A Study of ECG Heartbeat Categorization

Student Hoang Minh Chi
ID BA12-030

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 have experimented with LSTM to classify into 5 classes: 0: "Normal Beats", 1: "Supraventricular Ectopy Beats", 2: "Ventricular Ectopy Beats", 3: "Fusion Beats", 4: "Unclassifiable Beats".

Data Analysis

This study utilizes the MIT-BIH Arrhythmia Database, which contains five types of heartbeats: Normal, Supraventricular, Ventricular, Fusion, and Unknown. These heartbeat types are assigned numerical labels of 0, 1, 2, 3, and 4, respectively. Each data sample represents a heartbeat signal comprising 187 values.

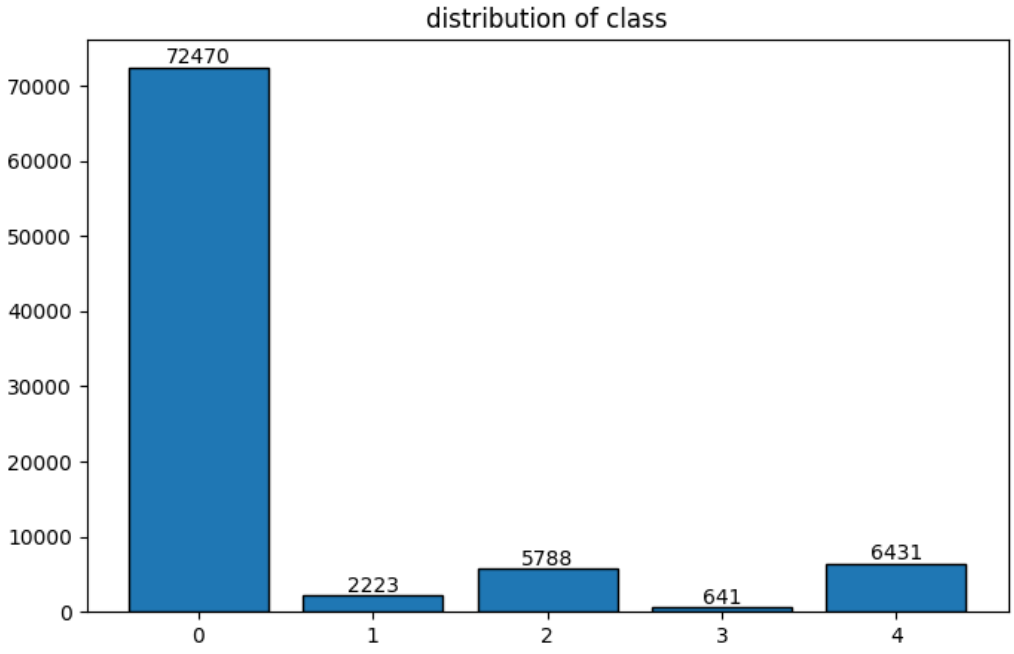

Figure 1: Frequency of each class in the dataset

As observed in the histogram in Figure 1, the training dataset from Kaggle is heavily skewed towards the Normal class, which is the most prevalent type of heartbeat. This imbalance may cause the model to overfit to the Normal class, reducing its ability to accurately classify other heartbeat types.
To mitigate overfitting, an up-sampling technique was applied to balance the dataset. By dividing the total number of files in the training dataset by 5, approximately 17,510 samples per class were obtained. Therefore, the dataset was balanced by ensuring that each class contained around 17,500 samples.

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.

