ECG Classification and the "Heart Age" Prediction using Machine learning

Yanting Shen

Trinity College

University of Oxford

A thesis submitted for the degree of

Doctor of Philosophy

Trinity 2020

Declaration

This dissertation is the result of my own work and includes nothing which is the outcome of work done in collaboration. It is not substantially the same as any that I have submitted, or am concurrently submitting, for a degree or diploma or other qualification at the University of Oxford or any other University or similar institution. I further state that no substantial part of my dissertation has already been submitted, or is being concurrently submitted, for any such degree, diploma or other qualification at the University of Oxford or any other University or similar institution. This dissertation does not exceed the prescribed limit of 40,000 words.

Acknowledgements

I want to offer my sincere gratitude to everyone who has helped me during my DPhil journey. Without you, this work would not be possible. I would like to especially thank my advisor, Prof. David A. Clifton, for his immense patience, encouragement, confidence, and trust in me, the freedom in my research directions, and the support for my entrepreneurial endeavours; co-supervisor Dr Tingting Zhu for many insightful discussions, proofreading of my writings, and being more than my co-supervisor but also a great friend in life and supporting me at my most difficult times; co-supervisor Prof. Robert Clarke for his kindness, patience, and deep knowledge in epidemiology and cardiology and proofreading many of my writings; and my first-year co-supervisor Dr Yang Yang for many insightful discussions and fun conversations outside of research.

Additionally, I would like to thank Prof. Zhengming Chen for his expertise in epidemiology, Prof. Sarah Parish for her expertise in statistics, Dr Marco Pimentel for his expertise in signal processing and his in-house software, Dr Girmaw Abebe Tadesse for his expertise in CNN. I have learned so much from you. Additional thanks go to Corinne Prescott and Alison Lewis for proofreading my manuscripts, and Dr Xinshao Wang for collaboration on publications; and folks in time series terrain: Heloise Greeff, Peter Gyring, Rasheed El-Bouri, Dr Farah Shamout, Dr Drew Birrenkott, Dr Glen Wright Colopy, Dr Kate Niehaus, Dr Rebecca Pullon, and Dr Johanna Ernst for many geeky conversations and lunchtime laughter at ORCRB atrium.

Being in the CDT Healthcare Innovation cohort offered me many unique experiences. I have developed a strong bonding with my cohort mates: Dr Brian Chu, Dr Cameron Higgins, Dr Estelle Beguin, Dr John Prince, Dr Kirubin Pillay, Dr Magda Abbas, Dr Navin Cooray, Dr Nicolas Basty, Dr Oliver Carr, Dr Phurit (Mookie) Bovornchutichai, Dr Sittichok (Bik) Chaichulee, Dr Yifan Cai, and Dr Yuan Gao, especially Magda, Mookie, and Bik who have become my closest friends at Oxford.

Special thanks to my friends at Oxford outside CDT: my 'good Chinese neighbour' at Trinity College: Qingyu Wang, Dr Hongjia Zhang, Lise Andersen, Dr Martin-Immanuel Bittner, Dr Tom Fleming, Dr Poppy Roworth, Shengyu Wang for making my social life so enjoyable.

Special thanks go to my business partners who have made my Oxford time truly exciting and unforgettable: Julian Lindloff, Dr Mun Ching Lee, Dr Martin-Emmanuel Bittner, Dr Tom Fleming, and Dr Sam Murphy. We become life long friends, even though our startups RIP. Special thanks to my colleague Sankranti Joshi at AIG who has also become one of my best friends and has helped me through thick and thin and helped me turn my spaghetti jupyter notebooks into something "classic".

And last but not least, thank you, my parents, for always loving me and being there for me. I love you.

To everyone above and many whose names are mistakenly missed, it is my greatest honour to have worked with you and have you in my life.

Abstract

Cardiovascular disease (CVD) is the leading cause of morbidity and mortality worldwide. Electrocardiogram (ECG) is an important clinical measurement of cardiac activity. The major challenge in incorporating ECG time-series into CVD risk metrics is extracting features and classifying the ECG time-series into appropriate ECG abnormality groups. Therefore we set out to use machine learning to address this challenge.

We used machine learning to analyse 12-lead, 500Hz, 10-s electrocardiogram (ECG) data provided by the Mortara device to perform ECG signal classification in a large cohort study of 25,019 participants in the China Kadoorie Biobank (CKB). We compared the performance of 11 representative traditional machine learning algorithms for four-class classification of normal, "arrhythmia", "ischemia", and "hypertrophy". We extracted 72 novel features and improved the 4-class classification accuracy from 53.5% using only Mortara features to 77.3%. We demonstrated that machine learning models could classify ECG with high accuracy without any knowledge of the diagnosis criteria, and the top features identified by the best model (SGB-F84) were very different from the ones commonly used in clinic.

We further proposed a novel neural network architecture family - the Layer-wise Convex Network (LCN), and a neural architecture search algorithm - the AutoNet, to classify the ECG raw signals end-to-end without signal denoising, preprocessing, nor feature extraction. We benchmarked the AutoNet-LCN with the state-of-the-art ResNet-based model on three datasets: CKB, PhysioNet, and ICBEB. The AutoNet generated LCNs has no more than 2% of the parameters compared to the state-of-the-art architectures, outperformed the latter on all three datasets by a wide margin (9-16% improvement in terms of F1 score) within 2 hours of architecture search time, in comparison to weeks to months of trial-and-error by human researchers in the conventional deep learning model development process. The neural networks found by AutoNet-LCN are robust to varying noise levels, ECG signal length, sampling frequency, number of leads, amplitude scale, ECG abnormality types, and cohort sizes of the study populations.

Finally, to address the issue that the labels in the CKB were provided by the deterministic rule-based Minnesota code, which in theory can be approximated to arbitrary precision by a neural network, we proposed a novel paradigm: learning from

alternative labels. We provided proof-of-concept by predicting the participants' age from the 10-s ECG waveforms in the CKB dataset using AutoNet-LCN. We trained the AutoNet-LCN on the normal population and tested on the normal, "arrhythmia", "ischemia", and "hypertrophy" classes. We developed the gender-agnostic model as well as the gender-stratified mode, achieving mean absolute error of 5.7 years (R^2 = 44.1%), 5.6 years (R^2 = 45.4%), and 6.2 years (R^2 = 34.7%) for gender agnostic, female, and male models in the normal class, respectively. The absolute deviation of the predicted "heart age" from one's chronological age suggests higher CVD risks, and a high "heart age" was associated with "hypertrophy", "ischemia", and "hypertension", while a low "heart age" was associated with "arrhythmia" and "hypotension". The "heart age" may be considered as an intuitive risk score for cardiovascular health and warrants further study of its associations with different CVD outcomes.

Contents

List of Figures

List of Tables

List of Abbreviations

AF . . . . . Atrial fibrillation

AR . . . . . auto-regression

BiLSTM . . . Bidirectional Long Short-term memory

CAD . . . . . coronary artery disease

CHF . . . . . congestive heart failure

CNN . . . . . convolutional neural network

CVD . . . . . cardiovascular disease

ECG . . . . . electrocardiogram

DBN . . . . . deep belief network

DWT . . . . . discrete wavelet transform

ELM . . . . . extreme learning machine

FNN . . . . . feed forward neural network

GMM . . . . . Gaussian mixture model

HBT . . . . . Hilbert-Huang Transform

HRV . . . . . heart rate variability

ISTC . . . . . ischemic ST changes

LQTS . . . . . long QT syndrome

LSTM . . . . . long short-term memory

MI . . . . . myocardial infarction

NA-MEMD . noise-assisted multivariate empirical model decomposition

NFIN . . . . . neural fuzzy inference network

NN . . . . . neural network

PNN . . . . . probabilistic neural network

PVC . . . . . premature ventricular contraction

RCR . . . . . resuscitation cardiac rhythms

ResNet . . . . residual network

SAE . . . . . sparse auto-encoder

SCD . . . . . sudden cardiac death

STFT . . . . . short-term Fourier Transform

SVEB . . . . . supra-ventricular ectopic beat

SVM . . . . . support vector machine

VF . . . . . ventricular fibrillation

VT . . . . . ventricular tachycardia

List of Mathematical Notations

$\mathbf{0}$	a tensor with all 0 elements
$\boldsymbol{a}$	the Lagrange multiplier
$\boldsymbol{A}^{[l]} \in \mathbb{R}^{n^{[l]} \times m}$	the activation of layer l
b	the bias parameter
$\tilde{A}$	$[\mathbf{1}; \boldsymbol{A}]$
$\boldsymbol{b} \in \mathbb{R}^{n^{[l]}}$	the bias vector of layer l
$\mathbb{B}$	Boolean domain
C_0	class 0 (in binary classification)
C_1	class 1 (in binary classification)
D	the dimension of a single feature vector
$E(\cdot)$	the loss/error function
$f : \boldsymbol{X} \mapsto \boldsymbol{Y}$	the data generating process
f_c	the number of channels in a convolutional kernel
f_h	the height of the kernel in a convolutional layer
f_w	the width of the kernel in a convolutional layer
f_s	sampling frequency
$g^{-1}(\cdot)$	the inverse function of $g(\cdot)$
$g(\cdot)^{[l]}$	the activation function of layer l
$\mathcal{H}$	hypothesis domain
$\boldsymbol{H}$	the Hessian matrix
k	the kernel size of an 1-D convolutional layer
$k(\cdot, \cdot)$	kernel function
l	the layer index of a neural network
m	the number of training examples
n	sample size

$\mathbb{N}$	natural number domain
$n^{[l]}$	the number of neurons in layer l of a neural network
n_c	the number of input channels to a convolutional layer
n_f	the number of filters in a CNN layer
n_h	the height of the input tensor to a convolutional layer
$n_{maxpool}$	the number of maxpooling layers in LCN
n_{repeat}	the number of convolutional layers
n_w	the width of the input tensor to a convolutional layer
$n_{\boldsymbol{\theta}}$	the number of parameters to stack between 2 maxpooling layers in LCN
$\mathcal{N}$	Gaussian distribution
$p(\cdot)$	the probability density function
p	the padding hyperparameter of a convolutional layer
s	the stride hyperparameter of a convolutional layer
$\boldsymbol{t} \in \{0,1\}^K$	the one-hot-encoded true label of a single training example in K-class classification
$\mathcal{U}$	the uniform distribution
$\boldsymbol{w}$	the weight parameters
$\boldsymbol{W}^{[l]} \in \mathbb{R}^{n^{[l]} \times n^{[l-1]}}$	the weight matrix between layer $l-1$ and layer l
$x_i \in \mathbb{R}$	the value of the ith feature of a feature vector
$\boldsymbol{x} \in \mathbb{R}^D$	a training example/feature vector
$\boldsymbol{x}_i \in \boldsymbol{R}^D$	the ith training example/feature vector
$\boldsymbol{X} \in \mathbb{R}^{D \times m}$	the design matrix
$y \in \mathbb{R}$	the true target of one training example
$\hat{y} \in \mathbb{R}$	the predicted target
$\boldsymbol{Y} \in \mathbb{R}^m$	the true target/label of the training set
$\hat{\boldsymbol{Y}} \in \mathbb{R}^m$	the predicted target of the training set
$\boldsymbol{Z}^{[l]}$	the affine transformation of the activations of layer $l-1$
α	the learning rate
α_0	the initial learning rate
$\boldsymbol{\theta}$	the learnable parameters of a model

ϵ	a small positive real scalar
$\boldsymbol{\lambda} = \{\lambda_i\}$	the eigenvector(s) of a matrix
μ	the uni-variate mean
$\boldsymbol{\mu} \in \mathbb{R}^D$	the multivariate mean
σ	the uni-variate standard deviation
$\boldsymbol{\sigma} \in \mathbb{R}^D$	the multivariate standard deviation
$\sigma(\cdot)$	the sigmoid function
$\boldsymbol{\Sigma} \in \mathbb{R}^{D \times D}$ . . .	the multivariate covariance matrix
τ	the timescale hyperparameter of LCN
$\boldsymbol{\phi}(\cdot)$	the basis function
ξ	the slack hyperparameter in soft-margin SVM

Unless stated otherwise, an unbold letter, capitalised or not, represents a scalar; a bold capitalised letter represents a matrix or a tensor (with more than two axes). A function with a scalar output is represented by an unbold letter; a function with a tensor output is represented by a bold letter. In neural networks, the superscript square bracket indexes the layer. For example, the number of neurons in layer l is denoted as $n^{[l]}$.

1 Introduction

1.1 Clinical Need

Cardiovascular diseases (CVDs), including coronary heart disease, cerebrovascular disease, peripheral arterial disease, rheumatic heart disease, congenital heart disease, deep vein thrombosis, and pulmonary embolism, are the leading causes of mortality worldwide and also in China. There are large geographical differences in CVD mortality rates in China, suggesting appropriate measures may be taken to prevent and effectively treat the disease. Identifying risk factors for CVD in the Chinese population could help in providing advice on lifestyle changes and enable clinicians to discover appropriate treatments for specific CVD outcomes to reduce mortality and healthcare expenditure. Many risk factors have been identified by long-term prospective studies, such as obesity (Arsanjani, Dey, et al. 2015), diabetes (Association et al. 2007), metabolic syndrome (Dekker et al. 2005), smoking (Tracy et al. 1997), hypertension (Colombet et al. 2000), and genetic risk factors (Gamberger, Lavrač, and Krstačić 2003).

Electrocardiogram (ECG) is a widely used screening and diagnostic tool for CVD, and ECG findings are additional CVD risk factors included in large cohort studies. For example, in the Framingham study, Kannel et al. reported that left ventricular hypertrophy on ECG tracings in asymptomatic adults was a strong

predictor of cardiac morbidity and mortality (Kannel, Gordon, and Offutt 1969); in another study of 12,142 patients with symptoms of cardiac ischaemia at rest and signs of myocardial ischaemia confirmed by ECG within 12 hours of admission, 22% had T wave inversion, 28% had ST-segment elevation, 35% had ST-segment depression, and 15% had a combination of ST-elevation and depression (Kannel, Anderson, et al. 1987). Savonitto demonstrated the use of ECG at presentation allowed immediate risk stratification of patients across the spectrum of the acute coronary syndrome (Kannel, Anderson, et al. 1987). Individual markers on ECG, such as spatial QRS angle has also been shown to be a stronger predictor of cardiac mortality than the conventional CVD risk factors in older population and provides additional value for prediction of fatal cardiac events (Kannel, Anderson, et al. 1987; Savonitto et al. 1999).

However, the interpretation of ECG requires clinical knowledge, which is subject to substantial inter-personal variation and human error. Computerised ECG has been developed to aid this process and is typically based on rule-based coding schemes such as the Minnesota Code (Prineas, Crow, and Z.-M. Zhang 2009), which was first developed in the 1960s, and few modifications were made since then. Minnesota Codes use common heuristics and fixed voltage and duration thresholds for diagnosis. For example, the Minnesota Code for high left R amplitude patterns is "if any of the following criteria are present: R amplitude $>$ 26 mm in either lead V5 or V6; R amplitude $>$ 20mm in any of leads I, II, III, have (see figure confirmation report 6.2); R amplitude $>$ 12mm in lead aVL" (Prineas, Crow, and Z.-M. Zhang 2009). This approach has a limited accuracy as it does not take interpersonal variation and signal quality into consideration. However, it can be normal for lean individuals as the electrodes are closer to the heart, and tall QRS may not be observed on the ECG of obese patients. An absolute voltage threshold would result in false-positive indications of sudden death for the lean individual and false-negative for the obese individual. In clinical practice, the ECG is not used as a standalone diagnosis tool but is combined with other medical data, such as age, gender, physical symptoms, CT, MRI scan, blood pressure, medical

history, genetics, and lifestyle information, which are available in electronic health records. While incorporating all this information can improve the computer-aided diagnosis, it is a cumbersome process to develop heuristic rules to include all the above data into the diagnosis criteria.

Risk stratification in large cohort studies typically uses statistical tests and simple regression models to study the significance and effect of risk factors on clinical outcomes. Cox proportional hazards regression model (Syed et al. 2011), the χ^2 test (Vapnik 2013), logistic regression (Consortium 2009), Fisher's exact test (Kubo et al. 2007; Wasan et al. 2013), the t-test (Kubo et al. 2007), and linear regression (J. M. Hill et al. 2003) are among the most familiar methods to the medical community. Scoring systems involving multiple risk factors (blood pressure, total cholesterol, high-density lipoprotein cholesterol, smoking, glucose intolerance, and left ventricular hypertrophy) have been evaluated to predict the risks of myocardial infarction, coronary heart disease, stroke, and death from these diseases. A limitation of conventional statistical methods is that when the feature dimension increases, commensurately larger sample sizes are required (McKinney et al. 2006). With an increasing number of risk factors being identified, and especially with abundant genetic and lifestyle data now available, it can be expected that statistical approach will face difficulty as the "healthy" range of the newly-identified factors are hard to obtain or quantify.

Machine learning has the advantage of estimating the associations between risk factors and disease without prior knowledge of accurate reference values of the risk factors. Such approaches have been widely used for risk evaluation and diagnosis of chronic diseases (Oresko et al. 2010; Rajkumar and Reena 2010; Katritsis et al. 2013). In CVD, Knuiman et al. have predicted coronary mortality in the Busselton cohort using a discriminative decision tree (Knuiman and Vu 1997); Lapuerta et al. used a neural network for prediction of coronary disease risk using serum and lipid profiles (Lapuerta, Azen, and LaBree 1995); Gamberger et al. used logistic regression and multilayer perceptrons to predict CVD risks from the INDIANA (Individual Data Analysis of Antihypertensive Intervention Trials) cohort (Gamberger, Lavrač, and

Krstačić 2003); and Das et al. performed heart disease diagnosis using ensembles of neural networks (Das, Turkoglu, and Sengur 2009). ECG information may be incorporated through categorised software findings (Rajkumar and Reena 2010; Berikol, Yildiz, and Özcan 2016) or heuristically extracted features (Alickovic and Subasi 2015; Arsanjani, Dey, et al. 2015; Mitra and Samanta 2013; Homaeinezhad et al. 2012). A classic example is to discover novel patterns in time series (Syed et al. 2011), where Syed et al. improved risk stratification after acute coronary syndrome by 7-13% using three computational ECG biomarkers: morphologic variability, symbolic mismatch, and heart rate motifs (Syed et al. 2011). These features, however, are all unintuitive to human inspection but were shown to be useful indicators of ECG risks.

1.2 ECG Risk Analysis using Machine Learning: Existing Methods

The typical 3-step framework for risk assessment using traditional machine learning on ECG data is feature extraction, classification, and model evaluation (Gamberger, Lavrač, and Krstačić 2003). The performance of the risk models is usually evaluated by how accurately the model predicts the labels that are regarded as the "gold standard" such as the clinical diagnosis. The aim is to improve clinically relevant evaluation metrics such as classification accuracy, sensitivity, and specificity with respect to the "gold standard".

