Inter-Patient Electrocardiogram Heartbeat Classification with 2-D Convolutional Neural Network

by

Kun Ye
B.Sc., University of Victoria, 2018

A Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of

MASTER OF APPLIED SCIENCE

in the Department of Electrical and Computer Engineering

Inter-Patient Electrocardiogram Heartbeat Classification with 2-D Convolutional Neural Network

by

Kun Ye
B.Sc., University of Victoria, 2018

Supervisory Committee

Dr. Xiaodai Dong, Supervisor
(Department of Electrical and Computer Engineering)

Dr. Hong-Chuan Yang, Department Member
(Department of Electrical and Computer Engineering)

Supervisory Committee

Dr. Xiaodai Dong, Supervisor
(Department of Electrical and Computer Engineering)

Dr. Hong-Chuan Yang, Department Member
(Department of Electrical and Computer Engineering)

ABSTRACT

Advanced computer technologies can transform the traditional electrocardiogram (ECG) monitoring system for better efficiency and accuracy. ECG records a heart's electrical activity using electrodes placed on the skin, and it has become an essential tool for arrhythmia detection. The complexity comes from the variety of patients' heartbeats and massive amounts of information for humans to process correctly. The first part of the thesis presents an image based two-dimensional convolution neural network (CNN) to classify the arrhythmia heartbeats with inter-patient paradigm. It includes a new data pre-processing method. The inter-patient paradigm simulates the practical use case of an ECG heartbeat classifier. Compared to the reported work in the literature, the proposed solution achieves superior experiment results. The rest of the thesis introduces the remote ECG monitoring system. The RESTful API design concepts of the system are described. The proposed API supports an efficient and secure way of interaction between each module in this remote monitoring system.

Contents

List of Tables

List of Figures

List of Abbreviations

ANN Artificial Neural Network

API Application Programming Interface

BiLSTM Bidirectional Long Short Term Memory

CNN Convolutional Neural Network

CVD Cardiovascular disease

DAO Data Access Object

ECG Electrocardiogram

GPU Graphics Processing Unit

GUI Graphical User Interface

HZ hertz

HTTP Hypertext Transfer Protocol

ICS Internal Covariate Shift

ILSVRC ImageNet Large Visual Perception Challenge

JSON JavaScript Object Notation

KNN K Nearest Neighbor

LSTM Long Short Term Memory

MLP Multi-layer Perceptron

QA Quality Assurance

RBFNN radial basis function neural network

RR R-peak R-peak

ReLU Rectified Linear Unit

REST Representational State Transfer

SDLC Software Development Life Cycle

SVEB Supraventricular Ectopic Beat

SVM Support Vector Machine

UI User Interface

URI Uniform Resource Identifier

VEB Venticular Ectopic Beat

WHO World Health Organization

ACKNOWLEDGEMENTS

I would like to thank:

My mother, father, and brother, for their love, caring, and sacrifices for my graduate education. It has been a changing two years for me, also the most meaningful. I am genuinely thankful to my parents for their understanding, patience, and continuing support to complete my research works. They help me when I am in low moments. I also express my special thanks to my brother to help and care for my student life in Canada.

Supervisor Dr. Xiaodai Dong, for her patient guidance, enthusiastic encouragement and useful critiques of this research work. She has continually encouraged me to think about this research, and when I have questions, she always answered patiently. Without her support, this research would not be possible. As an undergraduate student who wants to discover more in the medical software engineering world, she gives me the courage and confidence to keep studying and researching.

My colleagues and my friends, for their help and support to my research work in these two years. They provide many insightful research ideas to me and guide me in my graduate studies. It has been an honor for me to work with these colleagues, and I learned much useful knowledge through cooperation works with them. Without their help, I would not complete this thesis smoothly.

Kun Ye
Victoria, BC, Canada
June, 2020

DEDICATION

To my family, my supervisor
for
All the supports that you have given to me

Chapter 1

Introduction

1.1 Overview

Health issues have always been a primary concern of society. In 2017, the world health organization (WHO) listed cardiovascular diseases (CVDs) as the number one cause of death in the world [30]. About 17.9 million people died from CVDs in 2016, which counts for 31% of all deaths in the world of that year. Most of the CVDs happen in countries with low incomes. In these countries, people are usually not covered by the public health care system, and inadequate medical facilities cause hospitals not able to provide proper medical treatments for patients. Practical solutions for reducing deaths caused by CVDs are early detection and proper medical treatments. Usually, physicians diagnose a patient's cardiac problem based on analyzing the patients' ECG signal information. A physician reviews heartbeats' morphology information and rhythms to determine if this patient has abnormal heartbeats [37]. A long-time ECG recording contains a complete patient's ECG information for a long period of time, which is useful for a doctor to precisely diagnose a patient's heart situation. However, the traditional way of diagnosing arrhythmia is relatively inefficient for long-term ECG monitoring. A doctor can not analyze a massive amount of ECG information in a limited time [43]. This brings ideas of developing computer-based arrhythmia classification systems for helping doctors to diagnose abnormal heartbeats. There has been a long history of algorithm based automatic ECG data analysis since 1960s and a large amount of literature has been devoted to this area, ranging from interval determination to beat classifications.

The rapid advancement of machine learning models has enabled their wide use for

speech and face recognition, image identification, illnesses diagnosis, and etc [1]. In this thesis, we develop an effective two-dimensional autonomous convolutional neural network (CNN) arrhythmia classification system that can help physician accelerate the detection of abnormal heartbeats, thus improving early diagnosis rate to help reduce CVDs related deaths.

In order to accurately capture abnormal heartbeats, patients are often required for long-term ECG monitoring. During a long-term ECG monitoring period, ECG sensors record a patient's ECG signal information at different times of the day. Some ECG monitoring can take several days. Moreover, it requires remote ECG monitoring, and it is difficult for people who live far away from cities to get ECG monitoring. The advancement of wireless communication technologies provides real-time data transmission between portable devices and central servers [3]. This makes real-time remote ECG monitoring a feasible method to be applied in clinics. To solve these problems, our team has implemented an efficient remote ECG monitoring system. The software system involves interactions of mobile applications, the central server, and computer clients. We use hypertext transfer protocol (HTTP) request methods to implement all designed interactions. Consequently, we design a robust application programming interface (API) document to define all interface requirements for system modules. Our proposed solution allows physicians to establish long-term ECG remote monitoring for patients efficiently, and this system can be further developed by integrating with our proposed ECG arrhythmia classifier. In this way, it can help physicians diagnosing abnormal heartbeats after ECG monitoring is completed.

1.2 Summary of Contributions

In this thesis, contributions are presented in Chapters 3 and 4, which are summarized below.

This research's main contribution is that we improve classification accuracy on new patients by applying a light-weight two-dimensional convolutional neural network (CNN). Instead of using one-dimensional arrhythmia data, we apply the computer vision approach to classify ECG arrhythmia, which is similar to physicians diagnosing arrhythmia by reading ECG graphs. We design an algorithm to plot ECG images from ECG signals with reduced image sizes to lower processing time. Secondly, we separate the data sources into training and testing sets, each containing different patients' ECG information. This way, we can accurately evaluate the model performance when given new patients' ECG information. We have adopted reliable VGG network concepts for constructing our proposed model. Through various experiments, the hyper-parameters of the model structure are determined. Experiment results show that this model achieves excellent classification accuracy: 98.5% classification accuracy in the SVEB-type heartbeat and 98.4% prediction accuracy in VEB type heartbeat. We compare the proposed model with other arrhythmia classifiers. Our proposed model outperforms most of the compared models based on the same database with the inter-patient paradigm.

A second contribution of this research is the design of the remote ECG monitoring system with the representational state transfer (REST) API design. We introduce the essential module in our system, which is the central server that processes all requests from other modules. Our software development team has designed and implemented a REST-style framework. Specifically, we have built a login section, a nurse section, a patient section, an ECG test section, and an ECG raw data section. In each section, we define working logic and requirements for sending and receive hypertext transfer protocol (HTTP) requests. My contribution is to help the server RESTful API concept design. This REST framework can be further developed as an open-source framework that can be utilized by other ECG remote monitoring systems.

1.3 Organizations

Chapter 2 first introduces the neural network concept and structure and then describes the convolutional neural network's fundamental knowledge with a detailed example.

Chapter 3 describes current solutions and challenges in the electrocardiogram (ECG) arrhythmia classification. After the existing solutions are discussed, we compare current paradigms for creating training and testing subsets. Additionally, we discuss database information with data set partition strategies adopted by this research. ECG signals are converted to images for input to the subsequent neural network by a designed pre-processing procedure. We then present our proposed two-dimensional convolutional neural network (CNN) architecture by describing model layers and parameters. The experiment results are compared with other approaches. Finally, we summarize the proposed method's advancement and potential improvements.

Chapter 4 focuses on the ECG remote monitoring system. We introduce our ECG monitoring system, and explain the workflow in our system. The structure of our central server and the interactions among modules in the system. We then introduce the representational state transfer (REST) concept and explain the advantages of applying it in our system. The REST application programming interface (API) designs are provided with each API's functionality and design concept. Finally, we discuss the potential challenges in our software development process with future development plans.

Chapter 5 concludes this research work and discusses future research directions.

Chapter 2

Neural Network Overview

An artificial neural network (ANN) is a computational model inspired by how biological neural networks in the human brain process information, where neurons compute output values from inputs [33]. ANN models learn by studying training data. Typically, a neural network will contain an input layer, one or more hidden layers, and an output layer.

2.1 Perceptrons

In a neural network, an perceptron, also known as artificial neuron, is the fundamental calculating entity that computes several inputs using weighted sum. The sum is then compared with a threshold value to produce the output. The output can be a binary value or a continuous value. A typical perceptron includes input values, weights, net sum, and activation function. A perceptron works on these steps:

1. Input $X = (x_1, x_2, ..., x_n | x_i \in \mathbb{R})$.
2. Each value in the input X multiplies the corresponding weight in W where $W = (w_1, w_2, ..., w_n | w_i \in \mathbb{R})$.
3. Add all the multiplied values to obtain the weighted sum

$$\text{Result} = \sum_{i=1}^{n} w_i x_i$$

.

4. Input the result into the activation function and obtain the output, e.g., in a binary classification problem,

$$\text{Output} = \begin{cases} 0 & \sum_i w_i x_i \leq \text{ Threshold} \\ 1 & \sum_i w_i x_i > \text{ Threshold} \end{cases}$$

2.1.1 Example of Perceptron

Fig. 2.1 shows the structure of a simple perceptron.

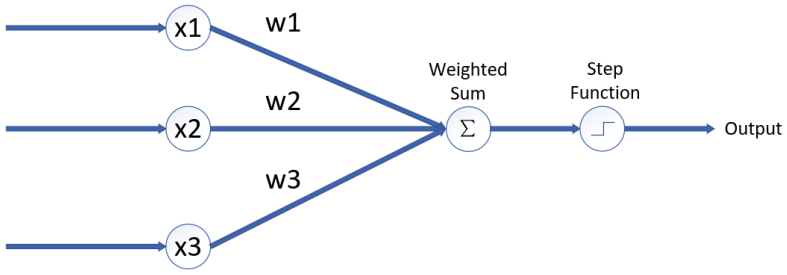

Figure 2.1: An example of a simple perceptron.

For example, let $X = (2, 1, 2), W = (0.25, 0.5, 0.1)$, and the threshold equals to one, we have:

$$\text{Sum} = \sum_{i=1}^{3} w_i x_i = 2 * 0.25 + 1 * 0.5 + 2 * 0.1 = 1.2$$

$$\text{Since } 1.2 > 1, \text{the output is one.}$$

2.2 Multilayer Perceptron

After understanding what is a perceptron, we can start to learn about multi-layer perceptron (MLP) [62]. The MLP is a type of artificial neural network (ANN), and it is explained with the example of classification digits.

2.2.1 Using Multi-Layer Perception to Classify Hand Recognize Digit

Each node in the output layer outputs a vector the size of total number of classes. For example, for classifying handwritten integer numbers ranging from 0 to 9, the output for an image i can be [0.05, 0.05, 0.6, 0.09, 0.01, 0.07, 0,03, 0.03, 0.06, 0.01]. Since the third element (0.6) in the sample vector is the largest, the image is classified to the number 2 and the label that corresponds to this image i is [0, 0, 1, 0, 0, 0, 0, 0, 0, 0]. Ideally, we want the output as close to the label as possible. The model performance is measured based on errors calculated using cross-entropy. Here is the cross-entropy formula for the distributions p and q over a given set:

$$H(p, q) = -\sum_{x} p(x) \log(q(x)),$$

where p is the expected output probability and q is the actual output probability. In our neural network, the error (cost) function of classifying one image is:

$$H(A, B) = -\sum_{i=1}^{n} A_i \log(B_i),$$

where B is the predicted label of the input image, and A is the actual label of the input image. The goal of training this ANN model is then to minimize the cost of classifying each image in the training set.

2.2.2 Neural Network Architecture

Before constructing a neural network, we can only know the number of input training features and samples and output classes. However, the number of neurons in hidden layers are unknown, thus requiring much adjustments in the training process. For example, we know there are m training samples for classifying handwritten numbers

and each of them is a 28×28 pixels gray-scale image. The goal is to identify each picture from a category of integer numbers from 0 to 9. Therefore, this is a classification problem with ten different classes.

Although determining the number of neurons and hidden layers of the neural network is mainly based on experiences, the number of neurons should be consistent with dimensions of input and output data. For example, an input image of 28×28 pixels can be converted to a one-dimensional column vector with 784 pixels, which corresponds to 784 features. If m images are fed in simultaneously, it is equivalent to have an input of size $784 \times m$ and Fig. 2.2 shows the neural network's structure.

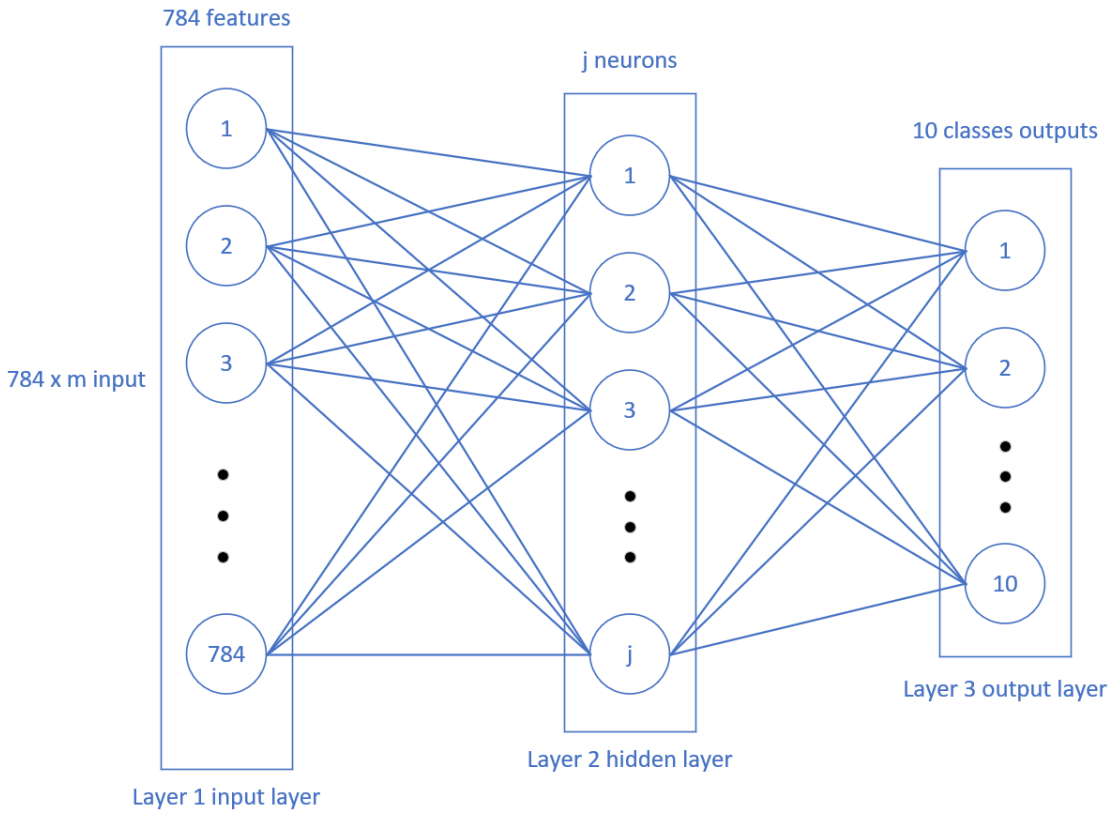

Figure 2.2: An overview of the neural network's structure.

