Heart Rate Variability Analysis of Post-Stroke Elderly People Using 1D Convolutional Neural Network Model

Honor Thesis by Zhaoyi Ding (Bio'21)
For the Academic Year (Fall 2020-Spring 2121)
Advisor: Dr. Eric Ho
Committee: Drs. James Dearworth and Allison Lewis

TABLE OF CONTENTS

Author Biography

Ms. Zhaoyi Ding is a Lafayette College student pursuing a Bachelor of Art in Biology with a minor in Mathematics. When not studying, she enjoys watching movies and cartoons, reading novels, and tasting food, while authentic Chinese food is always her favorite. After graduation, Zhaoyi will pursue her graduate study in Biostatistics at Johns Hopkins University.

Acknowledgment

Many people have contributed to my college life here at Lafayette and positively influenced me. Before I arrived at Lafayette College, I had never set foot on American soil. Moving from China to America and studying at Lafayette College for almost four years, I have experienced the obstacles and barriers like most international students. Despite that, I have adapted to the new environment relatively quickly with the kind support and encouragement from friends, family, and professors through this journey.

In particular, I would like to thank Dr. Eric Ho. He has supported me in overcoming the cultural and academic challenges faced by overseas students. Dr. Ho has shaped my academic career significantly and led me to discover my interest and strength in quantitative areas of biology. I cannot thank him enough for his kindness and patience in mentoring this project. Last but not least, the snacks (healthy and unhealthy) he supplied are life-saving in fueling the intense mental exercise in our weekly lab meetings.

I have been extremely grateful and fortunate to have my four years of education at Lafayette. It would never be possible for me to become the person I am today without all of these amazing people around me.

Abstract

Stroke is a leading cause of death and serious long-term disability in the US. There are two main types of stroke, ischemic stroke and hemorrhagic stroke - about 87% of all strokes are ischemic stroke. Besides age (65 and over), cardiovascular disease is a major cause of stroke. Thus, in this study, we analyzed electrocardiogram, abbreviated as ECG or EKG, of post-stroke people in order to identify distinctive ECG patterns among stroke survivors. As ECG is a non-invasive and well-established cardiovascular diagnosis, such analysis may become a potential preventive diagnostic method in the future.

Deep convolutional neural networks (CNNs) have been used to analyze ECG data including detection of real-time arrhythmia, early signals of hypertrophic cardiomyopathy (HCM), and ECG characteristic points. To our best knowledge, no research focused on using a one-dimensional CNN (1D CNN) to analyze ECG data recorded from post-stroke elderly people as this project. Subjects participated in this project were between 60 to 80 years of age, with the first hemispheric ischemic stroke, which to some extent, represents this high-risk elderly population of stroke.

In this project, we built two ID CNN models, one for quality checking and the other for distinguishing stroke from stroke-free people. The purpose of constructing a 1D CNN model for quality checking is to mitigate training noise present in ECG data such that the performance of the stroke model can be improved. Results show that the quality checking model has achieved a high accuracy of 97.5%. With that, high quality ECG segments were used to train another binary classifier ID CNN model with the goal of distinguishing stroke from stroke-free subjects. The model has achieved an accuracy of 90.5% on average.

In summary, 1D CNN is an effective method to identify distinctive ECG patterns among post-stroke elderly people, with a relatively high accuracy. With further refinements, it may hold promise for post-stroke monitoring.

摘要

在美国，中风是导致死亡和严重的长期残疾的主要原因之一。中风主要有两种类型，缺血性中风和出血性中风 — 大约 87%的中风患者是缺血性中风。除了年龄（65 岁及以上是患中风的高危人群），心血管疾病也是中风的主要原因之一。因此，在这项研究中，我们分析了中风后的人群的心电图（简称为 ECG 或 EKG），以识别中风幸存者中独特的心电图模式。由于心电图是一种无创且成熟的心血管诊断方法，我们的这项研究很可能在未来为预防性诊断中风的方法上提供有效的帮助。

尽管之前已经有许多研究利用了深度卷积神经网络（简称为 CNN）来分析心电图数据，其中包括用卷积神经网络检测实时心律失常、肥厚型心肌病（简称为 HCM）的早期信号、以及心电图的特征点。据我们所知，目前还没有任何研究像本项目一样专注于使用一维卷积神经网络模型（简称 1D CNN 模型）来分析中风后的老年人的心电图数据。本项目所使用到的心电图数据采集自 60 至 80 岁的受试者。受试者分为分为中分组（患过一次半球缺血性中风的受试者）和其对照组（没有患过中风的受试者）。因此，本项目所使用的实验数据在一定程度上代表了现实生活中更有可能患中风的高危人群，也就是 65 岁以上的老年人。

通过使用 1D CNN 模型，我们建立了一个二项分类器。这个分类器可以识别和去除质量差的心电图数据段，且准确率高达 97.5%。而识别后被定义为质量好的心电图数据段则被用来训练另一个二项分类器，该分类器可以使用 1D CNN 模型区分中风患者和健康患者。值得高兴的是该分类器同样取得了较高的准确率（平均 90.5%）。

综上所述，使用 1D CNN 模型对心电图数据进行分析从而区分实验对象是否中风可以达到相对较高的准确率，尤其是针对 65 岁及以上的老年人。虽然这个模型仍需要进一步的完善，但它极可能在未来对中风的监测上提供帮助。

Introduction

Stroke is the fifth leading cause of death in the United States (one death every 4 minutes) and is a major cause of long-term disability for adults (3.1% of adults) (Summary Health Statistics, 2018). Stroke occurs when blood supply to the brain is suddenly interrupted, leading to neuronal infarction. The two main types of stroke are ischemic stroke and hemorrhagic stroke. Most strokes, about 87% of stroke cases ("Stroke Facts", CDC, 2021), are caused by an ischemic stroke in which arteries leading to the brain are abruptly blocked by blood clots. The other type of stroke, namely hemorrhagic stroke, is caused by blood vessel bursts, which will cause bleeding into brain tissue. Common impairments of stroke include paralysis, and the loss of speech and language abilities. Stroke risk increases with age (particularly for people of age 65 and over), and the leading causes of stroke include high blood pressure, smoking, heart disease, diabetes, and obesity. Since cardioembolism is a major subtype of ischemic stroke, heartbeat analysis becomes a promising path to study stroke, may even becoming a potential preventive diagnostic method.

The standard method to capture heartbeat information is electrocardiogram, abbreviated as ECG or EKG. It is a non-invasive diagnosis that measures the electrical activity of the heartbeat. The heart is divided into four chambers (Figure 1). The upper chambers are called atria and the lower chambers are called ventricles. Blood flows from the atria to the ventricles through one-way valves. The atria and the ventricles orchestrate to pump blood through the heart, by a rhythmical series of contractions and relaxations through alternating electrical polarization and depolarization of myocardial cells. Resting myocardial cells are in a polarized state (-90mV). The influx of sodium ions in the myocardial cells at the right sinoatrial node, also known as the SA node or the pacemaker node, causes a rapid depolarization of the cells,

manifested as the “P wave” (Figure 2). The electrical signals pass through the atrioventricular, or AV node (after a delay) diverge, and are conducted through the left and right bundle of His to the respective Purkinje fibers for each side of the heart, leading to the ventricular endocardium, and epicardium that trigger rapid depolarization of these cells, and causing the muscle cells to contract, reflected in the “QRS complex” (Figure 2). To restore to the resting state, cells are repolarized by the outflow of potassium ions and the inflow of calcium ions, but with a rate slower than depolarization as indicated by the “T wave” (Figure 2).

As the heart undergoes cyclic depolarization and repolarization, the electrical currents that are generated within the heart also spread throughout the body, making it detectable by an array of electrodes placed on the body surface. Patterns of electrical signals revealed by ECG are used by physicians for diagnosing heart diseases such as atrial fibrillation, and myocardial infarction.

Nonlinear Dynamics Analysis, MFDFA, and Logistic Regression Model for ECG Analysis

A previous study suggested that heartbeat is nonstationary, nonlinear, and multifractal. (Baillie RT et.al. 2009). Therefore, in our initial proposal, we formulated an approach that characterizes ECG based on nonlinear dynamics time series analysis and multifractal detrended fluctuation analysis (MFDFA). Such characterizations will feed into a supervised machine learning classification model (logistic regression model) in order to determine if it is from a healthy or a post-ischemic stroke elderly people. This project was primarily implemented in R with additional packages, RHRV (García, Martínez, CA et al. 2017) and MFDFA.

Machine learning is the study of computer algorithms that make decisions or forecasts which are better by humans. The two types of machine learning are: supervised learning and

unsupervised learning. In essence, the major difference between the two is that supervised learning requires labeled data, e.g. healthy or stroke ECG, for training the machine model. In this project, a supervised machine learning approach was taken. Supervised machine learning algorithms build around a mathematical model (such as logistic regression) with the goal of predicting the response variable by the predictors with minimized prediction errors. The term predictor is the synonym of a feature or an independent variable in the context of machine learning. An effective learning algorithm harnesses misclassification error or prediction error during training as a feedback to fine-tune the model parameters such that the average error is gradually minimized.

Logistic regression was chosen for this project for its simplicity, interpretability, and suitability for two-class classification problems such as stroke versus healthy. At the core of it is the logistic function (Figure 3), also known as the sigmoid function:

$$f(x) = \frac{1}{1 + e^{-\vec{\beta t} \cdot \vec{x}}}$$

where $\vec{x}$ is a numerical vector representing input features, and f(x) is the predicted label of $\vec{x}$. The best coefficients (β's) would result in a model that would predict a value very close to 1 (e.g. stroke) for one class and a value very close to 0 (e.g. healthy) for the other class.

We characterized the ECG data into 11 values by using the nonlinear dynamics analysis and MFDFA as shown in Table 1 where they will be fed to the logistic regression model with the objective of optimizing β's such that misclassification error is minimized.

Figure 4 shows that the mean accuracy rate of the logistic regression model is around 67.8% ± 8.6. It is noteworthy that its performance varies in a wide range, from 30% to 90%. Although the performance of the model is not desirable, the logistic regression model can be served as the baseline due to its simplicity. To strike for better performance, we switched to

another supervised machine learning method, convolutional neural network, which is the focus of this report.