In the classification step, the most commonly used classifiers include support vector machine (SVM) (Berikol, Yildiz, and Özcan 2016; Übeyli 2008a; Özdemir and Barshan 2014; Kim et al. 2009; Q. Li, Rajagopalan, and Clifford 2013; Karpagachelvi, Arthanari, and Sivakumar 2011; C. Yu et al. 2006; Khandoker, Gubbi, and Palaniswami 2009; Kampouraki, Manis, and Nikou 2008; Bsoul, Minn, and Tamil 2010; Monte-Moreno 2011), neural networks (Özdemir and Barshan 2014; Kim et al. 2009; Mitra and Samanta 2013; Monte-Moreno 2011; Tantimongcolwat et al. 2008; Y. Sun and A. C. Cheng 2012), the extreme learning machine (ELM) (Zavar et al. 2011; Kim et al. 2009; Karpagachelvi, Arthanari, and Sivakumar 2011),

random forest (Monte-Moreno 2011; Hsich et al. 2011), k-nearest neighbour (KNN) (Özdemir and Barshan 2014; Karpagachelvi, Arthanari, and Sivakumar 2011), Bayesian decision making (BDM) (Özdemir and Barshan 2014; Luz et al. 2013), ensembles (Arsanjani, Xu, et al. 2013), fussy finite state machines (Pantelopoulos and Bourbakis 2010), stochastic Petro nets (Pantelopoulos and Bourbakis 2010), evolution algorithms (Zavar et al. 2011), self-organizing maps (Tantimongcolwat et al. 2008), radial basis function networks (Kim et al. 2009), and linear regression (Monte-Moreno 2011), and decision trees (Arsanjani, Dey, et al. 2015; Hsich et al. 2011; Kurz et al. 2009).

For feature extraction and selection, the most commonly used methods include the wavelet transform (El-Dahshan 2011; Zavar et al. 2011; Übeyli 2008b), genetic algorithms (El-Dahshan 2011), dynamic time warping (DTW, Syed et al. 2011; Özdemir and Barshan 2014), principle component analysis (PCA, Kim et al. 2009), symbolic aggregate approximation (SAX, Syed et al. 2011), correlation-based feature selection (Mitra and Samanta 2013), linear forward selection (Mitra and Samanta 2013), power-spectrum methods (Kim et al. 2009), Lyapunov exponents (Zavar et al. 2011) among many others.

Traditional machine learning models depend heavily on feature engineering; handcrafting salient features requires human time and effort as well as domain knowledge. For example, expertise in signal processing is required to obtain a good set of ECG features. The handcrafted features are often not transferable, and redesigning feature sets is required for different tasks. In contrast to traditional machine learning approaches, deep learning-based approaches can self-learn useful features from the ECG signals. Numerous deep learning models have been proposed for CVD detection in ECG. A comprehensive review is provided in Chapter 3.

Since the major challenge in incorporating ECG time-series into CVD risk metrics is extracting features and classifying the signals into appropriate ECG abnormality groups, this thesis looks at ECG-associated risk evaluation for CVD from the perspective of ECG time-series classification. There is a large body of literature concerning ECG classification, which reports high classification accuracy,

especially concerning arrhythmia beat classification. However, existing studies are mostly limited to small sample size in terms of recruited study participants. Also, most studies only concern the classification of normal vs abnormal beats from a single CVD group, such as normal vs different types of arrhythmia beats, or normal vs ischemic beats, while rarely study a dataset where ECG abnormalities associated with a broad range of CVD conditions such as arrhythmia, ischaemia, and hypertrophy co-exist. In comparison, the significant challenges of the main dataset this thesis studies - the China Kadoorie Biobank dataset (described in detail in Chapter 4) - has a large sample size ($n = 25,019$), short ECG recording length ($t = 10$s), noisy labels due to a lack of human expert labelling, and unbalanced classes with a large number of data points in the "middle ground" between different classes. This thesis aims to address these unique challenges.

1.3 Structure of the Thesis

The structure of this thesis is as follows: Chapter 2 provides an overview of the medical background of the CVD physiology and pathology, and an introduction to the ECG; Chapter 3 provides a comprehensive review of ECG classification using deep learning; Chapter 4 describes the data stricture of the three datasets studied in this thesis and provides descriptive statistical analysis of the CKB data. The next chapter studies a range of classical machine learning methods except for neural networks and uses extracted features from the "typical cycle", provided by the Mortara device, and analyse which features the machine learning models consider important; Chapter 6 analyses the raw ECG signals directly, in which we introduce a novel theorem, called the Layer-wise Convex Network, and a heuristic network architecture search algorithm - the AutoNet algorithm, which can design LCNs automatically end to end given any dataset and machine learning task. In Chapter 7, we address the issue that the targets in the CKB dataset are provided by the deterministic rule based Minnesota Code, and proposes a novel paradigm of learning using alternative labels, and built models using AutoNet-LCN to predict the ECG-derived age and analyse its association with CVD outcomes

and with blood pressure. Chapter 8 summarises the above work and outlines potential directions for the future work.

2
Medical Background

2.1 The Physiology of Human Heart

2.1.1 Anatomy of the Heart

The human heart has four chambers: right atrium, right ventricle, left atrium, and left ventricle. The right atrium receives venous blood from the superior vena cava, inferior vena cava, coronary sinus, and anterior cardiac veins. The right ventricle pumps blood into the pulmonary artery and is connected to the right atrium via the tricuspid valve and connected to the pulmonary trunk by the pulmonary valve. The left atrium receives oxygenated blood from the four pulmonary veins. The left ventricle pumps oxygenated blood into the aorta through the pulmonary valve and receives blood from the left atrium through the mitral valve.

2.1.2 Cardiac Myocytes

The heart muscle cells are known as cardiac myocytes. The electrical changes occurring during the activation phase of an excitable cell (nerve cells and muscle cells) are called “depolarisation”, and those during relaxation are called “repolarisation”. During depolarisation, a small amount of calcium cations (Ca^{2+}) enters the cell through the L-type voltage-gated Ca^{2+} channels, which increases the concentration of Ca^{2+} in the gap between the sarcolemma and the sarcoplasmic reticulum (SR),

which activates the Ca^{2+} sensitive Ca^{2+}-release channels in the SR to release a large amount of Ca^{2+} to enable the myocyte to contract. When the Ca^{2+} concentration exceeds the resting level, Ca^{2+}-ATPase pumps Ca^{2+} from the cytosol back to the SR, which reduces the concentration of Ca^{2+} to the resting level, leading to the relaxation of the myocyte.

2.1.3 The Cardiac Conduction System

The conduction sequence of a normal cardiac cycle begins with sinoatrial (SA) node depolarisation and progressively results in atrial contraction, atrioventricular (AV) node depolarisation, bundle of His depolarisation, left and right bundle branches of His depolarisation, Purkinje fibre depolarisation, and ventricular contraction, respectively.

The SA node is the pacemaker of the heart and is located at the junction of the superior vena cava and the right atrium. The AV node lies in the interatrial septum immediately above the opening of the coronary sinus. The normal heart beats at 60-100 beats per minute (bpm).

2.2 Introduction to Electrocardiogram

The electrical signals created by the cardiac myocytes can be detected by the electrodes placed on the body surface. An electrocardiogram (ECG) is a graphical interpretation of the electrical activity of the heart. The wave of depolarisation travelling towards an electrode produces a positive deflection and the wave of depolarisation travelling away from an electrode produces a negative deflection. A typical electrocardiograph (ECG) cycle is shown in figure 2.1. In clinical practice, the ECG is usually printed on standard ECG grids, as shown in figure 4.1. The smallest squares are 1 mm by 1 mm, and the horizontal 1 mm represents 0.04s, and the vertical 1 mm represents 0.1 mV.

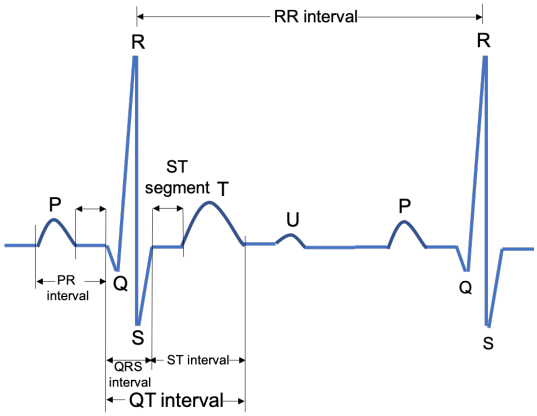


Figure 2.1: Schematic plot of an ECG cycle. (reproduced from Vaswani et al. 2015)

Table 2.1: The positions of the ten electrodes in a standard 12-lead ECG.

Electrode	Position
RA	right arm
LA	left arm
LL	left leg
RL	right leg (ground)
V1	right sternal edge, fourth intercostal space
V2	left sternal edge, fourth intercostal space
V3	midway between V2 and V4
V4	left mid clavicular line, fifth intercostal space
V5	midway between V4 and V6
V6	left mid-axillary line

2.2.1 The Standard 12-Lead ECG

The standard 12-lead ECG includes 10 electrodes: the 4 electrodes on the extremities of each limb yielding 6 limb leads (I, II, III, aVF, aVL, aVR), and the 6 electrodes on the precordium generate 6 precordial leads (V1, V2, V3, V4, V5, V6). The placement of the electrodes is shown in table 2.2. The 12 ECG leads are simply the voltage differences between the 10 electrodes. When calculating the augmented limb leads (aVR, aVL, and aVF), the average potential of the left arm (LA) and the right arm (RA) is used as the negative pole; when calculating the precordial leads, Wilson's central terminal ($V_W = \frac{RA+LA+LL}{3}$) is used as the negative pole. Because only nine electrodes are used to calculate the 12 leads, the 12 leads are not linearly independent - knowing any 9 of the 12 allowing calculations on the other 3 leads.

Table 2.2: The calculation of the 12 ECG leads. Note that to distinguish precordial leads and precordial nodes, the leads are capitalised while the nodes are not.

Lead	Corresponding Electrodes	Origin
I	$LA - RA$	lateral wall
II	$LL - RA$	inferior wall
III	$LL - LA$	inferior wall
aVR	$RA - \frac{1}{2}(LA + LL)$	
aVL	$LA - \frac{1}{2}(RA + LL)$	lateral wall
aVF	$LL - \frac{1}{2}(RA + LA)$	inferior wall
V1	$v1 - \frac{1}{3}(RA + LA + LL)$	anterior wall of right ventricle and the posterior wall
V2	$v2 - \frac{1}{3}(RA + LA + LL)$	anterior wall of right ventricle and the posterior wall
V3	$v3 - \frac{1}{3}(RA + LA + LL)$	anteroseptal and anterior walls of the left ventricle
V4	$v4 - \frac{1}{3}(RA + LA + LL)$	anteroseptal and anterior walls of the left ventricle
V5	$v5 - \frac{1}{3}(RA + LA + LL)$	lateral wall
V6	$v6 - \frac{1}{3}(RA + LA + LL)$	lateral wall

2.2.2 Waves, Segments, and Intervals

The cardiac cycle starts with the sinoatrial (SA) node on the wall of the right atrium, which sends depolarisation wave to the right and left atria, causing them to contract, represented as the P wave on the ECG (figure 2.1). Then the depolarisation wave reaches the atrioventricular (AV) node which delays for 100ms, represented as the PR interval, then causes a contraction in both ventricles. Meanwhile, the atria repolarise and relax, represented as the QRS wave. Finally, the ventricles repolarise and relax, represented as the T wave. Sometimes a U wave is also visible following a T wave, but the origin of U wave is uncertain.

An ECG segment is the period between the end of one wave and the beginning of the next wave. An interval contains at least one segment and at least one wave (figure 2.1). The PR segment is the segment between the end of P wave and the beginning of Q wave, and is usually flat and isoelectric. The ST segment is the interval between the end of S wave and the beginning of T wave, and represents ventricular repolarisation and should be isoelectric with the PR segment in healthy

individuals. The PR interval is the interval between the start of the P wave and the beginning of QRS complex, and represents the wave of depolarisation spreading from the SA node to the ventricles. The QT interval is the interval between the beginning of the QRS complex and the end of T wave and represents the time for ventricles to depolarise and subsequently repolarise.

The origins, normal morphology, and abnormal indications of the ECG waves, segments, and intervals are summarised in table 2.3.

Structure	Origin	Normal characteristics	Abnormalities
P wave	Atrial depolarisation	• Positive in leads I and II • Duration < 0.12s • Amplitude < 0.25mV	• Peaked P wave: right atrial hypertrophy • Bifid P wave (P mitrale): left atrial hypertrophy • Absent P wave: atrial fibrillation • Wide and flat P waves: hyperkalemia
QRS complex	Ventricular depolarization with submerged atrial repolarization	• Duration <0.12s • Amplitude: 10-35mm • Normal R progression	• Tall QRS: left ventricular hypertrophy or normal in slim or athletic people • Low QRS amplitude: obesity • Broad/bizarre shape/sine wave-like QRS: hyperkalemia
Q wave		• Duration<0.04s • Amplitude <0.2mV • Common in leads I, aVL, V4-6	• Pathological Q wave: previous myocardial infarction, fibrosis, or other causes
T wave	Ventricular repolarization	• Inversion can be normal in leads aVR, III, and V1	• Widespread, symmetrical inversion of T wave: ischaemia, infarction, or bundle branch block • Peaked T waves: hyperkalemia • Flattened T waves: hypokalemia
U wave	debatable	• Most common in V2-V4 • Maybe present in athletes	• Prominent U waves (amplitude>2mm): bradycardia, severe hypokalemia, digoxin toxicity • Inverted U waves: coronary artery disease, hypertension, valvular heart disease, congenital heart disease, cardiomyopathy, hyperthyroidism
PR segment		• Isoelectric	• PR segment depression: pericarditis
ST segment	Ventricular repolarization	• Isoelectric with PR segment	• ST elevation: MI, pericarditis • ST depression: MI (reciprocal changes), ischaemia, digoxin toxicity
PR interval	the wave of depolarization spreads from SA node to the ventricles	• Duration 0.12-0.20s	• Short PR interval • Long PR interval: hyperkalemia
QT interval	Time for ventricules to depolarise and subsequently repolarise	• Duration: 0.35-0.45s, increases as heart rate decreases	• Long QT: hypokalemia, hypomagnesaemia, hypocalcemia, hypothermia, congenital long QT syndrome, acute MI, subarachnoid haemorrhage, drugs

Structure	Origin	Normal characteristics	Abnormalities
			• Short QT: hypercalcemia, congenital short QT syndrome
P axis	Mean direction of atrial depolarization	• $-30°$ to $+90°$	
QRS axis	Mean direction of ventricular depolarization	• $-30°$ to $+90°$	• LAD: LVH, LBBB • RAD: right ventricular hypertrophy
T axis	Mean direction of ventricular repolarization		

Table 2.3: The origins and interpretations of ECG waves, segments, and interval interpretations

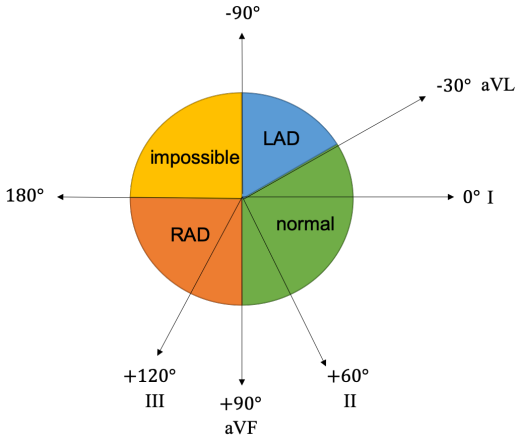


Figure 2.2: Cardiac axis. If the depolarization propagates towards the positive pole, it will yield the maximum positive amplitude on the relevant ECG lead. If the depolarization propagates more than 90^o away from the positive pole, the deflect will be negative on the relevant lead.

Table 2.4: Rules to determine axis deviation

	I	II	aVF
Normal	+	+	+
Right axis deviation	−	+	+
Left axis deviation	+	−	−

2.2.3 Cardiac Axes

The cardiac axes are the directions in which the depolarisation wave propagates (figure 2.2), and are influenced by the size of the muscles in different parts of the heart. Therefore, the cardiac axes indicate chamber enlargement and hypertrophy. Typically, the cardiac axes are evaluated on leads I, II, and aVF, shown in table 2.4. For example, in right ventricular hypertrophy, increased muscle thickness causes the wave of depolarisation to deviate to the right. Hence the QRS complex is negative in lead I and positive in II and aVF. In left ventricular hypertrophy, the wave of depolarisation deviates to the left, hence QRS is negative in lead II and aVF and positive in lead I.

2.2.4 ECG Interpretation

It is recommended in clinical practice to follow a systematic approach to interpret ECG (Vaswani et al. 2015), especially for the interpretation of ECG abnormalities

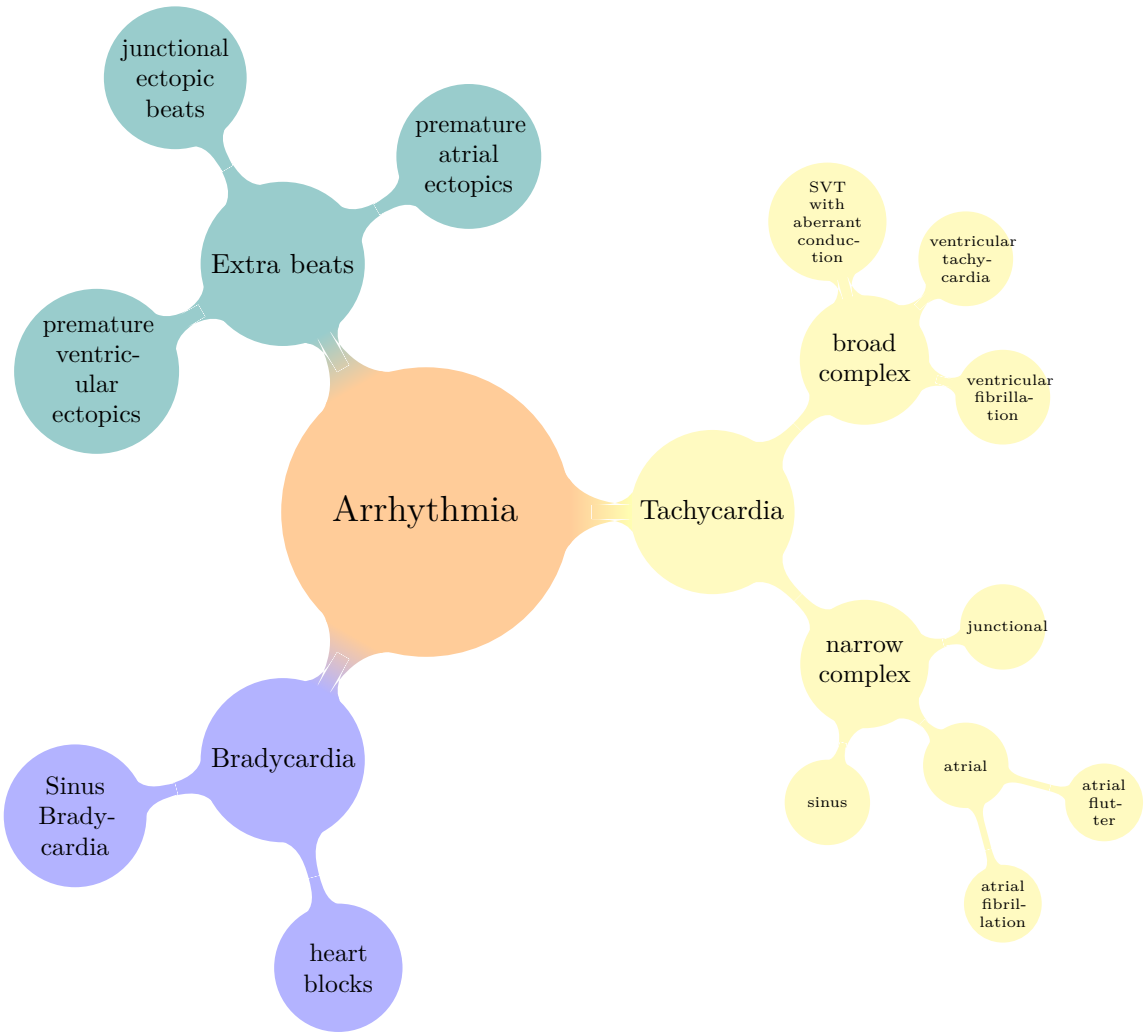


Figure 2.3: Taxonomy of arrhythmia

associated with arrhythmias. Generally, a medical practitioner examines the ECG for information including the heart rate, rhythm, cardiac axis, and wave morphology. The heart rate is usually taken by counting the heartbeats for 10s then multiplying by 6. The rhythm refers to whether the waves are in the P-QRS-T order (which is known as the sinus rhythm) and whether it is regular, regularly irregular, or irregularly irregular.

