University of Cincinnati

Date: 8/12/2019

I, Xingliang Shu, hereby submit this original work as part of the requirements for the degree of Master of Science in Electrical Engineering.

It is entitled:

Electrocardiograph Signal Classification By Using Neural Network

Student's name: **Xingliang Shu**

This work and its defense approved by:

Committee chair: William Wee, Ph.D.

Committee member: Chia Han, Ph.D.

Committee member: Xuefu Zhou, Ph.D.

UNIVERSITY OF Cincinnati

36343

Electrocardiograph Signal Classification by Using Neural Network

A thesis submitted to the

Division of Research and Advanced Studies of

the University of Cincinnati

in partial fulfillment of requirements for the degree of

Master of Science

in department of Electrical Engineering and Computer Science of

the College of Engineering

2019

By

Xingliang Shu

Committee Chair: William G. Wee, Ph.D.

ABSTRACT

Electrocardiogram reflects the electrical signal of the heart, and it is an important tool for diagnosing patients' cardiac conditions. The goal of our research is to develop a neural network system which can classify nine categories of Electrocardiograms. The dataset is from the 2018 China Psychology Competition [1]. In Chapter 1, we will start to introduce Electrocardiogram classification and why we want to use neural network approach for this work. In Chapter 2, we will describe our dataset and the evaluation metric. In Chapter 3, our system design will be explained in very detail. In Chapter 4, we will discuss the results and analyze the misclassified data. There are many systems that have been developed by using the dataset we have, and they are evaluated and ranked during the competition. Therefore, in Chapter 5, we will show the performance of our system by comparing with the models from the competition in a reasonable way. Finally, we will conclude our work.

ACKNOWLEDGEMENTS

Firstly, I would like to thank my advisor, Dr. William. G. Wee. He was very helpful, patient and brilliant. He can always give me some great ideas when I met difficulties. He suggested this project for my thesis topic.

Secondly, I would like to thank my thesis committee members, Dr. Chia Han and Dr. Xuefu Zhou. They gave me many valuable comments and ideas during my thesis defense.

Thirdly, I would also like to thank my colleagues, Ahmed Mostayed and Junye Luo. They were very kind, patient and helpful. They were always there for me when I had technical problems.

Lastly, I would like to thank my family. Without them, I cannot finish my master's degree. They were always there to support and encourage me.

Table of Contents

ABSTRACT ii

ACKNOWLEDGEMENTS iv

List of Figures viii

List of Tables x

Chapter 1 Introduction to Electrocardiogram Classification and Neural Networks 1

1.1 Electrocardiogram 1

1.2 Problem Statement 1

1.3 Objective 1

1.4 Other Methodologies vs Neural Networks 1

1.5 Neural Networks with More Details 2

1.5.1 Nonlinear activation function 3

1.5.2 Convolutional layer 3

1.5.3 Long-short-term-memory layer 5

1.5.4 Fully connected layer 5

Chapter 2 Introduction to Dataset, Difficulties, and Evaluation Metric 6

2.1 Dataset Description 6

2.2 Morphologies of Nine Types of ECGs 7

2.2.1 The functions of the heart and the PQRST waves of ECG 7

2.2.2 Normal (N) 9
2.2.3 Atrial Fibrillation (AF) 9
2.2.4 First degree atrioventricular block (IAVB) 10
2.2.5 Left bundle branch block (LBBB) 11
2.2.6 Right bundle branch block (RBBB) 11
2.2.8 Premature ventricular contraction (PVC) 13
2.2.9 ST-segment elevation (STE) 13
2.2.10 ST-segment depression (STD) 14
2.3 Difficulties of Using Dataset 14
2.4 Evaluation Metric 17
Chapter 3 System Design 19
3.1 A Concept of Our Neural Network Design 19
3.2 Data Preprocessing 20
3.3 Data Augmentation 21
3.4 Mini-batch 21
3.5 Network Hyperparameters and Data Processes 22
3.5.1 Feature extractor 23
3.5.2 Heartbeat analyzer 24
3.5.3 Classifier 25

3.5.4 An instantiation of the neural network design 27

3.6 Model Training and Testing 28

Chapter 4 Results and Discussion 29

4.1 Ensemble Model 29

4.2 Results and Analysis of Misclassified Data 30

4.3 Limitations of Our System and Suggestions for Improvement 35

Chapter 5 Model Comparison and Conclusion 37

5.1 Performances of Different Models from the Competition 37

5.2 Processing Speeds and the Sizes of Different Models 39

5.3 Summary of Our Work 40

Bibliography 42

List of Figures

Figure 1. 1: an illustration of linearly inseparable data points 3

Figure 1. 2: an illustration of convolutional layer 4

Figure 2. 1: non-invasive 12-lead ECG 7

Figure 2. 2: ECG of one single normal heartbeat and an anatomy map of the heart 8

Figure 2. 3: an example of normal ECGs 9

Figure 2. 4: an example of atrial fibrillation ECGs 10

Figure 2. 5: an example of first-degree atrioventricular block heartbeats 10

Figure 2. 6: an example of left bundle branch block ECGs 11

Figure 2. 7: an example of right bundle branch block ECGs 12

Figure 2. 8: an example of premature atrial contraction ECGs 12

Figure 2. 9: an example of premature *ventricular* contraction ECGs 13

Figure 2. 10: an example of ST-segment elevation ECGs 13

Figure 2. 11: an example of ST-segment depression ECGs 14

Figure 2. 12: a histogram of recording lengths 15

Figure 2. 13: a bar chart of the data distribution 15

Figure 2. 14: noisy signal from a lead 16

Figure 2. 15: noisy signal from a lead 16

Figure 3. 1: data preprocessing steps 21

Figure 3. 2: two branches of convolutional layers. 24

Figure 3. 3: process of heartbeat analyzer 25

Figure 3. 4: an instantiation of the neural network design 27

Figure 4. 1: ensemble model 30

Figure 4. 2: preprocessed PAC data which falls into N 32

Figure 4. 3: preprocessed N data which is classified correctly 32

Figure 4. 4: original PAC data refers to Figure 4.2 32

Figure 4. 5: another PAC data which demonstates the problem of our data preprocessing..... 33

Figure 4. 6: preprocessed N data with lead V5 34

Figure 4. 7: preprocessed STE data with lead V5 34

Figure 4. 8: preprocessed STE data with lead V5 34

Figure 5. 1: model ranks vs F1 score (refer to Table 5.1) 39

List of Tables

Table 2. 1: dataset after the rearranging process from section 3.4 6

Table 4. 1: results of each model and ensemble model 30

Table 4. 2: the result of testing dataset (average F1 score = 0.812) 31

Table 5. 1: top 15 models from the 2018 China Psychology Competition 38

Table 5. 2: comparison table of processing speed and model size 40

Chapter 1 Introduction to Electrocardiogram Classification and Neural Networks

1.1 Electrocardiogram

An electrical signal generated by the muscles of the heart is called electrocardiogram (ECG). ECG is used to diagnose patients' cardiac conditions; the ECG waveform represents the conditions of different cardiac tissues. There are three most important waveforms which we will explain in Chapter 2: P-wave, QRS complex, and T-wave. By analyzing these waveforms, we can determine the type of the cardiac conditions.

1.2 Problem Statement

It is important to figure out the abnormal symptoms of the heart as early as possible because some of them can be life-threatening. However, analyzing ECG with long recording time and classifying the type of the cardiac conditions are difficult and time-consuming. Therefore, it is necessary to develop a system which can accelerate the ECG diagnosis.

1.3 Objective

In this paper, our objective is to develop a system by using a neural network approach which can classify nine types of ECGs fast and accurately. Our system consists of four neural networks and a data preprocessing technique.

1.4 Other Methodologies vs Neural Networks

Many techniques have been developed to classify ECGs before the neural networks [2 – 8]. However, they have inconsistent performances when they meet new datasets which cause

unstable accuracies and efficiencies. Most of the non-neural-network techniques require domain knowledges to handcraft features from the dataset, and these features are fixed and static which cannot help the classifiers to solve generic classification problems. Thus, they are not practical to be used in real-life applications. Recently, many ECG classification systems [9 – 14] are using neural network approaches which have achieved the state-of-the-art performances. Neural networks can capture abundant underlying characteristics from the dataset in a much more sophisticated way. Those extracted features from the data by using neural networks are much more powerful than the handcrafted features, so the classifiers can perform much better. There is proof [12] that the neural network outperforms cardiologists on classifying ECG in both recalls and precisions.

1.5 Neural Networks with More Details

Neural network is a powerful learning algorithm which requires backpropagation algorithm to minimize the objective or loss functions during the training phase, for simplicity, it is essentially a matrix operation. Neural network is currently a cutting-edge technique which can be applied to many different fields, such as computer vision [15 – 19], natural language processing [20, 21], medical purposes [22, 23], information encoder and decoder [24, 25], etc. The reason why neural network is so powerful is because of its nonlinear activation functions and various types of layers which have different capabilities to learn the datasets. In this section we will briefly talk about the nonlinear activation functions first, then slightly introduce three types of layers which we are going to use in this paper: convolutional layer, long-short-term-memory layer, and fully connected layer.

1.5.1 Nonlinear activation function

In classification tasks, we usually meet a situation where the data is not linearly separable. To solve this problem, we can use nonlinear activation function to map the original data into another dimension which can be linearly separated. Figure 1.1 shows a linearly inseparable dataset. By using gaussian function, we can easily separate the red data points and the blue data points. In neural network a layer is usually followed by a nonlinear activation function.

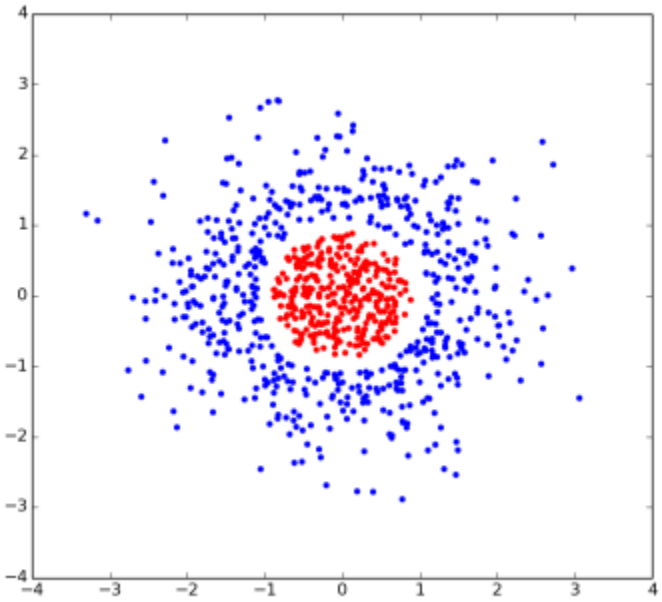

Figure 1. 1: an illustration of linearly inseparable data points, from https://stackoverflow.com/questions/1148513/difference-between-a-linear-problem-and-a-non-linear-problem-essence-of-dot-pro/1149330

1.5.2 Convolutional layer

Convolutional layer is broadly used in computer vision because of its outstanding ability of capturing features. By stacking convolutional layers, we can add levels of abstraction of the input data [26]. The most unique part of the convolutional layer is its weight sharing property. A

convolutional layer is comprised of many filters. The weight sharing property is that each filter will slide across the entire input data. In other words, each filter maps the input data into a feature map (Figure 1.2). The input has three dimensions: the height and the width are 32, and the depth is 3. A filter which has size of 5x5x3 is convolving over all spatial (height and width) locations with the stride size equals to 1. The stride size is the number of pixels skipped after convolving with a local spatial region. The output is an activation/feature map with size of 28x28x1. The equation of resizing the input by convolutional layer is provided below where N is the spatial size of the input data, P is the size of zero-padding on the board of the input, F is the spatial size of the filter, and S is the stride size.