2.3 Convolution Neural Network

The convolution neural network (CNN) consists of a sequence of layers [10]. Specifically, it includes convolutional layers, pooling layers, and fully connected layers. The majority of layers are convolution layers that execute convolutional mathematical

operations. In order to understand this neural network, it is essential to understand the concepts of CNN.

2.3.1 Kernels of Convolutional Layers

A convolutional layer contains a group of kernels (filters). These kernels are two-dimensional matrices that include specific integers, and Fig. 2.3 is an example of an input image and a filter. The left input image contains pixel values, and the right image is a filter with random weights.

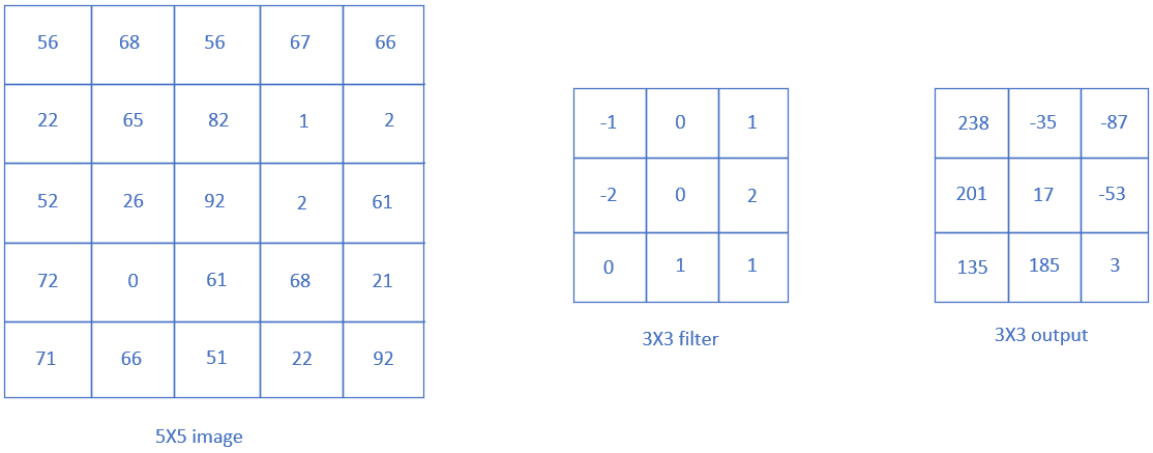

Figure 2.3: An example of an input image and a filter.

In a convolution operation, we put the filter on the left-top of the image, and we multiply values of filter cells with the corresponding pixel value in the input image. These steps are repeated for all input image pixel values and all multiplied values are summed to obtain the final output.

Usually, the output matrix should be the same size as the input matrix. To keep the identical size, zeros are added around the input image to increase the input size. In this way, an output matrix is kept the same size as an input matrix without changing any information in an input matrix.

2.3.2 Pooling

A pixel value in the input image tends to have a similar value to its neighboring pixels. This feature can cause a cell in a convolutional layer output being similar to its neighboring pixels, which means the output contains redundant information. This

redundancy makes critical feature extraction from an input image difficult. Therefore, pooling layers are applied to solve this problem. A pooling layer extracts feature value from a group of cells repeatedly, which are either the max value or the average value. By doing so, an input matrix size is reduced, which helps a model extract critical information from input images. Fig 2.4 shows an example of the max pooling and average pooling operations.

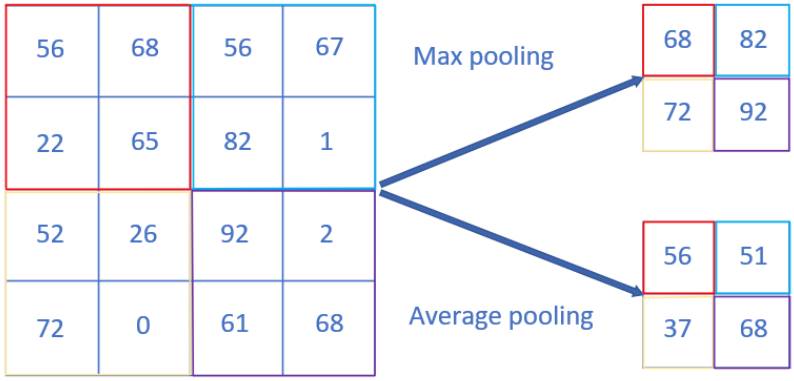

Figure 2.4: Examples of a max pooling operation and an average pooling operation.

2.3.3 Fully Connected Layer

Fully connected layers are the last part of CNN. In this section, the output matrix that comes from the pooling layer is flattened to a one-dimensional vector and used as input for the fully connected layers. The fully connected layers work the same as the multi-layer perceptron (MLP). The processed input is put into MLP and classified into a particular class. After combining all the layers, we can obtain a complete CNN, which is shown in Fig. 2.5 for the complete structure of a CNN.

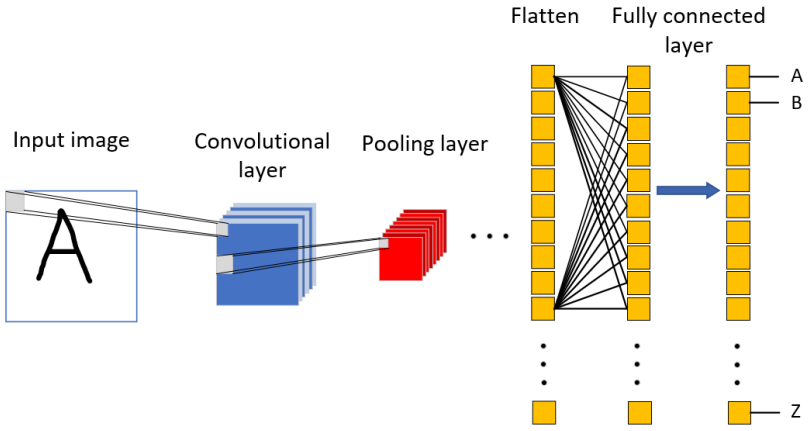

Figure 2.5: An overview of a complete CNN's structure.

Chapter 3

Deep Convolutional Neural Networks in ECG Arrhythmia Beat Classification

3.1 Introduction

In recent years, the quickened advancement in deep learning to solve various medical science problems is providing unprecedented assistance to medical field professionals [32]. There are a wide variety of potential applications to healthcare in terms of disease diagnostics, early detection, monitoring, etc. The machine learning technologies are currently adopted in image recognition, natural language processing, and self-driving automobile [33]. In general, a neural network is efficient in solving tasks with a massive amount of training data [34]. In the electrocardiogram (ECG) heartbeat recognition field [35], the classical way of monitoring a patient's heartbeat is by analyzing an ECG signal's morphological information manually. However, this approach is time-consuming and experience-based. When there are a large amount of ECG recordings, automatic data analysis through signal processing algorithms becomes the standard operation. The medical device industry has long utilized software to classify ECG data, the results of which reviewed and confirmed by cardiologists. To improve the accuracy of automatic classification has been continuously studied in the literature [19]-[28]. In recent years, the advancement of machine learning and deep learning has attracted significant attention in the field to achieve classification accuracy comparable to that of experts.

ECG is mainly used for cardiac abnormality identification [36]. As shown in Fig. 3.1, a typical ECG signal consists of three primary waves: P wave, QRS complex, and T wave [37]. An arrhythmia heartbeat is incurred by an abnormal heart which is usually caused by abnormal impulse information or transmission. By reading ECG information, a physician can diagnose a variety of arrhythmia heartbeats. Physicians make judgments based on the interval and morphological information of an ECG signal, such as the shape of these three original waves and the heartbeat's rhythm [38].

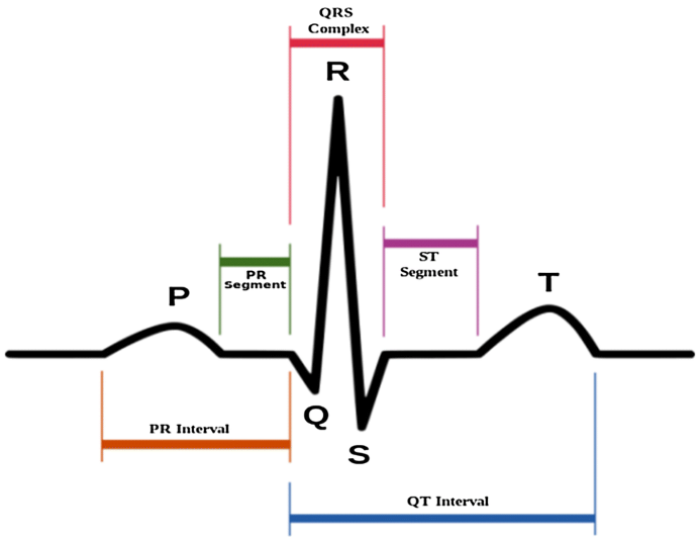

Figure 3.1: An ECG signal illustration [57].

In general, an ECG arrhythmia can be categorized as hazardous and the non-dangerous type. In order to detect hazardous arrhythmia heartbeats, a long-term ECG recording is required. However, it is relatively hard for a doctor to observe and analyze all the morphological information from long-term ECG records in limited amount of time. If a dangerous arrhythmia is detected, proper treatments needs to be applied immediately and any delay can negatively affect a patient's cardiac health. Therefore it is essential to establish an efficient arrhythmia detection solution for fast-paced arrhythmia heartbeat identification. Along with the development of portable sensor devices, many portable ECG devices are provided to patients [39]-[41].

Portable ECG recorders can help the clinics obtain efficient cardiac monitoring [42]. However, a physician needs to diagnose several patients simultaneously for an extended period. It is an impossible task for a physician to analyze the morphological information of a massive amount of patients' heartbeats in a limited time. This high

demand for quick arrhythmia heartbeat diagnosing can be fulfilled by computer-based ECG arrhythmia heartbeats diagnosis systems. However, there are several challenges in automatic ECG signal diagnosing because each patient has different morphological and temporal characteristics of ECG heartbeats. Therefore, it is relatively hard to define precise rules that can define all the arrhythmia types for all patients, and a patient's heartbeats have various morphological shapes when this patient participates in activities, such as exercising, relaxing, and sleeping. All these factors about patients' ECG morphological information uncertainty lead to automatic ECG classification is hard to achieve satisfying classification results. Currently, there are many research studies on computer-based ECG arrhythmia classification, such as RR interval-based classification system [43], SVM based classification system [44], ANN-based classification system [48], KNN-based classification system [45], swarm optimization with radial basis classification system [46], and conditional random fields classification system [47].

The most crucial part of ECG classification is feature extraction. Reported works introduce different approaches for feature extraction and feed these feature information into their proposed models. However, feature extractions can not obtain all the information of all patients' heartbeats. Since patients have different heartbeat shapes, existing models have relatively low classification accuracy for classifying a new patient. We develop a two-dimensional CNN abnormal heartbeats classification system to solve these challenging issues for automatic ECG arrhythmia detection. CNN is the most popular type of deep learning approach for image classification, and many ECG classification algorithms are based on one-dimensional CNN [27][49][50].

In this chapter, we construct a two-dimensional CNN system for ECG abnormal heartbeats classification. We use the MIT-BIH arrhythmia database [9] to evaluate the proposed model's performance. We first convert ECG signals to ECG heartbeat images; then, we feed the segmented heartbeat images into the proposed 2D CNN model for training and testing. In the experiments, detailed results are obtained to prove that our approaches for the proposed solution are effective, and we compare the proposed model with other reported works to evaluate this model's performance. This solution can be further developed and implemented in the remote ECG system to monitor many patients simultaneously.

The rest of this paper is organized as follows. Section 3.2 introduces reported works on automatic ECG arrhythmia classifications. Next, Section 3.3 explains partition paradigms, data labeling, database information, data pre-processing processes,

and detailed information about the proposed CNN model. Subsequently, Section 3.4 shows evaluation and validation progress for approaches and provides experiment results. Lastly, Section 3.5 presents the conclusion and discusses future research plans for the proposed model.

3.2 Related Work

The ECG waveform reflects a heart's electrical activity, and it is used for various heart conditions' detection. In long-term ECG monitoring, accurate ECG signals play an essential role in diagnosing a patient's present cardiac abnormality. With the development of algorithms in machine learning, many researchers focus on developing advanced machine learning algorithms to detect ECG abnormal heartbeats automatically. An review of the ECG arrhythmia classification is summarized next.

An effective linear discriminant classification system for identifying abnormal beats is reported in [43]. The authors obtained RR interval information by applying feature extraction techniques, and they use wavelet analysis and linear prediction modeling to extract morphological features. After that, the extracted features are combined with a discriminant classifier to classify arrhythmia heartbeats. The model is evaluated against the MIT-BIH arrhythmia database [9]. Based on the experiment results, the authors argue that the combination of wavelet and linear prediction features can improve the proposed model's classification accuracy.

A robust evidential k-nearest neighbors algorithm is presented in [45]. The authors followed the concept of Dempster Shafer Theory for classifying ECG irregular heartbeats [13]. The RR interval features are captured and fed into the proposed algorithm. The model was evaluated against the MIT-BIH arrhythmia database, and the model was compared with the traditional KNN method. Considering the error rates, the author argued that the proposed system outperforms the original KNN based classification system.

An effective SVM classifier is introduced in [51]. In this solution, the authors first detect and segment the QRS complexes. They then collect the frequency information, RR intervals information, and QRS information to characterize each beat. These features are fed into the SVM for classification. In the proposed model, the decision rule consists of dynamic reject thresholds with the cost of misclassifying or rejecting a sample. The model has a significant performance improvement when the model is evaluated in the MIT-BIH arrhythmia database. They obtain an average accuracy of

97.2% with no rejection.

The particle swarm optimization and radial basis function neural network were presented in [46]. The authors extracted four morphological features for each heartbeat. In the proposed model, the RBFNN structure with particle swarm optimization was used for the extracted features. They use the MIT-BIH arrhythmia database to test this model's performance. After several experiments, the proposed model obtains a relatively high classification performance, and the model's performance can be increased by applying additional feature extraction methods.

A useful one-dimensional CNN model for 17 classes of cardiac arrhythmia detection is presented in [49]. The proposed solution is based on extracting features from 10 second ECG signal fragments. The authors develop a specialized end-to-end structure for feature extraction instead of using classical segmentation methods. The proposed model is a one-dimensional CNN model, and the model's performance is evaluated in the MIT-BIH arrhythmia database. This solution is efficient and quick in the task of classifying the various classes of ECG arrhythmia. Also, the model's structure is straightforward, and the implementation of the solution is relatively simple. This model achieves an overall accuracy of 91.33% for 17 cardiac arrhythmia classes with a relatively short 0.015s classification time per single sample.