2.3 Major Cardiovascular Disease Types

2.3.1 Arrhythmias

Arrhythmias are the disorders in the cardiac rhythm, which are the most common cardiac abnormalities. The majority of cardiac arrhythmias are benign, but some arrhythmias are life-threatening, including ventricular fibrillation and ventricular tachycardia. A 12-lead ECG is routinely performed in all patients with suspected arrhythmias (Vaswani et al. 2015). The first line investigations of patients with suspected arrhythmias are 12-lead standard ECG, cardiac enzymes and selected cases have additional investigations including ambulatory 24-hour Holter recording, echocardiography, and electrophysiology studies. The taxonomy of arrhythmias is illustrated in figure 2.3 and introduced in the following sections.

Bradycardia

In bradycardia, the heart rate is below 50 bpm and it affects 20-25% of the people under 25 years old (Vaswani et al. 2015). Treatment is usually not required for asymptomatic patients. Heart block is a sub-type of bradycardia, which refers to the disorder in the cardiac conduction system. Heart block can be further divided according to its origin:

Atrioventricular Block

Atrioventricular (AV) block refers to abnormal conduction between the atria and the ventricles. Typically, AV block is classified into four categories:

- First-degree AV block (I-AVB): delayed atrioventricular conduction resulting in a constant prolonged PR interval (>0.2s) on ECG.
- Second-degree AV block (II-AVB) Mobitz type I (Wenckebach): an atrioventricular conduction disorder resulting in progressive prolongation of the PR interval until a beat is dropped.
- Second-degree AV block Mobitz type II (non-Wenckebach): an atrioventricular conduction disorder resulting in intermittently dropped beats without changes in the PR interval.

- Third-degree AV block (also known as complete heart block): the complete failure of AV conduction resulting in loss of communication between the atria and the ventricles, causing them to beat independently.

Bundle Branch Block (BBB)

The bundle of His splits into the left and right bundle branches. Bundle branch block refers to a disorder in the conduction pathways along the His-Purkinje system and results in asynchronous activation of the ventricles. BBB has two types:

- Right bundle branch block (RBBB): A conduction disorder in the right bundle branch of His resulting in a delay in right ventricular depolarisation. The ECG features are broad QRS complexes ($\geq$ 0.12s), RSR pattern in V1-V3 (M pattern), long S wave duration in leads V6 and I.

- Left bundle branch block (LBBB): a conduction disorder in the left bundle branch of His resulting in a delay in left ventricular depolarisation. Its ECG features are broad QRS complex ($\geq$ 0.12s), deep S wave in V1 and M-shaped R wave in V6, and R wave progression in chest leads. A new onset LBBB on ECG associated with chest pain should raise clinical suspicion of acute myocardial infarction (Vaswani et al. 2015).

Tachycardia

In tachycardia the heart rate is above 100 bpm. It can be further classified into narrow complex tachycardia, in which the QRS complex is less than 0.12s, and broad complex tachycardia, in which QRS complex is no less than 0.12s.

Narrow Complex Tachycardia

The majority of tachycardias are narrow complex in nature. All narrow complex tachycardias are supraventricular in origin and are benign. The sub-types of narrow complex tachycardia include:

- Atrial fibrillation: it is characterised by irregularly irregular heart rhythm and absent P waves on ECG and affects 1% of the population and has male dominance. The common causes of atrial fibrillation include ischemic heart disease, hypertension, and mitral pathologies.

- Atrial flutter: it is characterised by regular, rapid atrial rate. It should always be suspected in tachycardias with fixed atrioventricular conduction ratio (2:1). Atrial flutter typically has an atrial rate of approximately 300 bpm and a ventricular rate of 150 bpm. It can be caused by ischemic heart disease or can be a normal variant in tall males. Its ECG features "sawtooth" pattern of flutter waves.

Broad Complex Tachycardia

Broad complex tachycardia should be considered ventricular tachycardia (VT) or ventricular fibrillation (VF) until proven otherwise, as these two conditions are the most dangerous cardiac arrhythmias. Wide complex tachycardia is often ventricular in origin, but may also be supraventricular with aberrant conduction (usually a bundle branch block). They may be regular (monomorphic ventricular tachycardia) or irregular (Torsades de Pointes, or polymorphic ventricular tachycardia) in nature. Its subtypes include:

- Ventricular tachycardia (VT) is the tachyarrhythmia that originates from the ventricles producing three or more successive broad QRS complexes at a rate over 100bpm. Ventricular tachycardia and ventricular fibrillation account for the most common causes of sudden cardiac death. Common causes of VT are ischaemic heart disease (post-MI scarring), structural heart disease, and electrolyte disturbances (hyper-/hypokalaemia, hyper-/hypomagnesaemia).

- Ventricular fibrillation (VF) is a rapid, uncoordinated and life-threatening ventricular arrhythmia resulting in weak myocardial contraction, eventually leading to cardiac death. Ventricular fibrillation is usually a progression from ventricular tachycardias. Common causes include ischemic heart disease,

typically following an acute MI and electrolyte abnormalities (particularly hyperkalaemia). Its ECG demonstrates chaotic waveforms with varying amplitudes, unidentifiable P-waves, QRS complexes, or T waves.

2.3.2 Ischaemia

Ischaemia, or ischemic heart disease (IHD), refers to a group of diseases where there is an imbalance between myocardial oxygen demand and oxygen supply resulting in tissue hypoxia, which may progress to myocardial infarction. The discrepancy in supply and demand of oxygenated blood is most commonly caused by atherosclerotic diseases of the coronary arteries (Vaswani et al. 2015). Ischaemia on ECG is typically more common in men than in women. The first line investigations for ischaemia include ECG and blood tests for cardiac enzymes. The National Institute for Health and Care Excellence (NICE) recommends stratifying patients into risk categories and conduct further investigations accordingly:

- risk score <10%: consider an alternative diagnosis
- risk score 10-29%: CT calcium scoring
- risk score 30-60%: functional testing
- risk score 61-90%: coronary angiography/functional testing.

The acute coronary syndrome (ACS) is a sub-type of ischaemia in which a sudden disruption in the coronary blood supply to the heart happens after myocardial infarction. ACS ranges from the progression of tissue ischaemia to the development of infarction and necrosis, and is the most common cause of death in western countries (Vaswani et al. 2015). The majority of the affected individuals are male and mostly caused by atherosclerosis. The risk factors of ACS include old age, family history of coronary heart disease, diabetes mellitus, hypertension, smoking, obesity, male, ethnicity, and previous myocardial infarction (Vaswani et al. 2015). Blood tests, including cardiac troponin, and ECG are the first-line investigations for diagnosis of ACS.

In the ECG of ACS patients, ST-elevations typically appear for a few hours, T wave inversion appears for days, and pathological Q waves appears for days to months. Clinically, ACS is classified according to the changes in the ECG and biochemical markers of myocardial necrosis into unstable angina, non-ST elevation myocardial infarction (NSTEMI) and ST-elevation myocardial infarction (STEMI). STEMI is defined as ST-elevation $\geq$1mm in at least two adjacent limb leads or $\geq$ 2mm in 2 contiguous precordial leads or new-onset left bundle branch block. NSTEMI and unstable angina feature ST depression, and T wave inversion (T wave inversion is typical in aVR, III, and V2) on the ECG waveforms.

2.3.3 Hypertrophy

Hypertrophy is a compensatory enlargement of the heart muscles due to failure in other parts of the heart. It is frequently a symptom, rather than an underlying cause itself. Hypertrophy can be identified by studying the cardiac axis deviation on ECG.

A special case is hypertrophic cardiomyopathy, which is an autosomal dominant genetic disorder characterised by asymmetrical left ventricular hypertrophy with impaired diastolic filling. It is the most common cause of sudden death under 35 years, which is usually caused by arrhythmias or severe ventricular outflow tract obstruction (Vaswani et al. 2015). It occurs in 0.2% of the population and has male dominance (Ashish Vaswani et al. 2017). The majority of hypertrophic cardiomyopathy are asymptomatic, and sudden death may be the first presentation. The ECG features of hypertrophic cardiomyopathy include jerky pulses and/or double impulses at the apex, left ventricular hypertrophy, and T wave inversion.

3
Literature Review

A literature search was conducted using the key words such as “deep learning”, “neural network”, “ECG”, “EKG”, “electrocardiogram”, “electrocardiograph” “electrocardiography” of the past 10 years on human subjects and in English language, on the PubMed database on 28 May 2019, and 250 entries were obtained. Manual screening was performed to exclude studies that were unrelated to machine learning (e.g. “neural network” as a physiology term), unrelated to cardiovascular diseases (e.g. using ECG for biometric identification), or non-ECG classification studies (e.g. studies on ECG denoising). In the end, 70 publications remained and they are summarised in table 3.1. The works are organized by the datasets they studied and ordered chronically. The sample size refers to the number of recordings or beats in the training and test sets on which classification was performed. In other words, the sample size is the number to be considered when evaluating the statistical power of the study. These samples, however, typically come from much fewer human subjects. The asterisk next to the author-year means it is a signal classification (see section 3.4) study. The rest are beat classification.

Note that in this thesis the ECG abnormalities typically associated with arrhythmia, ischemia, and hypertrophy, and ischemia are frequently referred to “arritmia”, “iscemia”, and “hypertrophy” (inside double quotation marks), respectively, where

ambiguous, and not to be confused with the actual clinical diagnosis of arrhythmia, ischemia, and hypertrophy.

Work	CVD type	Inputs	Classifier	Sample size	n lead	n patient	n class	Results %
			MIT-BIH Arrhythmia Dataset					
Chudáček et al. 2009	VT	handcrafted features	SVM	215,035	single	-	2	sp 91.6, se 87.2
Kim et al. 2009	"arrhythmia"	handcrafted features	ELM	85,853	-	-	6	se 98.0, sp 98.0
Leite et al. 2010	"arrhythmia"	handcrafted features	FNN	-	lead II	-	2	ac 81.0
Yaghouby et al. 2010	AF	handcrafted features	genetic programming	680	lead II	-	2	se 99.1, sp 98.9
Osowski, Siwek, and Siroic 2011	"arrhythmia"	statistical features	ensemble	109,963	-	15	7	F_1 79.6
Kostka and Tkacz 2011	AF	wavelet features	MLP	40	single	-	2	sp 87.0, se 86.0
Hassan Hamsa Haseena, Mathew, and Paul 2011	"arrhythmia"	cluster centres	probabilistic NN	5357	singe	-	8	ac 99.6
Hassan H Haseena, Joseph, and Mathew 2011	"arrhythmia"	AR coefficients	probabilistic NN	1,920	lead II	20	8	ac 99.1
Javadi et al. 2011	PVC	wavelet features	ensemble of NN	15,566	II&V1/2/4/5	7	2	ac 87.6
Nejadgholi, Moradi, and Abdolali 2011	"arrhythmia"	phase space features	GMM-Bayesian	104,060	-	47	5	ac 92.5
Martis et al. 2012	"arrhythmia"	DWT features	SVM	-	II	65	2	ac 95.6
Benali, Reguig, and Slimane 2012	"arrhythmia"	handcrafted features	wavelet NN	27,280	II	47	5	F_1 98.2
Chikh, Ammar, and Marouf 2012	"arrhythmia"	handcrafted features	NFIN	18,663	II	47	2	se 97.9, sp 94.5
Y.-H. Chen and S.-N. Yu 2012	"arrhythmia"	handcrafted features	MLP	7,185	II	15	7	ac 97.5
Chakroborty 2013	"arrhythmia"	filtered beats	auto-regressive NN	32,940	single	47	5	F_1 76.9
S.-H. Liu, D.-C. Cheng, and C.-M. Lin 2013	"arrhythmia"	waveforms with R peak marked by SVM	NFIN	-	II&V1	33	5	ac 96.4
Prasad et al. 2013	"arrhythmia"	handcrafted features	KNN, CART, NN	2,383	II	47	3	se 98.8, sp 99.5
Javadi 2013	"arrhythmia"	DWT features	NFIN	106,647	II	47	7	ac 98.3

Work	CVD type	Inputs	Classifier	Sample size	n lead	n patient	n class	Results %
S.-N. Yu and F.-T. Liu 2014	"arrhythmia"	DWT features	-	7,187	II & VI	12	7	ac 98.3
Kiranyaz, Ince, Hamila, et al. 2015	SVEB	filtered beats	CNN	100,389	single	-	2	se 68.8, sp 99.5
Poddar, Kumar, and Sharma 2015	"arrhythmia"	handcrafted features	-	42,917	II	47	2	se 90.3, ppv 92.3
Yang et al. 2015	"arrhythmia"	SAE extracted features	softmax regression	80,740	II	47	2	ac 99.4
Kiranyaz, Ince, and Gabbouj 2015	"arrhythmia"	heavily filtered and segmented beats	CNN	633,410	2	34	5	F_1 77.0
Altan, Kutlu, and Allahverdi 2016	"arrhythmia"	HBT features	MLP	169	II	141	2	sp 98.2, se 97.1
Elhaj et al. 2016	"arrhythmia"	handcrafted features	SVM	11,094	II	47	5	ac 98.9
P. Li et al. 2016	"arrhythmia"	wavelet features	FNN	360	single	-	6	ac 99.3
Abdul-Kadir, Safri, and Othman 2016	AF	handcrafted features	SVM	35,034	single	41	3	ac 95.0
Kiranyaz, Ince, and Gabbouj 2016	"arrhythmia"	filtered beats	CNN	83,648	II	44	5	F_1 81.2
Sahoo et al. 2017	"arrhythmia"	wavelet features	PNN	108,374	single	47	6	se 100.0, ppv 99.8
P. Li et al. 2016	"arrhythmia"	handcrafted features	regression NN	12,572	single	17	5	ac 95.0
Acharya et al. 2017	"arrhythmia"	filtered beats	CNN	109,449	II	47	5	ac 94.0
F.-y. Zhou, Jin, and Dong 2017	PVC	filtered beats	CNN&LSTM	50,887	2	47	2	se 97.6, sp 99.5
Anwar et al. 2018	"arrhythmia"	DWT	-	10772	II	47	18	se 98.7, sp 99.9
Beritelli et al. 2018	"arrhythmia"	3 handcrafted features	radial basis PNN	2,175	II	47	3	ac 95.4
Yildirim 2018	"arrhythmia"	wavelet features	BiLSTM	7,376	II	47	5	ac 99.4
Z. He et al. 2018	"arrhythmia"	raw ECG waveforms after R peak detection	NN	109,449	II	47	5	F_1 99.7
Sayantan, Kien, and Kadambari 2018	"arrhythmia"	features learned from DBN	linear SVM	265,747	II	47	4	ac 99.4

Work	CVD type	Inputs	Classifier	Sample size	n lead	n patient	n class	Results %
Oliveira et al. 2019	"arrhythmia"	geometrical features	SVM	39,647	single	-	2	ac 99.0
MIT-BIH Atrial Fibrillation Dataset								
Kostka and Tkacz 2011	AF	wavelet features	MLP	40	single	-	2	sp 87.0, se 86.0
Abdul-Kadir, Safri, and Othman 2016	AF	handcrafted features	SVM	35,034	single	41	3	ac 95.0
Xia et al. 2018	AF	STFT matrices	CNN	162,536	single	23	-	se 98.3, sp 98.2
Physikalisch-Technische Bundesanstalt (PTB) Dataset								
Costantino et al. 2016*	MI	phase space features	ANN	104	3	104	2	se 92.0, sp 96.0
Masetic and Subasi 2016	CHF	AR coefficients	random forest	2,800	single	31	2	auc 99.9
Kora 2017	MI	raw ECG	HFPSO & NN	2,806	single	44	2	se 100.0, sp 98.7
W. Liu et al. 2018	MI	-	CNN	34,769	8	-	2	se 95.0, sp 97.4
Chinese Cardiovascular Disease Database								
Jin and Dong 2016*	abnormality	features extracted by NN	ensemble NN	179,130	12	-	2	auc 87.0
F.-y. Zhou, Jin, and Dong 2017*	PVC	filtered segments	FNN	176,886	12	-	2	se 96.3, sp 98.1
PhysioNet Atrial Fibrillation Detection 2017 Challenge Dataset								
Teijeiro et al. 2018*	AF	handcrafted features	RNN	8,528	single	-	4	F_1 83.0
Sadr et al. 2018*	AF	handcrafted features	quadratic NN	8,528	single	-	4	F_1 78.0
Kamaleswaran, Mahajan, and Akbilgic 2018*	AF	end to end	CNN	8,528	single	-	4	F_1 83.0
Hannun et al. 2019*	AF	end to end	ResNet	8,528	single	-	4	F_1 83.0
Other Datasets								
Kampouraki, Manis, and Nikou 2008	CAD	handcrafted features	SVM	12	single	12	2	ac 100.0
Ibaida and Khalil 2010	VT & VF	compressed ASCII codes	KNN	103	single	9	2	ac 93.3
Jovic and Bogunovic 2011	"arrhythmia", CHF	handcrafted features	random forest	100	single	100	4	ac 99.6

Work	CVD type	Inputs	Classifier	Sample size	n lead	n patient	n class	Results %
Tejera et al. 2011	hypertension	handcrafted HRV features	ANN	568	-	217	3	se 85.0, se 80.0
Park, Pedrycz, and Jeon 2012	ISTC	3 handcrafted features	SVM	367	2	-	2	se 94.1, sp 92.3
Ebrahimzadeh, Pooyan, and Bijar 2014	SCD	time frequency features	MLP	-	2-3	70	2	ac 84.0
L. Zhang et al. 2015	"arrhythmia"	Poincare plot	NN	674	-	674	5	ac 96.1
M. He, Lu, et al. 2016	cardiac arrest	time frequency features	NN	528	single	199	2	auc 81.9
M. He, Gong, et al. 2015	cardiac arrest	handcrafted features	logistic regression	3,828	single	1617	2	auc 87.6
M. Yu et al. 2016	VF	handcrafted features	NA-MEMD	1,017	single	-	2	se>95.0, sp>80.0
Immanuel et al. 2016	LQTS	handcrafted features	-	334	I	334	3	ac 61.2
Isler 2016	systolic and diastolic dysfunction	HRV features	FNN	30	-	30	2	ac 96.3
Mjahad et al. 2017	VF & VT	time frequency matrices	bagging	27,811	-	24	2	se 95.6, sp 98.8
Rad et al. 2017	RCR	wavelet features	FNN	1631	single	298	5	ac 78.5
Tan et al. 2018*	CAD	ECG segments	LSTM&CNN	38,120	II	47	2	ac 99.9
Amezquita-Sanchez et al. 2018	SCD	wavelet features	PNN	41	single	41	2	ac 95.8
Hannun et al. 2019*	"arrhythmia"	end to end	ResNet	91,232	single	53,549	12	F_1 83.7

Table 3.1: Summary of studies on ECG classification using neural networks. ac: accuracy, sp: specificity, se: sensitivity, -: no information. The asterisk means the study is signal classification (see section 3.4). The rest are beat classification.