$$The\ size\ of\ feature\ map = \frac{N + 2P - F}{S} + 1$$

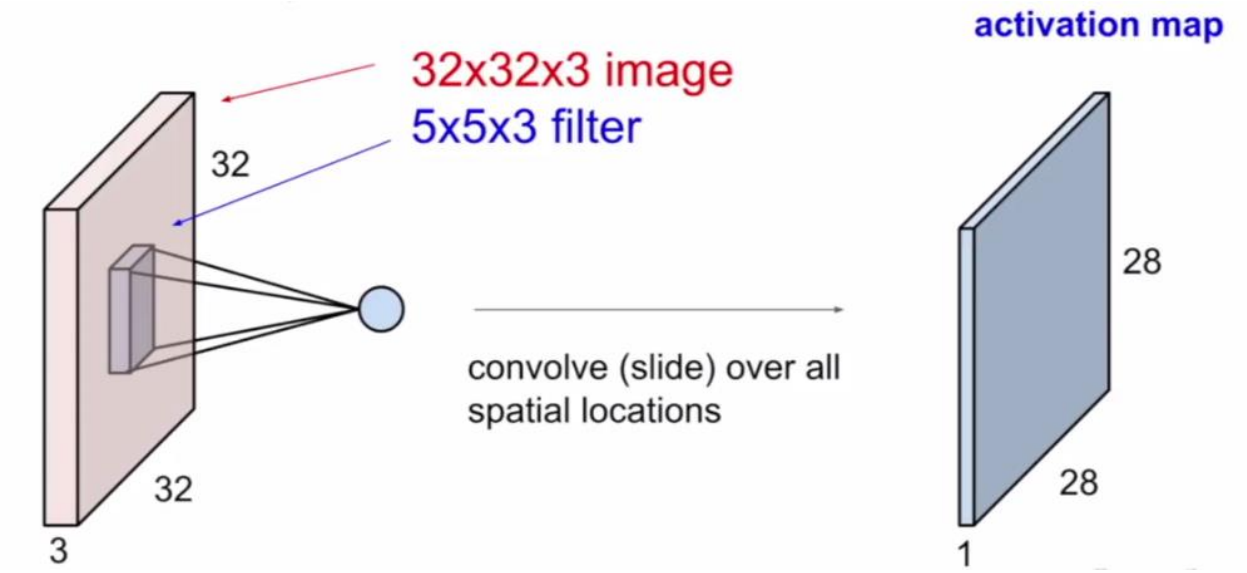

Figure 1. 2: an illustration of convolutional layer, from
http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture6.pdf

1.5.3 Long-short-term-memory layer

Long-short-term-memory (LSTM) layer [27] is used in sequence to sequence learning task. It can learn the information in time series which means it can remember, update, delete and predict information. There are four gates and a memory cell in a LSTM layer. The four gates are input gate, update gate, forget gate, and output gate. The input gate is used to input the information at the current time step. The update gate is to use the current information and the state (the knowledge the layer has learned or remembered) from the past time steps to update the information. The forget gate is to remove the useless information and keep the useful ones. All these operations are going into a mainstream, the memory cell. The memory cell holds the learned information over time and uses that information whenever it is ready to predict something at present time. The output gate is used to output something at the current time step.

1.5.4 Fully connected layer

A stack of fully connected layers (FCLs) with nonlinear activation functions is an extremely powerful algorithm which can approximate any function. However, it often suffers overfitting due to the large number of parameters. Fully connected means each neuron is connected with all the neurons in a preceding layer and a following layer. For example, three 1024-neuron FCLs can result 2,099,200 (1024x1024x2+1024x2) parameters. It is very necessary to reduce the dimensions of the input data before using FCLs so that the number of the parameters can be small. In addition, by using dropout [28], we can also reduce the overfitting degree.

Chapter 2 Introduction to Dataset, Difficulties, and Evaluation Metric

2.1 Dataset Description

There are two portions of datasets from the 2018 China Psychology Competition [1]: the training dataset and the hidden dataset. Both datasets were sampled at 500 Hz. The training dataset is provided for the participants to train their models, and the hidden dataset is for evaluating and ranking each trained model by judges. No participant is allowed to see and use the hidden dataset, so to train and evaluate our model the training dataset needs to be further split into 80% training and 20% testing. In the training dataset, 7% of the data which belongs to more than one category are removed before splitting. The summary of the dataset is in Table 2.1. There is also a rule to split the training dataset, and it will be explained in Chapter 3 at section 3.4. The training dataset contains nine categories: normal (N), atrial fibrillation (AF), first degree AV block (IAVB), left bundle branch block (LBBB), right bundle branch block (RBBB), premature atrial contraction (PAC), premature ventricular contraction (PVC), ST-segment elevation (STE), and ST-segment depression (STD). We will talk about the morphologies of each category in the next section.

Table 2. 1: dataset after the rearranging process from section 3.4

Nine Categories	**Total number of records**	**Training record (80%)**	**Testing record (20%)**
N	917	736	181
AF	975	780	195
IAVB	686	548	138
LBBB	179	145	34
RBBB	1533	1226	307
PAC	532	425	107
PVC	606	484	122
STE	185	146	39
STD	784	633	151
Entire dataset	6397	5123	1274

2.2 Morphologies of Nine Types of ECGs

12-lead ECG is non-invasive, each electrode is placed on the surface of the patient's skin to record the electrical activities of the heart. Figure 2.1 shows the location of each electrode. In this section, we will talk about the morphologies of ECGs in nine categories in depth. For simplicity we only use lead-1 to demonstrate all the morphologies. The lead-1 is placed on the left of the heart. In Figure 2.1, the right side of the picture is the left side of the patient.

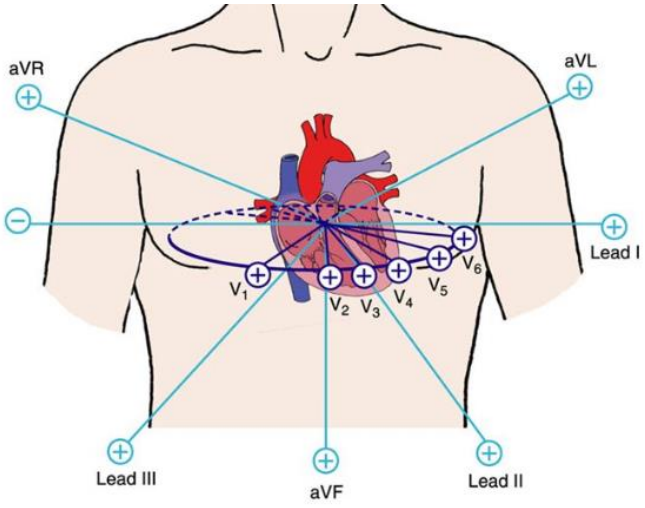

Figure 2. 1: non-invasive 12-lead ECG, from https://www.firstaidforfree.com/recording-a-12-lead-ecgekg/12-lead-ecg-planes/

2.2.1 The functions of the heart and the PQRST waves of ECG

We will start to explain the functions of a normal heart (Figure 2.2), then the morphology of each type ECG is discussed.

The starting point of a heartbeat is the sinoatrial (SA) node. The SA node is a natural peacemaker which initiates the heartbeat and determines the heart rate. The firing of the SA node spreads throughout both atriums and causes them to depolarize, the P-wave. The PR-segment is the

duration between the end of the P-wave and the beginning of the Q-wave, it represents the time the signal travels from the SA node to the atrioventricular (AV) node. The AV node is the electrical gateway to the ventricles. It receives the signals from the SA node and passes them onto the AV bundle (bundle of HIS). AV bundle divides into right and left bundle branches. The depolarization of the interventricular septum initiates from the left bundle to the right which causes a negative deflection of the Q-wave. Then the electrical signal moves to both left and right ventricles at the same time but the signal to the left ventricle is dominate, so the ECG displays a big positive deflection at the R-wave. S-wave represents the last phase of ventricular depolarization. The atrial repolarization is obscured by the ventricular depolarization, so we cannot see it in the QRS-complex. The ST-segment represents the plateau in the myocardial action potential. Finally, the T-wave reflects the repolarization of the ventricles. Based on the morphologies of these waves, intervals, and segments, we can analyze ECGs.

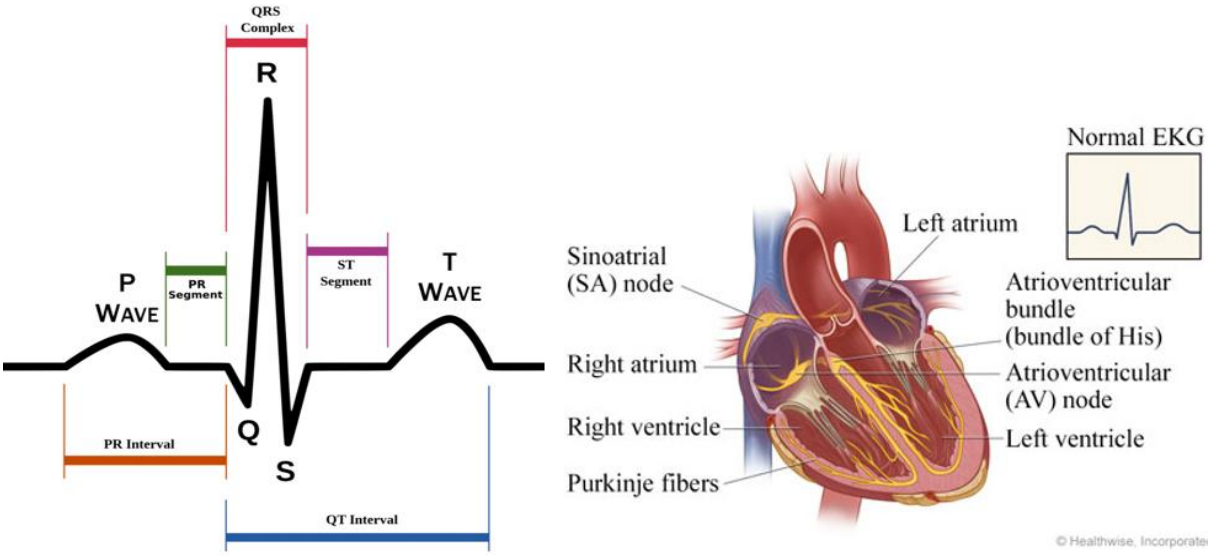

Figure 2. 2: ECG of one single normal heartbeat (left), and an anatomy map of the heart (right), from https://i1.wp.com/a-fib.com/wp-content/uploads/2012/08/Schematic-diagram-of-normal-sinus-rhythm-for-a-human-heart-as-seen-on-ECG-Wikipedia-free-to-use.png?ssl=1, and https://www.healthlinkbc.ca/health-topics/zm6205

2.2.2 Normal (N)

In normal ECGs, the RR-intervals are consistent which indicates that the patient has a regular heart rate. The morphologies of PQRST waves for each heartbeat are very similar to Figure 2.2. Figure 2.3 is an illustration of normal recordings from our dataset.