Convolutional Neural Network Model for ECG Analysis

Previous studies had used deep convolutional neural networks (CNNs) to process ECG. CNN is a specialized neural network that is designed to detect local features, such as the R-peaks in an ECG, their spatial relationship, and possible hierarchy. Detection of ECG characteristic points serves as the first step in automated ECG analysis technique (Chen M, et al. 2018). Arrhythmias is a common condition associated with irregular heart rhythm. One study employed a CNN-based model for real-time arrhythmia detection (Wu Q, et al. 2020). Another arrhythmia diagnosis study utilized a 9-layer deep CNN to capture the contrast between normal versus abnormal individual heartbeats to classify them into five categories including non-ectopic, supraventricular ectopic, ventricular ectopic, fusion, and unknown beats. (Acharya UR, et al. 2017)). The result can serve as a tool for screening arrhythmic heartbeats from patients' ECGs quickly. Another study also concentrated on the arrhythmia classification and presented a CNN model for accurate classification (Wang H, et al. 2019). Similarly, in this study, a CNN is proposed to automatically classify the heartbeat of arrhythmia into five categories (normal, supraventricular ectopic, ventricular, fusion, and unknown beats). The experiment demonstrated that its CNN model had high performance for arrhythmia detection of 99.06% accuracy.

The automatic detection of atrial fibrillation (AF) is essential for its association with the risk of embolic stroke. One study proposed an AF detection method based on 1D convolutional neural network to improve the detection accuracy and reduce network complexity (Hsieh CH, et al. 2020). Hypertrophic cardiomyopathy (HCM) is an uncommon but significant cause of sudden

cardiac death. One study aimed to develop a CNN model to detect early signals of HCM based on the 12-lead electrocardiography (Ko W-Y, et al. 2020). Given the dire consequence of HCM, the model could become a lifeline of high-risk people. Instead of processing the raw ECG signals, another study developed a 6-layer CNN model that recognizes ECG images and their patterns with the goal of identifying myocardial infarction (MI) (Makimoto H, et al. 2020).

There are similar studies used methods other than CNN to analyze ECG. ECG recordings are usually corrupted by large amounts of noise and artifacts, which not only interfere with the correct recognition of P, QRS, and T waves of the ECG, but also increases the rate of false alarms for cardiac monitors (Li Q, et al. 2014). Most current ECG signal quality assessment studies aimed to provide a two-level classification: clean or noisy. However, more specific noise level classification is demanded in clinical usage, such as whether or not the noise interferes with interpretation or recognition of P, T waves and QRS complexes. One study outlines a five-level ECG signal quality classification algorithm by training a support vector machine (SVM) (Li Q, et al. 2014). Besides ECG signal quality, various machine learning techniques are applied to distinguish abnormal (pathological) ECG signals from normal ones in another study (Celin S, et al. 2018). The input ECG signals are classified by using support vector machine, Adaboost, artificial neural network (ANN), and Naïve Bayes classifier into two categories, normal or abnormal ECG signal. Fragmented QRS (fQRS) is an accessible biomarker and indication of myocardial scarring that can be detected from the ECG. In one study, its objective is to detect and quantify fQRS via several well-known machine learning classifier such as SVM, K-nearest neighbors (KNN), Naïve Bayes and TreeBagger (Goovaerts G, et al. 2019). Besides ECG data, other health data, such as blood pressure, are also used to predict stroke via some machine learning algorithms (deep NN and other methods).

CNNs are widely used to process time series, images, and videos due to its flexibility in supporting multidimensional data such as 1D for time series, 2D for black and white images, and 3D for color images. A CNN uses a feature detector (filter or kernel) to summarize the characteristics of a small range of input data in a single value that represents their essence. For a time-series data example as shown in Figure 5, the filter examines three consecutive data points and returns the maximum value as the output. The filter will shift one position to the right until it has reached the rightmost boundary. By repeating this procedure for the whole time series from left to right, it produces an abstraction or summarization in a scale which is one-third of the original time series with spatial information being preserved. The output from the current layer will be passed as inputs to the next layer.

Besides the raw time series data, additional features are included with the aim of enhancing the characterization of the time series, leading to a better prediction accuracy. In this project, four features are devised to supplement the raw ECG signals including smoothed signal (filtered at 11 Hz), high-pass signal, first derivative and second derivative of the smoothed signal (Figure 6). More details will be given below.

A max pooling layer is often used after each CNN layer in order to capture high level spatial relationship among patterns. In this example, the pooling layer slides a window of height 15 across data and replaces it with the maximum value, thus discarding a majority of values from the previous layer and resulting in a smaller neuron matrix. The result from the first max pooling layer will be fed into the second CNN layer with 64 different filters (a kernel size of 30). Following the same logic as the first layer. This second convolutional layer allows the model to learn more complex features.

The global max pooling layer down samples the entire features map to a single value, summarizing the strongest activation. Results in an even smaller neuron matrix with only one row.

To further reduce overfitting, a dropout layer is added at the end. The dropout layer will randomly assign 0 weights to the neurons in the network. For example, a dropout rate of 0.5 means that weights of 50% of the neurons will be set to zero randomly. The different neurons are less likely to influence each other because any of them might be dropped out at random. Thus, the network becomes less sensitive to smaller variations in the data and further increases its accuracy. The output neuron matrix of this layer is the same as that of the global max pooling layer.

The final layer will reduce the vector of height to one since this example is a two-class problem, i.e., stroke versus control.

Learning Curves

Learning curves are plots that show changes in learning performance or loss (y-axis) over the number of rounds passing through the input data (epoch) on the x-axis (Figure 13). They are widely used in machine learning algorithms that illustrate incremental learning through optimization of the model's internal learning parameters over time, such as weights of a deep neural network.

The information from learning curves of model performance can help to monitor overfitting and underfitting of the model during training. Underfitting refers to a model that still has the room for improvement by learning more from the training data. A plot of learning curves shows the underfitting problem if the training loss remains flat regardless of training or the

training loss continues to decrease until the end of training. Overfitting refers to a model that has learned the training dataset too well – largely based on memorization. The symptom of overfitting is the loss of generalization when the model is used to predict unseen samples. A plot of learning curves shows the overfitting problem if the plot of training loss continues to decrease with experience and the plot of validation loss decreases to a point and begins to increase again simultaneously. A good fit is the goal of the machine learning algorithm and exists between an overfit model and an underfit model. A good fit is identified as the training and validation losses that decrease to a point of stability with a minimal gap between the two final loss values, and the loss of the model will almost always be lower on the training dataset than the validation dataset.

Giving the utility of 1D CNN in analyzing ECG, **the overarching goal of this project is to use features from ECG patterns that guide a supervised machine learning model (1D convolutional neural network model) to determine if it is from a post-stroke or a stroke-free elderly person.** A roadmap of the project can be found in Figure 7.

Aims

Aim 1: Collect and preprocess the ECG data downloaded from the study *Cerebral Vasoregulation in Elderly with Stroke* (Novak V. et.al. 2010).

Aim 2: Construct a 1D CNN model that detects poor quality ECG data segments.

Aim 3: Construct a 1D CNN model that distinguishes ECG of healthy people from post-stroke people.

Aim 4: Compare the performance of our classifier with existing methods.

Major Findings

By using a 1D CNN model, a binary classifier was developed that can weed out poor quality ECG data segments with a high accuracy of 97.5%. Good-quality ECG segments were used to train another classifier, which is utilized to distinguish stroke subjects from healthy ones, via a 1D CNN model. This stroke model has achieved a high accuracy (90.5%), showing a significant improvement from the baseline logistic model.

Materials and Methods

ECG Data

A freely accessible critical care ECG dataset of a large study (*Cerebral Vasoregulation in Elderly with Stroke*), which aimed to investigate the effects of ischemic stroke on cerebral vasoregulation, was obtained from PhysioNet Database (Novak V. et.al. 2010). This study recruited 60 subjects in the non-stroke group (control group) and 60 subjects in the post-stroke group (stroke group) from the greater Boston area, but only 48 control and 43 stroke participants completed the study. Subjects were 60 to 80 years of age, with a nearly even gender distribution (49 females vs. 42 males). For the stroke group, it included subjects with the first hemispheric ischemic stroke with documented neurological deficit persisting greater than 24 hours, and they were at least 6 months after stroke. Stroke etiology includes intracranial atherosclerosis (17 subjects), cardioembolism (5 subjects), atherothrombosis (5 subjects), and undetermined/unknown (16 subjects). For the control group, subjects with no clinical history of stroke and no focal deficit on the neurological exam were recruited. Without blood pressure controlling medications, subjects within stroke and control groups were stratified as "hypertensive" (BP $\geq$ 140/90 mmHg) and "normotensive" (BP < 140/100). The protocol involved a 2-day overnight stay in General Clinical Research Center (GCRC). The structured regime that simulated everyday activities such as active standing, walking, eating, and sleeping (Table 2) was implemented on Day 1 and Day 2. In our project, we focused on analyzing ECG segments that were randomly selected from the entire period of study. More details in below.

Logistic Regression Model

The preliminary data (Table 1) was divided into two parts, the training set containing 80% of the preliminary data and the rest of the 20% formed the test dataset. Training set was utilized to train our logistic regression model and the test set was utilized to test the accuracy of our model. The logistic regression model was built using R's glm package (version 3.6.2). The training and testing processes were repeated 10,000 times, and the average accuracy rate was 0.678 as shown in Figure 4.

One-dimensional Convolutional Neural Network Model

ECG Preprocessing

The ECG data downloaded from PhysioNet Database was in WFDB format (WFDB 2019). The ECG was captured by ME6000 device (MegaElectronics) at 1000 Hz. A custom Python script was developed to convert electrical signals recorded in WFDB format into plain text files, namely `.p_signal` files. In the two-day study, ECG of each subject was recorded in two to three separate files. In total, 402 `.p_signal` files from 91 subjects were created.

Due to the sheer volume of data, we randomly sampled 100 non-overlapping segments from each subject where each segment covers a period of 6000 milliseconds (ms). As a result, 35,000 segments were selected. This dataset formed the initial dataset for our 1D CNN model building process.

These segments were normalized and stored in separate files, namely `.normalized` files. Normalization reduces scale variation of the ECG signals. Importantly, normalization ensures neural network parameters converge during training. In this project, MinMaxScaler

method was used to normalize the data. Raw ECG electrical signals were rescaled to the range between 0 and 1 according to the formula below:

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

Where x_{min} and x_{max} are the minimum and maximum values of the segment, respectively.