An effective generative adversarial network is presented in [52]. In the proposed model, the authors design a generator section and a discriminator section. The generator contains several layers of a bidirectional long short-term memory (LSTM) network, and the discriminator is the structure of CNN. The proposed model is trained and tested by the MIT-BIH arrhythmia database. The model's performance is evaluated by comparing with recurrent neural network autoencoder and the recurrent neural network variational autoencoder. The experiment results show that this model's loss function has the fastest speed for converging to zero, and the BiLSTM-CNN generative adversarial network can generate the ECG data that is morphologically similar to real ECG data.

A powerful 16-layer one-dimensional CNN for classifying atrial fibrillation is presented in [53]. The proposed model is specifically designed for detecting atrial fibrillation. In this model, the skip connections technique improves the CNN model's feature learning capabilities and reduces the training time. The model is trained by 8528 ECG samples and tested by 3685 ECG samples, and each sample has a range from nine seconds to sixty seconds. The authors also implement RNN and spectrogram learning to compare with this model. After several experiments, the model

achieves 90% accuracy for identifying normal rhythm, 82% for identifying atrial fibrillation, and 75% for identifying other rhythms. The authors argue that this model can help diagnose the patient's heartbeats in real-time.

A relevant CNN that can identify arrhythmia based on different intervals of tachycardia ECG samples is introduced in [54]. The authors develop a computer-aided diagnosis system based on CNN, and the proposed model consists of an 11-layer CNN with an output layer that contains four neurons. The authors use ECG samples from range two seconds and five seconds without any QRS detection. The proposed model is evaluated in several databases, and this model achieves accuracy 92.50%, sensitivity 98.09%, and specificity 93.13% for two-second samples. Also, the model obtains relatively high accuracy for five-second samples. The authors argue that the proposed solution can be a useful tool to help clinicians for diagnosing the patient's abnormal heartbeats.

A useful attention-based time-incremental CNN is presented in [55]. The authors integrate CNN with recurrent cells and attention modules to combine the spatial and temporal information from ECG signals. This approach optimizes features input length and reduces a significant amount of parameters. The authors argue that this method reduces 90% of computation in real-time processing comparing with the original CNN model. The proposed model is evaluated in several data sources. The experiment results show that this model achieves an overall accuracy of 81.2%. Also, they compare this model with the original VGG network. This model's average accuracy is 7.7% higher than the VGG model, and the paroxysmal arrhythmias classifying accuracy is 26.8% higher than the VGG model. The authors argue that the proposed solution is a concrete example of all different length signal processing problems.

An efficient two-dimensional CNN for classifying ECG arrhythmia with information fusion and one-hot encoding techniques is presented in [56]. The authors combine the morphology information and rhythm information of heartbeats into a two-dimension vector, and the processed vectors are fed into the proposed CNN that contains specialized learning rate and dropout methods. The proposed model is evaluated in the MIT-BIH arrhythmia database, and this model has a better performance than other compared methods for classifying five and eight heartbeat categories. The proposed model has better performance in terms of the sensitivity and positive predictive rate for V-type beats and S-types beats than other solutions. The author argues that the proposed system is useful for classifying abnormal heartbeat, and it is an effective system that can be implemented on mobile devices for monitoring heart

health situations.

However, all these reported solutions do not perform well in some particular scenarios. Most of the solutions have inconsistent performance when classifying a new patient's ECG signals. In other words, if the patient's ECG heartbeats are not in the training set, these models obtain a decreased classification accuracy for this patient. The majority of models are tested on MIT-BIH arrhythmia database. However, many of the models do not follow the recommendation of AAMI [8] for ECG sample labeling. The ECG signal noise reduction process may lose important information about the patient's heartbeat and increasing the data pre-processing work. In this thesis, we convert the ECG signal classification task into a computer vision problem. We propose a two dimensional CNN system to solve the challenging issues not solved in the existing research works.

3.3 Methods

3.3.1 Paradigms

Many research studies dedicate to automatic ECG arrhythmia classification. Among all these studies, the intra-patient and inter-patient paradigms are the two mainly used data partition schemes.

- The intra-patient partitions methods randomly mix all the patients' heartbeats into a complete set and split it into a training subset and testing subset. Therefore, a patient's heartbeats can be in the training set and testing set simultaneously [5]. By applying this partitioning method, the model can obtain optimistic results. However, in the clinic, the classifier usually needs to predict a new patient's heartbeats. Therefore, this partition scheme can not evaluate the model's real classification accuracy for classifying a new patient's heartbeats.

- The inter-patient partitions method provides a more practical way of building the training subset and testing subset. Philip de Chazal proposes the inter-patient paradigm [7]. In this scheme, the training set comes from one group of patients' records, and the testing set comes from another group of patients' records. Therefore, it avoids the scenario that a patient's ECG information exists in training and testing subsets. The intra-patient method usually achieves a better result than the inter-patient method because the training and testing data can come from the same patient; therefore, the intra-patient scheme leads to model overfitting in the practical application scenarios.

After studying on other researchers' current achievements [50]-[52], we find that most reported works applied the intra-patient paradigm in their ECG classification systems. We adopt the inter-patient scheme as it is more suitable for real-life diagnosing scenarios. Although it is challenging to achieve high results, the experiment results are more reliable and meaningful because this paradigm is similar to when a physician diagnoses arrhythmia for a new patient.

3.3.2 Advancement of Medical Instrumentation Standards

A patient's heartbeats are categorized into five classes that are defined in the Advancement of Medical Instrumentation (AAMI) standard (IEC 60601-2-47:2012): normal

(N), supraventricular ectopic beat (SVEB), ventricular ectopic beat (VEB), fusion beat (F), and unknown beat (Q) [8]. Details of these classes are shown in Table 3.1. The standard recommends the training set labeled as DS1 (consisting of patients' record 101, 106, 108, 109, 112, 114, 115, 116, 118, 119, 122, 124, 201, 203, 205, 207, 208, 209, 215, 220, 223, and 230) and the testing set labeled as DS2 (consisting of record numbers 100, 103, 105, 111, 113, 117, 121, 123, 200, 202, 210, 212, 213, 214, 219, 221, 222, 228, 231, 232, 233, and 234). The standard also emphasizes that SVEB and VEB are the two most critical arrhythmia categories. For a given classification algorithm, the AAMI outlines the necessity to use a performance matrix that reveals classification performances for each of these four classes: N, SVEB, VEB, and F. We ignore the classification performances for the Q class because this class contains a few samples only, and classification results of this class are not suitable for evaluating proposed model's performance. We evaluate the model's performance using SVEB and VEB heartbeat classification results in the test. We also provide a confusion matrix to show the models' performance on classifying each of these four classes.

Class	Symbol	Members
Normal	N	Normal beat Left bundle branch block beat Right bundle branch block beat Atrial escape beat Nodal (junctional) escape beat
Supraventricular Ectopic Beat	SVEB	Atrial premature beat Aberrated atrial premature beat Nodal (junctional) premature beat Supraventricular premature or ectopic beat (atrial or nodal)
Venticular Ectopic Beat	VEB	Premature ventricular contraction Ventricular escape beat
Fusion beat	F	Fusion of ventricular and normal beat
Unknown beat	Q	Paced beat Fusion of paced and normal beat Unclassifiable beat

Table 3.1: Advancement of Medical Instrumentation recommended classes

3.3.3 Database Information

In the thesis, the MIT-BIH arrhythmia database is used to evaluate the proposed model [9]. The MIT-BIH arrhythmia database contains 48 recordings of two-channel ambulatory ECG. These records are obtained from 47 subjects studied by the BIH ar-

rhythmia laboratory between 1975 and 1979. Digitized ECG signals in the recordings are 360 Hz per channel with an 11-bit resolution over a 10 mV range. In total, there are approximately 110,000 beats in this database. There is an annotation record for each ECG record that contains QRS positions and the heartbeats' types, verified by at least two cardiologists. Therefore, the QRS detection process is applied to this database. These heartbeats are labeled along with their R peaks. A summary of the MIT-BIH arrhythmia database is listed in Table 3.2.

Group name	Labels	Number of beats	DS1	DS2
Normal	N	90042	45824	44218
Ventricular Ectopic	VEB	7007	3788	3219
Supraventricular Ectopic	SVEB	2779	943	1836
Fusion	F	802	414	388
	Total:	100630	50969	49661

Table 3.2: Summary of MIT-BIH arrhythmia database

3.3.4 ECG Data Pre-Processing

The two-dimensional CNN requires image inputs. We convert ECG signals into ECG images by plotting each ECG heartbeat as an individual 150 x 150 pixels gray-scale image. In the MIT-BIH arrhythmia database, every ECG beat is sliced based on R-R interval information, which is the time between QRS complexes. More specifically, the ECG heartbeat is labeled along with the heartbeat's R peak time. Thus, we determine a single ECG heartbeat by centering the heartbeat's R peak while excluding each record's first heartbeat and last heartbeat. These two heartbeats are not complete because of the ECG monitoring restriction. We also define each image only containing one heartbeat with this heartbeat label as the image's name. The data files from the MIT-BIH arrhythmia database are time-series data. So we apply data pre-processing methods for segmenting and converting the ECG signals to individual ECG heartbeat images.

Extracting individual ECG heartbeat from a time-series data

In ECG data files, every heartbeat is extracted based on the distance between the target heartbeat's R-peak and its adjacent heartbeats' R-peaks. Since each annota-

tion is located near each R-peak in the ECG data annotation file, we can easily find R-peaks of all heartbeats in the MIT-BIH database. The target heartbeat's starting point is the middle point between the target heartbeat's R-peak with the previous heartbeat's R-peak. The target heartbeat's ending point is the middle point between the target heartbeat's R-peak with the next heartbeat's R-peak. We store all the values between the starting pointing and ending point into an array. Here is the formula for calculating the starting and ending points:

$$\text{Starting point: } (\text{Rpeak}(n) - \text{Rpeak}(n-1))/2 + \text{Rpeak}(n-1)$$
$$\text{Ending point: } (\text{Rpeak}(n+1) - \text{Rpeak}(n))/2 + \text{Rpeak}(n)$$

Fig. 3.2 is the visual representation of this segmentation process. In this way, we can keep the heartbeat's R interval information. The R interval-based segmentation is also relatively simple to implement and it saves pre-processing time and computation resources.

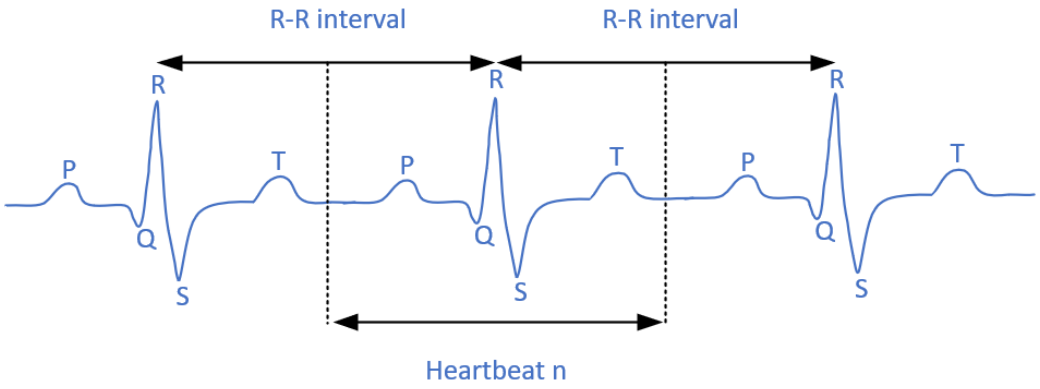

Figure 3.2: Segmenting a heartbeat from a series of ECG signal.

Plotting ECG heartbeat image

After obtaining all processed array objects, we use Python [58] to convert ECG signals into heartbeat images. We also find it essential to properly set a maximum x-axis value to plot the image through experiments. The image plotting function automatically fits the heartbeat's shape into the entire image, making all heartbeats appear to be the same length. This mechanism leads to heartbeat-length-information loss during the feature extraction process because it changes a heartbeat's R interval. In real life, the heartbeat's length is not the same and everyone has a different heartbeat shape.

For this reason, it is essential to set up the proper maximum x-axis value for each patient's ECG image. If it is too large, the heartbeat shape is hard to identify. If it is too small, the image does not cover the entire heartbeat interval. We develop a formula that can calculate the maximum value for each record, which provides the best model performance, as

$$S = A + A * 0.3$$
$$A = \frac{1}{n} * \sum_{i=1}^{n} length(i)$$

where n is the number of heartbeats of a patient's ECG record, $length(i)$ is the number of samples of each heartbeat, A is the average number of samples of a patients' record, and S is the maximum x-axis value for plotting heartbeats of a patients' record.

After feeding the patient's record to the proposed function, we can obtain the maximum x-axis value to plot this patient's ECG image. Fig. 3.3 shows the complete process of calculating the maximum value and applying it to plot the ECG heartbeat image.

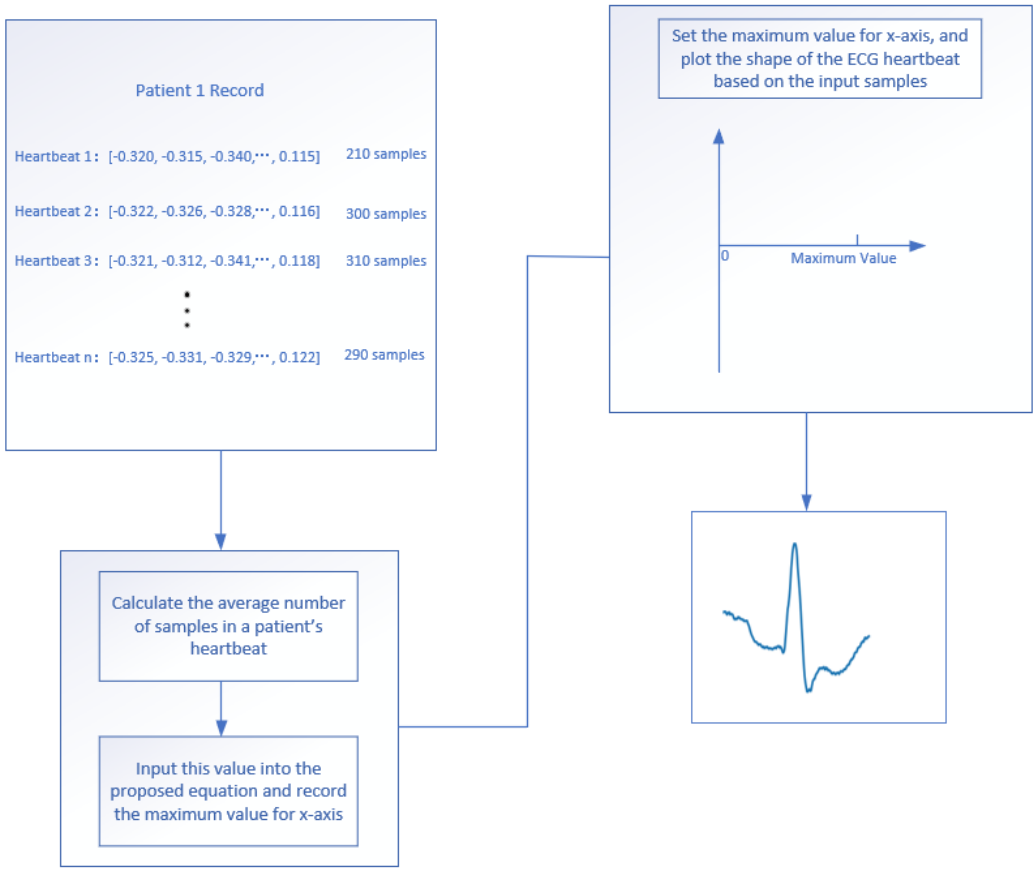

Figure 3.3: The processes of plotting ECG heartbeat images.