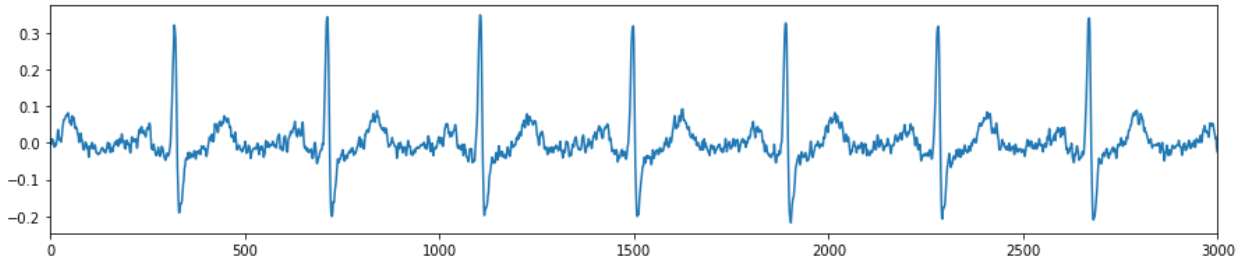

Figure 2. 3: an example of normal ECGs

2.2.3 Atrial Fibrillation (AF)

AF is the most common type of cardiac arrhythmias. The start of the heartbeat is not SA node, instead of an ectopic site. The electrical impulses are generated randomly from many ectopic sites which causes the fibrillation of atriums. In Figure 2.4, the oscillations before the QRS-complexes reflect the electrical impulses of many ectopic sites. Sometimes those ectopic sites activate the AV node, so the ventricles depolarize (the QRS-complex). Therefore, the P-wave is missing. AF can cause inefficient contractions of the atriums which in turn forms blood clots. The blood clots pass into blood stream and get stuck in small arteries of the brain can cause a stroke. Thus, it is life-threatening.

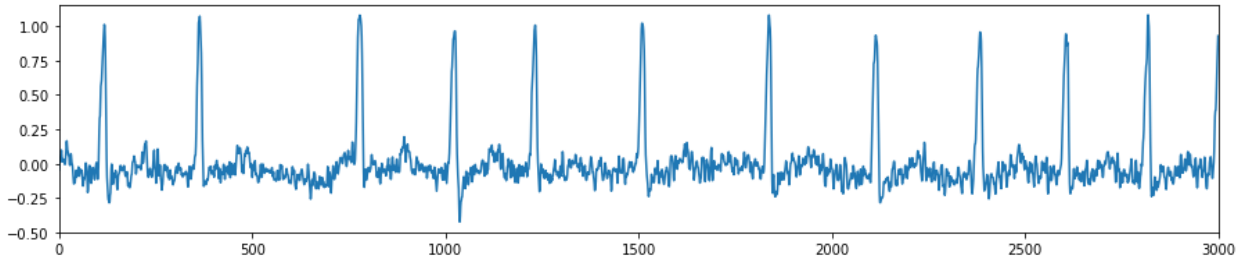

Figure 2. 4: an example of atrial fibrillation ECGs

2.2.4 First degree atrioventricular block (IAVB)

The symptom of IAVB is the delayed AV node conduction which results prolonged PR-interval (>0.2s) or prolonged PR-segment. The majority IAVBs are benign and require no treatment. Figure 2.5 shows an example of IAVB heartbeats, the PR-interval has 105 samples. By calculating the time in the green rectangle, the PR-interval is about 0.21 seconds, so the PR-interval is greater than 0.2 seconds.

$$\frac{500\ samples}{1\ second} = \frac{105\ samples}{?second} \quad \rightarrow \quad 0.21\ seconds$$

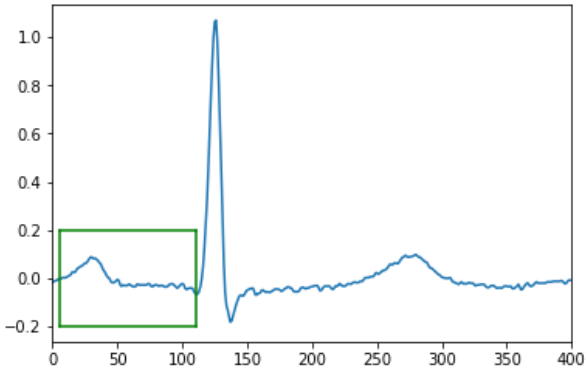

Figure 2. 5: an example of first-degree atrioventricular block heartbeats

2.2.5 Left bundle branch block (LBBB)

Normally the two ventricles depolarize simultaneously, then contract at the same time. In bundle branch blocks, the unaffected ventricle depolarizes first, then the other one depolarizes later. This results the delayed depolarization of the affected ventricle, so the morphologies of the QRS-complexes are abnormal.

For LBBB, we use Figure 2.6 to demonstrate the formation of the abnormal QRS-complexes. Abnormally, the depolarization of the septum initiates from the right to the left, therefore a positive deflection is showing in the beginning of the QRS complex. The right ventricle activates first which will cause a small downwards deflection. Then the electrical signal spreads to the large left ventricle which will cause another positive deflection, so a wide characteristic “bunny ears” QRS-complex. Sometimes, the small downwards deflection which is caused by the depolarization of the right ventricle is not visible, or the left ventricular depolarization has less amplitude than the septum depolarization.

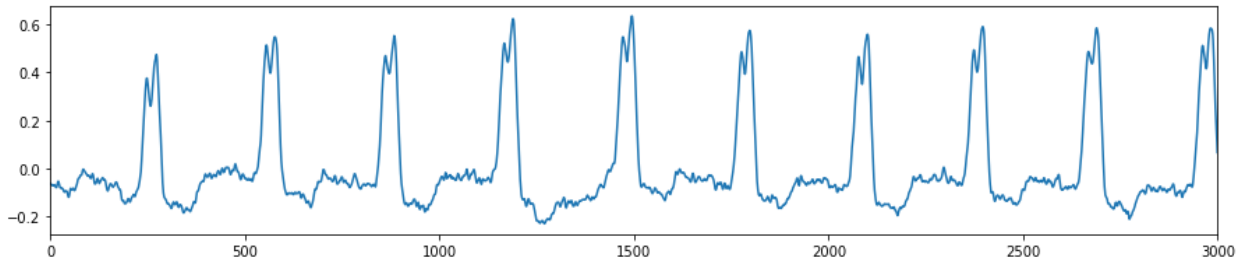

Figure 2. 6: an example of left bundle branch block ECGs

2.2.6 Right bundle branch block (RBBB)

In RBBB, the depolarization of the septum initiates from the left to the right as normal. The left ventricle depolarizes normally which results a normal R-wave. Then the electrical impulse

spreads to the right ventricle which results a large negative deflection and broader S-wave (Figure 2.7).

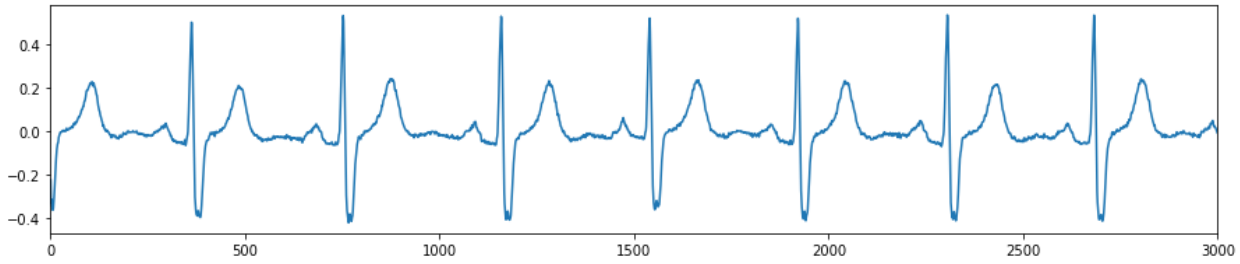

Figure 2. 7: an example of right bundle branch block ECGs

2.2.7 Premature atrial contraction (PAC)

PAC happens when there is an ectopic site firing in one of the atriums. There are several morphologies of PAC. Firstly, the P-wave can merge to the preceding T-wave due to the activation of the ectopic site. Secondly, the P-wave can be inverted, a negative deflection. Thirdly, the ectopic atrial activation can enter the SA node, depolarize it and reset its timing which changes the PP-interval. Figure 2.8 shows a merged PT-wave.

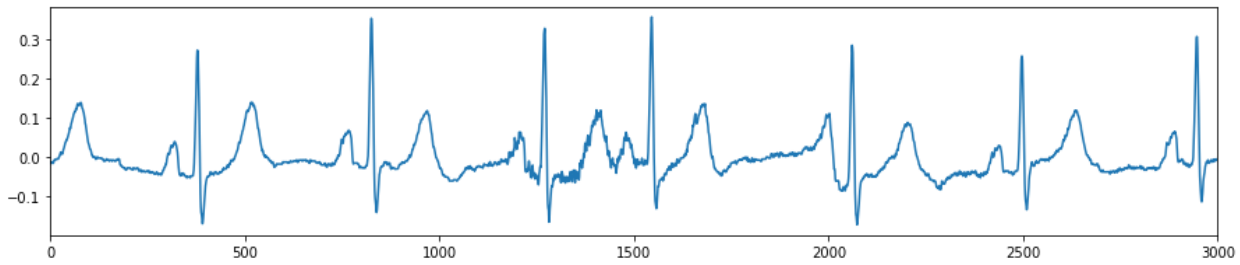

Figure 2. 8: an example of premature atrial contraction ECGs

2.2.8 Premature ventricular contraction (PVC)

PVC happens when there is an ectopic site firing prematurely in one of the ventricles. Because PVC does not affect the SA node, so the PP-interval remains unchanged. The morphology of QRS-complex is depending on the location of the ectopic site. The QRS complex can be broader, higher, or deeper (negative deflection in Figure 2.9).

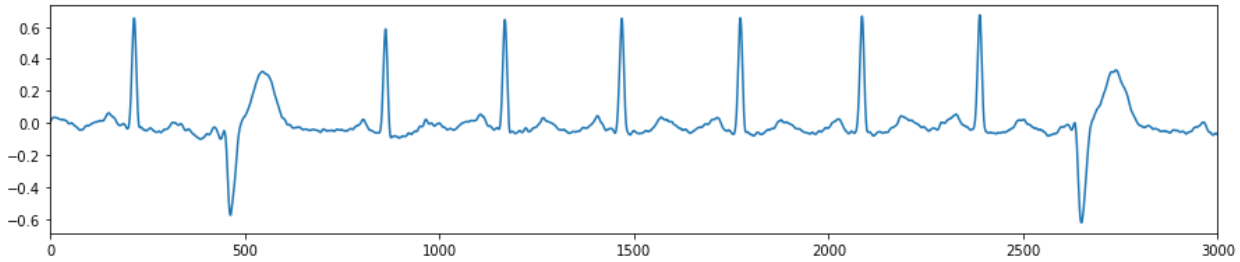

Figure 2. 9: an example of premature *ventricular* contraction ECGs

2.2.9 ST-segment elevation (STE)

The morphology of STE is that the ST-segment is elevated by referring to the baseline. Figure 2.10 shows each ST-segment has an upwards slope and larger amplitude compared to the PR-segment.