Here is the code for normalizing a 6000-ms ECG data segment using MinMaxScaler method:

```
import pandas as pd
from sklearn import preprocessing

df = pd.read_csv(data_home+"/"+fname)
normalized_data = min_max_scaler.fit_transform(df)
normalized_df = pd.DataFrame(data=normalized_data, columns=df.columns)
```

ECG Feature Extraction

Four features were extracted from the normalized segments: smoothed signal (at 11 Hz), high-pass signal, first derivative, and second derivative (Figure 6). Raw signals in our normalized segments possess random noise, and therefore smoothing out the raw signals in each normalized segment in the first step can get rid of most random fluctuations. Through try-and-error, the threshold frequency used to filter the raw signals was 11 Hz.

Butterworth filter method (`butter`) from `scipy` was used to implement the low-pass filter. Here is the code for generating the smoothed signal (11 Hz):

```
from scipy.signal import butter,filtfilt
def butter_lowpass_filter(data, cutoff, fs, order):

    nyq = 0.5*fs
    normal_cutoff = cutoff/ nyq
    b,a = butter(order, normal_cutoff, btype = 'low', analog = False)
    y = filtfilt(b, a, data)
    return y
```

In essence, the low-pass filter allows signals with frequency lower than a certain cutoff frequency to pass through unaltered but attenuating signals with a higher frequency. In the code above

- "data" was the normalized raw signals of a segment.
- "cutoff" was the desired cutoff frequency of the filter, i.e. 11 Hz.
- "fs" was the sampling frequency or rate of the input signal and it was 1000 Hz from above.
- "order" represented the polynomial order of the filtered signal and it was assigned to 2 since sinusoidal noise appearing in ECG data can be interpreted as quadratic.
- "nyq" represented the Nyquist frequency which was the minimum rate where a finite bandwidth signal needs to be sampled to retain all of the information. It was assigned as $0.5 * fs$.

As high-frequency signals (high-pass signals) may also reveal status of stroke, we isolated them from the normalized signals by subtracting the smoothed signals from the normalized signals per segment.

The rate in which the signals change may complement the smoothed and high-frequency features in differentiating stroke and stroke-free ECGs. Therefore, the first and second derivatives of the smoothed signal were considered. The first derivative was estimated by an imaginary slope triangle, with base length 10 units, sliding along the time axis point by point. The slope of the signal at a point equaled the value of the tenth data point minus the value of the first data point, and divided by 10. The second derivative was estimated in similar fashion except that the input data was the first derivative of the signals.

Here is the code for calculating the first and the second derivatives from the smoothed signals:

```
## 1st derivative
L = len(df.raw_signal)
lowps_1st = [0]*L

low_pass_series = df["low_pass_signal"].values
for n in range (0, L-delta_x+1):
    derivative_1 = (low_pass_series[n + delta_x-1] -
                    low_pass_series[n])/delta_x
    lowps_1st[n] = derivative_1
df['first_derivative_smooth'] = lowps_1st

## 2nd derivative
lowps_2nd = [0]*L

first_derivative_series = df["first_derivative_smooth"].values
for m in range (0, L-delta_x+1):
    derivative_2 = (first_derivative_series[m + delta_x-1] -
                    first_derivative_series[m])/delta_x
    lowps_2nd[m] = derivative_2
df['second_derivative_smooth'] = lowps_2nd
```

Quality Control

We discovered the ECG data was contaminated by bad signals as shown in Figure 7. To clean the data, we built a 1D CNN model to distinguish good from bad ECG segments such that low-quality segments can be removed before training the stroke model.

To train the classifier (good vs. bad), we used a semi-automatic approach to collect about 300 good and bad segments for training and testing the classifier. Four non-overlapping 6000-ms segments were randomly selected from each `.p_signal` files, making the total number of segments to almost 1608. A Python script was developed to plot those segments as shown in Figure 9 and 10. We manually classified segments into good and bad. If the figure was similar to Figure 10, it was labeled as a good segment. We repeated this process until we found at least 300 good segments. As for selecting bad segment samples, we observed that subject s0068 (`s0068-05022417-ecg0.p_signal` file) contained abundant bad segments, and therefore we randomly selected 600 non-overlapping 6000-ms segments from it. Following the same

procedure as finding good segments, we had found 300+ bad segments, constituting the training dataset for our 1D CNN model.

A binary classification model was built using a one-dimensional convolutional neural network (1D CNN) model with the goal of distinguishing good from bad ECG segments. The 1D CNN model was implemented in Python 3, Tensorflow (version 2.3.0), and Keras (version 2.4.3).

Here is the code for constructing the 1D CNN model that is utilized to check the data segment quality (good vs. bad):

```
model = models.Sequential()
model.add(layers.Conv1D(32, 30, activation='relu', input_shape=(6000,5)))
model.add(layers.MaxPooling1D(15))
model.add(layers.Conv1D(64, 30, activation='relu'))
model.add(layers.GlobalMaxPooling1D())
model.add(layers.Dense(1, activation='sigmoid'))

# Compile defines the loss function, the optimizer and the metrics.
model.compile(optimizer=RMSprop(lr=1e-4),
              loss='binary_crossentropy',
              metrics=['acc']
             )
```

Input data were good and bad 6000-ms data segments. The CNN comprised of two layers. The first 1D CNN layer defined a filter of length 30 (also called kernel size of 30). The filter moved along the time axis one data point each step. Only defining one filter would allow the neural network to learn only one single feature in the first layer, and this might not be sufficient. Through try-and-error, 32 filters have been defined, allowing the model to train 32 different features on the first layer of the network. This initial layer will learn basic features by decomposing the input signals into 32 different features. The term “filter” in the first 1D CNN layer refers to each filter that scan through the time series to learn one single feature. At the same time, it also means 32 output features (32 rows) of the neuron matrix after scanning as shown in

Figure 5. After the first 1D CNN layer, the max pooling layer sampled the maximum value of the first output layer within a stretch of 15 values. After that, it moved to the 16th position, sampled the maximum value as before. This process was repeated until the end of the time series. Another 1D convolutional layer similar to the first convolutional layer was added, except the number of output filters was increased to 64 from 32. The global max pooling downsampled the entire output features from the previous layer into a vector of 64 values and it was passed to a single-layer, fully connected network. The final output layer which consisted of one neuron ("good" vs. "bad"). It used these 64 values and the label (good or bad) to fine-tune parameters of a sigmoid function, namely activation function, with the goal of minimizing classification error. If the sigmoid function output a value that is greater than 0.5, the input will be assigned as good; otherwise, it will be assigned as bad.

The compilation is the final step in constructing a 1D CNN model. Once the compilation is done, we can move on to the training and testing phase. Important arguments include loss function, optimizer, and metrics. Loss function is utilized to find error or deviation in the learning process, and Keras requires loss function during the model compilation process. Among many loss functions supported by Keras, "`binary_crossentropy`" was chosen in this model. Optimization is an important process which optimizes the input weights by comparing the prediction and the loss function. In this model, `RMSProp` optimizer was used. Metrics was used to evaluate the performance of our model. It was similar to the loss function, but used in the training process. We chose "`accuracy`" as our metrics.

Here is the code for training and testing the 1D CNN model that is utilized to check the data segment quality (good vs. bad):

```
# Training & eviluation
history = model.fit(train_ecg, train_label,
                    epochs=10,
                    batch_size=20,
                    validation_split=0.2
                   )
# Calculate the model accuracy
test_loss, test_acc = model.evaluate(test_ecg, test_label)
print(f"test_accuracy = {test_acc}")
```

Model was trained using a fit function. The main purpose of this fit function is to evaluate our model on training, and it had following syntax:

- "train_ecg" and "train_label" were tuples to evaluate our data.
- "epochs" specifies the number of times the training dataset will be processed during training, and was set as 10. The number of epochs is related to the number of rounds of optimization which are applied during the training process. With more rounds of optimization, the error on training data will reduce further and further. However, there may come to a threshold where the network becomes over-fitting with respect to the training dataset and will start to lose performance in terms of generalization to unseen data (i.e. validation dataset and test dataset). In order to analyze this, we can monitor the error performance on separated training and validation dataset (more details are shown below), as the number of epochs increases.
- "batch_size" represents the number of training instances to be processed before refreshing the network weights. It was set to 20. A small batch size means that the weights will be updated more frequently but with small changes each time.
- "validation_split" specified randomly put aside 20% of the training data for validating the prediction accuracy of the model.

This 1D CNN model (also called good vs. bad classifier) scanned through each normalized segment, retaining good segments and discarding bad ones. The remaining data segments were all of good quality, namely `normalized_clean` files.

1D CNN Stroke Model

Another 1D CNN model (called 1D CNN stroke model) was built to distinguish ECG segments of a post-stroke person from a stroke-free person. Good-quality data segments classified by the 1D CNN good vs. bad model were selected for training the 1D CNN stroke model. In total, 14310 controls and 12193 strokes were fed into the model.

Here is the code for constructing the 1D CNN model that is utilized to categorize those good-quality data segments (stroke vs. control):

```
model = models.Sequential()
model.add(layers.Conv1D(32, 30, activation='relu', input_shape=(6000,5)))
model.add(layers.MaxPooling1D(15))
model.add(layers.Conv1D(64, 30, activation='relu'))
model.add(layers.GlobalMaxPooling1D())
model.add(layers.Dropout(0.5))
model.add(layers.Dense(1, activation='sigmoid'))

# Compile defines the loss function, the optimizer and the metrics.
model.compile(optimizer=RMSprop(lr=1e-4),
              loss='binary_crossentropy',
              metrics=['acc']
             )
```

Input data are all good-quality 6000-ms data segments. In total five rows, and each row represents one feature of ECG data segment. In total, two CNN layers were implemented. The first 1D convolutional layer with 32 filters with a kernel size of 30 applied to each filter, moving along on the input data. After the first 1D CNN layer, the max pooling layer slides a window of height 15 across the data and replaces it with the maximum value. A second 1D CNN layer with 64 filters was added, which was applied by the same concept as for the first 1D CNN layer. The

global max pooling down samples the entire features map to a single value summarizing the strongest activation. To further reduce overfitting, a dropout layer with a rate of 0.5 was added at the end. The dropout layer will randomly assign 0 weights to the neurons in the network. In this model, the rate is set to 0.5, and hence 50% of the neurons received a zero weight. With this operation, the network becomes less sensitive to react to smaller variations in the data. The final output layer consisted of one neuron (stroke vs. control). Sigmoid was used as the activation function. Calculated by the sigmoid function, the result value that is greater than 0.5 will be assigned as 1 (stroke group), or will be assigned as 0 (control group) if the result value is less than 0.5.

