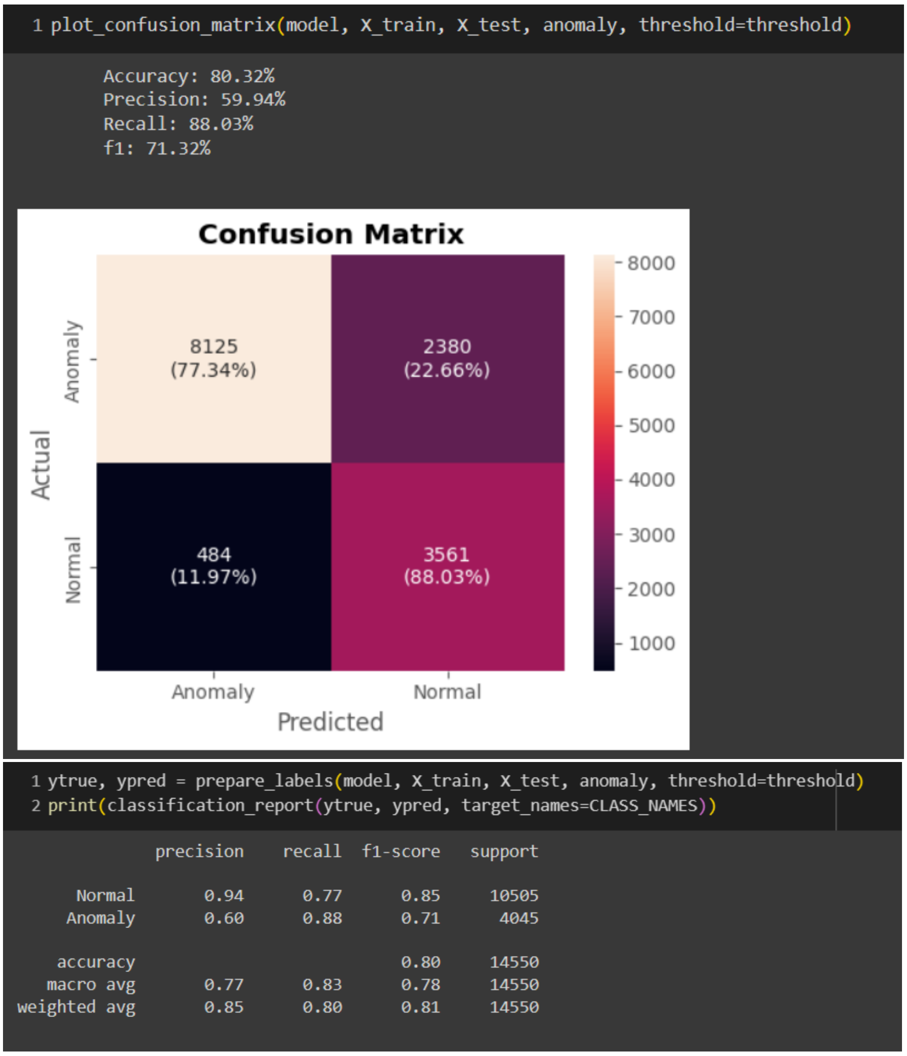
anomaly (loss: 0.030)

anomaly (loss: 0.016)

anomaly (loss: 0.033)

anomaly (loss: 0.031)

Figure 6: Sample plots for actual and reconstructed graphs



```
plot_confusion_matrix(model, X_train, X_test, anomaly, threshold=threshold)
```

```
Accuracy: 80.32%
Precision: 59.94%
Recall: 88.03%
f1: 71.32%
```

```
ytrue, ypred = prepare_labels(model, X_train, X_test, anomaly, threshold=threshold)
print(classification_report(ytrue, ypred, target_names=CLASS_NAMES))
```

```
              precision    recall  f1-score   support

      Normal       0.94      0.77      0.85     10505
     Anomaly       0.60      0.88      0.71      4045

    accuracy                           0.80     14550
   macro avg       0.77      0.83      0.78     14550
weighted avg       0.85      0.80      0.81     14550
```

Figure 7: Confusion Matrix

Project Link-

Google Drive-
https://drive.google.com/file/d/1A_HgBLN45imkRuKUGbxSod8CYgTXfcCA/view?usp=sharing

Github- https://github.com/arinp95/ECG-Anomaly-Detection-using-Autoencoders

Conclusion

The project successfully demonstrates the application of a CNN autoencoder for ECG anomaly detection. Despite the challenges posed by data imbalance and the complexity of ECG signals, the system achieves promising results in identifying abnormal ECG signals with 80.32% accuracy. This project has the potential to improve patient outcomes and streamline healthcare processes in clinical settings. The project's findings have significant implications for early detection of heart diseases, potentially saving lives. Future work could explore strategies to address the data imbalance issue and improve the model's performance further.

References:

1. https://www.kaggle.com/datasets/shayanfazeli/heartbeat

2. https://numpy.org/doc/stable/user/index.html#user

3. https://pandas.pydata.org/docs/user_guide/index.html

4. https://docs.python.org/3/
5. https://scikit-learn.org/stable/
6. https://www.analyticsvidhya.com/blog/2023/02/anomaly-detection-in-ecg-signals-identifying-abnormal-heart-patterns-using-deep-learning/
7. Understand Autoencoders in Machine Learning (analyticsvidhya.com)
8. https://www.cs.ucr.edu/~eamonn/neverending.pdf