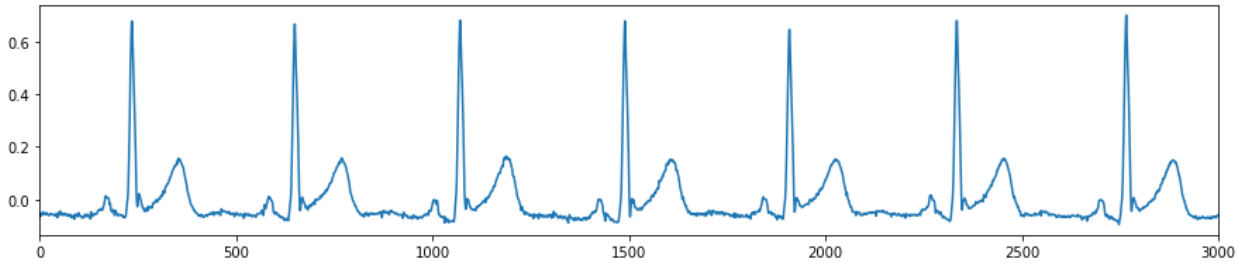

Figure 2. 10: an example of ST-segment elevation ECGs

2.2.10 ST-segment depression (STD)

The morphology of STD is that the ST-segment is depressed by referring to the baseline. Figure 2.11 shows each ST-segment has negative amplitudes, and they are below the baseline.

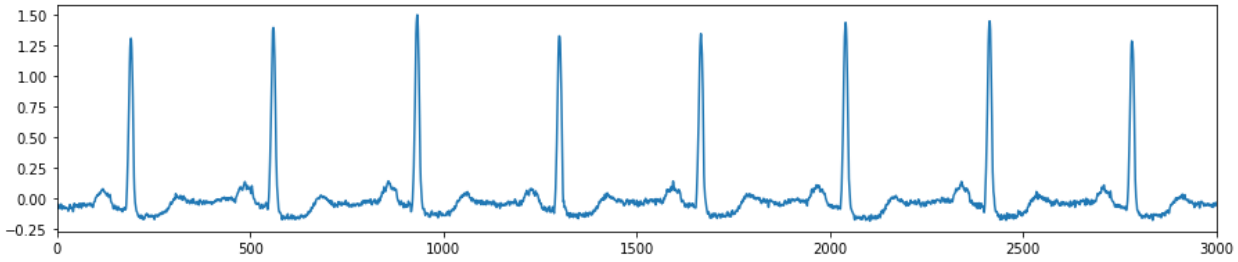

Figure 2. 11: an example of ST-segment depression ECGs

By studying the morphologies from different ECGs, we find it is helpful to do classification by analyzing the heartbeats. Therefore, in Chapter 3, we use R-peak detection algorithm in our data preprocessing to extract all the heartbeats from each recording.

2.3 Difficulties of Using Dataset

By comparing the datasets in [9 – 14] with ours, the dataset we have is harder to manipulate. Most of the datasets in [9 – 14] provides the labels of the heartbeats or the labels of the corresponding segments in ECGs. The format of the dataset (the way the labels and input data are provided) also highly affects the structure of the neural network design in many cases. Here we will talk about the difficulties of using our dataset. Firstly, it is not feasible for batch training because of the arbitrary lengths of the recordings. Figure 2.12 shows the histogram of the recording lengths. There are many recordings located around length 5000, but there are also many recordings distributed in long lengths. Secondly, the dataset is highly imbalanced. To reduce training bias, in each training iteration, a same amount recording need to be selected

from each category. Figure 2.13 shows the bar chart of our data distribution. Thirdly, each recording gives one label. For instance, in a recording, the type of each heartbeat is unknown, but we know there are certain types of abnormal heartbeats that exist in the recording. Therefore, inputting the whole sequence of a recording into the neural network system is crucial for classification. Finally, many recordings have high noise level, and some leads of a recording are useless (Figure 2.14, 2.15).

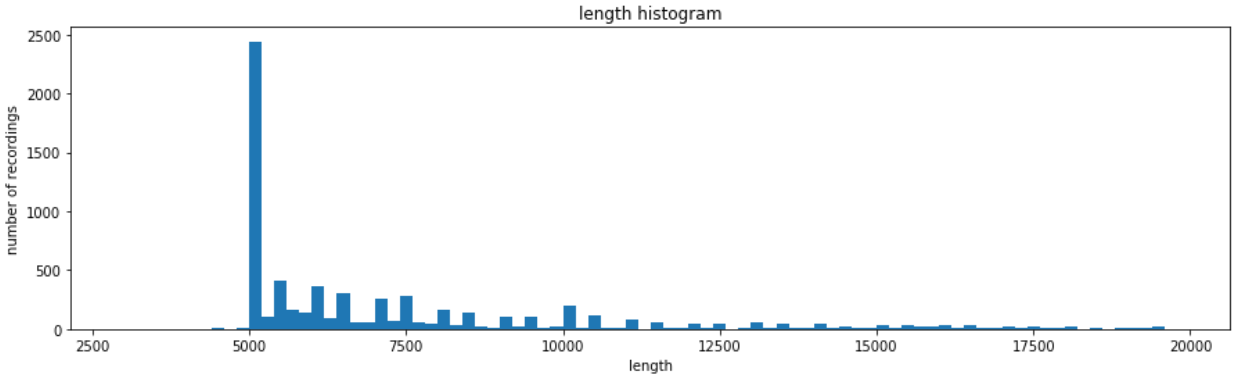

Figure 2. 12: a histogram of recording lengths

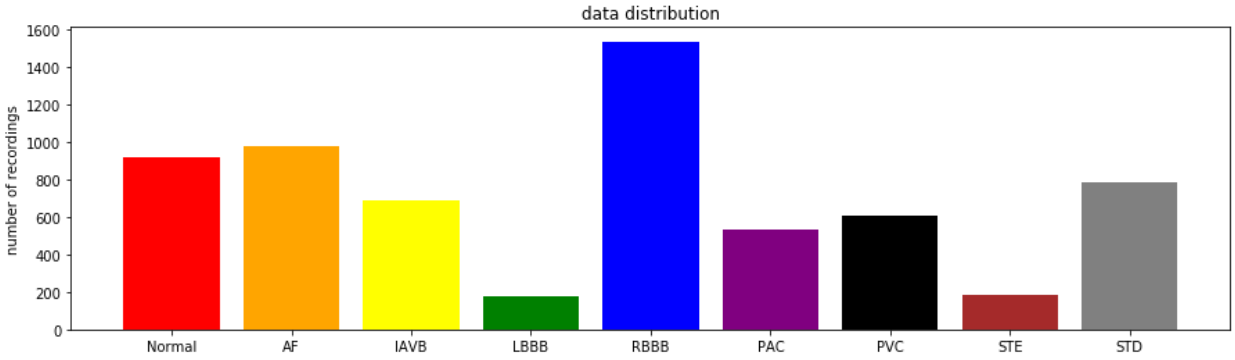

Figure 2. 13: a bar chart of the data distribution

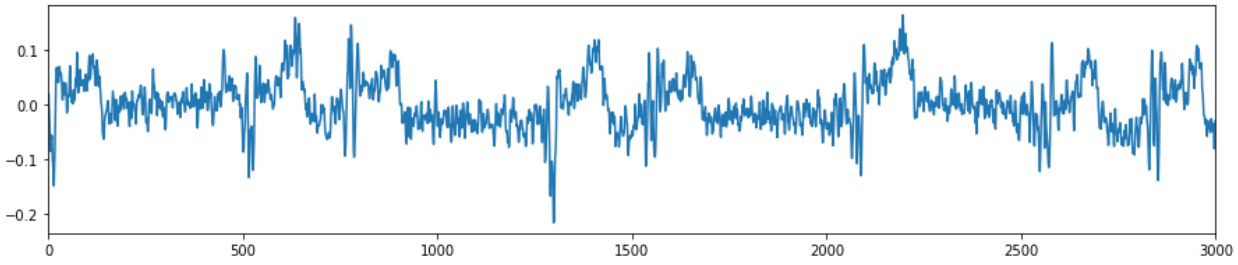

Figure 2. 14: noisy signal from a lead

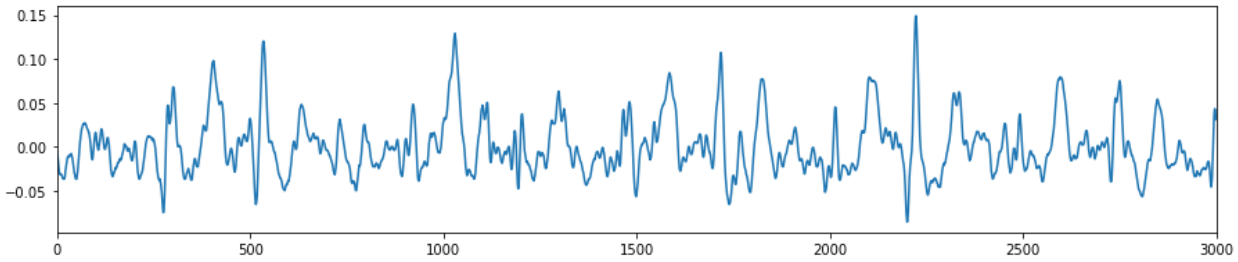

Figure 2. 15: noisy signal from a lead

2.4 Evaluation Metric

In this paper, we are using F1 scores as the evaluation metric. To calculate the F1 score for each category, we use the equation below:

$$F_{1,i} = \frac{2 \times N_{ii}}{\sum_{j=1}^{9}(N_{ij} + N_{ji})}$$

Where N_{ij} is the number of testing examples of category i predicted as category j. The numbers from 1 ~ 9 are corresponding to N, AF, I-AVB, LBBB, RBBB, PAC, PVC, STE, and STD.

The average F1 score is expressed in the equation below:

$$F_1 = \frac{\sum_{i=1}^{9} F_{1,i}}{9}$$

F_1 is the comprehensive score which decides the rank of the model during the competition.

The types of F1 scores are expressed in the equations below:

$$F_{af} = \frac{2N_{22}}{\sum_{j=1}^{9}(N_{2j} + N_{j2})}$$

$$F_{block} = \frac{2(N_{33} + N_{44} + N_{55})}{\sum_{i=3}^{5}\sum_{j=1}^{9}(N_{ij} + N_{ji})}$$

$$F_{pc} = \frac{2(N_{66} + N_{77})}{\sum_{i=6}^{7}\sum_{j=1}^{9}(N_{ij} + N_{ji})}$$

$$F_{st} = \frac{2(N_{88} + N_{99})}{\sum_{i=8}^{9}\sum_{j=1}^{9}(N_{ij} + N_{ji})}$$

F_{af} measures AF. F_{block} measures LBBB, RBBB and IAVB. F_{pc} measures PAC and PVC. F_{st} measures STD and STE.

Chapter 3 System Design

In this chapter, we will talk about how to design a neural network based on the dataset we have. Then according to the network design, a compatible data preprocessing technique is developed. To train the network much more efficiently, we rearrange our dataset based on the number of the R-peaks. We take the advantages of the data preprocessing to do data augmentation. Finally, we show how to train our model.

3.1 A Concept of Our Neural Network Design

In our dataset, each recording belongs to one category. According to the given label we know there are certain types of heartbeats in the recording, but we cannot extract only the abnormal heartbeats because the dataset does not provide the labels of the heartbeats. In addition, there are many abnormal recordings which contain normal heartbeats. Thus, to accurately classify recordings, the neural network needs to analyze all the heartbeats from the recording and summarize it to predict a label. Furthermore, we need to extract heartbeats to achieve batch training. For example, firstly we need to extract all the heartbeats from each recording and then randomly duplicate some heartbeats to a certain amount. Finally, the preprocessed recordings are equal sizes which can be wrapped into a single tensor for neural network training.

Because we are extracting heartbeats from each recording, and we want the neural network to detect the abnormal ones, the neural network needs to be designed in order to have the capability to analyze each single heartbeat from a recording, here we call this mechanism heartbeat analyzer. The heartbeat analyzer can be made by a stack of the LSTM layers. The heartbeat analyzer can map all the heartbeats from a recording into a low-dimensional fixed-size

aggregative feature vector. The fixed-size feature vector is a summary of the whole sequence of the recording, and it contains high activation values in respect to the abnormal heartbeats. Then the classifier can use this fixed-size feature vector for classification where the classifier is a stack of the FCLs. Finally, adding a feature extractor to extract characteristics of each single heartbeat can help the neural network learn the data better. The feature extractor can be made by a stack of the convolutional layers.