Synonymous with the model for data quality checking, loss function was set as "`binary_crossentropy`", optimizer is set as "`RMSprop`", and metrics was set as "`accuracy`".

Here is the code for training and testing the 1D CNN model that is utilized to categorize those good-quality data segments (stroke vs. control):

```
# Training & eviluation
history = model.fit(train_ecg, train_label,
                    epochs=100,
                    batch_size=50,
                    validation_split=0.2
                   )
# Calculate the model accuracy
test_loss, test_acc = model.evaluate(test_ecg, test_label)
print(f"test_accuracy = {test_acc}")
```

Same parameters as the data quality checking model were applied to the stroke model's fit function, but it used 100 as epochs, and 50 as batch size. Similarly, the model accuracy was calculated.

Finally, all good-quality normalized segments were fed into the stroke vs. control classifier model and it achieved a significant higher accuracy than 70% attained by the baseline logistic regression model.

Results

1D CNN Good vs. Bad Model with accuracy of 97.5%

A binary classifier with a high accuracy of 97.5% was built by training it with manually annotated bad and good segments. The network design can be found in Figure 11.

Our input data consisted of approximately 300 bad segments and 300 good segments. And each segment was further characterized by five features, resulting in an input shape of (6000, 5). The first and the second parameter values indicate that each sample is a 5x6000 matrix where each row represents a feature including raw signals, smoothed signals, high-pass signals, first derivatives, and second derivatives accordingly. The 6000 columns represent the temporal dimension of the ECG signal.

The input data were randomly apportioned into training and testing datasets with an 80-20 split, resulting in 480 and 120 ECG segments in the training set and the test set, respectively. Additionally, 20% of the training set was reserved for validation during model training (Figure 12). The validation set was used to test the model during training, and the test set was used to evaluate the final model after training.

With 32 filters and each filter having a size of 30 (kernel size), the input training dataset (6000,5) resulted in a shape of (5971, 32) after passing through the first convolutional layer. In other words, with a length of 6000 and a kernel size of 30, the window will slide through the input data for 5971 steps (6000-30+1), resulting in a 5971 x 32 output neuron matrix (Figure 5). For each input sample, each row of the output matrix holds the weights of one single filter containing 5971 weights, and in total 32 rows and 5971 columns.

A CNN layer is often followed by a pooling layer in order to achieve translational and scale invariance. It is also known as down-sampling. For example, a R-peak can occur in any position of an ECG with a variable height. The addition of the pooling layer enables the model to recognize the R-peak in any location of the sample. In this model, the max pooling layer slides a window of length 15 across our data and replaces it with the max value, therefore discarding 93.3% of the values from the previous layer. Results in a 398 x 32 neuron matrix.

The results from the max pooling layer were fed into the second CNN layer. The second CNN layer defined 64 filters with the same kernel size 30. Following the same logic as the first layer, the output matrix was of size 369 x 64. Having a second convolutional layer allowed the model to capture the spatial relationship between distinct patterns such as the relative location of a P-peak and a R-peak.

Similar to the first layer, another pooling layer (global max pooling layer) was added to further enhance translational and scale invariance. The max value was taken of each row with 369 weights within the neural network, meaning there was only one weight per filter remaining in this layer. In result, the output matrix had a size of 64 neurons.

The final CNN layer is a global pooling layer in which a two-dimensional network (e.g. 369x64) was reshaped to a one-dimensional vector of length 64, i.e. by finding the maximum weight per filter from the final layer of the network (see the bottom of Figure 5). And this one-dimensional vector was connected to a single neuron forming a small fully connected neural network. A sigmoid was used as the activation function to aggregate signals and weights into a single value representing “good” or “bad”.

Classification error (loss) is the sole indicator to gauge learning. According to Figure 13, training and validation loss decrease steadily, suggesting neither overfitting nor underfitting

problems. Also, the loss of the model almost always be lower on the training dataset than the validation dataset as samples from the validation data have not been seen by the model. The model was used to screen 35000 randomly selected ECG segments, and 1871 bad segments were identified and removed. In summary, the quality-checking model is well-performed and effective in identifying low-quality segments.

1D CNN Stroke vs. Control Model with average accuracy of 90.5%

A binary classifier, which distinguish post-stroke elderly people from stroke-free ones, with a high accuracy of 90.5% ± 2.3 (n=10) was built by training it with high-quality segments. The network design can be found in Figure 14.

Our input data are 33129 high-quality ECG segments with 5 features, resulting in an input shape of (6000, 5) similar to the quality-checking model. The only difference was the number of input samples. Similarly, the input data were randomly split into two parts, the training set and the testing set. The training set contained 80% of the input data (26503 ECG segments) and the rest of the 20% formed the testing dataset (6626 ECG segments), and 20% of the training set were further constructed as the validation dataset (Figure 12).

This model followed exactly the same learning processes with the same parameter values as the quality-checking model. The training dataset was fed into the first CNN layer, then the max pooling layer. The results from the max pooling layer served as the input data for the second CNN layer, and the global max pooling layer followed.

The main difference in this model's training process was that a dropout layer was added before the final output layer to further reduce the possible overfitting. The output of this layer is a matrix with a size of 64 neurons, which is synonymous with the output matrix size resulting

from the global max pooling layer. The result, one dimensional vector of length 64, from the dropout layer was connected to a single neuron forming a small fully connected neural network. Sigmoid function was also used as the activation function to aggregate signals and weights into a single value representing “stroke” or “control”.

Similarly, according to Figure 12, training and validation loss decrease steadily, suggesting neither serious overfitting nor underfitting occurred. The training and validation loss curves in Figure 15 decrease to a point of small number stably with a small gap between the two final loss values. The model can be used to screen good-quality ECG segments and determine whether it is from a post-stroke elderly or a healthy one. In summary, the 1D CNN stroke model is well-performed and effective in binary classification i.e. distinguish stroke from healthy subjects based on ECG data. However, the fluctuations existing in the validation loss curve need further investigation and explanation, which will be discussed below.

Discussion

The publicly available ECG dataset deposited the PhysioNet was used in this study. Recognizing the importance of maintaining high-quality training data, we first constructed a 1D CNN model to identify poor-quality ECG data segments and it has achieved a quite high accuracy (97.5%). We utilized this quality-checking model to scan through all input ECG segments such that bad segments were discarded in order to safeguard only good quality data was used for training the 1D CNN stroke model, resulting in a reliable model for distinguishing post-stroke from stroke-free ECGs. As a result, a 1D CNN model was built that has achieved a relatively high accuracy (90.5% on average).

For the binary classifier that distinguishes ECG segments from stroke and stroke-free subjects, the accuracy fluctuated as shown in the validation loss curve (Figure 15). That might be caused by several possible reasons.

A small degree of overfitting is a reason that causes fluctuations in the validation loss curve when the developing model is getting deep into the training process. In other words, the fluctuations increase as the loss values decrease in the validation loss curve. This most likely happened in the rounds of optimization during training as the number of epochs progress. With more rounds of optimization, the error on training data will reduce further and further. However, when the stroke model is almost optimized, though the error on the training dataset kept decreasing, further small changes to the network weights are likely to jeopardize rather than improve its performance.

Overfitting is an undesirable situation since the trained model will yield poor predictions on unseen observations, i.e., samples that were not part of the original training data set (James G, et al. 2013). The problem with overfitting is commonly called generalization error.

Another possible reason is due to the continuous optimization adjustments. When the error was about to reach the minimum value, the numerical optimization process may suffer from rounding error. Thus, the error values went back and forth, resulting in the increasing fluctuations in the validation loss curve.

However, those fluctuations in the validation loss curve represent no significant issues of this stroke versus control classifier model overall. Although this 1D CNN model requires further refinement and validation to improve its performance, it may hold promise for stroke risk assessment and post-stroke monitoring in the future.

Regarding the proposed aim 4: to compare the performance of our classifier with existing methods, there has not been any research focused on using a 1D convolutional neural network model to analyze the ECG data of post-stroke elderly people downloaded from PhysioNet (a large study named Cerebral Vasoregulation in Elderly with Stroke). However, there do exist many previous studies that used deep CNNs to analyze ECG data, including detection of ECG characteristic points (Chen M, et al. 2018), real-time arrhythmia detection (Wu Q, et al. 2020), and detection of early signals of HCM (Ko W-Y, et al. 2020). All these studies had achieved respectable performance. Previous studies can serve as baselines for further improvement in using deep CNNs to process ECG data. Similarly, our 1D CNN stroke model can also serve as a proper baseline for future studies concentrating on using a CNN model (1D, 2D, or 3D) to analyze the ECG data of post-stroke people from different databases.

Conclusion

The ECG data used in this project focused on subjects between 60 to 80 years of age, with the first hemispheric ischemic stroke. More than half of stroke cases happened to people of age 65 and over (Summary Health Statistics, 2018), and ischemic stroke (87%) accounts for the highest proportion of stroke in general ("Stroke Facts", CDC, 2021). The ECG database selected in this project, to some extent, represents this high-risk elderly population of stroke, and hence, the binary classifier (stroke versus control) presented here has established the potential utility in post-stroke monitoring for elderly people or people who are considered as high risk such as those who are older than 65.

From a broader perspective, this project has demonstrated the impact of artificial intelligence (AI) on medicine. Stroke reduces mobility in more than half of stroke survivors of age 65 and over (Virani SS, et al. 2020). However, in a survey, only 38% of respondents were aware of all major symptoms and knew to call 911 when someone was having a stroke and 93% of them only recognized one symptom i.e. numbness on one side as a symptom of stroke ("Stroke Facts", CDC, 2021), resulting in a delayed care. A delayed care will result in more disability than those who arrive at the emergency room within three hours of their first symptoms ("Stroke Facts", CDC, 2021). Thus, many elderly people who are not aware of all main symptoms of stroke may end up with delayed care and ultimate mobility-reducing. The 1D CNN stroke model with relatively high performance that is utilized to distinguish stroke from health works best in post-stroke elderly people. Though some elderly people may not realize that they had a stroke, their ECG data cannot disguise our model. Therefore, regular health checks of elderly people are indispensable, and our model may be a great help.