3.1 Types of Cardiovascular Diseases

Many studies used standard ECG databases such as the MIT-BIH database (Moody and Mark 2001), the MIT-BIH long-term ST database (Goldberger et al. 2000), the PhysioNet Atrial Fibrillation (AF) Detection Challenge (Clifford et al. 2017), and the Physikalisch-Technische Bundesanstalt (PTB) database (Bousseljot, Kreiseler, and Schnabel 1995; Goldberger et al. 2000). Each database focuses on different cardiovascular diseases or conditions, including "arrhythmias", ST changes, atrial fibrillation, and myocardial infarction (MI), respectively. As a result, few studies attempted to classify ECG abnormalities associated with diverse cardiovascular diseases, such as co-existence of arrhythmia, ischaemia, and hypertrophy in the same database using a single model. Most studies construct a dataset consisting of the normal class and an abnormal class, such as MI, and perform binary classification (Chudáček et al. 2009; Leite et al. 2010; Yaghouby et al. 2010). However, in a real-world application, the dataset typically contains many more types of medical conditions.

3.2 Number of ECG leads

Most studies only used one or two ECG leads, even when more leads are available in the dataset (Masetic and Subasi 2016; Kora 2017). Different ECG leads represent different parts of the heart, thus using single or few leads may not be sufficient when various cardiac conditions coexist in the same dataset. For example, a Q wave with more than 0.04s in duration and deeper than 0.2mV in amplitude is only considered pathological when it occurs in more than one lead, which is commonly a sign of previous myocardial infarction (Vaswani et al. 2015); R wave progression is the phenomenon that the amplitude of R wave increases gradually across chest leads V1-V4 and decreases gradually in leads V5-V6, which can only be observed when all six chest leads are present; T wave inversion is considered normal in leads aVR, III, and V1, but indicates ischaemia, infarction or bundle branch block when widespread

in the leads; QRS axis deviation can only be assessed by taking consideration of I, II, and aVF leads together (Vaswani et al. 2015).

3.3 Model Evaluation

Accuracy, sensitivity, specificity, area-under-receiver-operating-curve (AUROC), and F_1 are common model evaluation metrics for ECG classification. However, accuracy can be misleading when classes are highly unbalanced. For example, in a classification problem where 99% of the test set samples belong to class i, a trivial solution which classifies all samples to class i would yield 99% accuracy. AUC is more robust towards class skewness; however, it can only evaluate binary classifiers. Sensitivity, together with specificity, is a fair metric for skewed classification, yet two numbers are not as convenient as one number for model comparison. F_1 is a single-number metric that is robust to class skewness, therefore it is the evaluation criterion of many machine learning competitions, including PhysioNet AF Detection Challenge, and International Conference on Biomedical Engineering and Biotechnology (ICBEB) Physiological Signal Classification Challenge[1]. F_1 is almost always lower than accuracy, sensitivity, and specificity values on the same confusion matrix (the matrix whose element C_{ij} represents the number of samples known to be in class i being classified to class j). For example, in Luo et al. 2017, the authors reported an accuracy of 97.5% for 4-class classification; however, the equivalent F_1 value on the same confusion matrix was only 45.3%.

3.4 Beat Classification vs Signal Classification

In this thesis, we distinguish beat classification from signal classification in the literature. Beat classification aims to classify an ECG beat, while signal classification aims to classify a signal that consists of many ECG beats. Beat classification is useful for real-time monitoring, while signal classification is appropriate for screening and medical investigation. Beat classification often yields high accuracy, sensitivity,

[1] http://www.icbeb.org/CPSC2018Awards.html

and specificity, while signal classification renders much lower performance, due to the following difficulties faced by signal classification uniquely:

- Smaller labelled datasets: for example, a 30-s ECG signal typically has over 30 beats, which means it has 30 labelled examples for beat classification, but one labelled example for signal classification;
- Decision rules: It is difficult to decide whether a signal with majority normal beats and occasional abnormal beats should be classified as normal.
- Curse of dimensionality: while the number of labelled examples is much scarcer for signal classification than for beat classification, the dimensionality of each training example is much higher than that in beat classification; thus, the sample to dimension ratio for signal classification is more challenging to handle.

Comparing with signal classification, beat classification requires one label per beat, which is labour-intensive for human experts to provide in datasets containing hundreds of thousands of beats. Many studies approach signal classification problems by labelling all beats within a signal to be of the same class as the entire signal (Masetic and Subasi 2016; Kora 2017; W. Liu et al. 2018). For example, if a human expert labelled a 30-s segment as an atrial flutter episode, then the study would label all beats within the 30-s signal as atrial flutter beats. Beat classification also requires beat segmentation, which can be a challenging task in itself if the signal quality is low. Beat classification circumvents the challenges in beat segmentation by reporting post-segmentation classification results, which is another reason beat classification studies seem to have higher performance than signal classification studies. However, in real-world application, the beats are not readily segmented, thus signal classification is more relevant to the real-world medical problem. Signal classification is understudied in comparison to beat classification, as shown in table 3.1, where only 9 (studies with asterisk) out of 70 studies were signal classification.

3.5 Methods of ECG Classification

Before the era of deep learning, the typical workflow of ECG classification is signal quality analysis, involving discarding low-quality signals, denoising and removing baseline wander and artefacts, then a QRS detection is performed for beat segmentation. Furthermore, phase alignment and normalization are required to align the R peaks for feature extraction. Prior to classification, feature selection is performed. Although neural network was first applied to ECG classification in 1998 (Yao et al. 1998), most early studies tend to rely on handcrafted heuristic features (Chudáček et al. 2009; Kim et al. 2009; Leite et al. 2010); researchers gradually lean towards more principled feature extraction schemes, from using discrete wavelet transform coefficients as the input features to the classifier (Kostka and Tkacz 2011; Martis et al. 2012; Javadi 2013), to using a neural network to extract features automatically (Jin and Dong 2016; Sayantan, Kien, and Kadambari 2018).

From table 3.1 we can see that convolutional neural network (CNN), recurrent neural network (RNN), deep belief network (DBN), and auto-encoders are the most popular deep learning models for ECG classification. Regarding the inputs to the neural networks, very few studies use raw ECG time-series signals. In particular, time-frequency transform of the raw ECG waveforms is considered as an input which often yields good results. For example, Isin and Ozdalili 2017 used AlexNet to extract features from the time-frequency domain of ECG beats. Xiao et al. 2018 used a novel approach of overlaying raw ECG signals on standard ECG grid sheets and saved them as images, to mimic the same format as commonly interpreted by cardiologists. The authors then used the trained Google Inception v3 (Szegedy et al. 2015) as a feature extractor to detect sudden ST changes in an ambulatory setting.

End-to-end deep learning refers to the approach to use a single neural network to perform the full pipeline from feature extraction to ECG classification. The first end-to-end deep learning study was performed by Rajpurkar et al. 2017 who used a 34-layer ResNet-like CNN (K. He et al. 2016) to classify 14 rhythm classes, and outperformed average cardiologists. This work was the precursor of the model by Hannun et al. 2019, which is considered the state-of-the-art of ECG classification

nowadays. Hannun et al. 2019 reported cardiologist-level performance of their model on 12 "arrhythmia" classes that were trained on 91,232 signals from 53,549 patients and tested on 336 recordings and benchmarked against six cardiologists using 30s single-lead ECGs. The authors also tested their model post-hoc on the PhysioNet AF Detection Challenge and reported $F_1 = 0.83$, which is among the highest performances on the Challenge. Although none of the four official winners ($F_1 = 0.83$) of the PhysioNet AF Detection Challenge used end-to-end deep learning (Clifford et al. 2017), Kamaleswaran, Mahajan, and Akbilgic 2018 recently developed a 13-layer end-to-end CNN to obtain $F_1 = 0.83$ on the hidden test set, demonstrating again that an end-to-end CNN can perform as well as using handcrafted features and ensemble classifiers (Kamaleswaran, Mahajan, and Akbilgic 2018).

In the latest ECG classification competition held by ICBEB, the winning team Chen et al.[2] also used deep end-to-end learning. The power of deep learning is proven by all three state-of-the-art models (Chen's model, Rajpurkar-Hannun model, and Kamaleswaran's model), but all these models are computationally intensive: Rajpurkar-Hannun model has over 10 million parameters, while Kamaleswaran's model has 3 million parameters.

In the following chapters of this thesis, we study ECG classification on the Chinese population. We first look at the statistical characteristics of the CKB, then study ECG classification using traditional machine learning with handcrafted features. We then propose a novel neural network architecture family for time-series classification, called layer-wise convex networks, which characterises in parameter parsimony, and benchmark our results with Rajpurkar-Hannun model on 3 databases: the CKB, the PhysioNet AF Detection Challenge, and the ICBEB Physiological Signal Classification Challenge.

[2]Tsai-Min Chen, Chih-Han Huang, Edward S. C. Shih, Ming-Jing Hwang

4
Data Description

4.1 China Kadoorie Biobank

The China Kadoorie Biobank (CKB) is a prospective cohort study of 521,891 adults recruited from 10 areas in China during the years 2004 - 2008 (Z. Chen et al. 2011). Data were collected using questionnaires, and clinical measurements were recorded at baseline. After every five years, approximately 25,000 surviving participants were resurveyed using further questionnaires and clinical measurements. The second resurvey in 2013-2014 included a 12-lead ECG on 24,959 participants. Ethics approval was obtained from all relevant local and national committees. Public access to the CKB can be found at http://www.ckbiobank.org/site/Data+Access. The ECG data used for this thesis are described below:

4.1.1 ECG Time-Series

A standard 12-lead ECG (each with 10-s duration, sampled at 500Hz) was recorded on 24,959 participants using a Mortara ELIx50 device during the years 2013 - 2014. An ECG cycle template, representing a "typical" cycle, for an individual lead, was also generated by the Mortara device using the proprietary VERITAS^{TM} algorithm. The raw ECG time-series are overlaid on the standard ECG grids and are shown in figure 4.1.

Table 4.1: The Mortara features and the blood pressure features

Independent Features	**Explanation**
Age	
Average RR interval	Average distance between two R peaks
QRS offset	The end of the QRS complex
P wave duration	
PR interval	P onset to QRS offset
QRS duration	
QT interval	Q onset to T offset
P axis	Determined by P deflect in I, II, aVF
QRS axis	Determined by QRS deflect in I, II, aVF
T axis	Determined by T deflect in I, II, aVF
Dependent Features	**Explanation**
ventricular rate	$= \frac{6000}{average\ RR\ interval}$
R peak	$\equiv 500\ ms$
P wave onset	= Q offset − PR interval − QRS duration
P wave offset	= P onset + P duration
QRS onset	= PR interval+P onset
T wave offset	= Q onset + QT interval
QT_c duration	= QT interval $+154(1 - \frac{60}{ventricular\ rate})$
QT_{cB} duration	$= \frac{QT\ interval}{(average\ RR\ interval)^{\frac{1}{2}}}$
QT_{cF} duration	$= \frac{QT\ interval}{(average\ RR\ interval)^{\frac{1}{3}}}$
Blood Pressure	**Explanation**
SBP	Systolic blood pressure
DBP	Diastolic blood pressure

4.1.2 ECG Features

A total of 19 unique features were provided for each participant by the Mortara device (table 4.1). Ten features that could not be expressed as functions of the other features are collectively referred to as "independent features", and the remaining nine features which can be expressed as functions of the ten independent features or is constant for the duration of the typical cycle is referred to as the "dependent features".

4.1.3 Mortara Labels

For each participant, up to 10 textual descriptions describing the ECG were provided, such as "atrial fibrillation", "acute myocardial infarction", and "normal ECG", by the Mortara device using a propriety algorithm according to the Minnesota Code

Table 4.2: Grouping criteria and the number of participants in each group in the CKB

Class	N (%)	Inclusion Criterion
Normal	10,779 (43%)	Normal ECG
"Arrhythmia"	2,472 (10%)	Abnormal rhythm Atrial fibrillation Early repolarization Pre-excitation Premature ectopic beats Ectopic conditions Blocks Uncertain rhythm
"Ischaemia"	1,870 (8%)	Explicitly stated "ischaemia"
"Hypertrophy"	3,423 (14%)	"Hypertrophy" or enlargement
"Others"	6,362 (26%)	None of the above
All	24,906 (100%)	

(Prineas, Crow, and Z.-M. Zhang 2009). Each textual label is chosen from 236 possible values. We grouped the 236 Mortara labels into "normal", "arrhythmia", "ischaemia", "hypertrophy", and "others", according to (Ramrakha and J. Hill 2012). After removing 113 participants with incomplete records, the remaining 24,906 participants were grouped in the the five classes, shown in table 4.2. The complete mapping from the 236 Mortara labels to the 5 groups are shown in appendix F.

4.1.4 Blood Pressure Data

Systolic blood pressure (SBP) and diastolic blood pressure (DBP) were recorded twice on each participant after resting for at least 5 minutes using an Omron UA-779 digital sphygmomanometer. If the difference between the two measurements was more than 10mmHg, a third measurement was performed, and only the last two readings were recorded. Sphygmomanometers were supplied centrally, calibrated daily and only used by trained field workers. We use the average of the two blood pressure readings in our study.

4.1.5 Signal Quality

A signal quality index (SQI) was evaluated for all 12-lead ECG signals using in-house software. The SQI $\in [0, 1]$, depending on the agreement of two peak

detectors on the positions of the R peaks. High agreement yields a high SQI, which corresponds to high signal quality. 97% of all 12-lead waveforms were found to have high signal quality (SQI>0.9). The signal quality of the data was deemed sufficient to classify the 24,906 participants.

4.1.6 Descriptive Statistics of CKB

Age Group, Gender, and Class Distribution

The percentage of the participants in each age group and class is shown in figure 4.2. The exact numbers are provided in appendix A. The percentage is calculated relative to the total number of male or female participants in each age group. For example, there were 1,974 men aged under 50, and 133 of them had ECG abnormalities associated with arrhythmias, thus the percentage of males under 50 having "arrhythmia" is $\frac{133}{1974} = 7\%$.

We can see a higher percentage of males have "arrhythmia" and "hypertrophy" in their relevant age groups than their female counterparts, but the percentage of "ischaemia" is similar for men and women. In both men and women, as the age increases, the percentage of normal participants declines, and the percentage of "arrhythmia" increases. The percentage of women having "hypertrophy" also increases steadily with age. The percentage of men and women having "ischaemia", as well as men having "hypertrophy", are relatively stable across age groups. These observations are consistent with available evidence from epidemiology studies of heart disease, although the male dominance in "ischaemia" was not observed in the CKB dataset, perhaps because there were relatively few people ($n = 1,870$) having "ischaemia".

Blood Pressure Distribution

We performed Gaussian kernel density fitting (D. W. Scott 2015) using the Scipy package on the systolic and diastolic blood pressure to obtain the distribution of the blood pressure in the four classes, as shown in figure 4.3. For all participants, the mode of systolic blood pressure (SBP) of the normal class is lower than those of "arrhythmia", "ischaemia", and "hypertrophy" classes, meaning the abnormal classes

Table 4.3: Class size in ICBEB.

Class Index	Class	Number of Recordings
1	Normal (N)	918
2	Atrial Fibrillation (AF)	1,098
3	First-degree atrioventricular block (I-AVB)	704
4	Left bundle branch block (LBBB)	207
5	Right bundle branch block (RBBB)	1,695
6	Premature atrial contraction (PAC)	556
7	Premature ventricular contraction (PVC)	672
8	ST-segment depression (STD)	825
9	ST segment elevation (STE)	202
	Total	6,877

tend to have higher SBP. In female participants, the mode of SBP has the ascending order of normal < "arrhythmia" < "ischemic" < "hypertrophy", which agrees with medical knowledge. A similar trend can be observed in female diastolic blood pressure (DBP), but less obvious in male DBP. The reference values for normal SBP and DBP are 120 mmHg and 80 mmHg, respectively (B. Zhou et al. 2017).

4.2 The ICBEB Dataset

The publicly available training set of International Conference on Biomedical Engineering and Biotechnology (ICBEB) 2018 challenge [1] includes 12-lead 500Hz 5-143s [2] ECG time-series waveform from 6,877 participants (3,178 female and 3,699 male) obtained from 11 hospitals. The dataset has nine classes and the number of recordings in each class is shown in the table 4.3.

The hidden test set contains 2,964 ECG recordings of similar duration. The final evaluation is based on a balanced test set comprised of 50 samples randomly selected from each of the nine classes from the hidden test set. The training and test sets are mutually exclusive.

The primary evaluation criterion of the Challenge is the 9-class average F_1, calculated as equation 4.1 The secondary evaluation criteria are F_1 scores of sub-abnormal classes: F_{AF}, F_{Block}, F_{PC}, F_{ST} calculated as equations 4.2, 4.3, 4.4 and

[1] http://2018.icbeb.org/Challenge.html

[2] The website states 6-60s duration, however the actual signal duration in the dataset is 5-143s.

Table 4.4: The number of recording in each class in the PhysioNet dataset.

Class	Number of recordings
normal	5,050
atrial fibrillation (AF)	738
other rhythms	2,456
noise	284
total	8,528

4.5. The winning team was Chen et al.[3] who achieved 9-class average F_1 of 0.837, as well as the highest F_{AF} (0.933), F_{PC} (0.847), F_{ST} (0.779), and the 5th highest F_{Block} (0.899). They used bidirectional GRU and attention mechanism, and trained a different model for each lead as well as a 12-lead joint model, then performed 10-fold model averaging of each of the 13 models. Thus the final prediction was based on the average probability given by 10 × (12 single-lead models + the 12-lead joint model) = 130 models. The second place in terms of 9-class F_1 was Cai et al.[4], who also achieved the highest F_{Block} (0.912). They used Long Short-Term Memory (LSTM).

$$F_1 = \frac{1}{9}\sum_{i=1}^{9}\frac{2N_{ii}}{\sum_{j=1}^{9}(N_{ij}+N_{ji})} \tag{4.1}$$

$$F_{AF} = \frac{2N_{22}}{\sum_{j=1}^{9}(N_{2j}+N_{j2})} \tag{4.2}$$

$$F_{Block} = \frac{2\sum_{i=3}^{5}N_{ii}}{\sum_{i=3}^{5}\sum_{j=1}^{9}(N_{ij}+N_{ji})} \tag{4.3}$$

$$F_{PC} = \frac{2\sum_{i=6}^{7}N_{ii}}{\sum_{i=6}^{7}\sum_{j=1}^{9}(N_{ij}+N_{ji})} \tag{4.4}$$

$$F_{ST} = \frac{2\sum_{i=8}^{9}N_{ii}}{\sum_{i=8}^{9}\sum_{j=1}^{9}(N_{ij}+N_{ji})} \tag{4.5}$$

4.3 The PhysioNet Dataset

The publicly available training set of the PhysioNet 2017 Atrial Fibrillation Detection Challenge (Clifford et al. 2017) has 8,528 recordings, 9-60s in duration, 300Hz, single-lead ECG acquired using AliveCor. The dataset has four classes: normal, atrial fibrillation, "other rhythms", and noise. The number of recordings in each class is shown in table 4.4[5].