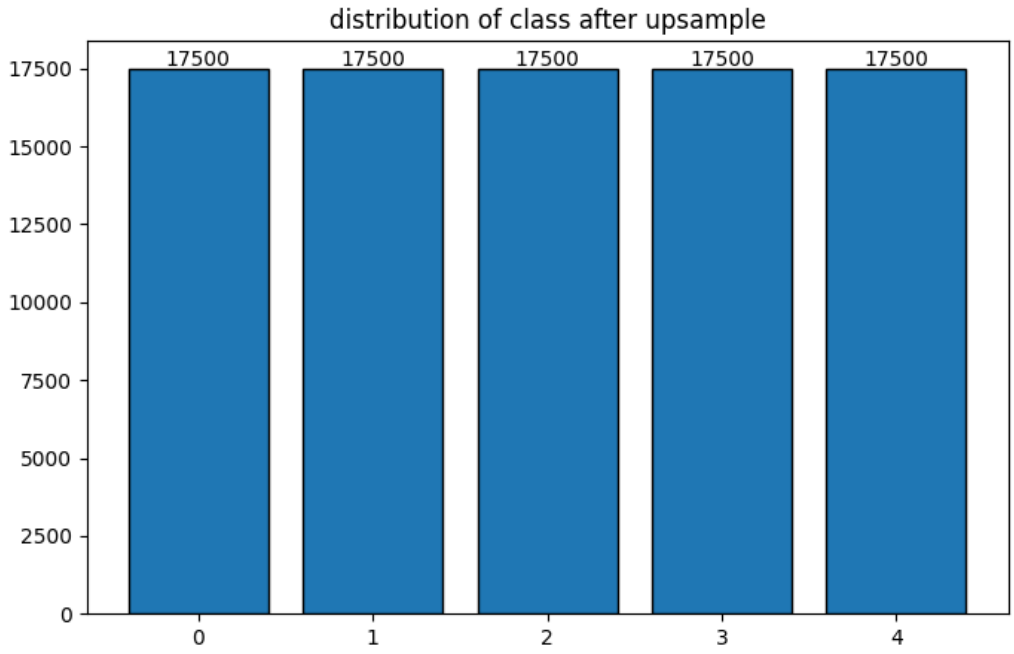

Figure 2: Up sample for each class

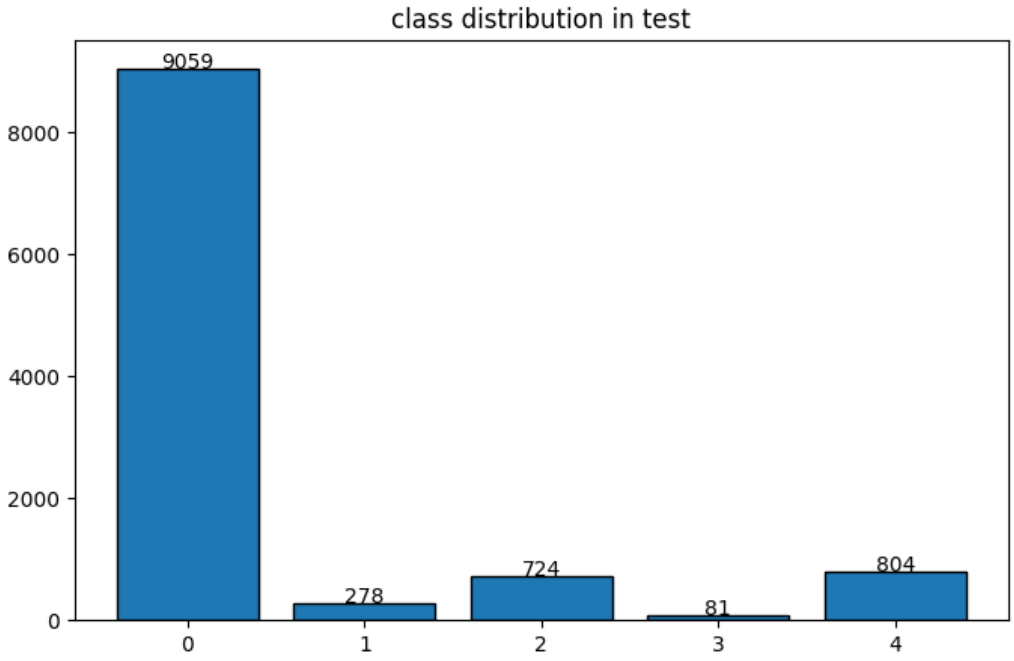

Figure 3: Distribution for each class in Test

Model

An RNN can handle sequential data, accepting the current input data, and previously received inputs. RNNs can memorize previous inputs due to their internal memory. RNN works on the principle of saving the output of a particular layer and feeding this back to the input in order to predict the output of the layer.
Before training the model, I encoded the labels and normalized the features within the range of -1 to 1 and used a dropout layer with a rate of 0.2 to prevent overfitting.