Augmenting the ECG heartbeat image

The matplotlib function [59] outputs ECG heartbeat images, and the default image resolution is 600 x 400 pixels, such as Fig. 3.4. However, due to computer memory limit, images need to be compressed. Since we only focus on the ECG heartbeat's morphological information, we can use gray-scale images. By converting colored images to gray-scale images, we can reduce the model's parameters and improve the model's training efficiency. The processed image resolution is 150 x 150 pixels, which is a massive drop-off for image resolution and the image's quality is relatively low as shown in Fig. 3.5. The reason is that the primary pixels representing the heartbeat shapes are also compressed which causes the compressed image to lose important information. We can see that the ECG heartbeat's shape is hard to recognize and the low-quality input images would lead to a significant drop in the model's classification accuracy. To solve this technical issue, we implement a method that can augment

the picture's heartbeat shape. The idea is to emphasize the shape of the heartbeat while compressing an ECG heartbeat image. When we convert heartbeat signals to images, we add one extra parameter that makes the plotted line thicker. Fig. 3.6 shows the augmented heartbeat image after the compressing process and we can see that the augmented image keeps most of the ECG heartbeat's shape. We can then obtain most of the ECG heartbeat's original information while reducing image size.

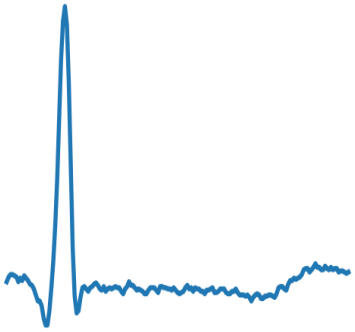

Figure 3.4: An original image.

Figure 3.5: A compressed image.

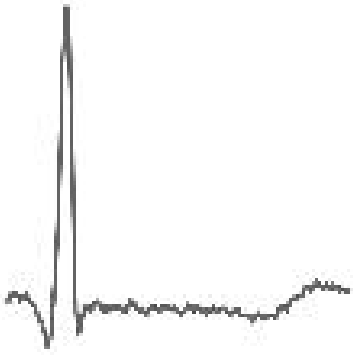

Figure 3.6: A compressed image (augmented).

3.3.5 The ECG Arrhythmia Classifier

The proposed solution uses a specifically designed two-dimensional CNN as the auto-ECG arrhythmia classifier. In 1989, LeCun brought a new type of neural network: the CNN model [10]. Comparing the performance differences between ANN and CNN in image classification tasks, the classical feed-forward neural network is not efficient when processing vast amounts of images since there are too many parameters. Therefore, we adopted the CNN as the proposed solution for ECG arrhythmia heartbeat classification. CNN can successfully capture the critical spatial and temporal connections in an image by applying relevant filters. For this reason, CNN architecture performs better in a image data set due to its reduced parameters. In other words, the network can be better trained to understand the content of a image. Most machine learning solutions for classifying ECG arrhythmia heartbeats are one-dimensional CNN [6][11][12]. We implement a two-dimensional CNN by converting the ECG signals into ECG images. The two-dimensional convolutional and pooling layers are more suitable for classifying the image-type data. Accordingly, we can obtain a higher accuracy of ECG arrhythmia classification. It is also similar to a doctor-diagnosing scenario because a doctor analyses the patient's ECG signal through two-dimensional measurement, i.e. analyzing a image. We decide to simulate a computer vision solution, which applies the two-dimensional CNN model to classify ECG heartbeats images.

Currently, the development of CNN has accomplished many outstanding achievements. There are many validated and effective CNN models in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC). It is a competition that evaluates algorithms for object detection and image classifications with millions of images. In 2014 ILSVRC, GoogLeNet takes first place [14], and VGG network takes second place [15]. Although GoogLeNet has better performance than VGG network, the VGG has a straightforward architecture and fewer layers than GoogLeNet. VGG can achieve relatively similar performance as the GoogLeNet in image classification; therefore, VGG network is often used for image recognition tasks. In the ECG arrhythmia classification task, the model only needs to classify 150 x 150 pixels grayscale images into several classes, which are much simpler than the classification task in ILSVRC. Consequently, the efficient and less complicated VGG-style network is a desirable solution for this classification work. We adopt the VGG network concepts for constructing our proposed model, and we modify parameters and layers of the proposed

model to achieve improved performance in ECG abnormal heartbeat classification. Fig. 3.7 is an overview of all the processes of proposed solution.

Figure 3.7: The overall workflow of the proposed model.

VGG network introduction

The VGG network is introduced in the paper [15], and VGG is the abbreviation of the visual geometry group that invents the VGG network. The VGG is a type of CNN structure, and it provides practical concepts to help developers build efficient CNN models for image classification. The fundamental idea of the VGG network is the VGG block, and a typical VGG block contains a sequence of convolutional layers, max-pooling layers, and activation functions. In the VGG paper, the original network includes 3 x 3 convolution kernels with padding of 1 and 2 x 2 max-pooling with a stride of 2. Fig. 3.8 is the visual representation of a VGG block.

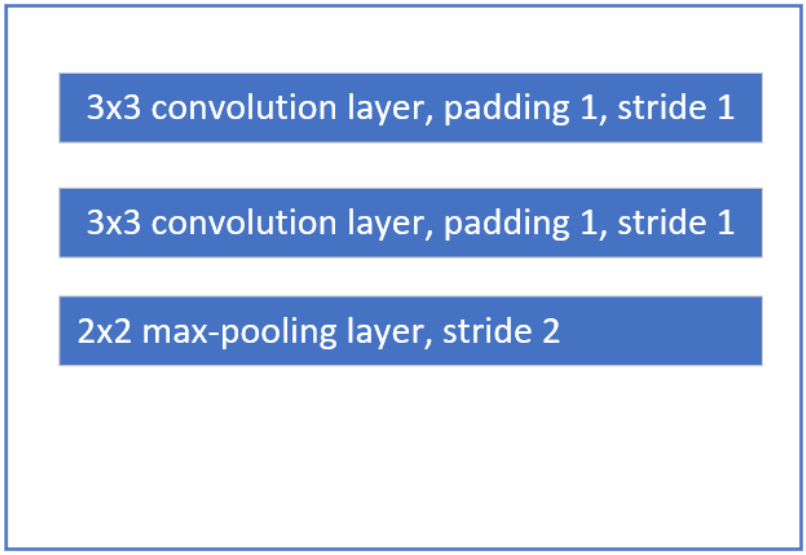

Figure 3.8: The structure of a VGG block.

Configurations of VGG

In general, there are six configurations for VGG networks, and each of them has a different size of convolution kernels. Fig. 3.9 is the summary graph of these configurations. All the configurations have five VGG blocks. Usually, the 16 weight layers and 19 weight layers are the most used VGG network structures. Although the VGG network has a straightforward structure, the model has numerous parameters that can cost many computation resources for training. For example, configuration D has 138 million parameters. Therefore, it is essential to add decent numbers of VGG blocks and adjust the proposed model's convolution layers. Through many studies and experiments, we construct the effective structure of our proposed model.

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224×224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Figure 3.9: Six configurations [15]

Adjusted VGG blocks

The proposed model is developed based on the VGG network. Although the proposed model's structure follows the same patterns as the VGG network, we apply many methods and modify the convolution layers to obtain the model's high performance. The original VGG network is used to classify millions of pictures into many categories. Therefore, the network has a deep structure with massive parameters. The ECG heartbeat classification problem has a small number of heartbeat categories. If we apply the original model to our problem, it can easily lead to model overfitting. It also wastes computation resources and long model training time. Therefore, we design a specialized network structure for the problem at hand. The following methods are the approaches for our proposed model.

The original VGG network includes five VGG blocks and these blocks make the VGG network a deep structure. In the ECG heartbeat classification, we decide to reduce the number of VGG blocks to improve model performance. After many ex-

periments, we conclude that the proposed model has the best performance with three VGG blocks. We also adjust the convolutional layers in the third VGG block. The difference between the modified VGG blocks and the original VGG blocks is shown in Fig. 3.10.

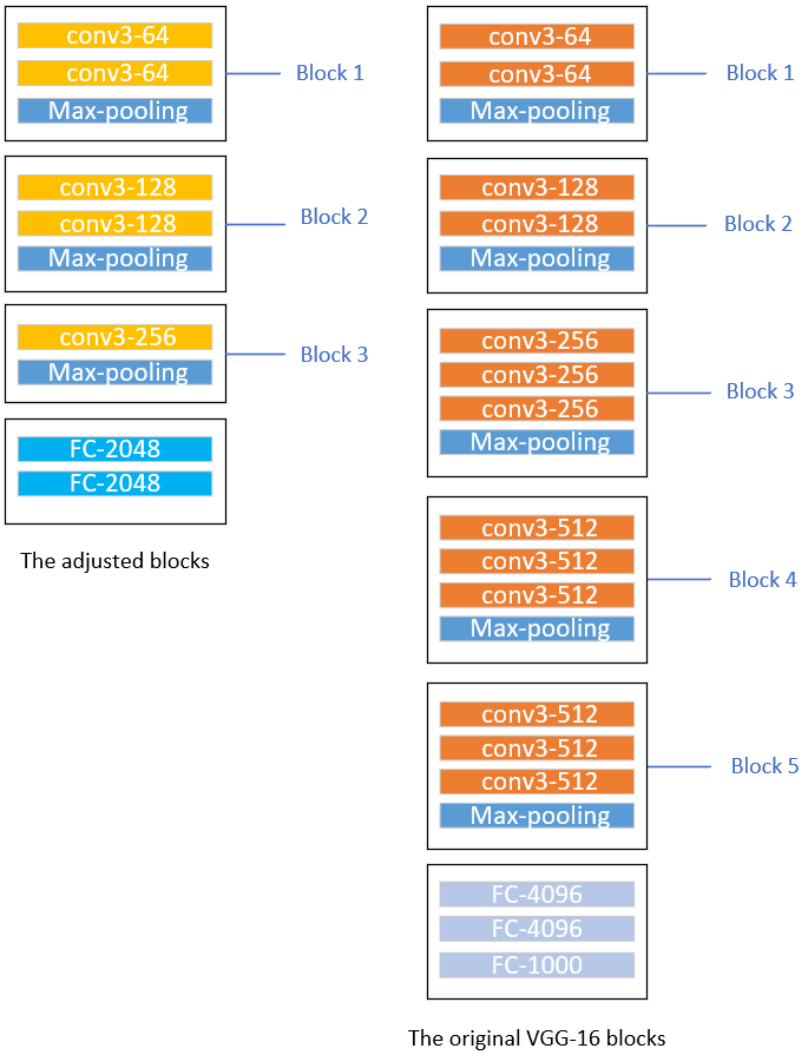

Figure 3.10: Differences between adjusted VGG blocks and original VGG-16 blocks.

Activation function

The activation function is an essential part of a neural network, which is a non-linear function that determines whether and how much to fire up the neuron output given the input. We adopt rectified linear unit (ReLU) as the activation function for our proposed model [55]. ReLU is the most commonly used activation function in CNN

models, given by

$$y = max(0, x)$$

The ReLU reduces the model's training time. The linearity of the ReLU makes it a fast converging algorithm because the slope remains the same when x increases, and Fig. 3.11 shows the graph of this function. With this feature, ReLU avoids the vanishing gradient problem during the model's training [2]. However, there is a disadvantage of using ReLU: it outputs zeros for all negative values. Since it outputs zeros for all negative values, it is unlikely for a neuron to change to other values if it has a negative value. Because the ReLU has a zero slope when neurons have negative values, neurons are stuck on the negative side and ReLu always outputs zero. Eventually, this property leads to many useless neurons in a neuron network, which lowers model classification accuracy. The existing solutions are lower learning rate or using exponential linear unit (ELU) as the activation function. We conclude that a lower learning rate can obtain better classification accuracy for this application. ELU does not improve the model's performance. For the above reasons, we apply ReLU as our activation function.

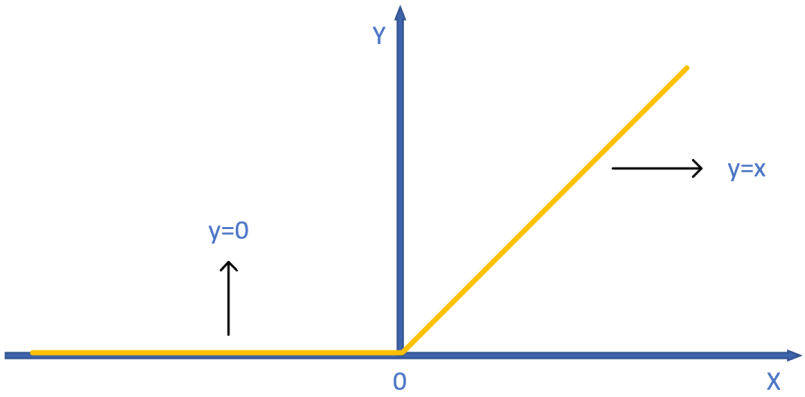

Figure 3.11: The graph of ReLU.

Batch normalization

In a deep CNN with many hidden layers, the hidden layer parameters are dependent on its previous layer. Therefore, even a small change in the last layer's parameters can strongly influence the next layer's input distribution. The changes in the input distributions of hidden layers in a neural network are technically named internal covariate

shift (ICS). It slows down the model training speed. Since we have a considerable amount of heartbeat images with limited computational resources, we adopt batch normalization for improving the training speed of our proposed model. Batch normalization reduces training time. It standardizes layer inputs. It does normalization on the output of a previous layer by calculating the batch mean and variance, and this will shift and scale the output of the layer. The batch normalization can apply before the active function of after it. Although the author who brought the idea of batch normalization puts this method before the active function [16], some people have reported better performance when placing the batch normalization after the active function. After many experiments, we conclude that the model can achieve a better classification result by placing the batch normalization before the active function.

Cost function

The softmax is a type of logistic regression that normalizes input data into a vector of probabilities, and the sum of probabilities is equal to 1. Softmax is implemented to compute costs in the proposed model's training process, and we use *tensorflow.keras* to implement the softmax layer [31]. In the neural network, the cost function measures the performance of the proposed model prediction accuracy. There are many types of cost functions, and we choose to use cross-entropy as our proposed model's cost function. Usually, the model outputs a probability value between 0 and 1. If the output value close to the actual label, we obtain a small cross-entropy loss. Otherwise, cross entropy-loss increases significantly. Here is the mathematical expression of cross-entropy [63]:

$$H(p,q) = -\sum_{i=0}^{n} p(x_i)\ \log(q(x_i))$$

Proposed model architecture

The proposed model structure adopts the VGG network. We apply the above approaches to improve the proposed model's performance, and Fig. 3.12 is the modified model's structure.