Besides, stroke risk increases with age, but the stroke can occur at any age. In 2018, around 33% of people having a stroke are less than 65 years old (Summary Health Statistics, 2018). Also, the risk of having a stroke varies with race and ethnicity. In particular, risk of having a first stroke is almost twice as high for blacks as for whites ("Stroke Facts", CDC, 2021). Thus, our future investigations can expand the experimental subjects to include younger age or different races.

Though the ECG data used in this project concentrated on the ischemic stroke, there are different stroke subtypes including intracranial atherosclerosis (17 subjects), cardioembolism (5 subjects), atherothrombosis (5 subjects), and undetermined/unknown (16 subjects). Different subtypes of stroke may possess distinct ECG patterns. However, the current model is not sensitive to different stroke subtypes, and therefore it may not be able to detect specific ECG patterns for each stroke subtypes respectively. Also, the sample size for each stroke subtypes in the ECG data we used is relatively small. Not enough samples for different stroke subtypes make it much more challenging to build a 1D CNN model sensitive to a specific type of ischemic stroke. As for the future improvements, enough samples for different stroke subtypes are indispensable, and can revise the 1D CNN stroke model to be more sensitive and accurate to various stroke subtypes.

Also, the current model only considered ECG segments of length 6000 ms. It might be interesting to investigate further if using ECG segments with different time length (i.e. 12000 ms or longer) can affect prediction accuracy. We had screened out poor quality segments before training the stroke model. However, it might be important to consider how low-quality segments can affect the performance of the model. Additionally, we fixed the number of sampling to be

around 33,000. Therefore, it is also important to find the minimum number of samples to be used to achieve comparable performance.

While this current 1D CNN stroke model has facilitated high accuracy in binary classification (stroke versus control), one noteworthy thing is model interpretability, a core component in model understanding. In the eyes of most ordinary people, CNNs are treated as "black box" technology, and people have no reasonable idea as to where the network is "looking" in the input data as shown in Figure 6. Hence, in the future, we are going to use a technique for visualization of CNNs. It was published in a paper called *Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization* (Selvaraju RR et al. 2016), which used heat maps to visualize neural networks. Using Grad-CAM, we can visualize our 1D CNN stroke model to see where our network is looking, discovering patterns in the input data (ECG data in this project) which may play an important role in CNN stroke model interpretation.

References

Stroke facts | cdc.Gov. Cdc.gov. 2021 Mar 17. https://www.cdc.gov/stroke/facts.htm

Summary Health Statistics: National Health Interview Survey, 2018, Table A-1a. Age-adjusted percentages (with standard errors) of selected circulatory diseases among adults aged 18 and over, by selected characteristics: United States, 2018 All types of heart Coronary heart. Cdc.gov. https://ftp.cdc.gov/pub/Health_Statistics/NCHS/NHIS/SHS/2018_SHS_Table_A-1.pdf

Baillie RT, Cecen AA, Erkal C. Normal heartbeat series are nonchaotic, nonlinear, and multifractal: New evidence from semiparametric and parametric tests. Chaos (Woodbury, N.Y.). 2009;19(2):028503. https://aip.scitation.org/doi/10.1063/1.3152006

García, Martínez, Constantino Antonio, et al. *Heart Rate Variability Analysis with the R Package RHRV*, Springer International Publishing AG, 2017. ProQuest Ebook Central, http://ebookcentral.proquest.com/lib/lafayettecol-ebooks/detail.action?docID=5049922.

Chen M, Wang G, Xie P, Sang Z, Lv T, Zhang P, Yang H. 2018. Region Aggregation Network: Improving Convolutional Neural Network for ECG characteristic detection. Annu Int Conf IEEE Eng Med Biol Soc. 2018:2559–2562.

Wu Q, Sun Y, Yan H, Wu X. 2020. ECG signal classification with binarized convolutional neural network. Comput Biol Med. 121(103800):103800.

Acharya UR, Oh SL, Hagiwara Y, Tan JH, Adam M, Gertych A, Tan RS. 2017. A deep convolutional neural network model to classify heartbeats. Comput Biol Med. 89:389–396.

Wang H, Shi H, Chen X, Zhao L, Huang Y, Liu C. An Improved Convolutional Neural Network Based Approach for Automated Heartbeat Classification. J Med Syst. 2019 Dec 18;44(2):35. doi: 10.1007/s10916-019-1511-2. PMID: 31853698.

Ko W-Y, Siontis KC, Attia ZI, Carter RE, Kapa S, Ommen SR, Demuth SJ, Ackerman MJ, Gersh BJ, Arruda-Olson AM, et al. 2020. Detection of hypertrophic cardiomyopathy using a convolutional neural network-enabled electrocardiogram. J Am Coll Cardiol. 75(7):722–733.

Makimoto H, Höckmann M, Lin T, Glöckner D, Gerguri S, Clasen L, Schmidt J, Assadi-Schmidt A, Bejinariu A, Müller P, et al. 2020. Performance of a convolutional neural network derived from an ECG database in recognizing myocardial infarction. Sci Rep. 10(1):8445.

Hsieh CH, Li YS, Hwang BJ, Hsiao CH. Detection of Atrial Fibrillation Using 1D Convolutional Neural Network. Sensors (Basel). 2020 Apr 10;20(7):2136. doi: 10.3390/s20072136. PMID: 32290113; PMCID: PMC7180882.

Celin S, Vasanth K. ECG Signal Classification Using Various Machine Learning Techniques. J Med Syst. 2018 Oct 18;42(12):241. doi: 10.1007/s10916-018-1083-6. PMID: 30334106

Goovaerts G, Padhy S, Vandenberk B, Varon C, Willems R, Van Huffel S. A Machine-Learning Approach for Detection and Quantification of QRS Fragmentation. IEEE J Biomed Health Inform. 2019 Sep;23(5):1980-1989. doi: 10.1109/JBHI.2018.2878492. Epub 2018 Oct 29. PMID: 30371397

Li Q, Rajagopalan C, Clifford GD. A machine learning approach to multi-level ECG signal quality classification. Comput Methods Programs Biomed. 2014 Dec;117(3):435-47. doi: 10.1016/j.cmpb.2014.09.002. Epub 2014 Sep 18. PMID: 25306242.

Novak V, Hu K, Desrochers L, Novak P, Caplan L, Lipsitz L, Selim M. Cerebral flow velocities during daily activities depend on blood pressure in patients with chronic ischemic infarctions. Stroke; a journal of cerebral circulation. 2010;41(1):61–66.

https://physionet.org/content/cves/1.0.0/. Last accessed on May 1, 2020

WFDB 2019. https://physionet.org/physiotools/wag/. Last accessed on May 1, 2020

James G, Witten D, Hastie T, Tibshirani R. 2013. An introduction to statistical learning: With applications in R. 2013th ed. New York, NY: Springer

Virani SS, Alonso A, Benjamin EJ, Bittencourt MS, Callaway CW, Carson AP, Chamberlain AM, Chang AR, Cheng S, Delling FN, et al. 2020. Heart disease and stroke statistics-2020 update: A report from the American Heart Association: A report from the American Heart Association. Circulation. 141(9): e139–e596.

Selvaraju RR, Cogswell M, Das A, Vedantam R, Parikh D, Batra D. Grad-CAM: Visual explanations from deep networks via Gradient-based localization. arXiv [cs.CV]. 2016. http://arxiv.org/abs/1610.02391

Tables

Table 1. Characteristic Features of Two Subjects (control vs. stroke)

Features	Subject s0030 (control)	Subject s0175 (stroke)
Time-lag (τ)	1	1
Embedding dimension (m)	16	16
Correlation dimension (C(r))	1.055	1.854
Lyapunov exponent (λ̂)	0.024	0.022
Sample entropy	0.007	0.023
SD1	17.459	40.831
SD2	123.126	99.784
Hurst exponent (H)	0.437	0.859
Spectral index	1.874	0.717
Delta α	0.181	0.515
Delta f(α)	0.909	0.891

Table 2. 2-Day Study Protocol.
In this study, we will consider the first period of walking (walk1, i.e. 11 am to noon on day 1).

Day 1 Protocol									
Hour	7:30	9:00	11:00	12:00	13:00-16:30	17:00	18:00	19:00	22:00
Test	Breakfast	24-hour BP 24-hour ECG,EMG (Me6000) Sit-to-stand (Labview)	Walk 12 min	Lunch	Sit-to-stand test Cognitive tests Beat-to-beat BP monitoring (Portapress)	Walk 12 min	Dinner	Sit-to-stand test	Sleep
Day 2 Protocol									
	Breakfast	Head-up tilt Sit-to-stand eyes open and closed with balance COP measures	MRI	Lunch	Discharge, resume antihypertensive medications				

Figures

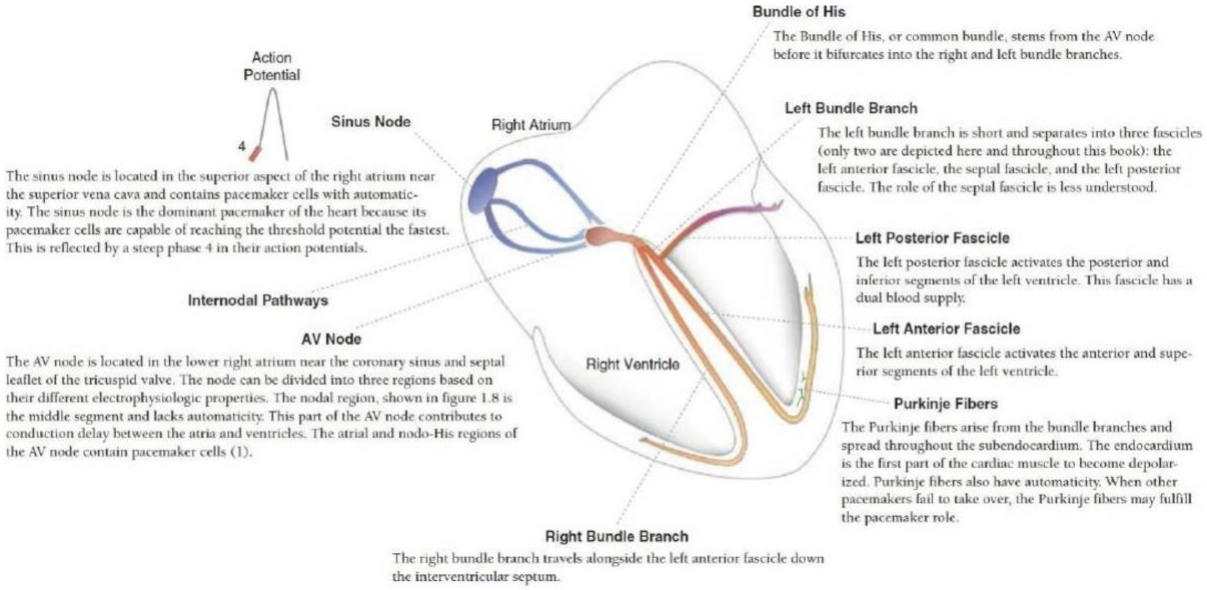

Figure 1. Anatomy of the heart conduction system (Jennifer L. Martindale et.al. A Visual Guide to ECG interpretation, 2016).

