[[1]](#footnote-0)

AI Final Project Report

*Name*:占陈郅 *Student ID*: 12012505

***Abstract*—This document provides a Report of AI final project assignment. Three methods, Fully connected feedforward deep network, CNN and RNN are used in these projects to analyze a ECG data. During the project, pre-processing the data and tune possible hyper-parameters and evaluate the model through loss and score are important.**

# I. Objectives

In this experiment, there are two objectives.

* Use Fully connected feedforward deep network method to analyze the cancer data, use loss function and score to describe it.
* Use CNN and RNN method to analyze the cancer data, use score to describe it.

# II. Fully connected feedforward deep network

1. Dataset

There are 4 kinds of signals to classify.

Normal: 1

Atrial Fibrillation (AF): 2

Non-AF related abnormal heart rhythms: 3

Noisy recording: 4

However, the performance metrics consider only include F1-score for normal (), AF (), other rhythms ().

1. Loss function

Now we talk about training. Using DataLoader to load the “origin\_breast\_cancer\_data.csv” as train data, as well as test data. For the target ‘diagnosis’ is **one-hot label**, the loss function is **Binary Cross Entropy Loss**.

# 损失函数

loss\_fn = nn.BCELoss(size\_average=False)

1. Learning rate

The value of learning rate will affect the train loss and test loss and the evaluate score. Change the value of learning rate and see the difference.

图表, 直方图

描述已自动生成

Fig.1 Train loss when learning rate = 1e-3

Precision: 0.607

# III. CNN Model

## A. Introduction

This is an introductory CNN convolutional neural network with 2 layers of convolution, 2 layers of pooling and 3 fully connected layers, all 3 inputs are the same data, except that the parameters of the convolutional kernels set for each branch are different, for example, the convolutional kernel sizes of the three channels in the first layer are 4, 6 and 8 respectively. A magnitude-based low-frequency data, which does not necessarily contain a large range of magnitude changes, i.e., a smaller local area, can hardly constitute a more characteristic waveform. And why not use large convolution kernels such as 16, 32, etc.? Because our heartbeat itself is a short time data, it is too much to focus on the overall variation and ignore the local one, so we choose three medium-sized convolution kernels. In general, tuning is part of the problem, and part is based on the problem itself, from data level analysis to model level analysis.

## B. Result

Table.1 F1-score of convolution model

|  |  |
| --- | --- |
| class | F1-score |
| 1 | 0.94 |
| 2 | 0.92 |
| 3 | 0.92 |
| 4 | 0.99 |

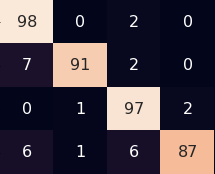


Fig.2 Normalized confusion matrix of CNN

# IV. RNN Model

## A. Introduction

Model consists of:

* Two stacked bidirectional GRU layers (input is masked to the variable dimension of the heartbeat vector)
* Two fully-connected layers connected to the last output-pair of the downstream (bidirectional) GRU layer
* A linear layer with softmax output
* Dropout regularization was used for the GRU layers

Since the model operates on segmented heartbeat samples, we can use a bidirectional RNN because the whole segment is available for processing at one time. It is also a more "fair" comparison with the CNN.

## B. Result

Table.1 F1-score of recurrent model

|  |  |
| --- | --- |
| class | F1-score |
| 1 | 0.94 |
| 2 | 0.92 |
| 3 | 0.92 |
| 4 | 0.99 |

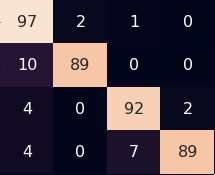


Fig.3 Normalized confusion matrix of CNN

# V. Discussion

**CNN versus RNN**

The CNN model has 53,957 parameters and the RNN model has 240,293. Moreover, the serial nature of the RNN causes it to be less parallelizable than the CNN. Given that the CNN is slightly more accurate than the RNN, it provides an all-around better solution.

1. [↑](#footnote-ref-0)