A Study of ECG Heartbeat Categorization

Nguyen Xuan Tung - BI12-479

March 7, 2024

1 Introduction

This report presents the results of my study on ECG heartbeat categorization taken from Kaggle. The data used in this study is the MIT-BIH Arrhythmia Database. I tried to analyze on the data and apply 5 models based on Random Forest, CNN and RNN to classify 5 types of heartbeats.

2 Data Analysis

The data used in this study is the MIT-BIH Arrhythmia Database. The data consists of 5 types of heartbeats: Normal, Supraventricular, Ventricular, Fusion and Unknown, which are encoded to 0, 1, 2, 3 and 4 respectively. Each data contains 187 values of the heartbeat signal.

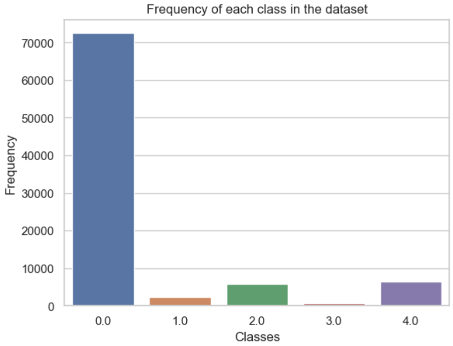

Figure 1: Frequency of each class in the dataset

As we can see from the histogram at figure 1, the training dataset in Kaggle is extremely bias to the Normal class, which is the most common type of heartbeat. This biasness can lead to the model being overfitted to the Normal class.

To avoid overfitting, the up-sample technique to balance the dataset was used. When divided the total number of files in the training dataset by 5, I obtained about 17510. So I balanced the dataset by sampling each class to have 17500 samples in each class.

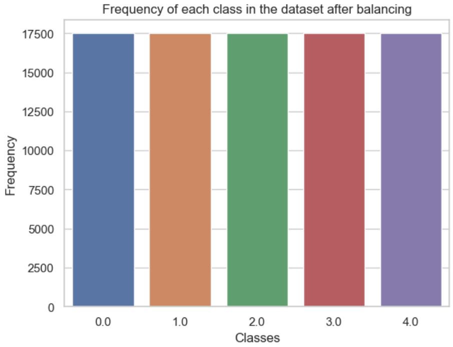

Figure 2: Frequency of each class in the dataset after balancing

Besides, I also split the test dataset in Kaggle into 2 parts: 50% for validation and 50% for testing. The distributions of 2 these datasets can be seen in the figure 3.

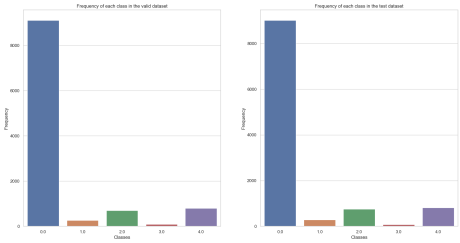

Figure 3: Frequency of each class in the valid and test dataset

3 Model

For categorization tasks, I tried to implement 5 different classification models which take the 187 values of the heartbeat signal as input and output the category of the heartbeat.

3.1 Random Forest

The first model I used is Random Forest Model, because it is a non high cost model for classifying structure data. When processing this model, it is not using the validation dataset. The number of estimators is 100 while I choose 42 for the random state.

3.2 Convolutional Neural Network

I tried to implement a simple convolutional neural network due to the powerful of the concept of convolution operation combined deep learning model which seem to be useful for signal processing in these days. The network contains three 1D convolutional layers for feature extraction of the signal and two fully connected layers.

3.3 Convolutional Neural Network with Residual Connection

The third model is a convolutional neural network with 1 first Conv1D layer and 5 residual blocks. The reason I used this model is that the skip connection can help it to prevent the vanishing gradient problem, which avoids the model from being overfitted. Besides, I also add some special layers such as BatchNormalization and Dropout to improve the model's performance.

3.4 Recurrent Neural Network

Besides using CNNs-based model, a recurrent neural network with 3 recurrent layers and 3 fully connected layers is also be used. The reason I used this model is that an RNN model can capture the long-term dependencies in the sequence data, which is the characteristic of the ECG signal.

3.5 Recurrent Neural Network with Residual Connection

Similarity to the idea of CNN with residual connection, I also tried to leverage an RNN model with residual connection.

4 Evaluation and Results

4.1 Loss Track of Deep Learning Models

As can be seen from the plots of loss for all models, the train loss of all models are decreased nearly to 0. However, the CNN with Residual Connection seems to tackle the overfitting problem. The reasons for that is easily understand by the mechanism of Residual Block which is avoiding the vanishing gradient problem.

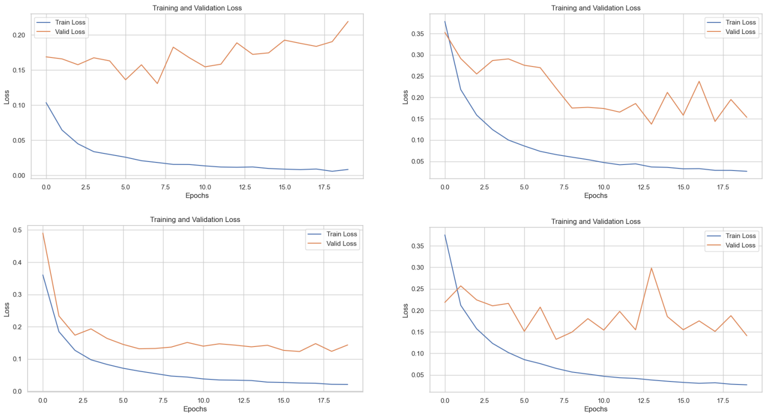

Table 1: Loss track of the Deep Learning models

4.2 Confusion Matrices of all Models

For the results recorded in the testing dataset, CNN with Residual Connection also shows the best prediction for unseen data.

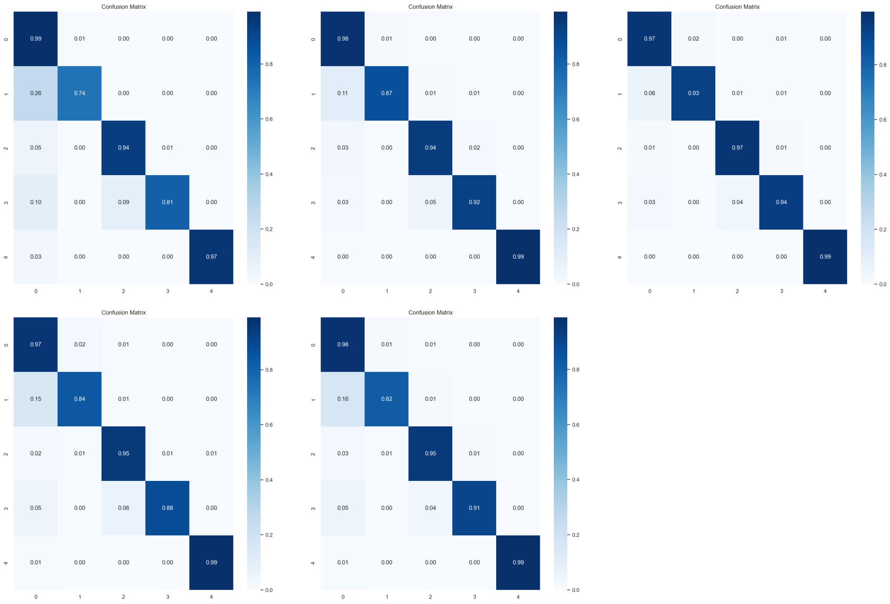

Table 2: Confusion Matrices of all Models