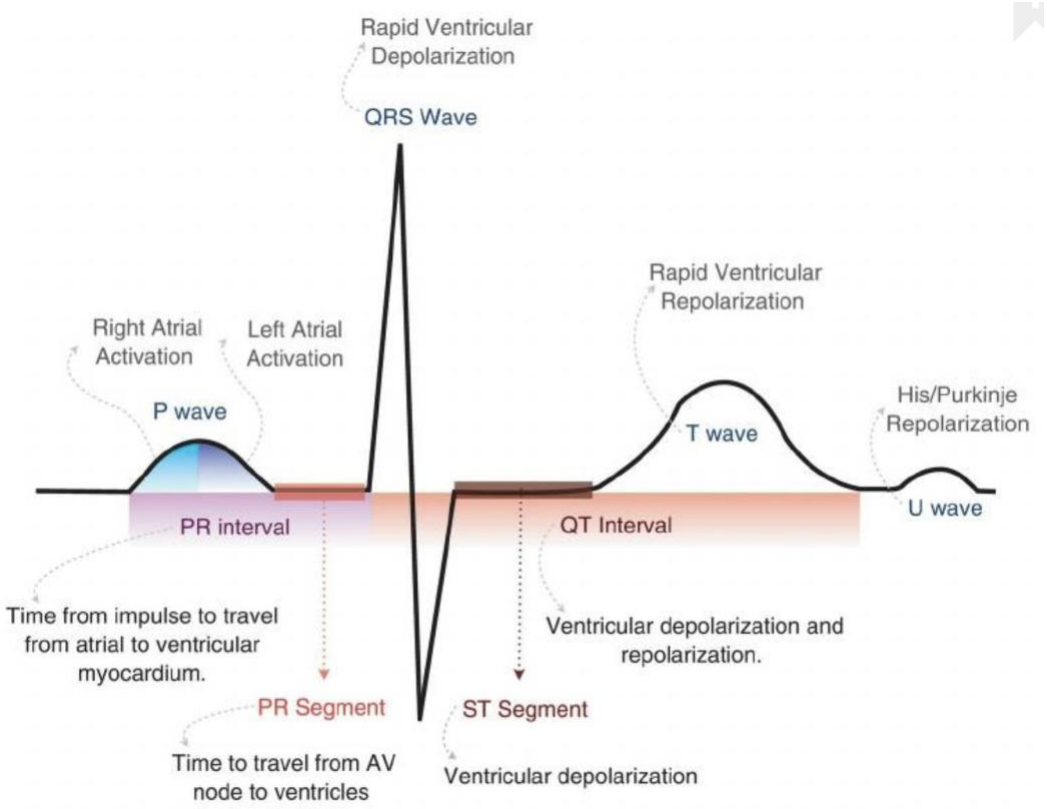

Figure 2. Electrocardiographic waves, intervals, and segments (Jennifer L. Martindale et.al. A Visual Guide to ECG interpretation, 2016).

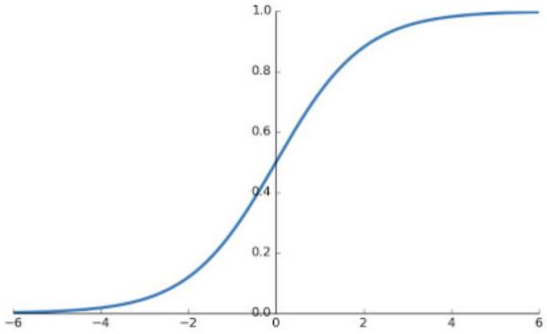

Figure 3. Logistic function plot.

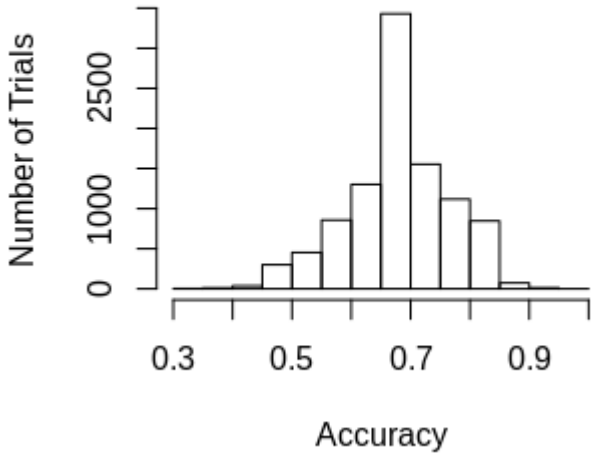

Figure 4. Prediction accuracy of the logistic regression model. In total, 10,000 trials were performed. The average accuracy was 0.678 ± 0.086.

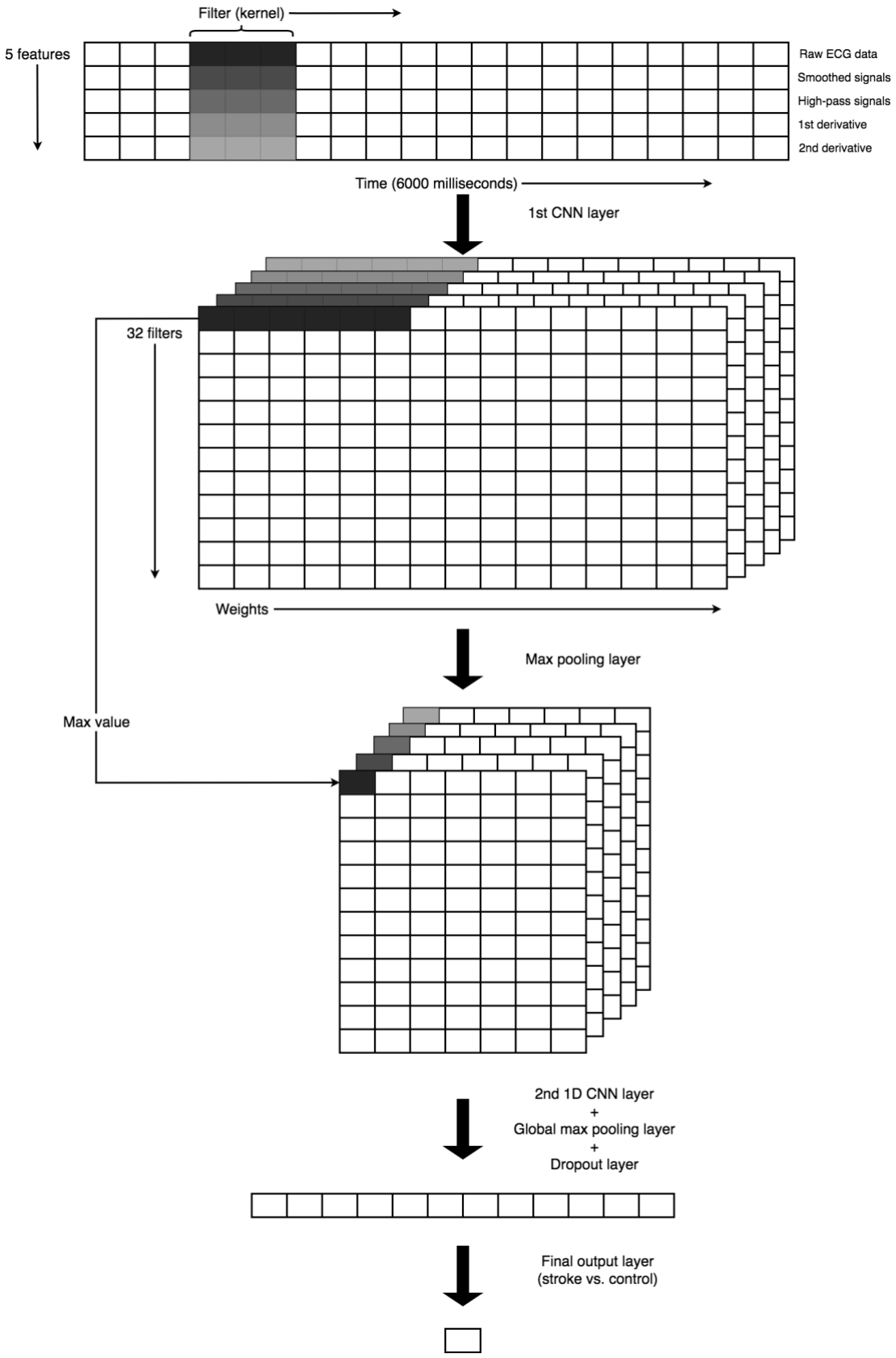

Figure 5. A 1D CNN model example.