Now we have a general concept of our neural network design. The designed neural network has three parts: a feature extractor, a heartbeat analyzer and a classifier. In section 3.5, we talk about how to choose hyperparameters of the neural network. The reason why we want to talk about the data preprocessing before section 3.5 is because it influences the hyperparameter selection.

3.2 Data Preprocessing

The key in this data preprocessing is to extract heartbeats. It is easier to manipulate the dataset once heartbeats of each recording are extracted. Figure 3.1 shows four steps of our data preprocessing. The original ECG recording is processed by using R-peak detection algorithm. The R-peak detection algorithm is beyond our scope in this paper so we will not talk about it in detail. Once the locations of the R-peaks are found, each heartbeat can be segmented or extracted by picking samples before and after the corresponding R-peak. In this paper, we pick 119 samples before the r-peak and 180 samples after the r-peak, so there are 300 samples for each heartbeat. Because there are 12 leads for each ECG recording, the shape of each extracted heartbeat is 300 by 12. Step 3, the duplication process, allows the neural network to train in batch manner, but firstly the dataset needs to be split into training and testing based on the number of R-peaks. We

will explain how to split the dataset in section 3.4. Finally, we concatenate all the extracted heartbeats from each recording. For each preprocessed data, it has shape 300n by 12 where n is the number of the heartbeats or R-peaks.

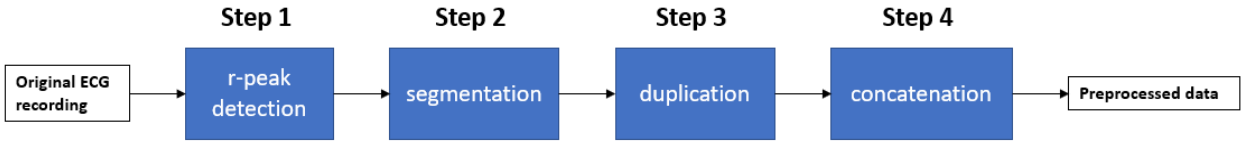

Figure 3. 1: data preprocessing steps

3.3 Data Augmentation

After heartbeat extraction, it is much easier to manipulate the data. We can take advantages from our data preprocessing to do data augmentation. Because the detected R-peak locations are always in order, the preprocessed data for training is usually the same. For data augmentation, we can randomly change the orders of the R-peaks, this can help the neural network enhance learning the shapes of the heartbeats, instead of memorizing the locations of the abnormal heartbeats. The duplication step in the data preprocessing also contributes to our data augmentation. This is because it randomly duplicates the existing heartbeats with respect to the recording. Then the duplicated heartbeats also change their orders randomly along with all the heartbeats before the concatenation.

3.4 Mini-batch

For batch training, we need to split the dataset into training and testing with respect to the number of R-peaks. In each category, we want to have 80% data for training and 20% data for testing. Since most recordings have less than 20 R-peaks, we decide to use 20 as the boundary to

split the dataset. If a recording has less than or equal to 20 R-peaks, the recording is moved to the training dataset, otherwise it is moved to the testing dataset.

The above process cannot promise 80% training and 20% testing. After splitting the dataset, one category can have 87% training data and 13% testing data, or 70% training data and 30% testing data. The first case is easy to deal with where the 7% training data can just move to the 13% testing data, so it causes no problem during the data preprocessing. However, the second case can cause problems. If the 10% testing data moves to the 70% training data, we no longer know the number of the heartbeats we want to duplicate. Therefore, we need to find the number of R-peaks which can be used as a reference to select some recordings from the testing dataset and move them to the training dataset. We go through the testing dataset in each category, and we find out 40 R-peaks and 60 R-peaks are the best options. Therefore, we further move the recordings who have less than or equal to 40 or 60 R-peaks from the testing dataset into the training dataset. Finally, for each category, the dataset split into 80% training and 20% testing.

To fetch and train those recordings more conveniently, we save them into different folders. The training recordings which have less than 20 R-peaks are saved in a folder. The training recordings which have more than 20 R-peaks and less than 40 R-peaks are saved in a different folder, and the training recordings which have more than 40 R-peaks and less than 60 R-peaks are saved in another folder. The training process will be explained in section 3.6.

3.5 Network Hyperparameters and Data Processes

In section 3.1, we give a general concept of our neural network design. The neural network has three parts: a feature extractor, a heartbeat analyzer and a classifier. In this section, we will talk

about how to choose hyperparameters for our neural network, especially for the feature extractor.

3.5.1 Feature extractor

The feature extractor is a stack of the convolutional layers connected serially. The preprocessed recording is a concatenation of the heartbeats, so between each two adjacent heartbeats the signal is not naturally connected. In the other words, we should avoid convolutional operation between each two adjacent heartbeats. To achieve this, the kernel size should be equal to the stride size so that during convolution there is no overlapping. In addition, there is a rule to pick the kernel size in each convolutional layer. We want the feature extractor to extract the characteristics of each heartbeat without involving other heartbeats so that the heartbeat analyzer can process features of each heartbeat at a time step. Because the sample number of each heartbeat is 300, the kernel size must be a divisor of 300. Before choosing the kernel size of each convolutional layer the number of the convolutional layers needs to be decided first, this is because the number of the convolutional layers is also the number of the divisors of 300. For example, if there are two convolutional layers, then the kernel sizes can be 10 and 30, or 12 and 25, etc. If there are three convolutional layers, then the kernel sizes can be 5 and 6 and 10, or 5 and 5 and 12, etc. The product of each set of numbers is 300.

The way we choose the hyperparameters in the feature extractor cannot capture all the underlying characteristics of the heartbeats, this is because there is no overlapping during convolution. The various abnormalities of the heartbeats require different scales of the filters to detect. To capture the features from the heartbeats better, we need to add convolutional layers with different kernel sizes parallelly along the other branches of the convolutional layers. Figure

3.2 shows two branches of convolutional layers with different kernel sizes. The output size from each branch is the same, so those outputs can be concatenated into one tensor.

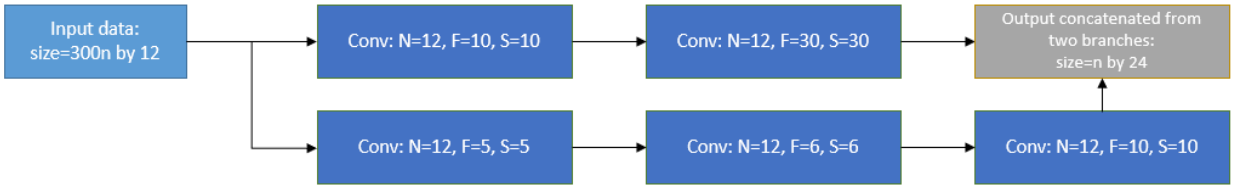

Figure 3. 2: two branches of convolutional layers assigned with different kernel sizes where n is the number of heartbeats, N is the number of filters, F is the kernel size, and S is the stride size. The outputs from botch branches can be concatenated into one tensor. In this case, the concatenated output has size n by 24 where n is the number of the heartbeats.

3.5.2 Heartbeat analyzer

Heartbeat analyzer is a stack of the LSTM layers. The output from the feature extractor has two dimensions: the first dimension represents the number of heartbeats, and the second dimension represents a set of features in different scales with respect to each heartbeat. Therefore, the first dimension decides the number of time steps in the process of the LSTM layers. In each time step the heartbeat analyzer takes a set of features of one heartbeat as an input. It produces a sequence of the outputs from each LSTM layer until it reaches the last one. The last LSTM layer outputs a fixed size vector. The fixed size vector is a set of aggregative features which summarizes the entire recording. Figure 3.3 is an indication of how the heartbeat analyzer processes the output of the feature extractor. In Figure 3.3, there are 20 heartbeats, so 20 time-steps for each LSTM layer. Each heartbeat has a set of features where the size of features is D. The third LSTM layer is the last layer of the heartbeat analyzer, so the output size is fixed. Furthermore, we

replace the basic LSTM layer with bidirectional LSTM layer [29], which enhances the sequential learning.

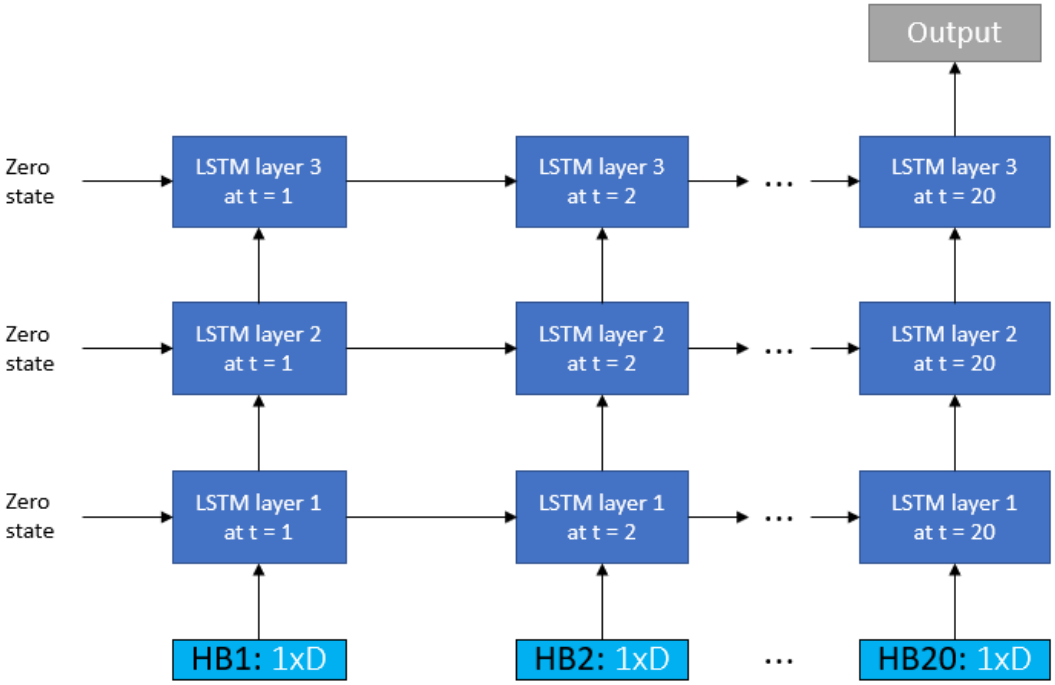

Figure 3. 3: process of heartbeat analyzer where “HB” means “heartbeat”.

3.5.3 Classifier

We can use the FCLs as our classifier because the input size is fixed. Fully connected layers are also called multi-layer perceptron. This layer does not share weights. Each neuron in a layer connects every single neuron in the preceding and following layers which can cause a tremendous number of parameters if the widths of the layers are large. It is impractical to use a large size classifier in our small dataset. However, the feature extractor and heartbeat analyzer vastly reduce the dimensions of the input data. Firstly, the feature extractor extracts the useful information of the heartbeats and abandons the redundant ones where the useful information

is sets of low-dimensional features which represents the original heartbeats. Then the heartbeat analyzer analyzes each set of features and further reduces the dimensions of the data. Finally, the output of the heartbeat analyzer is a compact feature which represents all heartbeats from a recording. By using the compact feature for classification, the size of the classifier can be small.