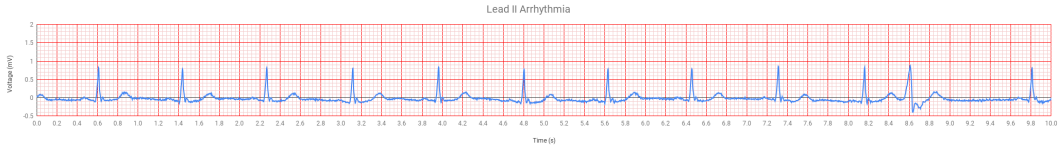
The hidden test set of the challenge had 3,658 recordings of similar duration. The final results were evaluated by the 3-class average F_1 of atrial fibrillation, normal, and "other rhythms" classes. The winning teams achieved 3-class average $F_1 = 0.83$ (Clifford et al. 2017). There are a few post-hoc studies including Hannun et al. 2019 and Kamaleswaran, Mahajan, and Akbilgic 2018, trained on the entire publicly available training set and achieved 0.83 three-class F_1 on the hidden test set as well. Both Kamaleswaran and Hannun used CNN-based architectures for their analysis.

[3]Tsai-Min Chen, Chih-Han Huang, Edward S. C. Shih, Ming-Jing Hwang

[4]Wenjie Cai, Jing Ma, Li Yang, Danqin Hu, Yanan Liu

[5]The numbers are counted from the downloaded dataset, which is very different from what is stated on the website.

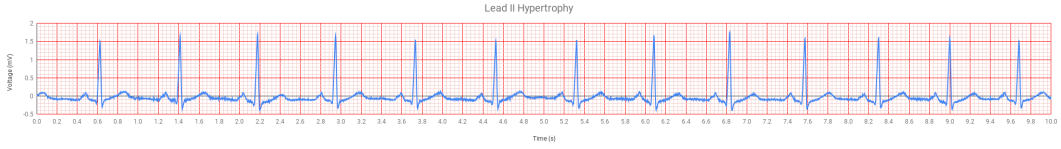
(a) Lead II Normal



(b) Lead II "Arrhythmia"

figures/Lead II Ischaemia (2).png

(c) Lead II "Ischaemia"



(d) Lead II "Hypertrophy"

Figure 4.1: Examples of Lead II ECG waveform for the four conditions

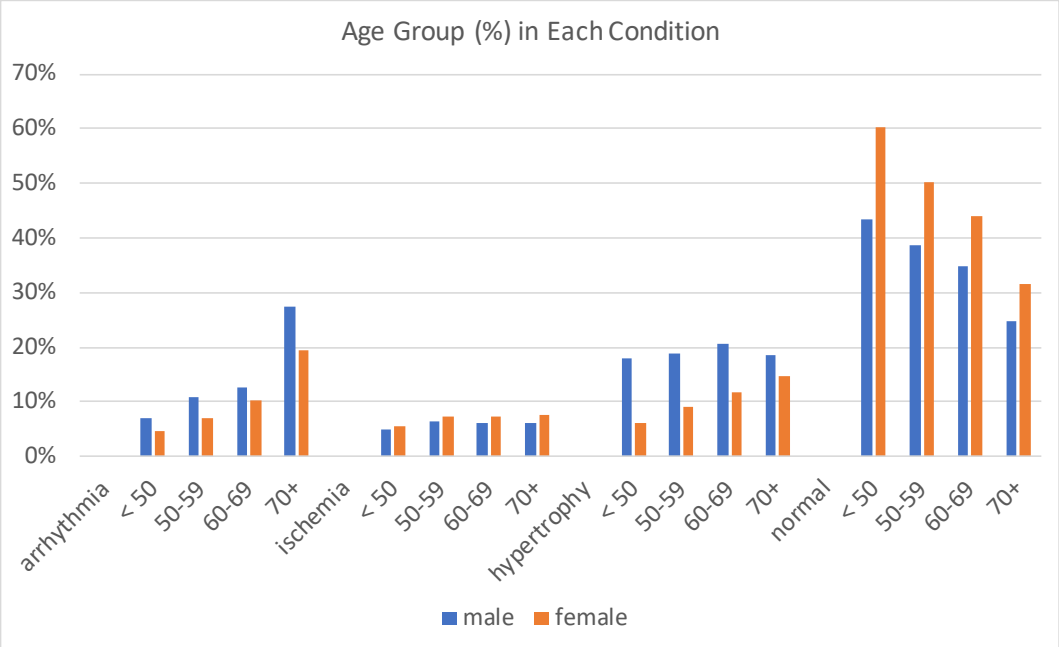
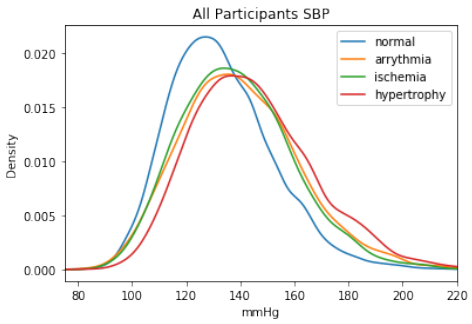
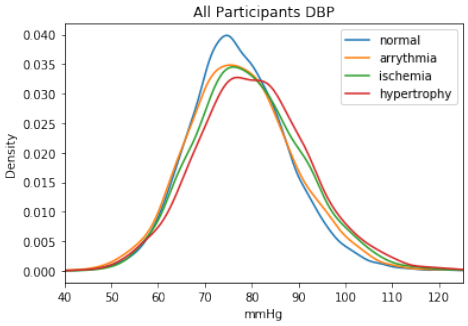


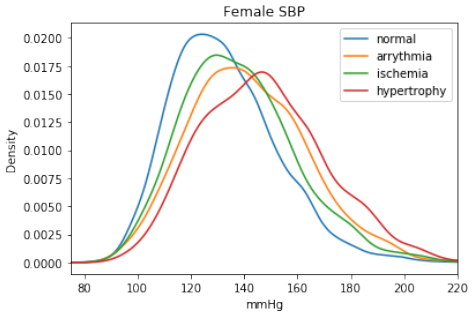
Figure 4.2: The percentage of participants in each age group and class



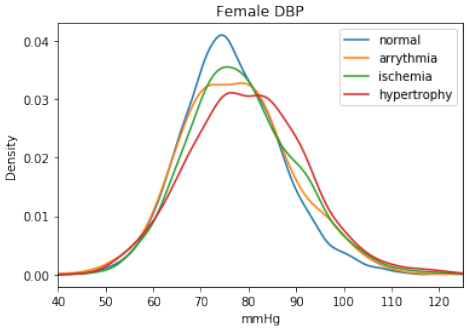
(a) SBP



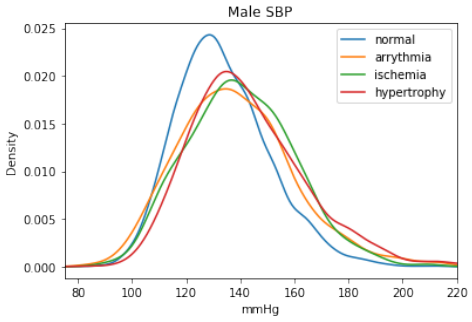
(b) DBP



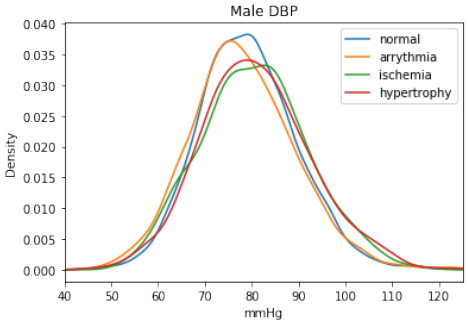
(c) SBP



(d) DBP



(e) SBP



(f) DBP

Figure 4.3: Distribution of SBP and DBP among the four classes

5

ECG Classification using Traditional Machine Learning Methods

5.1 Introduction

In this chapter, we present results of analysis of ECG data using traditional (i.e. non-deep) machine learning to classify the ECG signals into normal, "arrhythmia", "ischaemia", and "hypertrophy" classes. We start with a brief introduction to the principles of machine learning and the methods to be used. These methods are representative of all major machine learning model families except neural networks. As neural networks, or known by their legacy names artificial neural networks (ANN) or multilayer perceptron (MLP), are a very flexible model family and have now evolved into the regime of deep learning, they are not included in this chapter but will be the focus of Chapters 6 and 7. We will continue the chapter with data preprocessing and feature extraction, which are necessary for all the traditional machine learning models evaluated in this chapter. We then compare the performance of these methods on different combinations of the Mortara features, introduced in the previous chapter, and our new features, and finally perform feature ranking and conclude with comparative analysis of the Mortara features and the new features.

5.2 Introduction to Machine Learning

Mitchell et al. 1997 defined machine learning as "learning from experience E concerning some tasks T and performance measure P, if its performance at tasks T, as measured by P, improves with experience E". In this thesis we focus on supervised learning whose goal is to discover the data generating process $f : \boldsymbol{X} \mapsto \boldsymbol{Y}$, where $\boldsymbol{X}$ and $\boldsymbol{Y}$ represent the features and the labels, respectively.

5.2.1 Linear Models

Linear Discriminant Analysis

Linear discriminant analysis (LDA) assumes the likelihood $p(\boldsymbol{x}|C)$ is Gaussian, and all classes share the same covariance matrix. Formally, for K-class classification:

$$p(\boldsymbol{x}|C_k) = \frac{1}{(2\pi)^{\frac{D}{2}}|\boldsymbol{\Sigma}|^{\frac{1}{2}}} \exp\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_k)^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_k)\} \tag{5.1}$$

for $k = 1, ..., K$, where $\boldsymbol{x}$ is the feature vector of a single training example, C_k represents the true class membership of the training example, $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}$ are the mean and covariance of the multivariate Gaussian distribution for class k. Since linear discriminate analysis assumes all classes have the same covariance, the class index for $\boldsymbol{\Sigma}$ is omitted. At the decision boundary between two classes k and j, $p(C_k|\boldsymbol{x}) = p(C_j|\boldsymbol{x})$, we have

$$\begin{aligned} 1 = \frac{p(C_k|\boldsymbol{x})}{p(C_j|\boldsymbol{x})} &= \frac{p(C_k)}{p(C_j)} \exp\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_k)^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_k) + \frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_j)^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_j)\} \\ &= \frac{p(C_k)}{p(C_j)} \exp\{\frac{1}{2}\boldsymbol{\Sigma}^{-1}(2\boldsymbol{x}^T\boldsymbol{\mu}_j - 2\boldsymbol{x}^T\boldsymbol{\mu}_k + \boldsymbol{\mu}_k^T\boldsymbol{\mu}_k - \boldsymbol{\mu}_j^T\boldsymbol{\mu}_j)\} \end{aligned} \tag{5.2}$$

We can see that the decision boundary is linear with respect to $\boldsymbol{x}$. The parameters $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}$ can be estimated by the training set sample mean and covariance, respectively, and the class prior $p(C_k)$ can be estimated by the class ratios in the training set.

Logistic Regression

From Bayes theorem:

$$p(C_1|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|C)p(C_1)}{p(\boldsymbol{x})} = \frac{p(\boldsymbol{x}|C_1)p(C_1)}{p(\boldsymbol{x}|C_1)p(C_1) + p(\boldsymbol{x}|C_2)p(C_2)} \tag{5.3}$$

use $p(\boldsymbol{x}|C_1)p(C_1)$ to divide the nominator and denominator, we have

$$p(C_1|\boldsymbol{x}) = \frac{1}{1 + \frac{p(\boldsymbol{x}|C_2)p(C_2)}{p(\boldsymbol{x}|C_1)p(C_1)}} \tag{5.4}$$

If we denote $a = \ln \frac{p(\boldsymbol{x}|C_1)p(C_1)}{p(\boldsymbol{x}|C_2)p(C_2)}$ and substitute it into equation 5.4, we obtain

$$\sigma(a) = p(C_1|\boldsymbol{x}) = \frac{1}{1 + \exp(-a)} \tag{5.5}$$

a is also called log odds. If a is linear with respect to the input features, we obtain the formulation of logistic regression:

$$p(C_1|\boldsymbol{\phi}(\boldsymbol{x})) = \sigma(\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}) + b) = \frac{1}{1 + e^{-(\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x})+b)}} \tag{5.6}$$

where $\boldsymbol{w}$ is the weight vector, and $\boldsymbol{\phi}(\boldsymbol{x})$ is called the *basis function* of $\boldsymbol{x}$. $\boldsymbol{\phi}(\boldsymbol{x})$ is a fixed function that transforms the original data point $\boldsymbol{x}$ into a "feature space", thus can be seen as the feature extraction step. Logistic regression is said to be a linear classifier because the decision surface is linear with respect to the input feature vector $\boldsymbol{x}$. We can use a maximum likelihood approach to estimate $\boldsymbol{w}$ and b. If we use $t_i \in \{0, 1\}$ to denote the labels, with $t_i = 1$ denoting samples in class C_1, then the likelihood of the class membership of the entire training set $\boldsymbol{T} = \{t_i\}$ given the design matrix $\boldsymbol{X}$ and the model parameters $\boldsymbol{w}$ and b can be written as

$$p(\boldsymbol{T}|\boldsymbol{X}, \boldsymbol{w}, b) = \prod_{i=1}^{m} \sigma_i^{t_i}(1 - \sigma_i)^{1-t_i} \tag{5.7}$$

$$E = -\ln p(\boldsymbol{T}|\boldsymbol{X}, \boldsymbol{w}, b) = -\sum_{i=1}^{m} t_i \ln \sigma_i + (1 - t_i)\ln(1 - \sigma_i) \tag{5.8}$$

Take the first and second derivatives of E with respect to $\boldsymbol{w}$ and obtain

$$\frac{\partial E}{\partial \boldsymbol{w}} = \sum_{i=1}^{m}(\sigma_i - t_i)\boldsymbol{x}_i \tag{5.9}$$

$$\frac{\partial^2 E}{\partial \boldsymbol{w}^2} = \sum_{i=1}^{m} \frac{\boldsymbol{x}_i^T \boldsymbol{x}_i e^{\boldsymbol{w}^T \boldsymbol{x}_i + b}}{(1 + e^{-\boldsymbol{w}^T \boldsymbol{x}_i + b})^2} \geq 0 \tag{5.10}$$

we can see that the second derivative is non-negative, and only equals 0 when $\boldsymbol{x} = 0$. This means the loss is convex but not quadratic, as $\frac{\partial^2 E}{\partial \boldsymbol{w}^2}$ depends on $\boldsymbol{w}$, which means we can use iterative convex optimization to find the optimal $\boldsymbol{w}$, but there is no analytical solution for $\boldsymbol{w}$.

The multi-class logistic regression is similar to the binary logistic regression, except that we use softmax function (equation 5.11) instead of equation 5.5 to model $P(C_k|\boldsymbol{x})$. The maximum likelihood approach is similar.

$$\sigma(\boldsymbol{a})_i = \frac{e^{a_i}}{\sum_{i=1}^{K} e^{a_i}} \tag{5.11}$$

for $i = 1, .., K$ and $\boldsymbol{a} = \{z_i\} \in \mathbb{R}^K$. The output of softmax is a K-element vector.

5.2.2 Naïve Bayes

Naïve Bayes assumes that the distribution of the input features is conditionally independent, given the class. Suppose we are performing K class classification on data points with D features for each data point, we have

$$p(\boldsymbol{x}|C_k) = p(x_1, ..., x_D|C_k) = \prod_{i=1}^{D} p(x_i|C_k) \tag{5.12}$$

where $x_1, ..., x_D$ are the input features of a training example $\boldsymbol{x}$. To make predictions, we can calculate the posterior by

$$\begin{aligned} p(C_k|x_1, ..., x_D) &= \frac{p(x_1, ..., x_D|C_k)p(C_k)}{p(x_1, ..., x_D)} \\ &= \frac{\prod_{i=1}^{D} p(x_i|C_k)p(C_k)}{p(x_1, ..., x_D)} \\ &= \frac{\prod_{i=1}^{D} p(x_i|C_k)p(C_k)}{\sum_{j=1}^{K} p(x_1, ..., x_D|C_j)p(C_j)} \\ &= \frac{\prod_{i=1}^{D} p(x_i|C_k)p(C_k)}{\sum_{j=1}^{K} \prod_{i=1}^{D} p(x_i|C_j)p(C_j)} \end{aligned} \tag{5.13}$$

We can obtain $p(C_k)$ by calculating the class ratio in the training set, and fit $p(x_i|C_j)$ by maximum likelihood. Naive Bayes is suitable when the feature dimension is high and is an excellent way to link models trained on different features.

5.2.3 Kernel Methods

Support Vector Machine

Support vector machine (SVM, Taylor and Cristianini 2000; Müller et al. 2001; Schölkopf and Smola 2002; Herbrich 2001) aims to find the solution that separates the two classes with the largest "margin", defined as the minimum distance between the data points and the decision boundary. Suppose a binary classification where the decision boundary is expressed as

$$y = \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}) + b = 0 \tag{5.14}$$

The distance between a data point $\boldsymbol{x}$ and the decision boundary is

$$d = \frac{|\boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}) + b|}{\|\boldsymbol{w}\|} \tag{5.15}$$

Because the SVM is a "large margin" classifier, the convention of class membership notation is different. In SVM, the class labels are usually denoted as $t_n \in \{-1, 1\}$, then for correctly classified data points, we have:

$$t_n y_n = t_n(\boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}_n) + b) > 0 \tag{5.16}$$

SVM maximises the margin. Thus the optimisation problem can be written as

$$\boldsymbol{w}, b = \underset{\boldsymbol{w},b}{\operatorname{argmin}} \min_i \frac{t_n(\boldsymbol{w}\boldsymbol{\phi}(\boldsymbol{x}_i) + b)}{\|\boldsymbol{w}\|} \tag{5.17}$$

where i indexes the training examples. Because the distance does not change if we scale $\boldsymbol{w}$ and b by a factor, we can set $t_i(\boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}_i) + b) = 1$ for the data points closest to the decision boundary, and the optimisation problem can be written as

$$\boldsymbol{w} = \underset{\boldsymbol{w}}{\operatorname{argmax}} \frac{1}{\|\boldsymbol{w}\|} \tag{5.18}$$

subject to

$$t_i(\boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}_i) + b) \geq 1 \tag{5.19}$$

for $i = 1, 2, ..., m$. The margin boundaries can be expressed as $y = \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}_n) + b = \pm 1$.