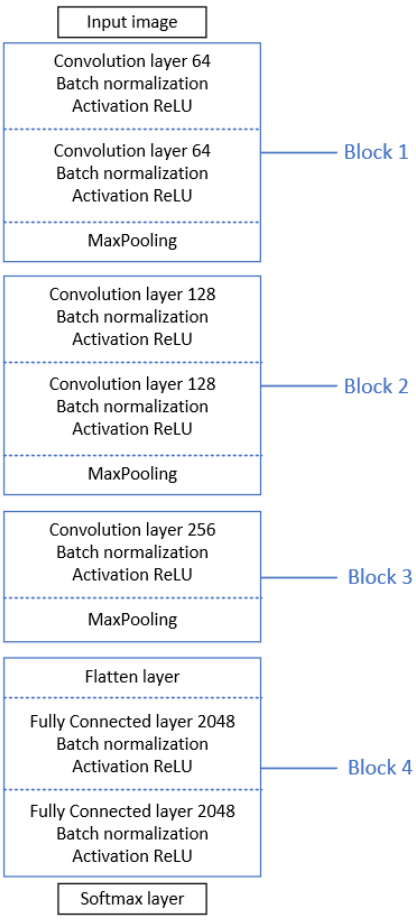

Figure 3.12: The proposed model blocks.

Basing on our input data's attributes, we modified the VGG structure to make it more efficient for solving our problem. We cut down the unnecessary VGG blocks and layers. Table 3.3 is the proposed CNN parameters.

Layers	Kernal Size	Stride	Channels
Conv2d	3 x 3	1	64
Conv2d	3 x 3	1	64
Max-Pooling	2 x 2	2	
Conv2d	3 x 3	1	128
Conv2d	3 x 3	1	128
Max-Pooling	2 x 2	2	
Conv2d	3 x 3	1	256
Max-Pooling	2 x 2	2	
Fully-Connect			2048
Fully-Connect			2048
Softmax			

Table 3.3: The proposed model parameters

Also, we apply the batch normalization into the model. With this method, we solve the internal covariate shift problem. It increases the proposed model's classification accuracy and reduces the model's training time. We conclude that when the batch normalization process starts earlier than the activation function process, the model can achieve better performance through many experiments. We use the cross-entropy as our model's loss function. The cross-entropy is often used in the deep learning model for categorical classification, and we obtain excellent classification results by applying cross-entropy as the loss function. Fig. 3.13 shows the visual representation of all the layers in the proposed model.

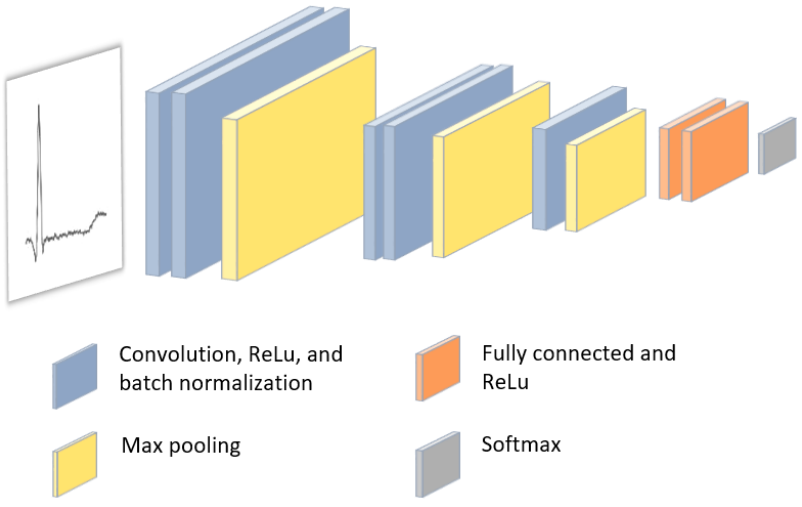

Figure 3.13: The proposed model layers.

We implement the proposed model using the TensorFlow framework [31]. We use *tensorflow.keras.models* to build the neural network structure. Here are the proposed model's training and evaluation steps.

1. Loading training set and testing set by *ImageDataGenerator*
2. Applying *tensorflow.keras.models* to construct the proposed CNN. We use this package to add convolution layers, batch normalization layers, max-pooling layers, and fully connected layers.
3. Loading the training data and testing data to the model by *model.fit_generator*
4. Recording the classification accuracy by *model.evaluate_generator*
5. Using *model.predict_generator* to obtain the prediction labels, and we compare the predicted outputs with the true labels of the images. Also, we build the confusion matrix of classification results.

3.4 Experiments And Analysis

3.4.1 Model Evaluation

The proposed ECG arrhythmia heartbeat classifier's performance has been evaluated under the inter-patient paradigm along with the AAMI standard. In the experiments, we train the proposed model with DS1, and we evaluate the model's performance by classifying the images in DS2. We run 120 epochs for model training, and we record the highest values among all the epochs' results. The model's classification accuracy becomes relative stables after 120 epochs. Therefore, we only train the model with this number of epochs for saving the experiment's time and computation resources. The model performance evaluation is calculated based on the sensitivity (SE), specificity (SP), the positive predictive value (PPV), and accuracy (ACC) defined as:

$$SE = \frac{TP}{TP + FN}$$

$$SP = TN/(TN + FP)$$

$$PPV = TP/(TP + FP)$$

$$ACC = (TP + TN)/(TP + TN + FP + FN)$$

where TP (True Positive) is the number of heartbeats correctly classified, TN (True Negative) is the number of heartbeats not belonging to the target class and not classified to target class, FP (False Positive) is the number of heartbeats that incorrectly classified into the target class, and FN (False Negative) is the number of target class heartbeats classified to a different class. After obtaining the model's output label, we calculate the result and analyze the model output. We also use the *classification_report* function from scikit-learn to evaluates the models' multi-class classification results [64]. In each experiment, model parameters are recorded with the model classification accuracy, and we keep adjusting the model parameters and optimizing the model structure to obtain the best classification accuracy that can be achieved. We also compare our model's performance with other reported models. In this way, we can evaluate the model's performance more accurately. The following table outlines testing environment for the experiment.

	CPU	GPU
Processor Name	AMD Ryzen 3600	GeForce RTX 2070
Clock frequency	4,200MHz	1,612MHz
Memory	16GB	8GB

3.4.2 Evaluation of Approaches

Impact of the input image resolution

The input image resolution can significantly affect the model's classification accuracy. The low-level image resolution can lose important information of the ECG heartbeat, and the high-level image resolution can take a relatively long training time or even run out of GPU's memory. Therefore, it is essential to determine the input image resolution size for achieving the best model's performance. We test various image resolution sizes in the experiments, and Fig. 3.14 shows the classification results of SVEB and VEB that correspond to different image resolutions.

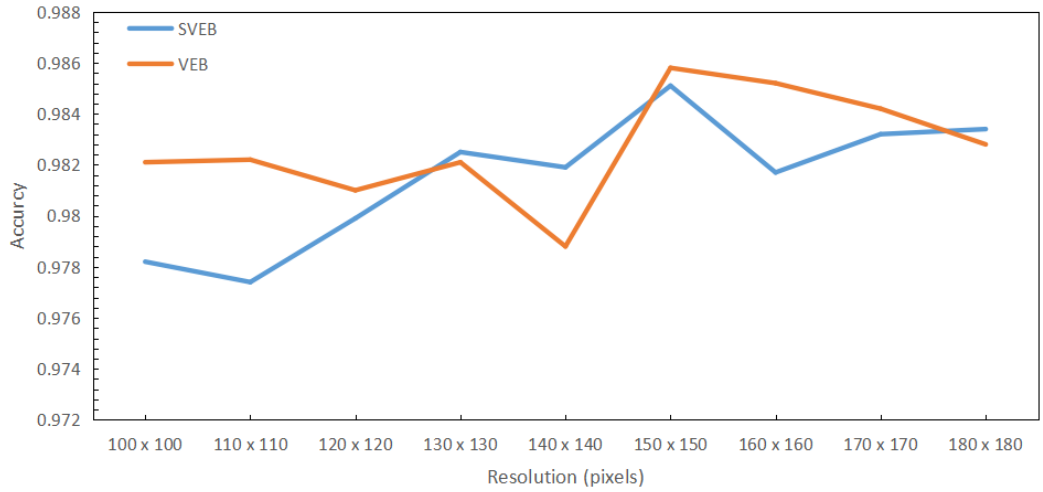

Figure 3.14: The model's classification accuracy with respect to image resolutions.

From on the experiment results, we can see that when the image resolution is 150 x 150 pixels, the model obtains the highest accuracy in both SVEB classification and VEB classification. Therefore, we set 150 x 150 pixels as the image resolution size for training images and testing images. We do not test image resolution that below 100 x 100 pixels because the image can not show the complete heartbeat's shape. Also, we do not test the image size above 190 x 190 pixels because it runs out of the GPU

memory, and the experiment results show that a higher image resolution does not necessarily increase classification accuracy.

Impact of the learning rate parameters

The proposed model is trained based on the stochastic gradient descent optimization algorithm, and the learning rate is the hyperparameter that controls the changes in the model's weights in the model training process. The learning rate decides how sensitive the model responds to the estimated error. A high learning rate means larger step size, and hence faster convergence but may be trapped to a local minimum, and a low learning rate makes the model insensitive to the error, so the model's weight does not change much, even the estimated error is big. A non-optimal learning rate can lower the model's performance or increase the model's training time. In the experiment, we test different learning rates, and Fig. 3.15 shows the model's classification accuracy with different learning rates.

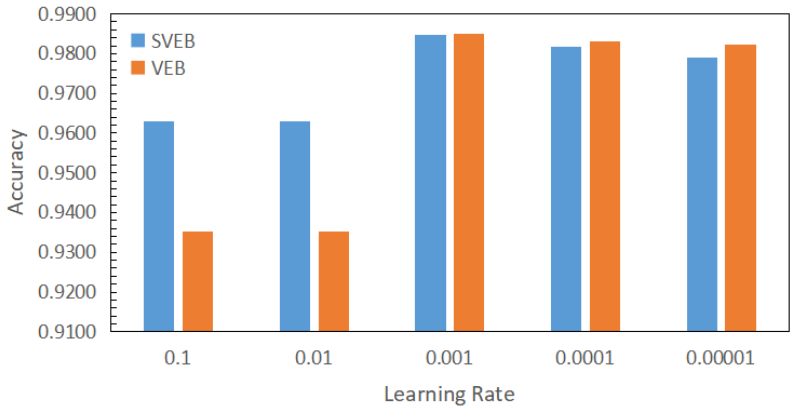

Figure 3.15: The model's classification accuracy with respect to learning rates.

Based on the experiment results, the proposed model achieves the highest classification result in both SVEB and VEB when the learning rate is 0.001. In the proposed model, we apply the ReLU as the activation function; the "stuck neurons" issue lowers the model's performance. The low learning rate is one of the satisfactory solutions for this problem. It is reasonable that the model achieves the best classification result with 0.001 as the learning rate value. Therefore, we set the model's learning rate to 0.001 for improving the model's training efficiency and classification performance.

Impact of batch normalization

We compare the model's performance difference between the model with batch normalization layers and without batch normalization layers in the experiment. Also, we record the experiment results of placing the batch normalization layer before the activation function and placing the batch normalization layer after the activation function. Fig. 3.16 shows the results of these cases.

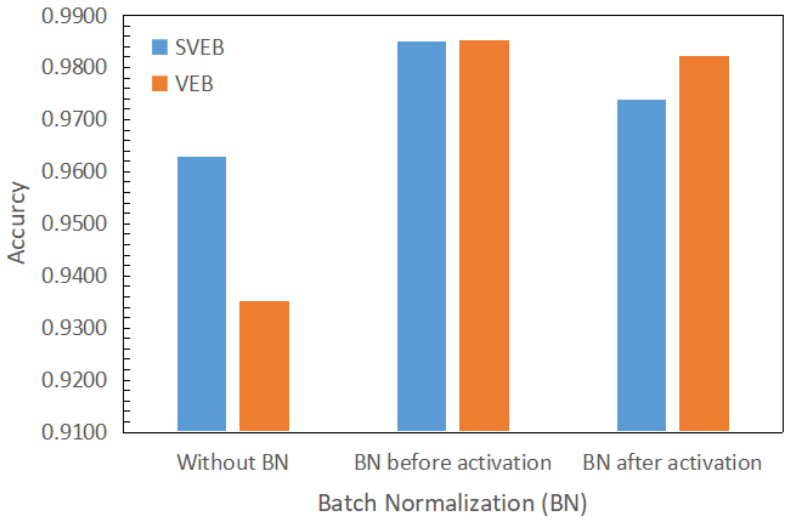

Figure 3.16: The model's classification accuracy with respect to batch normalization.

We can see that adding batch normalization layers into the proposed model can significantly improve the model's performance. In the previous section, we discuss the advantages of applying the batch normalization layer in the model. The experiment result proves that batch normalization layers can improve the model's classification accuracy. However, the order of a batch normalization layer is not fixed. It depends on the dataset, the model's parameters, and the model's structure. We can achieve a higher classification accuracy when we place the batch normalization layer before the activation function in the experiment.

Impact of activation functions

In the previous section, we discuss the proposed model's activation function, and there are two commonly used activation functions for CNN. We evaluate the model's performance with these two activation functions to determine the proposed model's activation function. Fig. 3.17 shows the results of using these two activation func-

tions. Basing on the experiment results, the model has similar classification accuracy when applying these two activation functions. However, the model has a higher classification accuracy in SVEB using ReLU than using ELU. Therefore, we choose to use ReLU as the activation function for the proposed model.

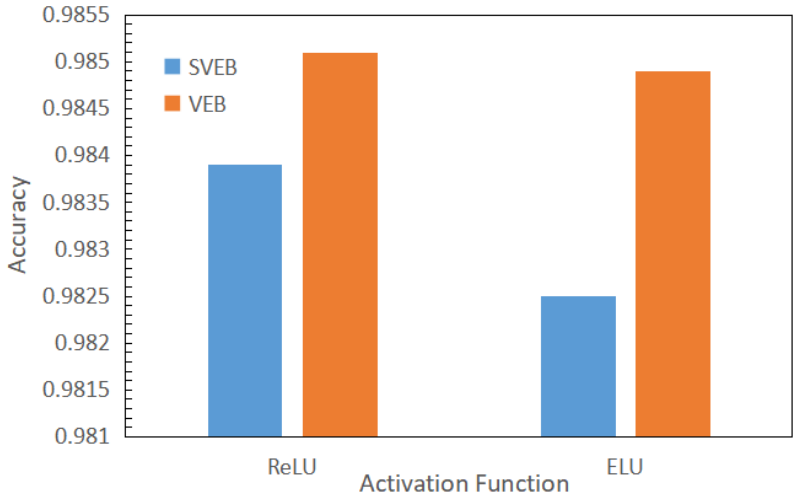

Figure 3.17: Comparison of the model's classification accuracy with two activation functions.

Comparison with VGG network

We compare the proposed model with the configuration A VGG network, and Fig. 3.18 shows the accuracy differences between these two networks. We can see that the proposed model outperforms the VGG model in the ECG arrhythmia classification task. The VGG model has relatively deep layers with a massive amount of parameters. Our experiments have relatively small samples with limited labels in the ECG arrhythmia heartbeats classification, so the VGG classifier suffers from overfitting issues. Eventually, this VGG model lowers the classification accuracy of abnormal heartbeats. Our proposed model removes unnecessary layers and reduces the model's parameters. We also apply batch normalization layers to obtain better classification results, and experiments validate that our approaches on the model's structure and parameters can significantly increase the classification accuracy compared with the original VGG model.