3.5.4 An instantiation of the neural network design

Figure 3.4 shows an instantiation of our neural network. The feature extractor has five branches of the convolutional layers, the number of parameters is 23,184. The heartbeat analyzer has 74,240 parameters, and the classifier has 3,689 parameters. Except the last FCL, each layer in Figure 3.4 is followed by a batch-normalization layer [30] and ReLu [31] activation function. The last layer is not followed by batch-normalization layer, but Softmax [32] activation function. The loss function is categorical cross-entropy loss function [33].

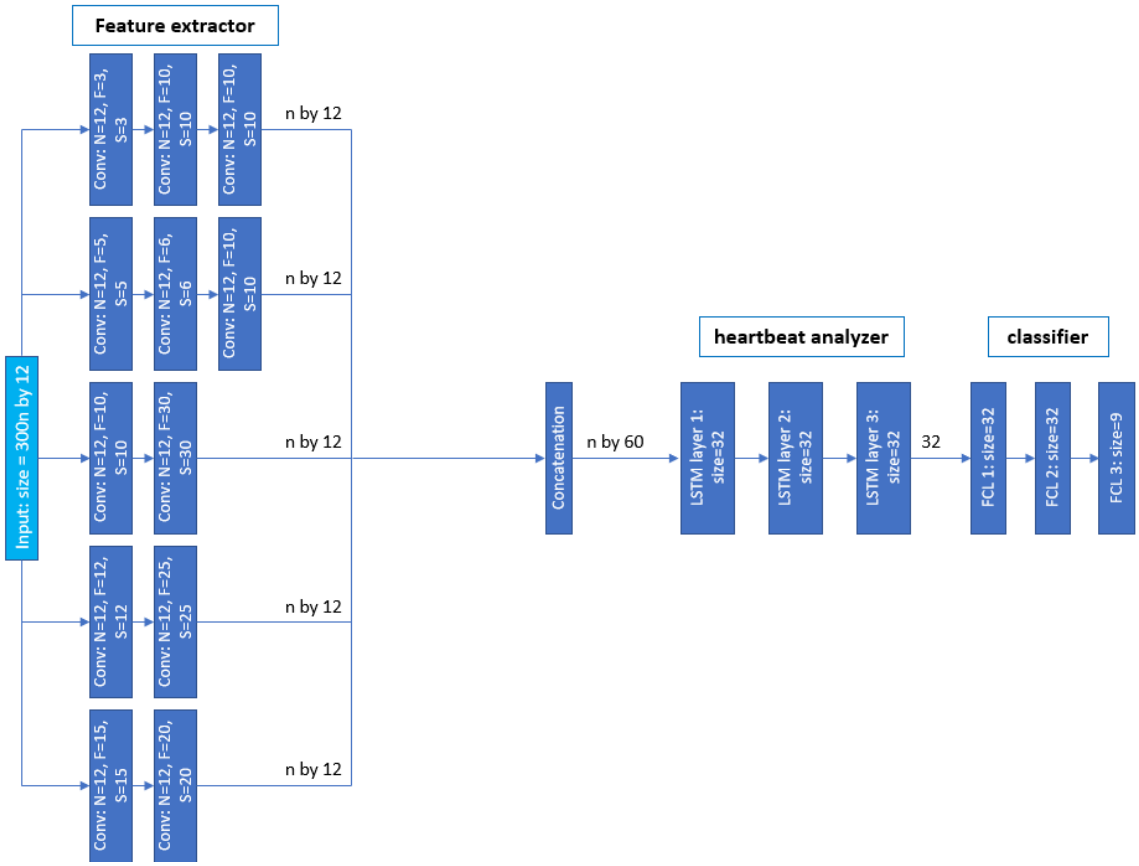

Figure 3. 4: an instantiation of the neural network design

3.6 Model Training and Testing

Now it is the time to talk about how we train the neural network. We have training recordings in three different folders: the 20 R-peaks, the 40 R-peaks, and the 60 R-peaks. So, there are three training phases in each training iteration. To avoid neural network bias to one category than another, we want to evenly train each category. The detail of the training process is shown below:

For each iteration:

In 20-heartbeats recordings (1st phase):

Randomly pick 128 recordings in each category

Data preprocessing and augmenting

*Train 128*9 recordings all together*

In 40-heartbeats recordings (2nd phase):

Randomly select 32 recordings in each category

Data preprocessing and augmenting

*Train 32*9 recordings all together*

In 60-heartbeats recordings (3rd phase):

Randomly select 16 recordings in each category

Data preprocessing and augmenting

*Train 16*9 recordings all together*

Reducing learning rate by multiplying by 0.1 if the accuracy saturates

For testing dataset, we do not use duplication step and data augmentation.

Chapter 4 Results and Discussion

In this chapter, we first show the results of our model. Then based on the results, we discuss some misclassified data. With the insight of knowing the reason for misclassification, we talk about the shortcomings of our system design and give suggestions to improve the design in the future.

4.1 Ensemble Model

Ensemble model is a combination of multiple learning algorithms which has better predictive performance than a single learning algorithm (see Table 4.1). We independently train four neural networks with the same design concepts from Chapter 3. Each neural network has a different number of parameters (see Table 4.1). For example, a feature extractor of the neural network can have more or less branches of convolutional layers. The size of bidirectional-LSTM layer can be 32, or 50. The point is to reduce the objective loss function of the neural network to different local minima so that each neural network can learn the dataset in different directions. The training dataset and testing dataset for all the models are the same, we do not use cross validation here. The output of the last fully connected layer is a vector of nine probabilities, called scores. We combine four sets of scores by piece-wise addition which results in a vector with nine values. The index of the largest value decides the category of the input data. Figure 4.1 shows our ensemble model. 9 in Figure 4.1 is the size of the output from each neural network. The size of our ensemble model is 703,153 parameters which is relatively small by comparing with other models in the competition.

Table 4. 1: results of each model and ensemble model

	F_1	F_{af}	F_{block}	F_{pc}	F_{st}	# of params
Neural network 1	**0.779**	0.895	0.874	0.686	0.766	258,485
Neural network 2	**0.767**	0.882	0.876	0.686	0.706	96,785
Neural network 3	**0.793**	0.894	0.892	0.682	0.732	121,873
Neural network 4	**0.756**	0.901	0.885	0.639	0.689	226,010
Ensemble model	**0.812**	0.911	0.899	0.723	0.765	703,153

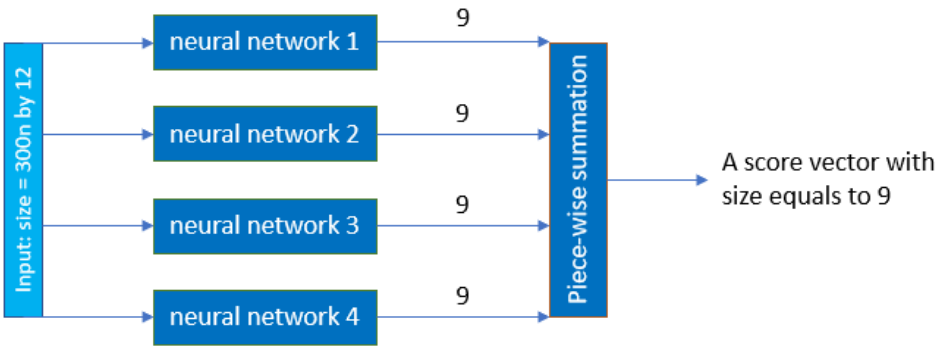

Figure 4. 1: ensemble model

4.2 Results and Analysis of Misclassified Data

In Table 4.1, we can see all models suffer poor performances on PC and ST type ECGs. This is a very common phenomenon once we look at the results of the models in Table 4.3, but firstly we need to analyze why PC and ST are difficult for our system to learn. The testing result of our ensemble model is in Table 4.2. We use the average F1 score to measure the performance of the model, and it is 0.812. According to the F1 scores in Table 4.2, we can see our model does not do a very good job on classifying PAC and STE data. In this section, we will visualize and analyze the misclassified data, particularly on PAC, and STE.

Table 4. 2: the result of testing dataset (average F1 score = 0.812)

	N	AF	IAVB	LBBB	RBBB	PAC	PVC	STE	STD	F1score
N	**159**	1	4	0	6	3	2	1	5	0.805
AF	1	**185**	1	0	2	1	5	0	0	0.911
IAVB	4	3	**120**	0	4	2	1	0	4	0.876
LBBB	0	0	1	**30**	0	0	3	0	0	0.896
RBBB	5	7	4	0	**275**	4	3	2	7	0.909
PAC	18	5	0	1	9	**58**	6	3	7	0.614
PVC	6	4	2	1	0	5	**101**	0	3	0.805
STE	8	1	2	0	1	1	0	**25**	1	0.714
STD	13	5	2	1	1	8	8	0	**113**	0.777

Table 4.2 shows the worst F1 score is in PAC where most data fall into N. In section 2.2, we mentioned the morphologies of PAC includes the changes of PP-intervals. Our data preprocessing is to extract heartbeats which removes the correlations between heartbeat to heartbeat. Thus, our system cannot learn the time between P-wave to P-wave. To understand why most PAC data fall into N, we visualize the PAC data which falls into N (Figure 4.2) and the N data which is classified correctly (Figure 4.3 Here, we are using lead I to do analyzation). By comparing Figure 4.2 with Figure 4.3, we can see they look alike. If we take a look at the original signal of this PAC data, we will see the essential problem is from data preprocessing. Figure 4.4 is the PAC data from Figure 4.2 without data preprocessing. In Figure 4.4, the green rectangle is the abnormal part where the P-wave is merging to the preceding T-wave. At segmentation of data preprocessing, we are using the fixed threshold values to extract heartbeats, so the abnormality disappears after the segmentation step. Figure 4.5 demonstrates this problem. In Figure 4.5, we can see the P-wave comes earlier in the rectangle area, but the segmented part looks just like N data. Thus, this explains why most PAC data are classified as N.