Because maximising $\frac{1}{\|\boldsymbol{w}\|}$ is equivalent to minimising$\|\boldsymbol{w}\|^2$, we obtain the canonical optimisation representation of SVM:

$$\boldsymbol{w} = \underset{\boldsymbol{w}}{\operatorname{argmin}} \frac{1}{2}\|\boldsymbol{w}\|^2 \tag{5.20}$$

subject to equation 5.19, and $\frac{1}{2}$ is for convenience of derivation. This is a constrained optimisation problem, corresponding to minimising

$$L(\boldsymbol{w}, b, \boldsymbol{a}) = \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{m} \boldsymbol{a}_i\{t_i(\boldsymbol{w}^T\phi(\boldsymbol{x}_i) + b) - 1\} \tag{5.21}$$

with Karush-Kuhn-Tucker (KKT) conditions:

$$\boldsymbol{a}_i \geq 0 \tag{5.22}$$

$$t_i(\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}) + b - 1) \geq 1 \tag{5.23}$$

$$\boldsymbol{a}_i(t_i(\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}) + b) - 1) = 0 \tag{5.24}$$

where $\boldsymbol{a}_n$ are called Lagrange multipliers. Setting derivative of $L(\boldsymbol{w}, b, \boldsymbol{a})$ w.r.t $\boldsymbol{w}$ and b to 0, we obtain

$$\boldsymbol{w} = \sum_{i=1}^{m} \boldsymbol{a}_i t_i \boldsymbol{\phi}(\boldsymbol{x}) \tag{5.25}$$

$$0 = \sum_{i=1}^{N} \boldsymbol{a}_i t_i \tag{5.26}$$

and substitute into equation 5.21, we obtain the dual representation of the optimisation problem:

$$\begin{aligned} \tilde{L}(\boldsymbol{a}) &= -\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \boldsymbol{a}_i\boldsymbol{a}_j t_i t_j k(\boldsymbol{x}_i, \boldsymbol{x}_j) + \sum_{i=1}^{m} \boldsymbol{a}_i t_i b + \sum_{i=1}^{m} \boldsymbol{a}_i \\ &= \sum_{i=1}^{m} \boldsymbol{a}_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \boldsymbol{a}_i\boldsymbol{a}_j t_i t_j k(\boldsymbol{x}_i, \boldsymbol{x}_j) \end{aligned} \tag{5.27}$$

subject to

$$\boldsymbol{a}_i \geq 0, \forall n \tag{5.28}$$

$$\sum_{i=1}^{m} \boldsymbol{a}_i t_i = 0 \tag{5.29}$$

where $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{\phi}(\boldsymbol{x}_i)^T \boldsymbol{\phi}(\boldsymbol{x}_j)$ is called the kernel function. The benefit of dual representation is that we do not need to represent $\boldsymbol{\phi}(\boldsymbol{x})$ explicitly. Also, if the dimension of feature space is larger than the number of training examples, optimizing the dual representation 5.27 is computationally more efficient than optimising the canonical representation 5.21. We can also represent y using dual formulation by substituting equation 5.25 into equation 5.14, and obtain

$$y(\boldsymbol{x}) = \sum_{i=1}^{m} \boldsymbol{a}_i t_i k(\boldsymbol{x}, \boldsymbol{x}_i) + b \tag{5.30}$$

From equation 5.30 we can see that either $\boldsymbol{a}_n = 0$ or $t_n y_n - 1 = 0$, which means only points on the margin contribute to the prediction. These points are called "support vectors" hence the name of the model. For datasets that are not linearly separable in the feature space ($\boldsymbol{\phi}$ space), we introduce a slack parameter ξ_n and replace constraint 5.19 by equation 5.31:

$$t_i(\boldsymbol{w}^T \xi(\boldsymbol{x})) \geq 1 - \xi_i \tag{5.31}$$

$\xi_i = 0$ for points on the margin or on the correct side of the margin, $0 < \xi_i \leq 1$ for points within the margin but on the correct side of the decision boundary, and $\xi_i > 1$ for points on the wrong side of the decision boundary. Therefore we try to minimise:

$$C \sum_{i=1}^{m} \xi_i + \frac{1}{2} \|\boldsymbol{w}\|^2 \tag{5.32}$$

Subject to equation 5.31, where $C > 0$ is a hyperparameter controlling the trade-off between training error and model complexity. This is called "soft-margin" SVM. We can see that both the canonical representation and the dual representation losses are quadratic with respect to the parameters, thus SVM loss is quadratic.

The original SVM does not make probabilistic predictions, but makes classifications by the sign of y. Platt, Cristianini, and Shawe-Taylor 2000 proposed probabilistic SVM by "squashing" y in equation 5.14 by a logistic function, i.e.

equation 5.33, and the parameters A and B are learned from minimising the cross-entropy loss using data not used when training the SVM. Because this two-step approach does not jointly optimise the SVM parameters and A and B, it may give sub-optimal posteriors (Tipping 2001).

$$p(t = 1|\boldsymbol{x}) = \frac{1}{1 + e^{-(Ay(\boldsymbol{x})+B)}} \tag{5.33}$$

SVM does not readily extend to $K > 2$ classification scenarios, and one common approach is constructing K separate one-vs-rest classifiers. SVM can also be extended to regression. ϵ-insensitive SVM replaces the mean squared loss used in linear regression by

$$E_\epsilon(y, t) = \begin{cases} 0 & \text{if } |y - t| < \epsilon \\ |\hat{y} - y| - \epsilon & \text{if } |\hat{y} - y| \geq \epsilon \end{cases} \tag{5.34}$$

and minimises

$$C \sum_{i=1}^{N} E_\epsilon(y_i, \hat{y}_i) + \frac{1}{2}\|\boldsymbol{w}\|^2 \tag{5.35}$$

5.2.4 K-Nearest Neighbours

K nearest neighbours (KNN) comes from density estimation, where the density of a class in a small region R is

$$P = \frac{K}{NV} \tag{5.36}$$

Where K is the number of data points of class K in the region, N is the total number of data points, and V is the volume of the small region. KNN partitions the entire feature space into hyperspheres each having exactly K data points and the class membership of the region is assigned to the class that is most represented, i.e. to the class with the most number of data points within the region, and ties are broken at random. KNN is quick to train but slow to predict, because given a new test data point, it needs to calculate the distances of the new test point to every training data point.

5.2.5 Decision Trees

Tree-based models, also known as decision trees, partitions the input space into axis-aligned regions and assigns a simple model to each region. The process of selecting the region given an input data point $\boldsymbol{x}$ can be described as traversal of a binary tree. Classification and regression trees (CART Breiman et al. 1984) is a widely used framework of decision trees. Variations of CART including ID3, and C4.5 (J. Quinlan 1993; J. Ross Quinlan 1986). The training procedure of CART involves growing the tree by exhaustive search of the input variables and the solution thresholds that minimise the residual loss, typically mean squared error for regression and cross-entropy for classification. Decision trees are popular in medical diagnosis as it is intuitive to interpret and similar to the typical medical diagnosis process. However, the split of the tree is very sensitive to the input data: small changes in the training data can result in a very different split of the tree (Hastie, Tibshirani, and J. Friedman 2001). Another drawback of decision trees is that the region boundaries are aligned to the feature axis. Thus the performance of decision trees relies heavily on the input features.

5.2.6 Ensembles

Bagging

Bagging (Breiman et al. 1984), also called bootstrap aggregation, trains a different model on M bootstrapped datasets $\boldsymbol{X}_i$ from the original dataset $\boldsymbol{X}$, then average the predictions of the N models:

$$y_{bag}(\boldsymbol{X}) = \frac{1}{N}\sum_{i=1}^{N} y_i(\boldsymbol{X}_i) \tag{5.37}$$

Averaging the predictions from independently trained models is called committee, and boosting is a type of committee.

Random Forest

Random forest is an ensemble of decision trees. Because a single decision tree is sensitive to the feature split, the random forest is an ensemble of decision trees

that introduces variations among the models by bootstrapping the dataset and constructing a different tree on each bootstrapped subset, then use majority voting to predict the class. In a random forest, the optimal feature selection is made on a randomly selected feature set, and the threshold is chosen to be the optimal split.

Extra Trees

Extra trees, short for "extremely random trees", is similar to random forest, except that 1) it is trained on datasets drawn from the original training sets without replacements, 2) it chooses both the features and the split thresholds at random (Geurts, Ernst, and Wehenkel 2006).

AdaBoost

Boosting differs from committee methods by training a sequence of classifiers each minimising a weighted loss function, instead of training the base classifiers independently. Data points misclassified by the previous classifiers will be given a higher weight in the subsequent classifier. Boosting can give good results even if the base classifiers are weak classifiers, i.e. they are only slightly better than random. Originally designed for classification, boosting can also be extended for regression (J. H. Friedman 2001). AdaBoost (Freund, Schapire, et al. 1996), short for "adaptive boosting", is the most widely used form of boosting (Bishop 2006). Suppose we train a binary AdaBoost classifier containing N base classifiers on a dataset containing m data points. The weight $\boldsymbol{w}_i^{(0)}$ for each data point is initialised as $\frac{1}{m}$. For $j = 1, ..., N$, we train a base classifier to minimise the loss function

$$E_j = \sum_{i=1}^{m} \boldsymbol{w}_i^{(j)} I(\hat{y}_j(\boldsymbol{x}_i) \neq y_i) \tag{5.38}$$

where $\boldsymbol{x}$ represents a single data point, I is the indicator function. $I(\hat{y}_j(\boldsymbol{x}_i) \neq y_i) = 1$ and $I(\hat{y}_j(\boldsymbol{x}_i) = y_i) = 1$.

We then calculate ϵ_j and α_j:

$$\epsilon_j = \frac{\sum_{i=1}^{m} \boldsymbol{w}_i^{(j)} I(\hat{y}_j(\boldsymbol{x}_i) \neq y_i)}{\sum_{i=1}^{m} \boldsymbol{w}_i^{(j)}} \tag{5.39}$$

$$\alpha_j = \ln \frac{1 - \epsilon_j}{\epsilon_j} \tag{5.40}$$

and update the weights for each data point:

$$\boldsymbol{w}_i^{(j+1)} = \boldsymbol{w}_i^{(j)} \exp\{\alpha_j I(\hat{y}_j(\boldsymbol{x}_i) \neq y_i)\} \tag{5.41}$$

and finally make predictions:

$$\boldsymbol{Y}_j(\boldsymbol{X}) = sign(\sum_{j=1}^{N} \alpha_j y_j(\boldsymbol{X})) \tag{5.42}$$

J. Friedman, Hastie, Tibshirani, et al. 2000 interpreted binary AdaBoost classifier as sequential minimisation of the exponential loss:

$$E = \sum_{i=1}^{m} \exp\{-t_i f_j(\boldsymbol{x}_i)\} \tag{5.43}$$

The exponential loss is exponential with respect to $-t_i f_j(\boldsymbol{x}_i)$, which means it penalises large negative $t_i f_j(x_i)$ heavily, thus sensitive to outliers or mislabeled data points. Another drawback of exponential loss is that it cannot extend to $K > 2$ classification scenarios, nor has negative-log likelihood interpretation of any probability distribution.

Stochastic Gradient Boosting

Stochastic gradient boosting (SGB, J. H. Friedman 2002) is analogous to stochastic gradient descent, but in the function space. It initialises with a base learner, typically a decision tree, and in each iteration, uses another learner trained on a bootstrapped subset of the original dataset to minimise the residual error. Formally, assume the ground truth is $F(\boldsymbol{x}, \boldsymbol{\theta})$, the base learner is initialised to minimise the loss E as

$$F_0(\boldsymbol{x}, \boldsymbol{\theta}) = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \sum_{i=1}^{m} E(y_i, \boldsymbol{\theta}) \tag{5.44}$$

where $\boldsymbol{\theta}$ are the parameters of the base learner F_0, m is the number of data points in the bootstrapped subset. For $j = 1, .., N$,

$$F_j(\boldsymbol{x}) = F_{j-1}(\boldsymbol{x}) + \underset{h_j \in \mathcal{H}}{\operatorname{argmin}} \sum_{i=1}^{m} L(y_i, F_{j-1}(\boldsymbol{x}_i) + h_j(\boldsymbol{x}_i)) \tag{5.45}$$

where $h_j \in \mathcal{H}$ is a base learner.

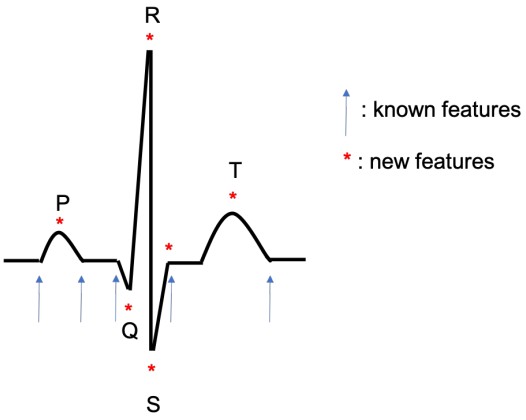


Figure 5.1: Feature extraction. The 12-lead amplitudes of P, Q, R, S, T waves and the baseline level (approximated as the voltage level at the QRS offset) were extracted for each participant from the "typical cycles". The timestamps of onsets and offsets of these waves were given by the Mortara device and are shown by the blue arrows.

5.3 Methods

5.3.1 Feature Extraction

Six additional features were extracted from a "typical cycle" from each of the 12 leads, forming a total of 72 new features for each participant (figure 5.1). They are the P, Q, R, S, T wave amplitudes in the 12 ECG leads and the baseline levels, which are approximated as the voltage level at QRS offset. The positions of the onset and offset of the waves were provided by the Mortara device.

5.3.2 The 11 Machine Learning Models

We selected 11 representative machine learning models from the major machine learning families except neural networks which we will study in the next chapter. They are linear models (Logistic Regression, Linear Discriminant Analysis (LDA)), Naive Bayes, kernel models (SVM), decision trees (CART), neighbours models (KNN), and ensembles (SGB, Bagging, Random Forest, AdaBoost, Extra Trees). Section 5.2 provided a detailed summary of the models used in the analysis. All hyperparameters were left as default as used in the sklearn package (version 0.19.1). We studied the effects of different combinations of the Mortara features, blood

pressure features, and the 72 new ECG features on the classification accuracy, and denote the following feature sets:

- F10: The 10 independent Mortara features described in table 4.1.
- F12: F10, SBP and DBP
- F19: All features described in table 4.1
- F82: F10 and the 72 new features (P, Q, R, S, T, baseline level (see section 5.3.1) $\times$ 12 leads)
- F84: F82, SBP, and DBP

5.3.3 Five-Fold Cross-Validation

We used a standard machine learning approach K-fold cross-validation to separate the training and test sets. In brief, the dataset was divided into K equal portions, and each portion becomes the test set once and only once, while the rest of the dataset becomes the training set. The model was trained on the training set and evaluated on the test set. The mean and standard deviation of the K accuracy values on K test sets were reported as the final results of K-fold cross-validation. We used 5-fold cross-validation in this study.

5.3.4 Normalisation

Machine learning models are sensitive to the scale of the input features. Therefore, we normalised the features according to the mean and standard deviation (SD) of the training set, according to equations 5.46 and 5.47.

$$\boldsymbol{x}_{train} := \frac{\boldsymbol{x}_{train} - \boldsymbol{\mu}_{train}}{\sigma_{train}} \tag{5.46}$$

$$\boldsymbol{x}_{test} := \frac{\boldsymbol{x}_{test} - \boldsymbol{\mu}_{train}}{\sigma_{train}} \tag{5.47}$$

5.3.5 Four-Class Classification

We constructed balanced four-class classification dataset by sampling n samples from each of the "normal", "arrhythmia", "ischaemia", and "hypertrophy" classes [1], with n being the size of the smallest class, i.e. 1,870 (table 4.2), to build a balanced four-class dataset of 7,480 individuals for five-fold cross-validation. The down-sampling and five fold-cross validation were repeated 100 times, and the means and standard deviations of the 100 repeats of the 5-fold cross-validation mean accuracy were reported. In other words, the final result mean μ, and standard deviation σ were calculated as follows:

$$\overline{a}_j = \frac{1}{5}\sum_{i=1}^{5} a_{ij} \tag{5.48}$$

$$\mu = \frac{1}{100}\sum_{j=1}^{100} \overline{a}_j \tag{5.49}$$

$$\sigma = (\frac{1}{100}\sum_{j=1}^{100}(\overline{a}_j - \mu)^2)^{\frac{1}{2}} \tag{5.50}$$

where a_{ij} is the accuracy of ith fold cross-validation accuracy in jth repeat. The above process was repeated for the 11 machine learning models on the five feature sets to find out the best combination of the machine learning model and the feature set for the subsequent analysis.

5.3.6 One-vs-Rest Classification

To study the performance of the identified best machine learning model and feature set to identify normal, "arrhythmia", "ischaemia", and "hypertrophy" from a general population containing "borderline" participants, we performed one-vs-rest classification. The "rest" class included participants from the "other" class (table 4.2). We sampled participants randomly from the class of interest and the "rest" class, with n being the size of the smaller of the two. Then mean and standard deviation of the 100 repeats of the sampling and 5-fold cross-validation accuracy were reported.

[1] Note that here the "arrhythmia", "ischaemia", and "hypertrophy" classes refer to the ECG abnormality groups that are typically associated with the clinical CVD conditions of "arrhythmia", "ischaemia", and "hypertrophy", not to be confused with actual clinical diagnosis, hence the quotation marks.

Table 5.1: Four class classification results.. Results are shown as the mean and std of 100 repeats of 5-fold cross-validation mean accuracy.

Rank	Model	F19	F10	F12	F82	F84
1	SGB	53.5±0.4	53.1±0.4	54.0±0.4	**77.3±0.4**	**77.3±0.4**
2	SVM	51.4±0.4	51.8±0.4	52.6±0.4	73.3±0.3	73.1±0.3
3	Bagging	48.0±0.6	47.9±0.5	48.5±0.5	71.9±0.5	71.8±0.4
4	Random Forest	47.4±0.5	47.8±0.5	48.2±0.5	70.0±0.5	69.9±0.5
5	AdaBoost	50.6±0.6	50.5±0.6	51.3±0.6	69.8±0.5	69.7±0.6
6	Logistic Regression	43.9±0.4	42.4±0.4	43.3±0.4	66.4±0.3	66.3±0.4
7	LDA	44.4±0.4	42.2±0.4	43.3±0.4	65.6±0.4	65.5±0.4
8	Extra Trees	45.2±0.6	46.2±0.5	46.1±0.5	64.9±0.5	64.7±0.6
9	CART	41.2±0.6	41.9±0.6	41.3±0.5	63.2±0.6	63.0±0.6
10	KNN	43.9±0.5	43.5±0.5	43.9±0.5	58.7±0.5	58.4±0.5
11	Naive Bayes	46.9±0.4	46.5±0.4	47.5±0.4	58.3±1.0	58.4±1.0

5.3.7 Feature Ranking

To find out which features the identified best model considered as the most important, we ranked the features in descending order of the average weights of 100 repeats of the 4-class classification and one-vs-rest classification

5.4 Results

5.4.1 Accuracy of Four-Class Classification

Table 5.1 ranked the 11 models in descending order of the average accuracy of the 100 repeats of the 5-fold cross-validation using the F82 set. Of the machine learning models, stochastic gradient boosting (SGB) performed consistently better than other algorithms. SVM and ensembles (bagging, random forest, AdaBoost) generally performed well. F82 with SGB yielded the highest mean accuracy, although no significant difference was found between SGB-F82 and SGB-F84 models (p-value $= 0.75 > 0.05$, unpaired two-tail t-test, $n = 200$). Comparing F19 and F10, removal of the nine dependent Mortara features did not influence classification accuracy significantly ($p = 0.29 > 0.05$, Wilcoxon signed-rank test on means, $n = 22$). Comparing F10 and F82, the addition of 72 new features significantly improved classification accuracy (p-value $= 0.0017 < 0.05$, one-sided Wilcoxon ranked test

Table 5.2: One-vs-rest classification, including the borderline participants. Results are shown as the mean and STD of 100 repeats of five-fold cross-validation mean accuracy.