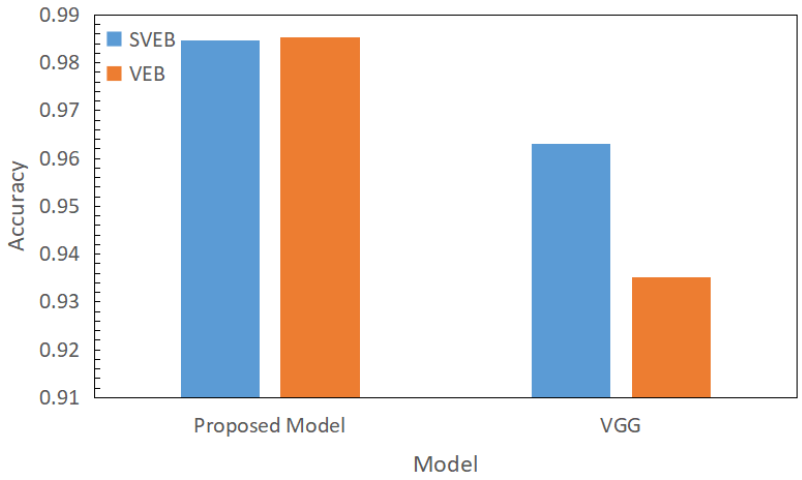

Figure 3.18: The performance difference between the proposed model and the original VGG network.

3.4.3 Results

The complete classification assessment results are presented in Table 3.4, and we can see that the proposed classifier shows a 93.7% overall heartbeat classification accuracy. The model achieves outstanding results in classifying N-type and VEB-type heartbeats. However, the classification accuracy of other types of heartbeats are relatively low. In compliance with the AAMI recommendations, we focus on assessing an algorithm's ability to discriminate VEB beats from non-VEB beats and SVEB beats from non-SVEB beats. Other reported works in [7], [19]–[27] also apply this performance assessment.

		Predicted class				
		F	N	SVEB	VEB	Total
True Class	F	1	366	1	20	388
	N	583	42848	339	448	44218
	SVEB	2	819	984	31	1836
	VEB	3	476	33	2707	3219
	Total	589	44509	1357	3206	49661
	Accuracy = 93.7%					

Table 3.4: The four classes classification result

We compare the proposed model and other classification models. In Table 3.5, we can see the proposed model obtains better classification results in the inter-patient paradigm, with a 98.4% overall accuracy in VEB-type heartbeat and a 98.5% overall accuracy in SVEB-type heartbeat. Notably, a 98.5% positive predict value of SVEB indicates that the proposed model has an outstanding performance in identifying SVEB-type heartbeats. Also, the 75.8% sensitivity of SVEB is higher than most of the other solutions. These results prove that the proposed model can effectively distinguish SVEB heartbeats from non-SVEB heartbeats and VEB from non-VEB heartbeats. The comparison results of the four-class heartbeat classification are presented in Table 3.6. We compare the results of sensitivity and positive predict value with other existing solutions. Since there are few F-type heartbeats samples, we only show the classification accuracy of N, SVEB, and VEB. We can see that the proposed model achieves the highest overall classification accuracy among all compared models. It obtains the highest sensitivity in the N-type and highest positive predictive value in SVEB type. By comparing with other solutions, we argue that converting the

ECG signals to ECG heartbeat images and using two-dimensional CNN for classifying is a practical approach for ECG arrhythmia classification under the inter-patient paradigm.

Method	SVEB				VEB			
	ACC(%)	Sen(%)	Spe(%)	Ppv(%)	ACC(%)	Sen(%)	Spe(%)	Ppv(%)
Proposed	**98.5**	**75.8**	**99.3**	**81.1**	**98.4**	**84.6**	**99.3**	**89.9**
Chazel et al. [7]	94.6	75.9	N/A	38.5	97.4	77.7	N/A	81.9
Ye et al. [19]	97.4	56.4	98.6	55.1	94.6	84.7	95.4	59.5
Alvarado et al. [20]	97.0	86.2	97.5	56.7	99.1	92.4	99.5	93.4
Mar et al. [21]	93.3	83.2	93.7	33.5	97.4	86.8	98.1	75.9
Rahlal et al. [22]	94.9	37.8	97.5	40.5	97.8	90.1	98.6	87.1
Garcia et al. [23]	96.6	62.0	97.9	53.0	95.4	87.3	95.9	59.4
Zhang et al. [24]	93.3	79.1	93.9	36.0	98.6	85.5	99.5	92.7
Kan Luo et al. [25]	96.2	15.4	99.3	47.3	95.5	60.4	97.9	66.8
L. Zaor´alek et al. [26]	96.0	6.0	99.5	33.5	99.0	91.6	99.5	93.2
Ince et al. [27]	96.1	62.1	98.5	56.7	97.6	83.4	98.1	87.4

Table 3.5: Results of SVEB and VEB classification

Methods	Accuracy	N		SVEB		VEB	
		Se(%)	Ppv(%)	Se(%)	Ppv(%)	Se(%)	Ppv(%)
Proposed	**93.7**	**96.9**	**96.3**	**53.6**	**72.5**	**84.1**	**84.4**
Mar et al. [21]	89.0	94.2	99.2	86.2	56.7	92.4	93.4
Alvarado et al. [20]	93.6	94.2	99.2	86.2	56.7	92.4	93.4
Ye et al. [19]	88.2	90.0	98.2	56.4	55.1	84.7	59.5
Zhang et al. [24]	88.3	88.9	99.0	79.1	36.0	85.5	92.8

Table 3.6: Results of four-classes heartbeats classification

3.5 Conclusions

In conclusion, we have proposed a lightweight two-dimensional CNN ECG abnormal heartbeat classifier. We have developed algorithms to generate the proper ECG beat images during the ECG signal pre-processing stage. Instead of using the intra-patient paradigm to obtain overly optimistic results, we have implemented the inter-patient paradigm for partitioning the data set to evaluate the model's relatively real classification performance when given a new patient.

We have adopted VGG network concepts and VGG block as the core element of the proposed model. We have also applied methods that add batch normalization layers to the model, reduce unnecessary layers, and adjust the model's hyper-parameters. We have compared with other reported arrhythmia classification solutions. Experiment results show that our proposed classifier achieves outstanding classification accuracy: 98.5% classification accuracy in SVEB-type heartbeat and 98.4% classification accuracy in VEB-type heartbeat. The proposed model's performance is better than most of the models in SVEB-type and VEB-type classification based on the same database with the inter-patient paradigm. The proposed model has a significant performance improvement than the configuration-A VGG network in the same testing environment. We plan to modify the model's structure and parameters to achieve better classification results for various abnormal heartbeats in future research.

Chapter 4

Remote ECG Monitoring System with API Design

4.1 Introduction

Internet technology development and high-speed wireless networks have provided many new potentials in the medical field [3]. Researches on medical software development have contributed to our society significantly. The remote electrocardiogram (ECG) monitoring is one application of the advanced computer technology to provide better medical treatments [40]. Some clinics plan to monitor patients' cardiac health remotely now that steady wireless internet services are widely available [42]. In long-term ECG monitoring, physicians can collect patients' ECG information remotely. Our team develops an effective system for physicians to monitor patients' ECG information remotely. Since this system will be used in hospitals and clinics, primary goals in developing this system are to be reliable and secure.

The entire system consists of four modules: mobile applications (APP), the central server, ECG sensors, and clients. The proposed system relies heavily on interactions among modules, making it a challenge to implement interaction methods that support stable and secure data transmissions between modules. We adopt the hypertext transfer protocol (HTTP) to implement interactions between modules in our system. The application programming interface (API) that uses HTTP plays an essential role in the interaction. Hence, it is critical to design secure and secure APIs to accomplish software development goals. We adopt the representational state transfer (REST) style as our API design guideline. REST APIs have unique and advanced

resource focusing techniques [29], which can connect the web framework, servers, and clients in a modern practical way. It also presents a secure and straightforward web service structure. In our software development team, Linfeng Xu works on front-end development, Zhilun Liu works on back-end development, and I work on APIs design.

In this chapter, We first explain the modules and workflows in our proposed system. The central server structure is then discussed in detail. After which the REST-style web structure for ECG monitoring is defined to cover all essential components of the ECG monitoring. Specifically, the considerations during the design stage are clearly explained and discussed. This chapter aims to provide a comprehensive overview of the proposed system and the ECG APIs design concepts.

The chapter is organized as follows: Section 4.2 introduces the remote ECG monitoring system. Section 4.3 shows all details of the proposed REST APIs. Lastly, the conclusions of the proposed work are given in Section 4.4.

4.2 Remote ECG Monitoring System

The remote electrocardiogram monitoring system provides an efficient way for doctors to establish remote long-term ECG monitoring for patients. In this system, doctors set up a monitoring appointment through a proposed client software, and patients pick up ECG sensor devices and smartphones with ECG application (APP) installed in the hospital or specific appointed locations. When patients arrive home, they start the video call to their physicians, and they follow physicians' instructions to set up an ECG hookup connection. It includes wearing the ECG sensors, pairing the sensors with the proposed APP, testing the connection with the server, and checking the ECG signal transfer by reading the ECG signal images in the APP. After patients complete all these processes, patients wear ECG sensors during the entire monitoring period. Eventually, physicians analyze patients' ECG information in the proposed client software, and they use this software to generate a diagnosing report. This report sends to the patient by mail or email. Fig. 4.2 is an overview of the proposed system's workflow.

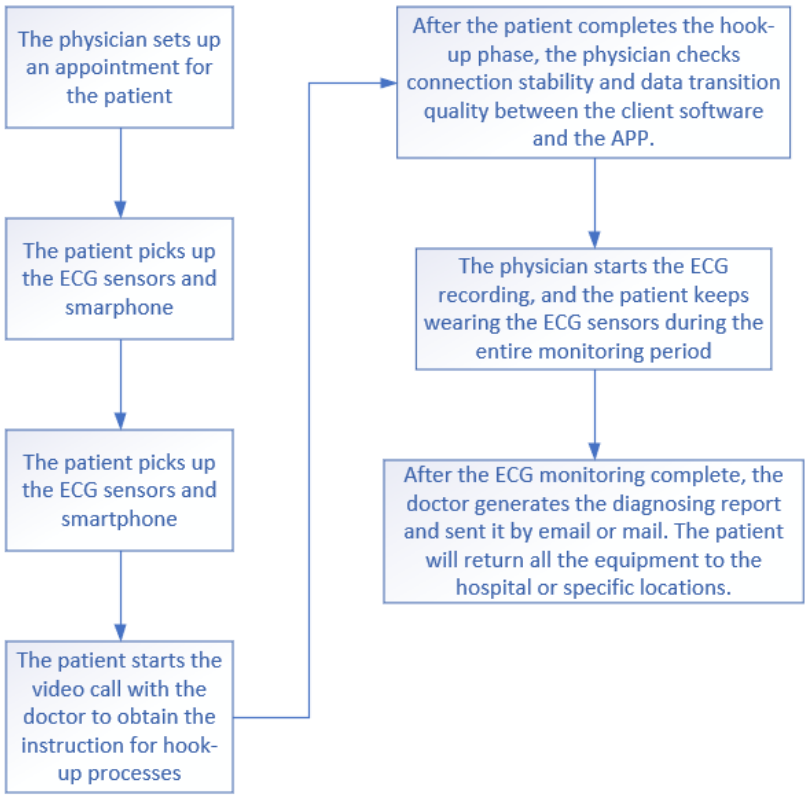

Figure 4.1: The ECG software flow chart.

The proposed system involves several modules: ECG sensor, portable APP, the central server, database, and client software. The ECG sensor is wearable equipment that collects the patients' ECG signal information, and it sends the ECG data through the bluetooth connection with the smartphone. The smartphone's APP receives ECG signals that come from the sensors. After the APP obtains ECG information, it starts plotting ECG signal graphs. At the same time, it sends the ECG data to the central server. The server receives ECG data from APPs. Subsequently, the server processes and stores the ECG data in the database. The server is also responsible for processing all requests from the front-end, such as user login, user account registration, creating an appointment for a patient, and related requests. The client software provides a graphical user interface (GUI) for physicians to control ECG monitoring processes. It provides several functions to help physicians to establish an ECG monitoring test for a patient. The software's primary goals are creating appointments, arranging the appointments, and receiving the ECG data. The complete system can be separated into two parts, the front-end section and the back-end section. Fig. 4.3 is the visual presentation of these two sections.

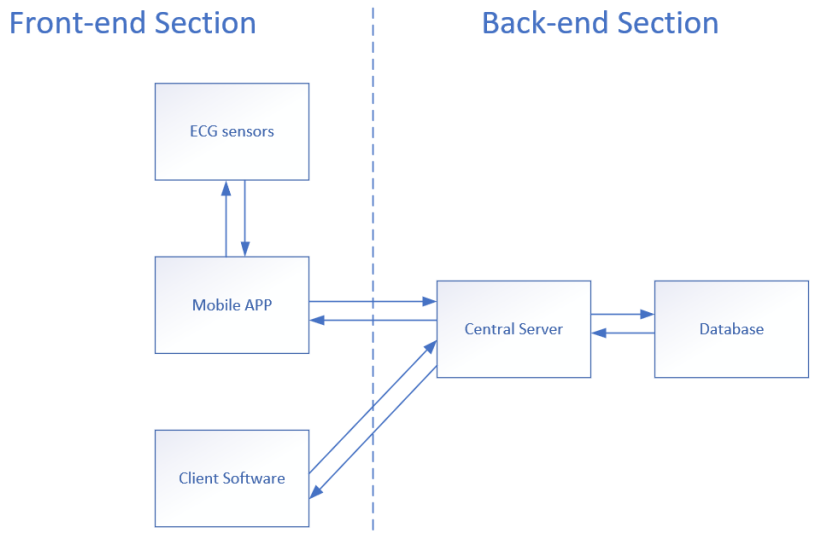

Figure 4.2: The ECG software flow chart.

4.2.1 The Central Server

The most important part of the proposed system is the central server, and it is the engine that keeps the whole system running functionally. All modules in the front-end operate request sending and receiving functions. Without the central server

process these requests, these modules are isolated units, and the system stops running. Therefore, it is essential to design a solid structure for the server to process all the requests from the front-end consistently [18]. We apply the industry-standard spring framework for building our proposed server program [17]. This framework reduces the coding works for developing functions in the server program. In this way, it saves the back-end developer's valuable time. The proposed server consists of the data access object (DAO) tier, the service tire, and the controller tire. The DAO is responsible for interactions between the server functions and databases. In this layer, functions perform data interactions with databases, such as input data to the database, receive data from the database, and update the database's information. The proposed database is implemented in MySQL [60]. The service layer is responsible for processing all requests from the front-end. The back-end developer uses Java [61] to implement all the service functions. For each request, there is a corresponding function to process this request. The controller layer responsible for receiving the front-end's requests, and it analyzes the received request and communicates with the corresponding functions in the service layer. After that, it returns specific values to this request's sender. Fig. 4.4 shows the structure of the back-end.

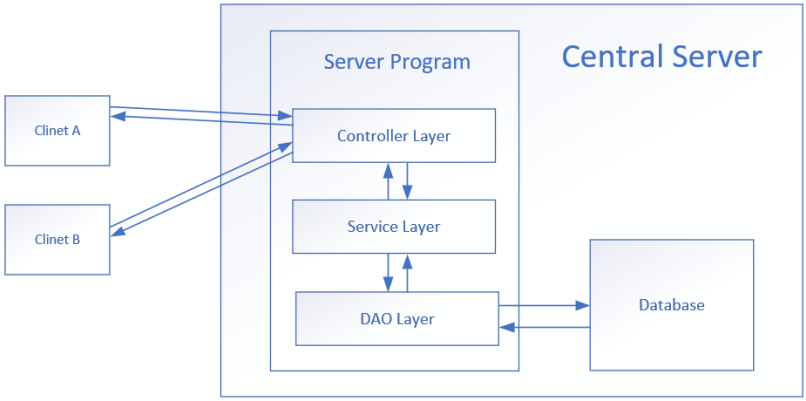

Figure 4.3: The ECG software flow chart.