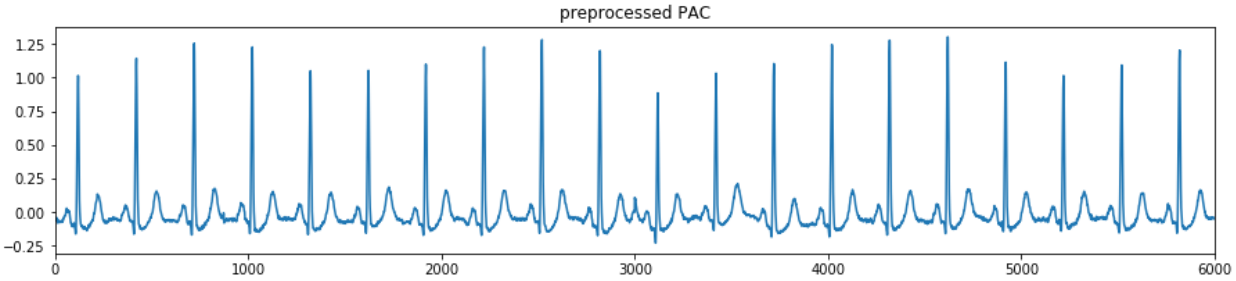

Figure 4. 2: preprocessed PAC data which falls into N

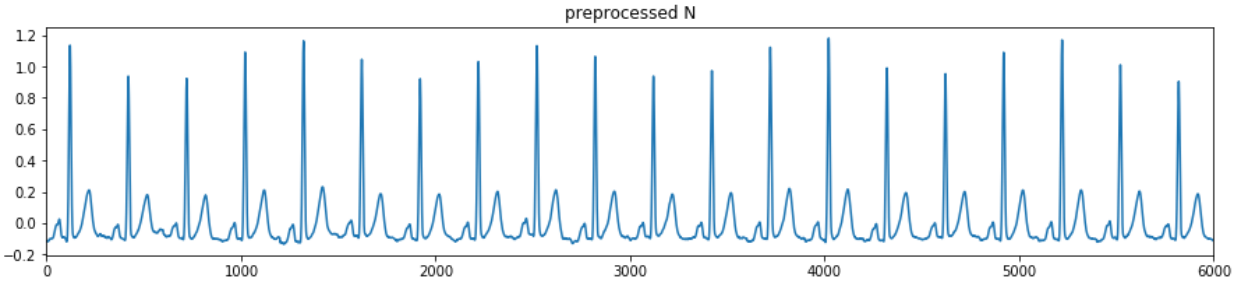

Figure 4. 3: preprocessed N data which is classified correctly

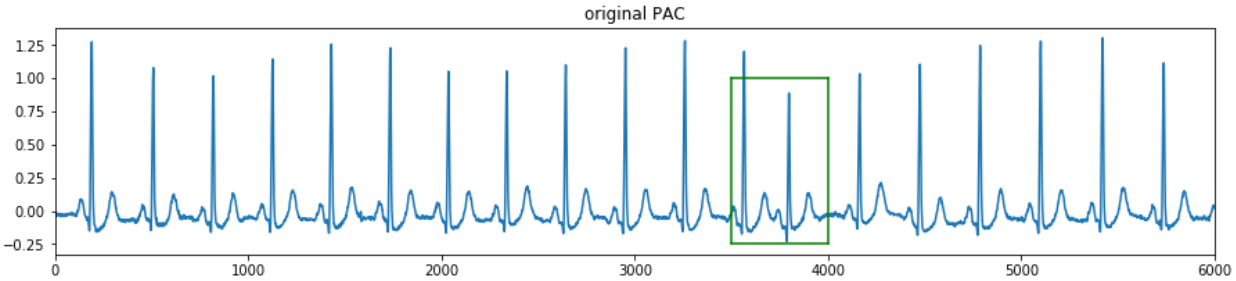

Figure 4. 4: original PAC data refers to Figure 4.2

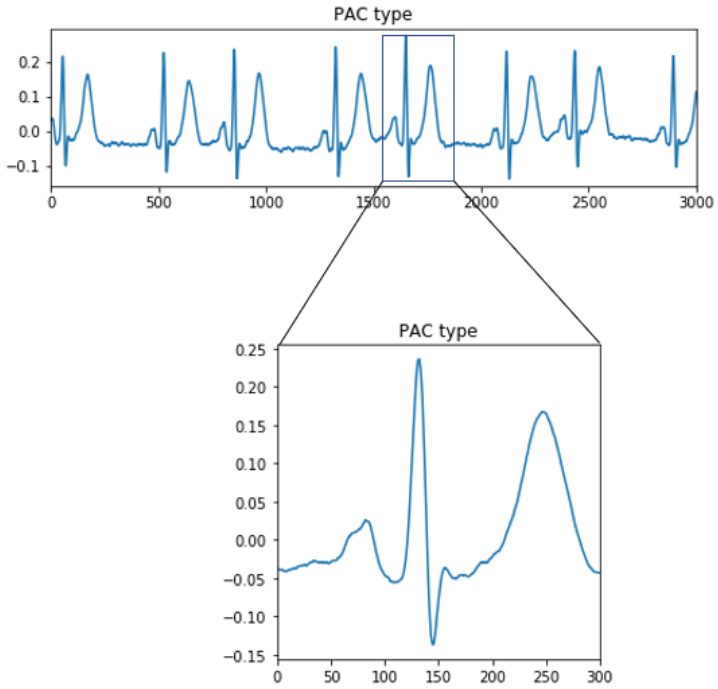

Figure 4. 5: another PAC data which demonstrates the problem of our data preprocessing

According to Table 4.2, STE and STD also have relatively low F1 scores which are below 80%. Most of their data are misclassified as N. We will use lead V5 to visualize the data in this case. We compare the N data which is classified correctly (Figure 4.6) and the STE data which falls into N (Figure 4.7, 4.8). There are upwards slopes at ST-segments in all those figures. Figure 4.6 and Figure 4.8 are very similar in terms of the morphology of each heartbeat, which explains why this STE data is misclassified as N. However, Figure 4.7 shows the steepest upwards slopes at ST-segment, it is easy to be recognized as STE, but the model classifies it as N. The only reason we can think of is that the STE category does not have enough data (Figure 2.13) to train, which causes learning bias of our system. The F1 score of STD is 6% higher than STE's because STD has much more data to train, but STD is still difficult for our system to learn.

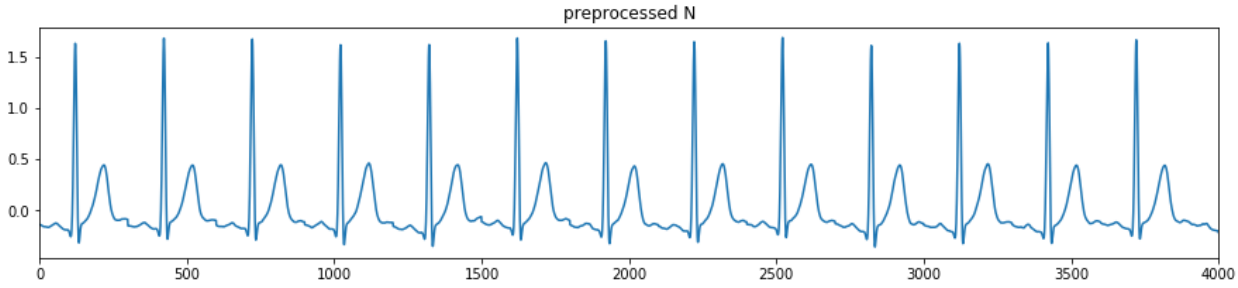

Figure 4. 6: preprocessed N data with lead V5

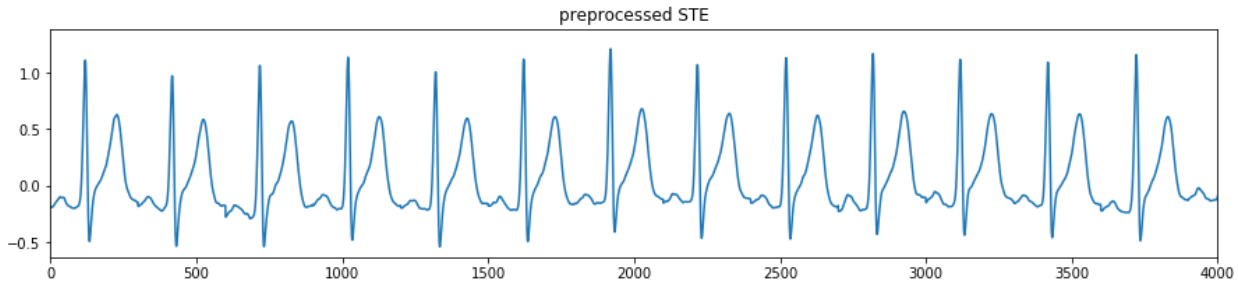

Figure 4. 7: preprocessed STE data with lead V5

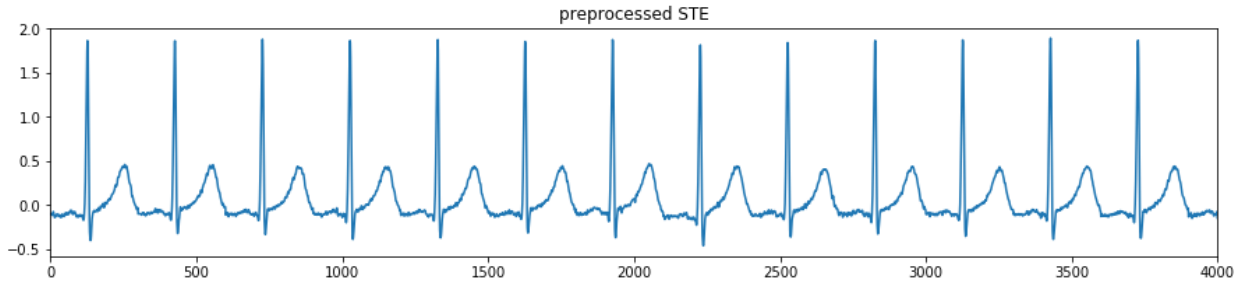

Figure 4. 8: preprocessed STE data with lead V5

According to Table 2.1, LBBB has the least data, but the F1 score is very high, 0.896. This is because LBBB has the most unique morphology among all nine categories which is very easy for our system to learn.

Furthermore, training neural network requires big dataset and sophisticated data augmenting techniques so that the neural network can capture some types of invariances from the data. Our

dataset is small, and it is hard to apply data augmenting techniques on the data. Although we use data preprocessing to ease the training of the system and data augmenting technique to enhance the learning of the system, the system cannot fully capture invariances, such as signal distortion invariances. The signal distortions are noises on the original ECGs which cause corruptions of the information. Those noises include baseline wander, motion artifacts, powerline interference, muscle noise, etc. A good performance of the model also requires good qualities of the dataset. The dataset we have contains a lot of noises and some leads do not have useful information (Figure 2.14, 2.15). In addition, it is possible that the label can be wrong.

4.3 Limitations of Our System and Suggestions for Improvement

In section 4.2, we have mentioned the shortcoming of our data preprocessing. Since we are extracting individual heartbeats, the preprocessed data does not provide information of the relationship between heartbeats, such as PP-intervals. Therefore, our neural network cannot learn the statistics of PP-intervals which cause poor performance in PAC. To improve the result of PAC, we need to segment two to three consecutive heartbeats.

There are also flaws in our neural network design. In each convolutional layer, the kernel size equals the stride size, so there is a limitation to capture all the underlying characteristics of the heartbeats. The reason for same sized kernel and stride is that the convolutional operation should not convolve between two adjacent segmented heartbeats (they are not naturally connected). However, if we apply the convolutional operation in the heartbeat analyzer, we can use any stride size. This type of design is called recurrent convolutional neural network where

the convolutional operation is used in the recurrent layers. However, the trade-off is the processing speed.

Chapter 5 Model Comparison and Conclusion

The dataset is from the 2018 China Psychology Competition [1], so there are models which have been evaluated by the hidden dataset. Because our model was designed after the competition, we were not able to get an evaluation from the hidden dataset. In this chapter, we will first talk about the performances of the top 15 models from the competition. Then we compare our system with some models from the competition in terms of the processing speed and the size of the model. In the end, we conclude our work.

5.1 Performances of Different Models from the Competition

Table 5.1 is a summary of competition result of the hidden dataset from different models. The Serial No. indicates the rank of each model. There are 34 models in the competition, but we only take the top 15 models. F1 in Table 5.1 is the average F1 score which determines the placement of the model.