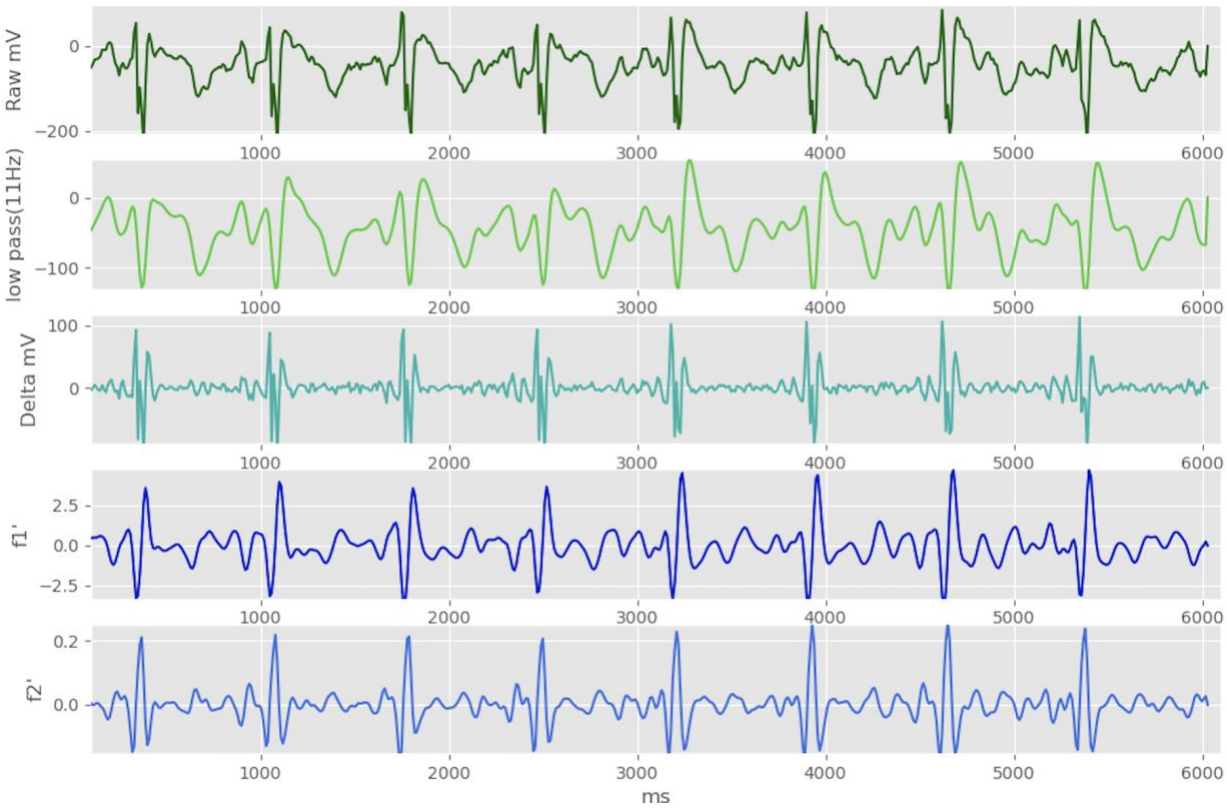

Figure 6. An example of ECG data with features.

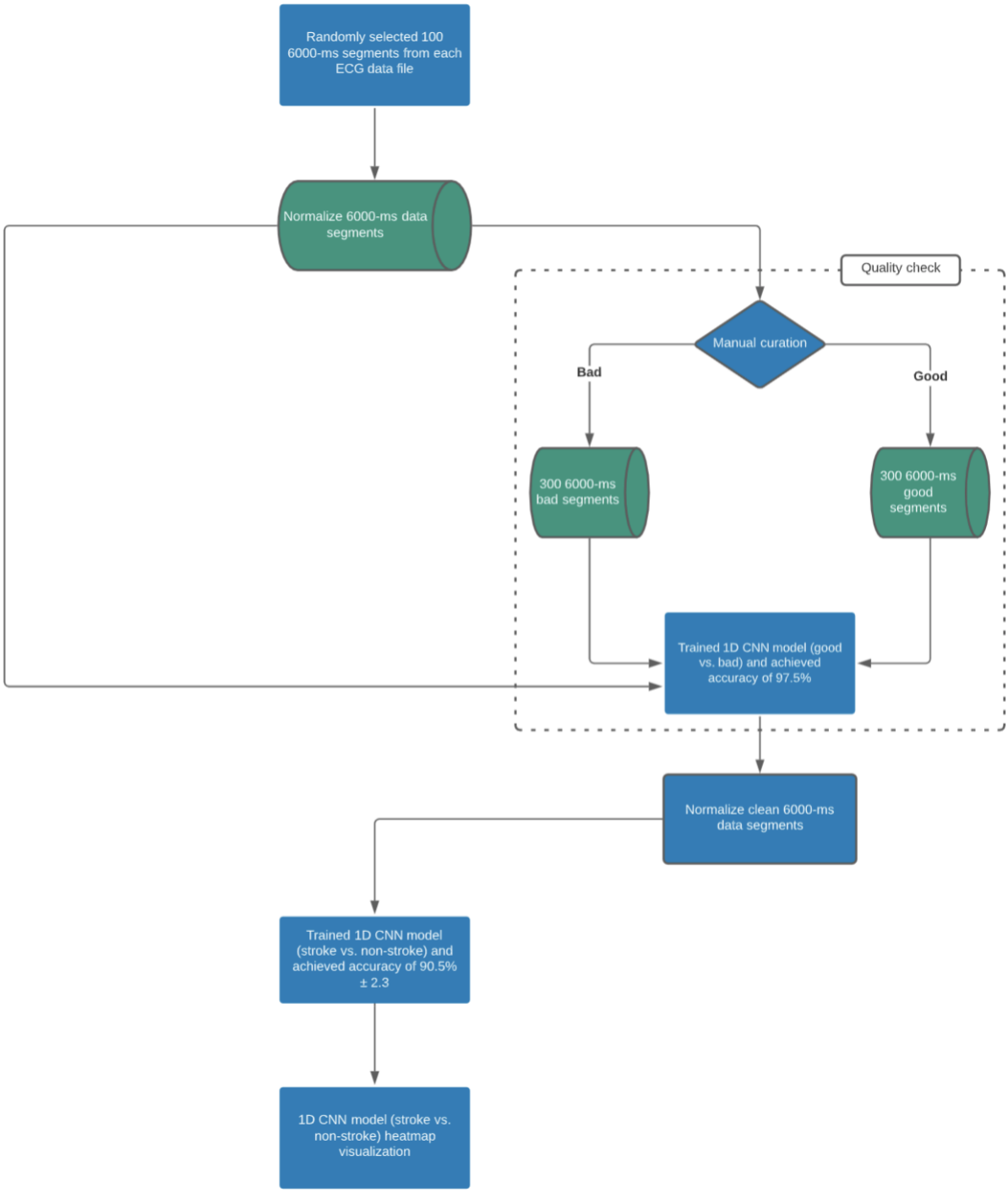

Figure 7. Project roadmap.

```
raw_signal, low_pass_signal, delta, first_derivative_smooth, second_derivative_smooth
0.330183988065639,0.48021145655466546,0.35503928216452074,0.6504286168303892,0.6030791302109901
0.3336648433615117,0.48311586463334955,0.3578569315756494,0.6498335451742602,0.6007033726845861
0.33714569865738436,0.48602762036017233,0.36066819639499187,0.6490542558010082,0.5984736599884105
0.34012928891098954,0.4889381498331415,0.3627174729317774,0.6480950823567408,0.5964059873943621
0.3431128791645947,0.49183884143447665,0.3647752978902134,0.6469617796097589,0.5945140598665068
0.3460964694181999,0.49472110610198955,0.3668491345639774,0.6456614921751938,0.5928091786106606
0.34808552958727,0.4975764302987269 5,0.36742027299800684,0.6442027397050215,0.5913001343381519
0.3510691198408752,0.5003964206221853,0.36954822171661017,0.642595387126501,0.5899931979652928
0.3540472401790154,0.5031728465191018,0.37324013249430954,0.6408505497285292,0.5888922995263561
0.3590253605171556,0.5058976988585915,0.3769768570824853,0.6389803984923221,0.5879993737479589
0.3630034808552959,0.508563272077392,0.38076509107610657,0.6369978841669038,0.5873147812124858
```

Figure 8. A glimpse of the normalized data used for building the two 1D CNN models. Each row represents a record per millisecond.

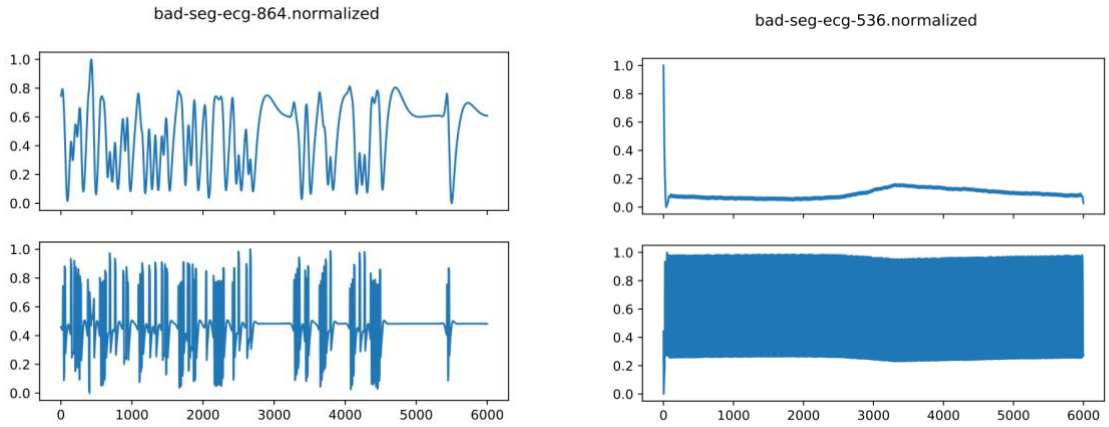

Figure 9. Two samples of poor ECG data segments.