Class	Sample Size	Accuracy (%)
Normal	21,446	83.3±0.2
"Arrhythmia"	4,294	84.1±0.4
"Ischaemia"	3,616	95.3±0.2
"Hypertrophy"	6,680	95.7±0.2

on means, $n = 22$). Similarly, comparing F82 and F84, the addition of SGB significantly lowered classification accuracy, although by a slight margin (Wilcoxon ranked test in means, $n = 22$, p-value $= 0.002 < 0.05$), suggesting blood pressure data is a confounder at the presence of the 72 additional features.

5.4.2 One-vs-Rest Classification

We used the SGB-F82 model to perform one-vs-rest classification. The purpose of this experiment was to see how well the model identified the four classes from a general population that includes participants in the "borderline" area that cannot be categorised into any of the four categories.

Table 5.2 shows that the SGB-F82 model performed well for all one-vs-rest classification, achieving over 80% mean accuracy for all experiments. "Hypertrophy" was reliably detected, followed by "ischaemia", "arrhythmia", and normal. The relatively low performance on the normal class may reflect the fact that borderline participants belong to the "sub-healthy" group, and are therefore, closer to the "normal" class than the "arrhythmia", "ischaemia", and "hypertrophy" classes in the latent space. To test this hypothesis, we performed one-vs-rest classification excluding the borderline participants, and the results are shown in table 5.3. Comparing table 5.2 with 5.3, the removal of the borderline participants from the "rest" class indeed boosted normal class classification, but also made the other three disease classification more complicated because it is hard to distinguish them from each other. This led to another question: which of the three disease classes (arrhythmia, "ischaemia", and "hypertrophy") are more likely to be confused with another disease class by the SGB-F82 model?

Table 5.3: One-vs-rest classification excluding the borderline participants. Results are shown as the mean and std of 100 repeats of five-fold cross-validation mean accuracy.

Class	Sample Size	Accuracy (%)
Normal	13590	89.1±0.1
"Arrhythmia"	4294	79.5±0.4
"Ischaemia"	3616	86.6±0.4
"Hypertrophy"	6680	92.1±0.2

Table 5.4: 2- and 3-class classification, excluding the borderline participants. Results are mean±std of 5-fold cross-validation.

	Experiments	n	Accuracy (%)
2-class	Normal vs "Hypertrophy"	6680	94.1±0.7
	"Ischaemia" vs "Hypertrophy"	3616	91.5±1.3
	Normal vs "Ischaemia"	3616	91.0±0.6
	"Arrhythmia" vs "Hypertrophy"	4294	88.9±1.2
	Normal vs "Arrhythmia"	4294	85.8±0.5
	"Arrhythmia" vs "Ischaemia"	3616	83.1±0.5
3-class	Normal vs "Ischaemia" vs "Hypertrophy"	5424	86.6±0.8
	Normal vs "Arrhythmia" vs "Hypertrophy"	6441	83.3±0.7
	"Arrhythmia" vs "Ischaemia" vs "Hypertrophy"	5425	81.3±1.0
	Normal vs "Arrhythmia" vs "Ischaemia"	5425	78.9±1.1
4-class	Normal vs "Arrhythmia" vs "Ischaemia" vs "Hypertrophy"	7232	77.5±0.8

5.4.3 Two- and Three- Class Classification

To answer this question, we conducted 2- and 3- class balanced classification. An equal number of samples were selected from the larger classes to match the smallest class in order to construct balanced datasets for five-fold cross-validation. Results are shown in table 5.4 as the mean and standard deviation of the 5-fold cross-validation accuracy and ranked in descending order of the means in each of the 2- and 3-class categories. In 2-class classification, classification of "arrhythmia" and "ischaemia" yielded the lowest accuracy, suggesting SGB-F82 has difficulty in distinguishing "arrhythmia" and "ischaemia". It is validated in 3-class classification results where the absence of either "arrhythmia" or "ischaemia" produced better performance.

5.4.4 Feature Ranking

Although SGB-F82 yielded the highest mean accuracy in section 5.4.1, there is very little difference between SGB-F82 and SGB-F84, and we wish to see the ranking of blood pressure in feature ranking analysis. Therefore, we included SBP and DBP, using SGB-F84 for feature ranking.

We listed the top 10 features from each of the one-vs-rest classifications, and the four-class classification in descending order of the mean weights in 100 repeats. The 72 new features are shown in bold font on table 5.5.

Table 5.5 shows that the Mortara features are most important for distinguishing individuals with normal, "arrhythmia", or "ischaemia", while the new features are particularly important for "hypertrophy" classification.

Of all the 84 features, QRS duration appeared most frequently (5 times), followed by QRS axis (4 times), average RR (4 times), PR duration (4 times), R amplitude in V5 (4 times), S amplitude (V1, 4 times), and age (3 times). Blood pressure features only appeared once in top 10 features, ranking 10th in "ischaemia" classification. Lead V5 appeared most frequently among the 12 leads, for 5 times, followed by leads V1 (4 times), and (3 times), II (twice), III (twice), aVF (twice), and V6 (twice). Lead I did not appear at all. In terms of the amplitude of the waves, T wave amplitude appeared most frequently, for 9 times, followed by Q (7 times), S (5 times), and R (4 times). P wave amplitude did not appear at all. While R and S wave amplitudes concentrate in lead V5 and V1, respectively, Q and T wave amplitudes spread to many leads.

These features are quite different from the clinical criteria. For example, the three most common clinical criteria for left ventricular hypertrophy - Sokolow-Lyon index (Sokolow and Lyon 1949), Cornell voltage criterion (Casale et al. 1987), and Romhilt-Lyon point score system (Romhilt and Estes Jr 1968) - all consider S and R amplitudes exclusively, while SGB-F84 model identified mostly Q and T amplitudes in the top 10 features to detect "hypertrophy", suggesting our model may have discovered different patterns than those already identified in clinic.

Table 5.5: Feature ranking. Features are ranked in descending order of the mean weights of 100 repeats. The borderline participants were included in the rest class of one-vs-rest classification. New features are in bold font.

Rank	**Normal**	**"Arrhythmia"**	**"Ischaemia"**	**"Hypertrophy"**	**Overall**
1	QRS axis	QRS duration	average RR	**Q amp (avF)**	QRS duration
2	QRS duration	average RR	QRS duration	T axis	**R amp(V5)**
3	average RR	PR interval	P axis	**Q amp (aVR)**	average RR
4	PR interval	QRS axis	**S amp (V1)**	**T amp (V3)**	**S amp (V1)**
5	**T amp (V5)**	**T amp (II)**	**S amp (V2)**	**Q amp (aVL)**	QRS axis
6	**T amp (V6)**	age	PR interval	**T amp (V4)**	**Q amp (III)**
7	**R amp (V5)**	**R amp (V5)**	QRS axis	**Q amp (III)**	**Q amp (aVR)**
8	**S amp (V1)**	baseline (V3)	age	QRS duration	PR interval
9	**T amp (II)**	**S amp (V1)**	Q offset	**T amp (V2)**	age
10	**T amp (aVR)**	**T amp (V6)**	SBP	**R amp (V5)**	**Q amp (aVF)**

5.5 Discussion and Conclusion

In this chapter, we demonstrated that machine learning models could indeed classify ECG features with high accuracy without any knowledge of the diagnosis criteria - all they need are relevant features. The 77.3% four-class classification accuracy by SGB-F84 and SGB-F82 is encouraging, especially considering the "arrhythmia", "ischaemia", and "hypertrophy" classes are not mutually exclusive. In fact, they may be the underlying causes of one another. For example, a subclass of arrhythmia, ventricular fibrillation, is often caused by ischemic heart disease (Vaswani et al. 2015). Although for the ease of presentation, the results given in this chapter are classification accuracy, but in fact, the machine learning models give a probability score for each of the target class. We can further validate the models by comparing the probabilities for the four classes with the actual clinical diagnosis for each patient.

The dependent features in table 4.1 can be all expressed as transformations of the independent features, thus contain no additional information beyond the independent features. The comparison between F10 and F19 in table 5.1 validates that there are no improvements in classification accuracy by introducing the dependent features models, and the addition of dependent features may serve as a confounder when there are limited training examples.

The significant improvement using the 72 additional features compared to the Mortara features and the blood pressure features is especially encouraging, considering the naïve feature extraction scheme we used in this chapter. We did not perform any denoising nor advanced signal processing. However, all features were extracted from the "typical cycle" thus did not include much rhythmic information, which may explain the relatively low accuracy in classification of "arrhythmia" (table 5.4). In the next chapter, we will use deep learning to analyse the raw 10-s ECG signals directly.

This chapter has several surprising observations. The top features identified by SGB-F84 are quite different from those that are commonly used in clinical practice. We discovered that lead I was not selected in the top features at all, which may suggest many studies that used only single lead, typically lead I or II, even when the

12-lead ECG is available, have sub-optimal performances and our findings suggest using lead V5 instead of lead I when the single-lead analysis is inevitable due to resource constraints. This can also be understood by looking at table 2.2, which shows lead I is solely determined by the voltages of the left and right arm electrodes, which also contribute to aVR, aVL, aVF, and V1-V6 leads. In other words, the information contained in lead I is already contained in aVR, aVL, aVF, and V1-V6 leads, while V1-V5 contain information from the precordial electrodes (v1-v6) which are not shared with another lead. Thus it makes sense for a machine learning model to exclude redundant information and include independent information (i.e., information that cannot be obtained from other sources).

On further analysis of the features provided by the Mortara device (table 4.1) and compare with the features commonly used in clinical ECG interpretation (table 2.3), we can see that, similarly, Mortara device included many "redundant" features (i.e. features that could be expressed as functions of other features, such as P wave onset) and did not include clinically-relevant features such as ST-segment, which is likely due to the difficulty in detecting the onset of T waves in ECG. This is a general problem in ECG classification using the traditional machine learning pipeline of beat-and-wave segmentation $\rightarrow$ feature extraction $\rightarrow$ classification, as the difficulties in beat-and-wave segmentation would prevent accurate information to flow to the next step of the pipeline. For example, ST-segment feature may be left out because of the challenges in T wave onset detection, and even if the researchers detected T wave onset and extracted the ST segment feature, the error in T wave detection might propagate to the downstream steps via the ST segment feature. This motivates the need to apply machine learning methods to unsegmented signals, essentially learning the feature extraction step and classification step jointly. Deep learning is most powerful in this respect, which will be the focus of our next two chapters.

The analysis in this chapter has several limitations. For example, we used a point estimate of the voltage level of Q offset as the baseline level, while in theory, we should have used the average level of P offset to Q onset segment and Q offset

to T onset segment. This is also due to the difficulty in T wave segmentation and will be addressed in the next chapter using deep learning.

Another limitation of this chapter is that our labels were provided by the Mortara machine, which is based on the deterministic rule-based Minnesota Code (Prineas, Crow, and Z.-M. Zhang 2009). In theory, machine learning models can discover the rules given enough training data and training time. To address this issue, in the next chapter, we will perform classification using machine learning models on additional datasets (ICBEB and PhysioNet) where the labels were provided by the cardiologists.

Another limitation of our study is that we did not build gender-stratified models, nor did we include gender as a feature. As introduced in section 2.3, men and women have different risks in many cardiac diseases. For example, men have predominance in ischemic heart disease, we can expect that a gender-stratified model would perform better and be of higher clinical relevance than the unstratified model. We will look at gender-stratified models in Chapter 7.

6 Deep Learning ECG Classification

6.1 Introduction

In the previous chapter, we studied 11 representative traditional machine learning models, although some of them obtained good ECG classification, their performance is sensitive to the choice of the features. Also, these methods need feature extraction from the raw ECG signals. In this chapter, we use end-to-end deep learning to classify ECG signals, taking the raw ECG signals as input, without preprocessing or feature extraction steps. We start with an overview of the core principles of deep learning, followed by a proposal to use a novel deep learning architecture family, called Layer-Wise Convex Networks (LCNs), and a theorem by the same name. Then we introduce a heuristic algorithm - the AutoNet - designed to automatically generate LCNs based on the characteristics of the training set. Finally, we demonstrate the performance of AutoNet-generated LCNs compared to the state-of-the-art end-to-end deep learning model for ECG classification on three datasets: (i) International Conference on Biomedical Engineering and Biotechnology (ICBEB)[1] Physiological Signal Challenge 2018, (ii) the PhysioNet Atrial Fibrillation Detection Challenge 2017 (Clifford et al. 2017), and (iii) the China Kadoorie Biobank (CKB)[2]. The data description for the three datasets are provided in Chapter 4.

[1] http://2018.icbeb.org/Challenge.html
[2] https://www.ckbiobank.org/site/

6.2 Introduction to Deep Learning

A comprehensive introduction to deep learning merits a textbook in itself. This section summarises the core principles of deep learning to enable the readers to understand this thesis. Interested readers are encouraged to refer to I. Goodfellow, Bengio, and Courville 2016 for further details.

The name "deep learning" was given to neural networks after the rediscovery of their power in pattern recognition since 2012, thanks to the growing amount of training data, increasing computational capacity, and theoretical and algorithmic advances which have enabled successful training of much deeper neural networks than what was previously possible. In this thesis, we use the term "deep learning" and "neural networks" interchangeably.

6.2.1 Basic Formulation

Let us use supervised K-class classification as an example, and denote the design matrix with $\boldsymbol{X} \in \mathbb{R}^{D \times m}$, where D is the dimension of the feature vector, m is the number of training examples, $\boldsymbol{Y} \in \mathbb{R}^{K \times m}$ represents the one-hot-encoded training targets (in unsupervised learning, $\boldsymbol{Y}$ may be equal to $\boldsymbol{X}$ or some function of $\boldsymbol{X}$), where K is the number of classes. Let $\hat{\boldsymbol{Y}}$ represent the prediction of $\boldsymbol{Y}$ given by an L-layer neural network, then each layer of the network computes:

$$\boldsymbol{Z}^{[l]} = \boldsymbol{W}^{[l]} \boldsymbol{A}^{[l-1]} + \boldsymbol{b}^{[l]} \tag{6.1}$$

$$\boldsymbol{A}^{[l]} = g^{[l]}(\boldsymbol{Z}^{[l]}) \tag{6.2}$$

for $l = 0, 1, ..., L$. Layer 0 and layer L represent the input and the output layers, respectively; in other words, $\boldsymbol{A}^{[0]} = \boldsymbol{X}$, and $\boldsymbol{A}^{[L]} = \boldsymbol{Y}$. $\boldsymbol{A}^{[l]} \in \mathbb{R}^{n^{[l]} \times m}$ is called the *activation* or *output* of layer l; $g^{[l]}$ is (usually) the non-linear activation function of layer l; $\boldsymbol{Z}^{[l]} \in \mathbb{R}^{n^{[l]} \times m}$ is the affine transformation of the activations of layer $l-1$; $\boldsymbol{W}^{[l]} \in \mathbb{R}^{n^{[l]} \times n^{[l-1]}}$ is the weight matrix pointing from layer $l-1$ to layer l in the

forward pass; $n^{[l-1]}$ and $n^{[l]}$ are the number of neurons in layer $l-1$ and layer l, respectively. $\boldsymbol{b}^{[l]} \in \mathbb{R}^{n^{[l]}}$ is the bias vector of layer l.

Loss Functions and Output Activations

The choice of the loss functions and the output activation functions are closely linked to the machine learning problem. For binary classification, the default choice is the binary cross-entropy loss (equation 6.3) with a sigmoid output; for K-class ($K > 2$) classification, the default choice is the multi-class cross-entropy loss (equation 6.4) with a softmax output; and for regression problems, the default choice is the mean squared error (equation 7.4), and linear output (identify mapping). These choices correspond to the maximum likelihood approach.

$$E = -\frac{1}{m}\sum_{i=1}^{m}[y_i \log \hat{y}_i + (1-y_i)\log(1-\hat{y}_i)] \tag{6.3}$$

$$E = -\frac{1}{m}\sum_{i=1}^{m}\sum_{k}^{K} y_{ik} \log \hat{y}_{ik} \tag{6.4}$$

$$E = \frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2 \tag{6.5}$$

To see this, we first look at binary classification and assume we have m training examples and denote the two classes as class 0 and class 1, and the training target for the ith training example as $y_i \in \{0, 1\}$. We interpret the output of the neural network as the estimate of the posterior for class 1, i.e. $\hat{y}_i = p(y_i = 1|\boldsymbol{x}_i)$, then we can write down the posterior for each training example $\boldsymbol{x}_i$ as:

$$p(y_i|\boldsymbol{x}_i, \boldsymbol{\theta}) = \hat{y}_i{}^{y_i}(1-\hat{y}_i)^{1-y_i} \tag{6.6}$$

where $\hat{y}_i = \hat{y}_i(\boldsymbol{x}_i, \boldsymbol{\theta})$. Assume the training examples are identically and independently distributed (i.i.d.), using the Bayes rule, we can write down the likelihood of the entire training set as

$$p(\boldsymbol{X}|\boldsymbol{Y}, \boldsymbol{\theta}) = \prod_{i=1}^{m} p(\boldsymbol{x}_i|y_i, \boldsymbol{\theta}) = \prod_{i=1}^{m} \frac{p(y_i|\boldsymbol{x}_i, \boldsymbol{\theta})p(\boldsymbol{x}_i)}{p(y_i)} = \prod_{i=1}^{m} \hat{y}_i^{y_i}(1-\hat{y}_i)^{1-y_i}\frac{p(\boldsymbol{x}_i)}{p(y_i)} \tag{6.7}$$

Taking the negative logarithm of equation 6.7, we have

$$-\log p(\mathbf{X}|\boldsymbol{Y},\boldsymbol{\theta}) = -\sum_{i=1}^{m}[y_i \log \hat{y}_i + (1-y_i)\log(1-\hat{y}_i)] - \sum_{i=1}^{m}\log p(\boldsymbol{x}_i) + m\log C \quad (6.8)$$

Because the logarithm function is monotonically increasing, maximising the likelihood (equation 6.7) is equivalent to minimising the negative log-likelihood (equation 6.8); and because the term $-\sum_{i=1}^{m}\log p(\boldsymbol{x}_n) + m\log C$ in equation 6.8 is invariant to $\boldsymbol{\theta}$, maximising the likelihood is equivalent to minimising the first term on the right-hand side of equation 6.8, which is equivalent to minimising the binary cross-entropy loss (equation 6.3).