4.2.2 API Explanation

Another essential part of the ECG remote monitoring system is the application programming interface design. The proposed system consists of various modules, and we use the hypertext transfer protocol requests to implement interaction among these modules. Proper API designs can help server programs process requests efficiently,

and a well-written API document is a software development guide for front-end developers and back-end developers. The document defines each API's requirements and structure. Accordingly, software developers can follow these requirements to write corresponding functions. A well-designed API is always straightforward to use and improves developers' work efficiency. The API can be considered as a graphical user interface (GUI) for developers. Consequently, they can quickly know the requirements for implementing different HTTP requests in the system. Usually, the API documentation is designed ahead of the actual development of the software.

Representational state transfer provides design concepts for building efficient APIs between multi-platforms in decentralized systems [29]. A REST API defines the results of an interaction between applications that satisfies the REST standards. Also, REST is an open concept that is not restricted by any specific implementation methods. REST APIs are mainly used for defining HTTP request methods that interact between different modules in a system, and they have a straightforward structure with practical concepts. Applying REST APIs in software development can significantly reduce works between front-end developers and back-end development developers.

4.2.3 Principles of RESTful APIs

- The important part of a REST API is the resource. Resources are data that can be accessed by clients. REST APIs focus on methodologies of accessing these resources. All resources have a unique identifier corresponding to them. Practically, we use uniform resource identifiers (URI) as the unique identifier of the resource. Fig. 4.5 is an example of accessing a student's information through URI.

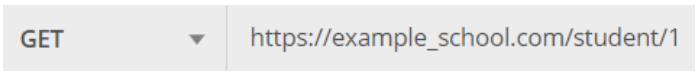

Figure 4.4: An example of accessing a student information through URI.

- Clients interact with a server program by exchanging representations of resources. Many web APIs use javascript object notations (JSON) as the exchanged data format [4], and Fig. 4.6 is an example of JSON format data in a request.

Figure 4.5: An example of JSON data format.

- In REST APIs, interactions between the server program and clients are defined by HTTP methods. These methods determine the proposed actions of HTTP requests. Usually, standard actions are GET, POST, PUT, and DELETE [61]. By applying these approaches, interactions between the server program and clients are more efficient.
- REST is stateless, which means each request between server and client can only contain the necessary information for this request. The server program can not use information that comes from previous requests. Therefore, clients need to store all the state information and provide sufficient information for sending an HTTP request to the server program.

4.2.4 REST Parameters

In an HTTP request, we have options to specify the requirements of the request. Commonly, there are three parameters: header parameters, path parameters, and query parameters. By defining these parameters, we can make sure a request contains all the proposed request information to the server program.

Header parameters

Header parameters should only be contained in the request header section. A header parameter defines the authorization information. Usually, it shows the authorization type. It can also include the user's unique token for accessing the specific resource in a server program.

Path parameters

Path parameters are part of the URI itself and are not optional. Fig. 4.7 shows a path parameter in a URI. In this example, studentId is the required path parameter.

Figure 4.6: An example of a path parameter in a URI.

Query string parameters

In a URI, query string parameters are placed after a question mark. When a request contains multiple query string parameters, we use a special annotation to connect these parameters. Fig 4.8 is query string parameters in a URI. We can see these two parameters are connected by the annotation & in this URI.

Figure 4.7: An example of query string parameters in a URI.

4.3 ECG REST API Framework Design

4.3.1 Login Section

The nurse logins to the software

Instead of using GET as the HTTP request method, the POST has better security for this request because it can prevent the information from being accessed by other users during the process of sending the request information to the server. This request contains the user's email and password; then, this request is sent to the server by POST method. If the email and password are correct, a success status code 200 is sent back to the client; otherwise, an error status code 400 with specific error information is returned to the client.

4.3.2 Nurse Section

The nurse registers to the software

The goal of this request is to process creating a new user account. The request contains nurses' required information for creating a new account; then, this request is sent to the nurses resource in the server program by the POST method. If a new user account is successfully created, a success status code 200 is sent back to the client; otherwise, an error status code 400 with the specific error information is returned to the client.

The nurse resets password

The goal of this request is to reset the user account password. The request contains the user's email address; then, the request is sent to nurses resource in the server program by the POST method. If the email is registered, a success status code 200 and reset password information is sent back to the client; otherwise, an error status code 400 with specific error information is returned to the client.

The nurse starts recording device

The goal of this request is to start ECG recording with an ECG device. The request contains the ECG device id; then, the request is sent to the nurses resource in the server program by the PATCH method. If the device starts recording, a success status

code 200 and ECG data is sent back to the client; otherwise, an error status code 400 with the specific error information is returned to the client.

The nurse stops recording device

The goal of this request is to stop an ECG recording device. The request contains the ECG device id; then, the request is sent to the nurses resource in the server program by the PATCH method. If the device stops recording and sending data, a success status code 200 is sent back to the client; otherwise, an error status code 400 with specific error information is returned to the client.

4.3.3 Patient Section

Searching a patient's record

The goal of this request is to search for a patient's record. The request contains required search terms; then, the request is sent to the patients resource in the server program by the GET method. If the target patient's information is successfully found, a success status code 200 and the patient's record information is sent back to the client; otherwise, an error status code 400 with specific error information is returned to the client.

Creating a patient's information

The goal of this request is to create a patient's record. The request contains the patient's information; then, the request is sent to the patients resource in the server program by the POST method. If the patient's record is successfully created, a success status code 200 is sent back to the client; otherwise, an error status code 400 with specific error information is returned to the client.

Updating a patient's record

The goal of this request is to update a patient's record. The request contains the new patient's information; then, the request is sent to the patients resource in the server program by the PUT method. If the patient's record is successfully updated, a success status code 200 is sent back to the client; otherwise, an error status code 400 with specific error information is returned to the client.

4.3.4 ECG Test Section

Searching all ECG tests of a patient

The goal of this request is to search for all ECG tests of a patient. The request contains the patient id; then, the request is sent to the ECG resource in the server program by the GET method. If the patient's tests are successfully found, a success status code 200 and the ECG tests are sent back to the client; otherwise, an error status code 400 with specific error information is returned to the client.

Searching the ECG test of a patient

The goal of this request is to search for an ECG test of a patient. The request contains the patient ID and test ID; then, the request is sent to the ECG resource in the server program by the GET method. If the patient's test is successfully found, a success status code 200 and the ECG test is sent back to the client; otherwise, an error status code 400 with specific error information is returned to the client.

4.3.5 ECG Raw Data Section

Obtaining ECG raw data sets from ECG test resource

The goal of this request is to obtain a patient's ECG raw data. The request contains the patient id and test id; then, the request is sent to the patients resource in the server program by the GET method. If the patient's test is successfully found, a success status code 200 and ECG raw data is sent back to the client; otherwise, an error status code 400 with specific error information is returned to the client.

Mobile APP sends ECG raw data to the server

The goal of this request is that the server receives ECG raw data from a mobile APP. The request contains the patient id and test id; then, the request is sent to the data resource in the server program by the POST method. If the data is successfully sent to the server, a success status code 200 is sent back to the APP; otherwise, an error status code 400 with specific error information is returned to the APP.

4.4 Conclusions

In this chapter, we explain our proposed model's details, and present the workflow in this system. The server's structure and layers are illustrated with a specific diagram. We explain the functionalities of each layer and how the central server keeps the system running functionally. After that, we emphasize the importance of designing APIs to implement interactions in the server. We introduce the REST concepts and how these concepts can improve our APIs quality, and we provide examples to explain REST standards. Eventually, we present our proposed REST APIs design. We discuss the idea, requirements, and return information of each HTTP request. Our proposed system is still in the development phase, so the APIs may change in the future to meet new users' requirements. However, our proposed APIs' core design ideas are properly introduced in this chapter. The proposed API framework can also be implemented in other remote ECG monitoring systems to help the development of ECG remote monitoring software. Some aspects of our API design document are available in the Appendix section.

Chapter 5

Conclusions

This thesis studies deep learning approaches for electrocardiogram arrhythmia classifications and API design for constructing a remote ECG monitoring system. We have proposed a two-dimensional convolutional neural network solution for classifying patients' abnormal ECG heartbeats using the inter-patient paradigm. We have introduced our remote ECG monitoring system with the proposed application programming interface design. This chapter concludes the research results of these topics and discusses some future research works.

5.1 Inter-Patient ECG Classification Using Deep Convolutional Neural Networks

Many existing solutions are based on one-dimensional convolutional neural network due to difficulties in pre-processing. This thesis has proposed a two-dimensional CNN-based ECG arrhythmia heartbeat classifier using the inter-patient scheme. The inter-patient scheme, where patients in training set differ from patients in testing set, has ensured the proposed model closely reassembles real life applications. The proposed pre-processing algorithm has successfully generated input images from one-dimensional ECG recordings, which mimics doctors' diagnosing processes. By adjusting CNN layers and hyper-parameters based on input images and output classes, the proposed model has improved classification rates. With careful handle of pre-processing, two-dimensional CNN has been proved to be an effective solution for ECG arrhythmia classification.

5.2 ECG REST API Design

In this section, we have proposed the remote ECG monitoring system. The practical structure of the server program has been shown to be able to correctly process all requests from clients. The REST APIs have provided a straightforward approach for implementing interactions between modules in the system. We have observed that a well-designed APIs documentation can improve software development efficiency and present clear requirements for programs.

5.3 Future Work

These are future research plans.

- The proposed model has relatively low performance on four-class classifications. In the future, we will keep modifying the model's structure to achieve better classification results on four-class classifications. We will test the model with more databases accurately evaluate our model's performance by comparing different experimental results.
- The remote ECG monitoring system will connect to more clients in the future. The server will process more HTTP requests from different applications. Therefore, URIs naming will need to be carefully determined and we will carefully design APIs to avoid conflicts between different requests. User feedback will be collected to improve our system.

Bibliography

[1] I. E. Naqa and M. J. Murphy. What Is Machine Learning?. *Machine Learning in Radiation Oncology*, pages 3-11, 2015.

[2] S. Hochreiter. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *World Scientific Publishing Co., Inc.*, vol. 2, pages 107-116, 1998.

[3] M. Shafi. et al. 5G: A Tutorial Overview of Standards, Trials, Challenges, Deployment, and Practice. *IEEE Journal on Selected Areas in Communications*, vol.5, no. 6, pages 1201-1221, 2017.

[4] L. Bassett. ntroduction to JavaScript object notation: a to-the-point guide to JSON. *O'Reilly Media, Inc*, 2015.

[5] L. Zaorálek, J. Platoš, and V. Snasel. PATIENT-ADAPTED AND INTER-PATIENT ECG CLASSIFICATION USING NEURAL NETWORK AND GRADIENT BOOSTING. *Neural Network World*, vol. 28, pages 241-254, 2018.

[6] Li Guo, Gavin Sim, and Bogdan Matuszewski. Inter-Patient ECG Classification with Convolutional and Recurrent Neural Networks. *Biocybernetics and Biomedical Engineering*, vol. 39, pages 868-879, 2019.

[7] P. de Chazal, M. O'Dwyer, and R.B. Reilly. Automatic Classification of Heartbeats Using ECG Morphology and Heartbeat Interval Features. *IEEE transactions on bio-medical engineering*, vol. 51, pages 1196-1206, 2004.

[8] Association for the Advancement of Medical Instrumentation and American National Standards Institute. Testing and reporting performance results of cardiac rhythm and st segment measurement algorithms, ANSI/AAMI EC57:2012.

[9] A. Goldberger, L. Amaral, L. Glass, J. Hausdorff, PC. Ivanov, R. Mark, JE. Mietus, GB. Moody, CK. Peng, HE. Stanley. PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals. Circulation [Online]. vol. 101, pages e215–e220.

[10] A. Khan, A. Sohail, U. Zahoora. et al. A survey of the recent architectures of deep convolutional neural networks. *Artif Intell Rev*, 2020.

[11] J. Takalo-Mattila, J. Kiljander and J. Soininen. Inter-Patient ECG Classification Using Deep Convolutional Neural Networks. *21st Euromicro Conference on Digital System Design (DSD)*, pages 421-425, 2018.

[12] S. Kiranyaz, T. Ince, R. Hamila and M. Gabbouj. Convolutional Neural Networks for patient-specific ECG classification. *37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 2608-2611, 2015.

[13] J. Gordon and E. H. Shortliffe. The Dempster-Shafer theory of evidence. *Readings in uncertain reasoning*, pages 529-539, 1990.

[14] C. Szegedy et al. Going deeper with convolutions. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1-9, 2015.

[15] K. Simonyan and A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. *CoRR*, 2014.

[16] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015

[17] C. Walls, R and Breidenbach. Spring in action. *Dreamtech Press*, 2005.

[18] A. M. Magdaleno, C. M. L. Werner and R. M. de Araujo. Reconciling software development models: A quasi-systematic review. *Journal of Systems and Software*. vol. 85, issue. 2, pages 351-369, 2012.

[19] C. Ye, B. V. K. Vijaya Kumar and M. T. Coimbra. Heartbeat Classification Using Morphological and Dynamic Features of ECG Signals. *IEEE Transactions on Biomedical Engineering*, vol. 59, no. 10, pages 2930-2941, 2012.

[20] A. S. Alvarado, C. Lakshminarayan and J. C. Principe. Time-Based Compression and Classification of Heartbeats. *IEEE Transactions on Biomedical Engineering*, vol. 59, no. 6, pages 1641-1648, 2012.

[21] T. Mar, S. Zaunseder, J. P. Martínez, M. Llamedo and R. Poll. Optimization of ECG Classification by Means of Feature Selection. *IEEE Transactions on Biomedical Engineering*, vol. 58, no. 8, pages 2168-2177, 2011.

[22] M. M. Al Rahhal, Y. Bazi, H. AlHichri, N. Alajlan, F. Melgani, and R. R. Yager. Deep learning approach for active classification of electrocardiogram signals. *Inf. Sci*, vol. 345, pages 340–354, 2016.

[23] G. Garcia, G. Moreira, D. Menotti. et al. Inter-Patient ECG Heartbeat Classification with Temporal VCG Optimized by PSO. *Scientific Reports*, pages 10543, 2017.

[24] Z. Zhang, J. Dong, X. Luo, KS. Choi, X. Wu. Heartbeat classification using disease-specific feature selection. *Comput Biol Med*, vol. 46, pages 79–89, 2014.

[25] K. Luo, J. Li, Z. Wang, and A. Cuschieri. Patient-Specific Deep Architectural Model for ECG Classification. *Journal of Healthcare Engineering*, 2017.

[26] L. Zaoralek, J. Platos, and V. Snásel. Patient-adapted and inter-patient ecg classification using neural network and gradient boosting. *Neural Network World*, vol. 28, pages 241-254, 2018.

[27] S. Kiranyaz, T. Ince and M. Gabbouj. Real-Time Patient-Specific ECG Classification by 1-D Convolutional Neural Networks. *IEEE Transactions on Biomedical Engineering*, vol. 63, no. 3, pages 664-675, 2016.

[28] M. Llamedo, JP. Martinez. Heartbeat classification using feature selection driven by database generalization criteria. *IEEE Trans Biomed Eng*, vol. 58, no. 3, pages 616-625. 2011.

[29] S. Tilko. A brief introduction to REST. Dec 10, 2007.