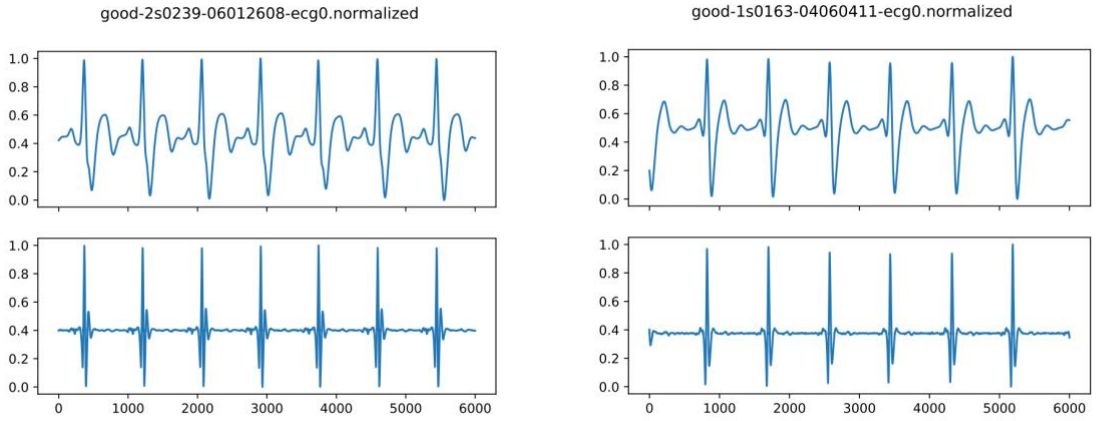

Figure 10. Two samples of good ECG data segments.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d (Conv1D)              (None, 5971, 32)          4832
_________________________________________________________________
max_pooling1d (MaxPooling1D) (None, 398, 32)           0
_________________________________________________________________
conv1d_1 (Conv1D)            (None, 369, 64)           61504
_________________________________________________________________
global_max_pooling1d (Global (None, 64)                0
_________________________________________________________________
dense (Dense)                (None, 1)                 65
=================================================================
Total params: 66,401
Trainable params: 66,401
Non-trainable params: 0
_________________________________________________________________
```

Figure 11. A deep neural network of the quality-checking model.

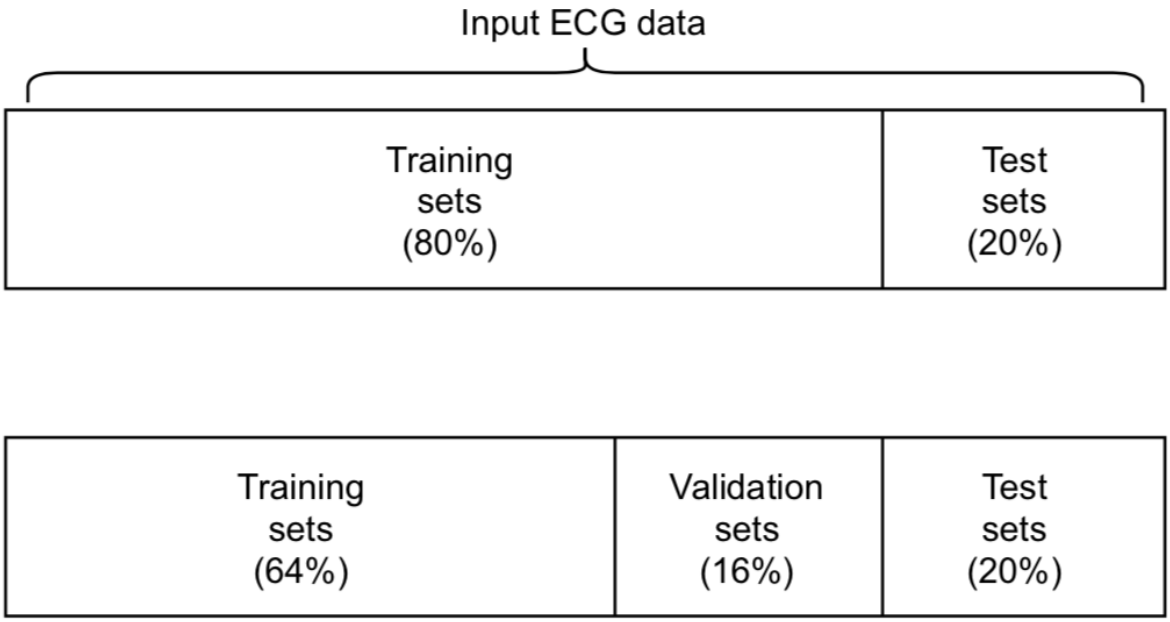

Figure 12. The way to partition an input dataset. The input is first divided into two the training and test sets, with a 80-20 split, and 20% of the training set form the validation set.

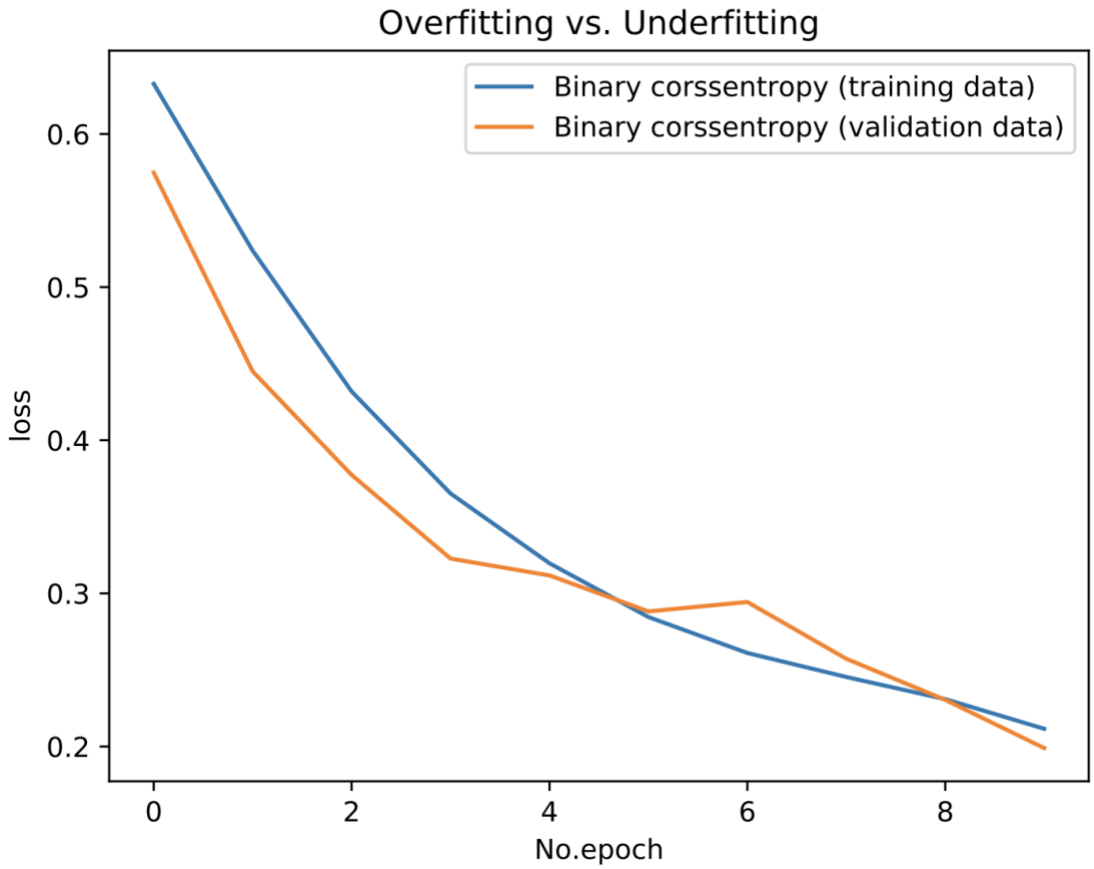

Figure 13. Learning curves of the quality-checking model.

Layer (type)	Output Shape	Param #
conv1d (Conv1D)	(None, 5971, 32)	4832
max_pooling1d (MaxPooling1D)	(None, 398, 32)	0
conv1d_1 (Conv1D)	(None, 369, 64)	61504
global_max_pooling1d (Global	(None, 64)	0
dropout (Dropout)	(None, 64)	0
dense (Dense)	(None, 1)	65

Total params: 66,401
Trainable params: 66,401
Non-trainable params: 0

Figure 14. A deep neural network of the 1D CNN stroke model.

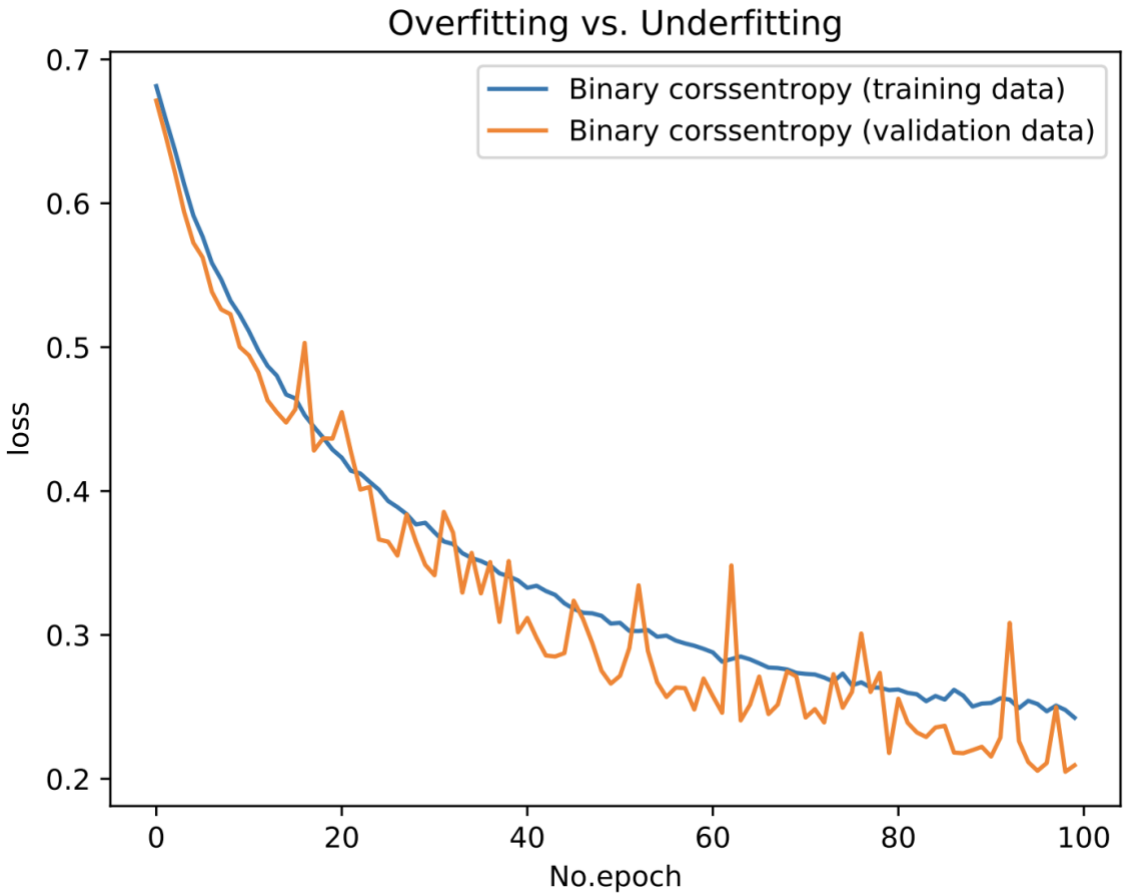

Figure 15. Learning curves of the 1D CNN stroke model.