Table 5. 1: top 15 models from the 2018 China Psychology Competition

Serial No.	F1	Faf	Fblock	Fpc	Fst
1	**0.837**	0.933	0.899	0.847	0.779
2	**0.830**	0.931	0.912	0.817	0.761
3	**0.806**	0.914	0.879	0.801	0.742
4	**0.802**	0.918	0.890	0.789	0.718
5	**0.791**	0.924	0.882	0.779	0.709
6	**0.783**	0.905	0.902	0.722	0.708
7	**0.782**	0.911	0.891	0.775	0.670
8	**0.778**	0.921	0.858	0.797	0.676
9	**0.776**	0.906	0.876	0.773	0.711
10	**0.766**	0.894	0.857	0.733	0.683
11	**0.764**	0.911	0.901	0.662	0.709
12	**0.756**	0.918	0.867	0.785	0.617
13	**0.752**	0.888	0.845	0.709	0.697
14	**0.749**	0.884	0.848	0.712	0.711
15	**0.720**	0.878	0.871	0.618	0.666

Figure 5.1 is made according to Table 5.1; it shows the ranks of models versus the types of F1 scores. The horizontal dash lines are the average values of certain types of F1 scores. As we can see all the models have difficulties to learn the PC type and ST type recordings. The performances of the PC type and the ST type have a strong influence on the rank of the model. The star markers are the results of our ensemble model (Figure 4.1) by using the testing dataset (Table 2.1). The star markers are only for the y-axis visualization, the x-axis is not applicable to the star markers. Although the results of our model are not comparable with others due to different testing

dataset, it gives a general idea of how the performance of our model is. Our F1 score of ST type is quite high, but the F1 score of PA type is lower than the average.

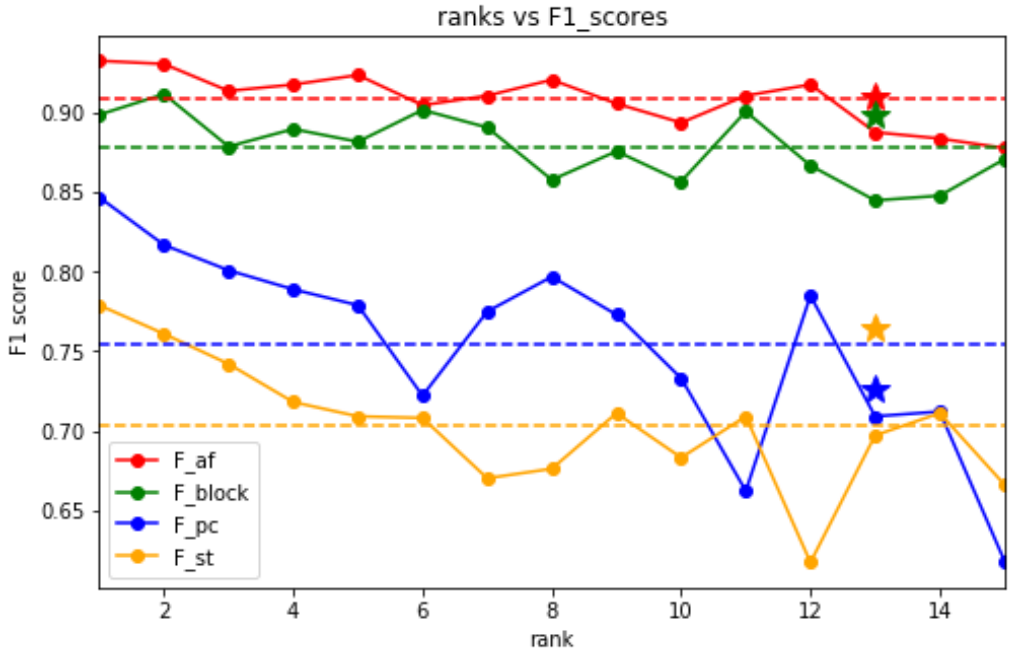

Figure 5. 1: model ranks vs F1 score (refer to Table 3)

5.2 Processing Speeds and the Sizes of Different Models

In many real-life applications, it is impractical to have a large-sized model and a slow processing speed to do tasks. In this section, we try to show the sizes and processing speeds of some neural networks from the competition. Many models from the competition are not running successfully on our computer, so we only have four models presented. The way we evaluate the processing speed is to feed our testing dataset (Table 2.1) into their models, one recording at a time. In Table 5.2 we can see the top 3 models have large numbers of parameters, especially the model with rank 2. The processing speed of our model is relatively fast, but the model with rank 3 is twice as fast as ours. In addition, the model that is rank 3 also uses gender and age as inputs for the neural

network, this is smart because the cardiac arrhythmias are also highly related to these two factors.

Table 5. 2: comparison table of processing speed and model size

Serial No.	Size of the model (param #)	Processing speed (mins)
1	3,646,890	None
2	22,569,166	22
3	1,163,913	2
12	333,009	9
Not ranked (our model)	703,153	4

5.3 Summary of Our Work

In this paper, we build a system to classify nine types of ECGs. The system can extract features from each heartbeat in different scales, then it is able to analyze features of each heartbeat and detect the abnormal ones. After analyzing features of each heartbeat, the heartbeat analyzer gives us a summary which is a fixed size of comprehensive features of the recording. Finally, the classifier tries to predict the category of the recording based on the comprehensive feature. Our system eliminates the relationship between heartbeat to heartbeat which causes poor performance in PAC, but the overall performance is good where our system achieves average F1 scores of 81.2%. We also talk about the results of the models from the competition and compare them with our system in a reasonable manner. Our system has a relatively faster processing

speed and smaller size. To improve the results of our model, we need our system to learn the statistics of PP-intervals, and the stride sizes need to be reduced.

Bibliography

[1] The 2018 China Psychology Competition. [Online]. Available: http://2018.icbeb.org/Challenge.html

[2] K. Minami et al., “Real-time discrimination of ventricular tachyarrhythmia with Fourier-transform neural network,” IEEE Trans. Biomed. Eng., vol. 46, no. 2, pp. 179–185, Feb. 1999.

[3] Inan et al., “Robust neural-network based classification of PVCs using wavelet transform and timing interval features,” IEEE Trans. Biomed. Eng., vol. 53, no. 12, pp. 2507–2515, Dec. 2006.

[4] X. Alfonso and T. Q. Nguyen, “ECG beat detection using filter banks,” IEEE Trans. Biomed. Eng., vol. 46, no. 2, pp. 192–202, Feb. 1999.

[5] J. L. Willems and E. Lesaffre, “Comparison of multigroup logisitic and linear discriminant ECG and VCG classification,” J. Electrocardiol., vol. 20, pp. 83–92, 1987.

[6] J. L. Talmon, Pattern Recognition of the ECG. Berlin, Germany: Akademisch Proefscrift, 1983.

[7] D. A. Coast et al., “An approach to cardiac arrhythmia analysis using hidden Markov models,” IEEE Trans. Biomed. Eng., vol. 37, no. 9, pp. 826–836, Sep. 1990.

[8] S. Osowski et al., “Support vector machine based expert system for reliable heartbeat recognition,” IEEE Trans.Biomed. Eng., vol. 51, no. 4, pp. 582–589, Apr. 2004.

[9] Zubair, M., Kim, J., & Yoon, C. (2016). An Automated ECG Beat Classification System Using Convolutional Neural Networks. 2016 6th International Conference on IT Convergence and Security (ICITCS). doi: 10.1109/icitcs.2016.7740310

[10] Acharya, U. R., Fujita, H., Lih, O. S., Hagiwara, Y., Tan, J. H., & Adam, M. (2017). Automated detection of arrhythmias using different intervals of tachycardia ECG segments with convolutional neural network. Information Sciences, 405, 81–90. doi: 10.1016/j.ins.2017.04.012

[11] Acharya, U. R., Fujita, H., Oh, S. L., Hagiwara, Y., Tan, J. H., & Adam, M. (2017). Application of deep convolutional neural network for automated detection of myocardial infarction using ECG signals. Information Sciences, 415-416, 190–198. doi: 10.1016/j.ins.2017.06.027

[12] Hannun, A. Y., Rajpurkar, P., Haghpanahi, M., Tison, G. H., Bourn, C., Turakhia, M. P., & Ng, A. Y. (2019). Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network. Nature Medicine, 25(1), 65–69. doi: 10.1038/s41591-018-0268-3

[13] Xiong, Z., Nash, M. P., Cheng, E., Fedorov, V. V., Stiles, M. K., & Zhao, J. (2018). ECG signal classification for the detection of cardiac arrhythmias using a convolutional recurrent neural network. Physiological Measurement, 39(9), 094006. doi: 10.1088/1361-6579/aad9ed

[14] Pyakillya, B., Kazachenko, N., & Mikhailovsky, N. (2017). Deep Learning for ECG Classification. Journal of Physics: Conference Series, 913, 012004. doi: 10.1088/1742-6596/913/1/012004

[15] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). doi: 10.1109/cvpr.2016.91

[16] Girshick, R. (2015). Fast R-CNN. 2015 IEEE International Conference on Computer Vision (ICCV). doi: 10.1109/iccv.2015.169

[17] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). doi: 10.1109/cvpr.2016.90

[18] He, K., Gkioxari, G., Dollar, P., & Girshick, R. (2017). Mask R-CNN. 2017 IEEE International Conference on Computer Vision (ICCV). doi: 10.1109/iccv.2017.322

[19] Chollet, F. (2017). Xception: Deep Learning with Depthwise Separable Convolutions. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). doi: 10.1109/cvpr.2017.195

[20] Collobert, R., & Weston, J. (2008). A unified architecture for natural language processing. Proceedings of the 25th International Conference on Machine Learning - ICML 08. doi: 10.1145/1390156.1390177

[21] Qian, Y., & Woodland, P. C. (2016). Very deep convolutional neural networks for robust speech recognition. 2016 IEEE Spoken Language Technology Workshop (SLT). doi: 10.1109/slt.2016.7846307

[22] Milletari, F., Navab, N., & Ahmadi, S.-A. (2016). V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation. 2016 Fourth International Conference on 3D Vision (3DV). doi: 10.1109/3dv.2016.79

[23] Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. Lecture Notes in Computer Science Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015, 234–241. doi: 10.1007/978-3-319-24574-4_28

[24] Nifong, N. (2000). Learning General Features From Images and Audio With Stacked Denoising Autoencoders. doi: 10.15760/etd.1549

[25] Sun, W. (2016). Induction Motor Fault Diagnosis Based on Deep Neural Network of Sparse Auto-encoder. Journal of Mechanical Engineering, 52(9), 65. doi: 10.3901/jme.2016.09.065

[26] Zeiler, M. D., & Fergus, R. (2014). Visualizing and Understanding Convolutional Networks. Computer Vision – ECCV 2014 Lecture Notes in Computer Science, 818–833. doi: 10.1007/978-3-319-10590-1_53

[27] Understanding LSTM Networks. (n.d.). Retrieved from http://colah.github.io/posts/2015-08-Understanding-LSTMs/.

[28] Santos, L. A. dos. (n.d.). Dropout Layer. Retrieved from https://leonardoaraujosantos.gitbooks.io/artificial-inteligence/content/dropout_layer.html.

[29] Graves, A., & Schmidhuber, J. (2005). Framewise phoneme classification with bidirectional LSTM and other neural network architectures. Neural Networks, 18(5-6), 602–610. doi: 10.1016/j.neunet.2005.06.042

[30] Yuan, X., Feng, Z., Norton, M., & Li, X. (2019). Generalized Batch Normalization: Towards Accelerating Deep Neural Networks. Proceedings of the AAAI Conference on Artificial Intelligence, 33, 1682–1689. doi: 10.1609/aaai.v33i01.33011682

[31] Wu, Y., Zhao, H., & Zhang, L. (2014). Image Denoising with Rectified Linear Units. Neural Information Processing Lecture Notes in Computer Science, 142–149. doi: 10.1007/978-3-319-12643-2_18

[32] Softmax function. (2019, November 7). Retrieved from https://en.wikipedia.org/wiki/Softmax_function.

[33] Entropy and Cross Entropy as Diversity and Distance Measures. (n.d.). *SpringerReference*. doi: 10.1007/springerreference_205700