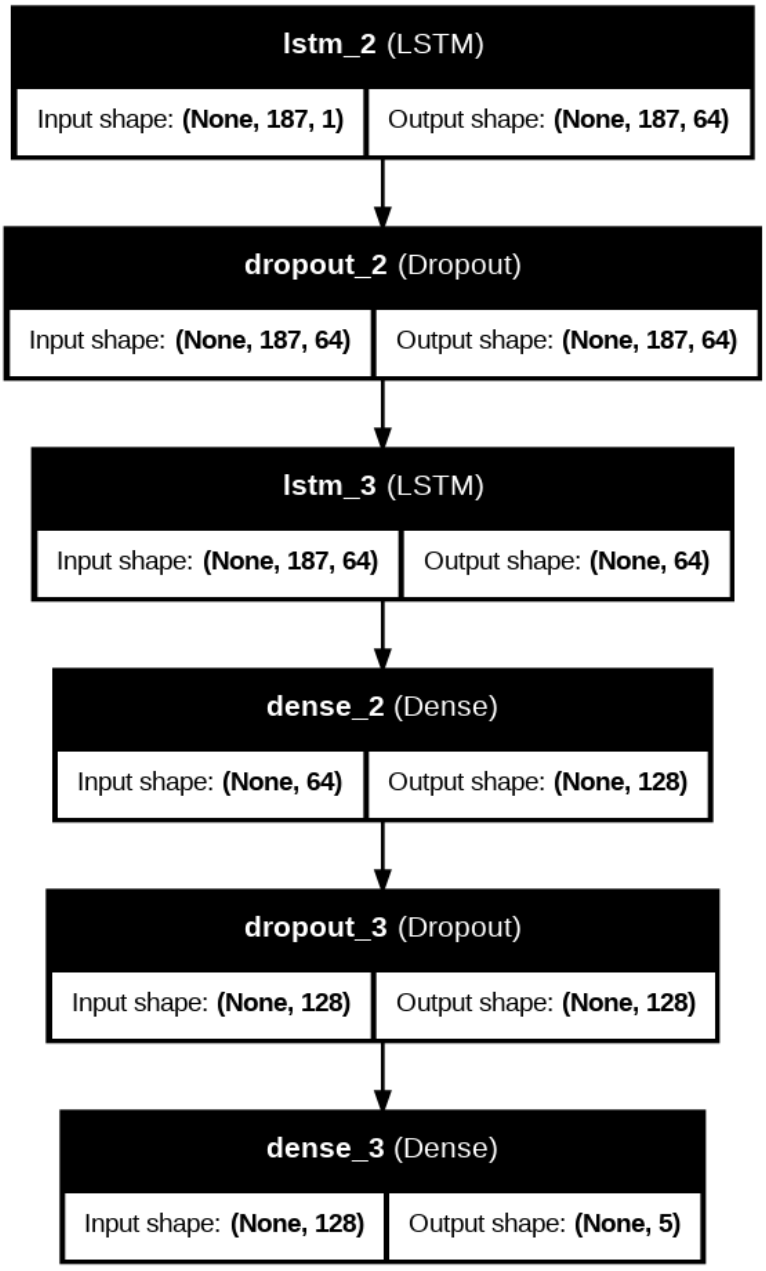

Figure 4: Architecture of model

Evaluation and Results

As you can see, the accuracy of the training and validation sets is quite high, but the loss continues to decrease, indicating that the model initially performs well, but gradually begins to overfit.
A confusion matrix is a tool used to evaluate the performance of a classification model by comparing the predicted labels with the true labels. It is an N x N matrix where each element represents the number of samples with the true label being one class but predicted as another. The diagonal of the matrix represents the correctly classified samples, showing how many samples were correctly predicted for each class. The higher the values on the diagonal,