[30] World Health Organization. Cardiovascular diseases (CVDs). https://www.who.int/news-room/fact-sheets/detail/cardiovascular-diseases-(cvds), May 2017. Last accessed on 2020-09-19.

[31] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. Corrado, A. Davis, J. Dean, M. Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. arXiv preprint arXiv:1603.04467, 2016.

[32] G. Litjens, T. Kooi. et al. A survey on deep learning in medical image analysis. *Medical Image Analysis*, vol. 42, pages 60-88, 2017.

[33] S. Shanmuganathan. Artificial Neural Network Modelling: An Introduction. *Artificial Neural Network Modelling*, pages 1-14, 2016.

[34] F. Jia, Y. Lei, J. Lin, X. Zhou and N. Lu. Deep neural networks: A promising tool for fault characteristic mining and intelligent diagnosis of rotating machinery with massive data. *Mechanical Systems and Signal Processing*, vol. 72-73, pages 303-315, 2016.

[35] I. Odinaka, P. Lai, A.D, Kaplan, J. A. O'Sullivan, E. J. Sirevaag and J. W. Rohrbaugh. ECG Biometric Recognition: A Comparative Analysis. *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 6, pages 1812-1824, 2012.

[36] Jen Hong Tan, Yuki Hagiwara, Winnie Pang, Ivy Lim, Shu Lih Oh, Muhammad Adam, Ru San Tan, Ming Chen, U. Rajendra Acharya. Application of stacked convolutional and long short-term memory network for accurate identification of CAD ECG signals. *Computers in Biology and Medicine*, vol. 94, pages 19-26, 2018. 37

[37] I. Odinaka, P. Lai, A. D. Kaplan, J. A. O'Sullivan, E. J. Sirevaag and J. W. Rohrbaugh. ECG Biometric Recognition: A Comparative Analysis. *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 66, pages 1812-1824, 2012.

[38] S. Sachin Kumar, Neethu Mohan, P. Prabaharan, K.P. Soman. Total Variation Denoising Based Approach for R-peak Detection in ECG Signals. *Procedia Computer Science*, vol. 93, pages 697-705, 2016.

[39] D. Lucani, G. Cataldo, J. Cruz, G. Villegas, S. Wong. A portable ECG monitoring device with Bluetooth and Holter capabilities for telemedicine applications. *Conf Proc IEEE Eng Med Biol Soc*, pages 5244-5247, 2006.

[40] T. Jeon, B. Kim, M. Jeon, et al. Implementation of a portable device for real-time ECG signal analysis. *BioMed Eng OnLine*, 13(1):160, 2014.

[41] L. Kai. et al. A system of portable ECG monitoring based on Bluetooth mobile phone. *2011 IEEE International Symposium on IT in Medicine and Education, Cuangzhou*, pages 309-312, 2011.

[42] A. Bansal, R. Joshi. Portable out-of-hospital electrocardiography: A review of current technologies. *J Arrhythm*, pages 129-138, 2018.

[43] C. C. Lin and C. M. Yang. Heartbeat Classification Using Normalized RR Intervals and Morphological Features. *Mathematical Problems in Engineering*, pages 1-11, 2014.

[44] A. Khazaee and A. E. Zadeh. ECG beat classification using particle swarm optimization and support vector machine. *Frontiers of Computer Science*, vol. 8, pages 217-231, 2014.

[45] S. Faziludeen and P. Sankaran. ECG Beat Classification Using Evidential K -Nearest Neighbours. *Procedia Computer Science*, vol. 89, Pages 499-505, 2016

[46] M. Korürek and B. Doğan. ECG beat classification using particle swarm optimization and radial basis function neural network. *Expert Systems with Applications*, vol. 37, issue. 12, pages 7563-7569, 2010.

[47] G. de Lannoy, D. Francois, J. Delbeke and M. Verleysen. Weighted conditional random fields for supervised interpatient heartbeat classification. *IEEE Trans Biomed Eng*, vol. 59, no. 1, pages 241-247, 2012.

[48] T. Mar, S. Zaunseder, J. P. Martínez, M. Llamedo and R. Poll. Optimization of ECG Classification by Means of Feature Selection. *IEEE Transactions on Biomedical Engineering*, vol. 58, no. 8, pages 2168-2177, 2011.

[49] Ö. Yıldırım, P. Pławiak, R. Tan and U. R. Acharya. Arrhythmia detection using deep convolutional neural network with long duration ECG signals. *Computers in Biology and Medicine*, vol. 102, pages 411-420, 2018.

[50] M. Zubair, J. Kim and C. Yoon. An Automated ECG Beat Classification System Using Convolutional Neural Networks. *2016 6th International Conference on IT Convergence and Security*, pages 1-5, 2016.

[51] Z. Zidelmal, A. Amirou, D. O. Abdeslam, J. Merckle. ECG beat classification using a cost sensitive classifier. *Comput Methods Programs Biomed*, vol. 111, no. 3, pages 570-577, 2013.

[52] F. Zhu, F. Ye, Y. Fu. et al. Electrocardiogram generation with a bidirectional LSTM-CNN generative adversarial network. *Scientific Reports*, vol. 9, 2019.

[53] Z. Xiong, M. K. Stiles and J. Zhao. Robust ECG signal classification for detection of atrial fibrillation using a novel neural network. *2017 Computing in Cardiology*, pages 1-4, 2017.

[54] U. R. Acharya, H. Fujita, O. S. Lih, Y. Hagiwara, J. H. Tan and M. Adam. Automated detection of arrhythmias using different intervals of tachycardia ECG segments with convolutional neural network. *Information Sciences*, vol. 405, pages 81-90, 2017.

[55] Q. Yao, R. Wang, X. Fan, J. Liu and Y. Li. Multi-class Arrhythmia detection from 12-lead varied-length ECG using Attention-based Time-Incremental Convolutional Neural Network. *Information Fusion*, vol. 53, pages 174-182, 2020.

[56] J. Li, Y. Si, and T. Xu. Deep Convolutional Neural Network Based ECG Classification System Using Information Fusion and One-Hot Encoding Techniques. *Mathematical Problems in Engineering*, pages 1-10, 2018.

[57] A. Sheetal, H. Singh and A. Kaur. A. QRS detection of ECG signal using hybrid derivative and MaMeMi filter by effectively eliminating the baseline wander. *Analog Integr Circ Sig Process*, vol. 98, pages 1-9, 2019.

[58] G. Van Rossum and FL. Drake. Python 3 Reference Manual.*Scotts Valley*, CA: CreateSpace; 2009.

[59] J. D. Hunter. Matplotlib: A 2D graphics environment. *Computing in Science & amp; Engineering*, vol.9, pages 90-95, 2007.

[60] M. Widenius, D. Axmark and P. DuBois. Mysql Reference Manual, 2002.

[61] K. Arnold, J. Gosling, D. Holmes. The Java programming language. *Addison Wesley Professional*, 2005.

[62] M. W. Gardnera1 and S. R. Dorling. Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences. *Atmospheric environment*, vol. 32, pages 14-15, 1998.

[63] A. Thomas. An introduction to entropy, cross entropy and KL divergence in machine learning. https://machinelearningmastery.com/cross-entropy-for-machine-learning/, October 2019. Last accessed on 2020-09-19.

[64] Pedregosa. et. al. Scikit-learn: Machine learning in Python. *Journal of machine learning research*, pages 2825-2830, 2011.

Appendix A

Python code of Inter-patient ECG Classification Using Deep Convolutional Neural Networks

```
for i in range(num_images):
    current_length =len(signals[i])
    total_length=current_length+total_length

avg_length=math.ceil(total_length/num_images)
avg_length=math.ceil(avg_length*0.3)+avg_length


for i in range(num_images):

    if labels[i] in N :

        fig = plt.figure(frameon=False)
        plt.plot(signals[i],linewidth=3.9)
        plt.xticks([]), plt.yticks([])
        for spine in plt.gca().spines.values():
            spine.set_visible(False)

        imgIndex=get_image_num_by_type('N')

        filename = path_name+'/'+'N'+'/'+'N'+str(imgIndex)+'.png'
        fig.savefig(filename)
        plt.close(fig)
```

```
if labels[i] in SVEB :

    fig = plt.figure(frameon=False)
    plt.plot(signals[i],linewidth=3.9)
    plt.xlim([-1,(avg_length+1)])
    plt.xticks([]), plt.yticks([])
    for spine in plt.gca().spines.values():
        spine.set_visible(False)

    imgIndex=get_image_num_by_type('SVEB')

    filename = path_name+'/'+'SVEB'+'/'+'SVEB'+str(imgIndex)+'.png'
    fig.savefig(filename)
    plt.close(fig)


if labels[i] in VEB :

    fig = plt.figure(frameon=False)
    plt.plot(signals[i],linewidth=3.9)
    plt.xlim([-1,(avg_length+1)])
    plt.xticks([]), plt.yticks([])
    for spine in plt.gca().spines.values():
        spine.set_visible(False)

    imgIndex=get_image_num_by_type('VEB')

    filename = path_name+'/'+'VEB'+'/'+'VEB'+str(imgIndex)+'.png'
    fig.savefig(filename)
    plt.close(fig)
```

```
model = Sequential()

model.add(Conv2D(64, (3, 3), input_shape=input_shape))
model.add(BatchNormalization())
model.add(Activation('relu'))


model.add(Conv2D(64, (3, 3)))
model.add(BatchNormalization())
model.add(Activation('relu'))


model.add(MaxPooling2D(pool_size=(2, 2)))


model.add(Conv2D(128, (3, 3)))
model.add(BatchNormalization())
model.add(Activation('relu'))


model.add(Conv2D(128, (3, 3)))
model.add(BatchNormalization())
model.add(Activation('relu'))


model.add(MaxPooling2D(pool_size=(2, 2)))


model.add(Conv2D(256, (3, 3)))
model.add(BatchNormalization())
model.add(Activation('relu'))


model.add(MaxPooling2D(pool_size=(2, 2)))
```

```
model.add(Flatten())
model.add(Dense(2048))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(Dropout(0.5))


model.add(Dense(2048))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(Dropout(0.5))


model.add(Dense(2, activation='softmax'))

model.compile(loss='categorical_crossentropy',
              optimizer=tf.keras.optimizers.Adam(0.001), metrics=['accuracy'])
```

Appendix B

Remote ECG Monitoring Software API document

Path Parameter	
clinic	Refers to the clinic's name that uses the ecg software

Required JSON Body

email	String
password	String

Response Body

Http status code-200

```
{
        "success":  "the user has successfully logged into the software"
        "nurse-id":  2
}
```

Parameter	
clinic	Refers to the clinic's name that uses the ecg software

Request body

lastname	String
midname	String
firstname	String
phone	Integer
clinic-id	Integer
password	String
email(username)	String

Response Body(email= username)

Http status code-201

```
{
        "success":  "the new nurse account has successfully created",
        "nurse id":  2
}
```

Parameter	
clinic	Refers to the clinic's name that uses the ecg software

Request body

email	String

Response Body

Http status code-200

```
{
        "The nurse email address has successfully sent to the server, the server will send
         a reset password link to the nurse email"
 }
```

Parameter	
clinic	Refers to the clinic's name that uses the ecg software

Query parameter	
lastname	The last name of the patient
birth	The birthday of the patient, format:YYYYMMDD
phn(optional)	The patient's personal health insurance number

Response Body

Http status code-200


```
[       {
        "patient-id": 1,
        "lastname": "Steve",
        "midname": "",
        "firstname": "Kin",
        "birthday": "1982/03/22",
        "address1": "122 Test Ave",

        "province": "BC",
        "city": "Victoria",
        "email": "test1@gmail.com"
        "phn": "s123456789",
        "phone-number": "2501234567",
        "work-number": "",
        "home-number": "",
        "gender": "male",
        "postcode": "T1R 6V2",
        "deleted": fasle,
        "clinic_id": 1,
        "pacemaker": 1,
        "supervising_physician": "content",
        "age": 19,
        "weight": "weight",
        "height": "height",
        "indications": "content",
        "medications": "content",
        "referring_physician": "content",
        "remark": "content",
        },
        ...
    ]
```

Parameter	
clinic	Refers to the clinic's name that uses the ecg software

Request Body

patient-id	Integer
lastname	String
midname	String
firstname	String
birthday	String
address1	String
postcode	String
province	String
city	String
email	String
phn	String
phone-number	String
work-number	String
home-number	String
gender	String

Response Body

Http status code-201

```
{
        "success":  "the patient's information has successfully created",
        "patient-id":  2
}
```

Parameter	
clinic	Refers to the clinic's name that uses the ecg software
patient-id	The patient's id

Request Body

patient-id	Integer
lastname	String
midname	String
firstname	String
birthday	String
address1	String
postcode	String
province	String
city	String
email	String
phn	String
phone-number	String
work-number	String
home-number	String
gender	String

Response Body

Http status code-201

```
{
        "success":  "the patient's information has successfully created",
        "patient-id":  2
}
```

Path Parameter	
clinic	Refers to the clinic's name that uses the ecg software
patient-id	The patient id

Response Body

Http status code-200

```
[
        {
                "ecg-test-id":  1,
                "start-time": "6 May 2019 22:41:09 UTC 8",
                 "scheduled-end-time": "7 May 2019 22:42:09 UTC 8",
                "actual-end-time":  "7 May 2019 22:42:09 UTC 8",
                 "deleted": false
                "patient-id":1
                "nurse-id":1
                "phone-id":1
                "device_id": 1
                "clinic_id": 1
                 "comment": "comment on the ecg test"

        }
        ...
        ...
        ...

]
```

Path Parameter	
clinic	Refers to the clinic's name that uses the ecg software
patient-id	The patient id
ecg-test-id	The ecg test id

Response Body

Http status code-200

```
[
        {
                "ecg-test-id":  1,
                "start-time": "6 May 2019 22:41:09 UTC 8",
                 "scheduled-end-time": "7 May 2019 22:42:09 UTC 8",
                "actual-end-time":  "7 May 2019 22:42:09 UTC 8",
                 "deleted": false
                "patient-id":1
                "nurse-id":1
                "phone-id":1
                "device_id": 1
                "clinic_id": 1
                 "comment": "comment on the ecg test"

        }

]
```

Path Parameter	
clinic	Refers to the clinic's name that uses the ecg software
patient-id	The patient id
ecg-test-id	The ecg test id

Query Parameter	
patient-id	The patient id.
page	Divide the data set into each individual pages, The default is 1 if no page number indicated in the URI. Page size 50
page-size	The number of entries to return. The default value is 50 if no page_size number indicated in the URI. If page-size greater than 50, automatically convert the page_size value to 50

Response Body

Http status code-200

```
[
        {
                "ecg-raw-data-id":  1,
                "ecg_test_id":1,
                 "received-time":  "6 May 2019 22:41:09 UTC 8",
                "ecg-raw-data":  (the ecg raw data that is received from ecg devices ),
                "deleted": false
                "start-time": "6 May 2019 22:41:09 UTC 8",
                 "end-time": "6 May 2019 22:42:09 UTC 8"
                 "clinic-id": 1

        }
        ...

]
```

Path Parameter	
clinic	Refers to the clinic's name that uses the ecg software

Request Header

Authorization uses **verification code**

Request Body

```
[
        {
                "startTime": "2019-07-27T12:00:00.000-08",
                "endTime":  "2019-07-27T12:00:00.000-08",
        }

        (object) file
         The ecg raw data file
]
```

*** Time format: "yyyy-mm-ddThh:mm:ss.SSS+xx" and "+xx" or "-xx" is the time zone difference between local and UTC; for example, PST is -08***