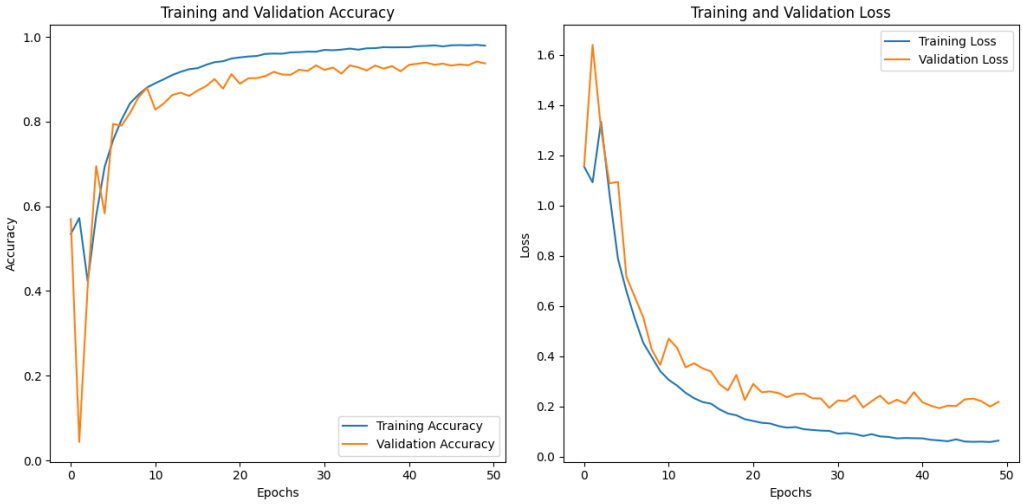

Figure 5: History of RNN

the better the model's performance.
As in the results, you can see that in the diagonal, the percentage of prediction to each class reach more than 90%

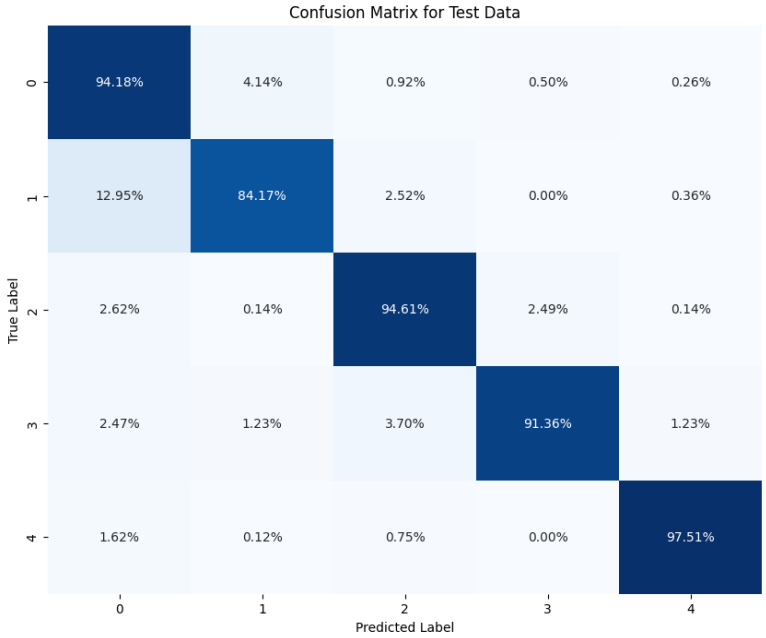

Figure 6: metrics