One justification for using the sigmoid output for binary classification is that the output of the sigmoid function lies in the open interval of $(0, 1)$. Another justification is that we can rearrange the posterior $p(y = 1|\boldsymbol{x})$ into a sigmoid function, the proof of which has been given in Chapter 5.

The similar argument applies to softmax output layers with multi-class cross-entropy loss for multi-class classification, as the sigmoid function can be seen as a particular case of softmax when $K = 2$. Similarly, the mean squared error loss with linear output can be derived from taking the negative log of Gaussian likelihood $\prod_{i=1}^{m}\mathcal{N}(\hat{y}_i, \beta^{-1})$, where β^{-1} is the precision and is invariant to $\boldsymbol{\theta}$. We can see that these choices mean that the loss is non-convex with respect to the parameters if the network has at least one non-linear hidden layer.

Hidden Layer Activation

In principle, the aforementioned activation functions - sigmoid, softmax, and linear - can also be used in hidden layers, although now the use of sigmoidal activations is discouraged in feed-forward hidden layers. Hidden layers with linear activations have the effect of dimension reduction, and two consecutive linear layers are not equivalent to a single linear layer with the same number of parameters, as the former approach requires the resulting weight matrix to be decomposable into two real matrices, while the latter approach has no such constraint.

Rectified linear unit (ReLU, Jarrett et al. 2009; Nair and G. E. Hinton 2010; Glorot, Bordes, and Bengio 2011, equation 6.9) remains the default choice when

building neural networks. The drawback of ReLU is that it has large regions with 0 gradient, therefore researchers have come up with piece-wise linear activations that have gradients everywhere, such as leaky ReLU (Maas, Hannun, and A. Y. Ng 2013), parametric ReLU (K. He et al. 2015) and maxout (I. J. Goodfellow, Warde-Farley, et al. 2013), where the domain of the function is divided into K regions, with each region having increasingly larger positive gradient than the previous region, and the gradient can be learned or fixed.

$$y = max\{x, 0\} \tag{6.9}$$

It was once believed that only the everywhere-differentiable functions could be valid hidden layer activations, and sigmoidal functions were the most popular choices. However, sigmoidal functions have large regions of saturation, where the gradient is very small, which hindered the training of deep neural networks. Nowadays, it is found that functions with defined left and right gradients everywhere are sufficient to act as the hidden activations, while everywhere-differentiability is not necessary. Now the choice of hidden layer activation has the trend of having mostly piece-wise linear regions. This is because piece-wise linear activation functions do not introduce second-order effects. Also, almost all commonly used activation functions are monotonic. I. Goodfellow, Bengio, and Courville 2016 observed that non-monotonic activation functions make training extremely difficult. The switch from sigmoidal functions to piece-wise linear functions as hidden activation is one of the key factors contributing to the recent advancements in deep learning. Jarrett et al. 2009 observed that in small datasets, using piece-wise linear hidden units is more important than actually learning the appropriate weights: ReLU nets with random weights are sufficient to propagate useful information.

Although the use of sigmoidal activations are discouraged in feed-forward hidden layers, they are useful in specialised architectures. For example, tanh and sigmoid can act as "gates" in gated architectures such as the long sort-term memory (LSTM) and the gated recurrent units (GRU).

Width and Depth

The primary neural network architecture design consideration, after deciding on the model family (e.g. feed-forward, recurrent, or convolutional neural networks), is the width and depth of the network. The width refers to the number of neurons in each layer of the network, and the depth refers to how many layers the network contains. There is no consensus as to how to count the layers: some authors count only one of the output and input layers, while others count both; some authors only count layers with learnable parameters, while others also count layers without learnable parameters, such as pooling layers; some authors count the convolutional layers and activation layers separately, while others consider the convolutional and activation a single layer and call it convolutional layer. There is also no consensus as to how many layers qualify as deep.

There is also little theoretical guidance on the choices of the width and depth of the network. A narrow and deep network is generally believed to generalise better than a broad and shallow network, given a fixed total number of parameters. A deep network, compared to a shallower one, also encodes the practitioners' preference for learning hierarchical factors of variations over independent factors of variations, meaning more complicated factors of variations may be built upon simpler factors of variations. However, a deep network can also be more challenging to train, due to vanishing and exploding gradients, and the worsening of Hessian conditioning (in more detail in section 6.2.2) as the depth increases. Another intuition that depth may not always help is that the human brain only has six layers of neurons (Marieb and Hoehn 2007), although there is an abundance of interconnections and feedback loops. Hanin 2018 et al. proved that for ReLU nets, given a fixed total number of parameters, the network with all identical width layers is the least susceptible to vanishing and exploding gradients, suggesting deep networks with the same number of neurons in each layer may have desirable properties.

The choice of depth and width of an neural network is mostly designed by trial and error. In this thesis, we attempt to determine them, based on principles of information theory. We regard each training example as one piece of information,

and our goal is to create a model that makes the most use of the training set while also facilitate optimisation. We determine the depth using principles of reinforcement learning and adapt the model size according to training and validation losses.

6.2.2 Optimisation

From section 6.2.1 we learned that the common choices of loss functions are no longer quadratic with respect to the parameters to be optimised if the network has at least one non-linear hidden layer. Also, neural networks have the property of weight space symmetry, which means swapping two neurons of the same layer and their input and output weights, we can obtain an equivalent neural network with different parameters. Thus we cannot solve for the parameters that minimise the loss. Instead, we must resort to iterative numerical optimisation to reduce the loss.

Gradient Descent

Let $\boldsymbol{\theta}$ denote the vector collecting all parameters of a neural network (including weights and biases), the directional gradient of loss E with respect to any unit vector $\boldsymbol{u}$ is $\nabla_{\boldsymbol{u}} E(\theta)$. To minimise $E(\boldsymbol{\theta})$, we follow the direction that decreases E the fastest, i.e.

$$\boldsymbol{u}^* = \operatorname{argmin} \nabla_{\boldsymbol{u}} E(\boldsymbol{\theta}) \tag{6.10}$$

from vector calculus, we have

$$\boldsymbol{u}^* = -\nabla_{\boldsymbol{\theta}} E(\boldsymbol{\theta}) \tag{6.11}$$

which means the direction that minimises E the most is the negative gradient. The iterative optimisation following the negative gradient of the loss is called *steepest descent*, *gradient descent*, *full-batch gradient descent*, or *batch gradient descent*. Formally, it updates the parameters by

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \alpha \nabla_{\boldsymbol{\theta}} E(\boldsymbol{\theta}) = \boldsymbol{\theta} - \frac{\alpha}{m} \sum_{i=1}^{m} \nabla_{\boldsymbol{\theta}} E_i(\boldsymbol{\theta}) \tag{6.12}$$

where α is the learning rate. Since the loss term is usually a sum over all training examples, one update step of steepest descent would require $O(m)$ computation just for the summation operation. Deep learning typically handles millions of training examples, thus $O(m)$ complexity is undesirable. Researchers realised the loss term could be interpreted as an expectation over the training set. Therefore the expectation can be estimated with much fewer training examples, leading to *stochastic gradient descent*. There is some inconsistency in the literature regarding the use of the term "batch". Here we follow the convention by I. Goodfellow, Bengio, and Courville 2016 and define the following concepts:

- Batch size: the number of training examples used to make an update of the parameters, demoted as m_b.
- Batch gradient descent: same as steepest gradient descent, using all training examples to make one update of the parameters, i.e. $m_b = m$.
- Stochastic gradient descent: using less than all training examples to make one update of the parameters, i.e. $1 \leq m_b < m$.
- Mini-batch gradient descent: using less than all and more than one training examples to make one update of the parameters, i.e. $1 < m_b < m$.
- Online learning: using only one training example to make one update of the parameters, i.e. $m_b = 1$.

Almost all modern deep learning is powered by stochastic gradient descent (I. Goodfellow, Bengio, and Courville 2016). A full pass through the entire training set is called an *epoch*. The batch size influences generalisation error and training speed: a small batch size has a regularisation effect to reduce overfitting, but it takes longer than a large batch size to go through all training examples. The batch size is often set as powers of 2 to take computational advantage of multi-core hardware.

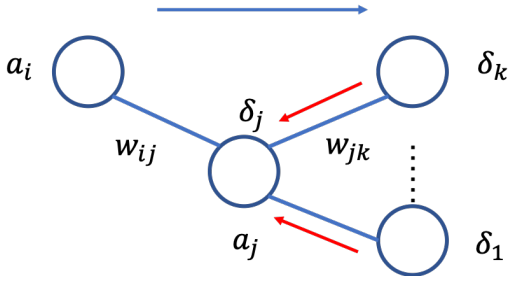


Figure 6.1: Back-propagation expressed as propagation of δ. Blue arrow: forward pass; red arrow: backward pass. Reproduced from Bishop 2006.

Backpropagation

In the last section, we can see that gradient descent requires the evaluation of $\nabla_{\boldsymbol{\theta}} E(\boldsymbol{\theta})$. This is implemented in deep learning using backpropagation, or backprop for short. Backprop is not an optimisation algorithm, but a way to calculate the gradient iteratively using the chain rule of calculus and smart representation of recursive entities, and can be used outside the deep learning context.

The mechanism of backprop is shown in Figure 6.1. Let us denote the weight pointing from unit i to unit j in forward propagation as w_{ij}, then the gradient of the nth training example with respect to w_{ij} can be obtained by the chain rule:

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial z_j}\frac{\partial z_j}{\partial w_{ij}} \tag{6.13}$$

We aim to find an entity that can be expressed recursively from the output layer to the input layer, much like z and a in forwarding propagation, but in reverse. If we use δ_j to denote $\frac{\partial E}{\partial z_j}$, we have

$$\delta_j = \frac{\partial E}{\partial z_j} = \sum_k \frac{\partial E}{\partial z_k}\frac{\partial z_k}{\partial z_j} = \sum_k \delta_k(\frac{\partial z_k}{\partial a_j}\frac{\partial a_j}{\partial z_j}) = \sum_k \delta_k(w_{jk}g_j') = g_j' \sum_k w_{jk}\delta_k \tag{6.14}$$

where k denotes the neurons of the layer above, and we have made use of equations 6.1 and 6.2 and the fact that any changes in neuron j of layer l will influence the loss E via all neurons of the layer above (layer $l+1$). We can see that δ is indeed expressed as a weighted sum over the same entity of the layer above. Finally, we substitute equation 6.14 into equation 6.13 and obtain

$$\frac{\partial E}{\partial w_{ij}} = \delta_j a_i \tag{6.15}$$

Comparing equations 6.14 and 6.15 with equations 6.1 and 6.2, we can see that backprop is analogous to propagating the "differential" information (δ) from the output to the input layer, and δ is calculated as the weighted sum of δ of the layer above, multiplied by the derivative of the activation function of the current layer.

The most common implementation of backpropagation updates all parameters together in each iteration. Block optimisation updates a subset of the parameters in each iteration, for example, layer-by-layer.

Challenges Faced by Gradient Descent

Almost all deep learning is powered by stochastic gradient descent and its variations (I. Goodfellow, Bengio, and Courville 2016). Gradient descent, including stochastic gradient descent, faces several challenges, presented as follows:

Learning Rate

The learning rate (α) is arguably the most critical hyperparameter to tune (I. Goodfellow, Bengio, and Courville 2016). If the learning rate is too high, the training can miss the minimum or even diverge, but if the learning rate is too low, training is prolonged or can get stuck at local minima or plateaus. The choice of the learning rate is highly associated with the conditioning of Hessian, as to be discussed shortly. The most common practice is to have an initial learning rate α_0 and reduce it as the training progresses. There are several protocols: exponential decay ($\alpha = \alpha_0 \times 0.95^{epoch\ number}$), $\alpha = \frac{\alpha_0}{1+decay\_rate \times epoch\ number}$, $\alpha = \frac{\kappa\alpha_0}{(epoch\ number)^{\frac{1}{2}}}$, step decay, and manual decay. Ng recommends prioritising tuning α_0 over the other hyperparameters in the learning rate schedule (A. Ng 2015). I. Goodfellow, Bengio, and Courville 2016 suggests training the model for a few epochs with different learning rates, then initialise the learning rate to be slightly higher than the best-performing learning rate. Learning rate can also increase as training progresses: cyclic learning rate scheduling (Smith and Topin 2017; Smith 2017) periodically

increases and decreases the learning rate, aiming to escape poorly-conditioned areas and local minima.

Poor Conditioning of the Hessian

The Hessian matrix is the second derivative matrix of the loss with respect to the parameters. The conditioning of Hessian is quantified by the condition number of the matrix, calculated as $\max_{i,j} |\frac{\lambda_i}{\lambda_j}|$, where $\boldsymbol{\lambda} = \{\lambda_i\}$ are the eigenvalues of the Hessian matrix. Since the second derivative of any continuous function is permutable, and we rarely use non-continuous activation functions, if at all, the Hessian we encounter in deep learning is real symmetric. Any real symmetric matrix has real eigendecomposition, so the condition number is defined unless the Hessian is singular.

The conditioning of Hessian describes the "curvature" of the loss surface. If the condition number is large, the gradient in the directions corresponding to large-magnitude eigenvalues changes fast, while the gradient in the directions with small eigenvalues changes slowly. Gradient descent has no information regarding the second-order behaviour of the loss surface, thus will take a long time "zigzagging" along the fast-changing directions and make little progress in the slow-changing directions. Training can waste much time if the direction leading to a minimum has a slow-changing gradient. Poor conditioning can also make choosing learning rate difficult, as some directions require a high learning rate, while other directions require a low learning rate.

The Hessian can even be singular, meaning its determinant is 0. Singular Hessian completely degenerates along one or more dimensions, which may be caused by redundancies in the training data (meaning some of the training data are co-linear of each other). In practice, the Hessian can also be singular due to numerical rounding errors and underflow. Singular Hessian's condition number is effectively infinite. The above problems will cause numerical instability and often result in errors in the program.

Critical Points other than Global Minima

Critical points are the points at which all derivatives are **0**. In addition, if the Hessian is positive definite, the critical point is a minimum; if the Hessian is negative definite, the critical point is a maximum; if the determinant of the Hessian is negative, meaning the Hessian has both positive and negative eigenvalues, the point is a saddle point. Finally, if at least one eigenvalue is 0 while other eigenvalues have the same sign, then the point is inconclusive.

For a long time, researchers attributed the difficulty in training deep neural networks to the presence of local minima. This is now found not the case, especially when the parameter space is of high dimension: the probability of encountering minima and maxima are exponentially lower than encountering saddle points. I. Goodfellow, Bengio, and Courville 2016 concluded that local minima typically have low loss rather than high cost; critical points with high loss are typically saddle points, while critical points with very high loss are typically local maxima.

Local maxima rarely cause problems in neural network training using first-order methods, as gradient descent follows the negative gradient downhill, rather than solving for a critical point, where the gradient is **0**. Saddle points may cause problems because training can get stuck at the saddle points where the gradient is **0**, but the loss is still high, as saddle points are maxima in the directions with negative eigenvalues. I. J. Goodfellow, Vinyals, and Saxe 2014 observed that stochastic gradient descent can escape saddle points relatively quickly, perhaps thanks to the noisy gradient estimation introduced by the mini-batches, resulting in the estimated gradient not precisely **0** even at saddle points, thus training can follow the negative gradient to reduce the loss further.

Plateaus are more problematic. At plateaus, gradients of all orders are **0**; training no longer has a guide as to which direction to travel in order to reduce loss further.

Exploding and Vanishing Gradients

Another problem plaguing deep neural network training is the vanishing and exploding gradient problem, especially in architectures that reuse the weight

matrices over may layers (equivalent to unrolling recurrent neural networks along the time axis).

Suppose an architecture reuses $\boldsymbol{W}$ for t layers (or time steps in RNN case), then $\boldsymbol{W}^t$ term will exist in the function it represents. Using eigendecomposition of $\boldsymbol{W}^t$ (equation 6.16), we see that if the diagonal matrix formed by the eigenvalues $\boldsymbol{\lambda}$ of $\boldsymbol{W}$ deviate slightly from the identity matrix, the resulting $\boldsymbol{\lambda}^t$ will have very large or very small values for large t, causing numerical overflow or underflow. This is the reason that learning long-term dependencies is difficult for recurrent neural networks.

$$\boldsymbol{W}^t = (\boldsymbol{V} diag(\boldsymbol{\lambda}) \boldsymbol{V}^{-1})^t = \boldsymbol{V} diag(\boldsymbol{\lambda})^t \boldsymbol{V}^{-1} \tag{6.16}$$

Even if these values can be represented in a computer, they will result in very small or very large gradients (equation 6.14 and 6.15), causing the training to make infinitesimal steps or "jump off cliffs". Researchers realised that it is the direction, rather than the magnitude of the gradient, that matters, and proposed gradient clipping heuristic to mitigate the exploding gradient problem, by capping the magnitude of the gradient to a predefined value. A similar idea inspired optimisation algorithms that adapt the learning rate or re-scale the gradient according to the local gradient magnitude, presented next.

Adaptive Learning Rate Algorithms

In this section, we present algorithms using adaptive learning rates adjusted for local gradient or curvature. They are primarily designed to mitigate the poor conditioning of the Hessian and the vanishing and exploding gradient problems, and to facilitates the choice of learning rate.

Momentum

The intuition of momentum (Polyak 1964) is that we can avoid oscillation along the direction with large Hessian eigenvalues by using an exponentially weighted moving average of the gradient, rather than the gradient *in situ*, to make parameter updates. The momentum update rule is as follows:

Initialise $\boldsymbol{v} = \mathbf{0}$. In each iteration,

$$\boldsymbol{v} = \beta \boldsymbol{v} + (1 - \beta) \nabla_{\boldsymbol{\theta}} E(\boldsymbol{\theta}) \tag{6.17}$$

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \alpha \boldsymbol{v} \tag{6.18}$$

where $\nabla_{\boldsymbol{\theta}} E$ is calculated on the current mini-batch. The default choice of β is 0.9. Bias correction is usually not implemented. Momentum is named after the analogy of Newtonian motion in physics. $\boldsymbol{v}$ is analogous to velocity, $\nabla_{\boldsymbol{\theta}} E$ is analogous to acceleration, $\boldsymbol{\theta}$ is analogous to the position, and α is analogous to the time interval. In physics, momentum equals mass times velocity, and here we assume unit mass. Therefore $\boldsymbol{v}$ is also the value of momentum. Momentum can work with full batch gradient descent or stochastic gradient descent.

RMSprop

Hinton proposed Root Mean Squared Propagation (RMSprop) in the Coursera course *Neural Networks and Machine Learning* (G. Hinton, Nitsh Srivastava, and Swersky 2012). RMSprop rescales the gradient by its magnitude, which is calculated as the square root of an exponentially weighted moving average of the element-wise square of the gradient. The update rule of RMSprop is as follows:

Initialise $\boldsymbol{S}_{\boldsymbol{\theta}} = 0$. in each iteration,

$$\boldsymbol{S}_{\boldsymbol{\theta}} = \beta \boldsymbol{S}_{\boldsymbol{\theta}} + (1 - \beta)(\nabla_{\boldsymbol{\theta}} E)^2 \tag{6.19}$$

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \frac{\alpha}{(\boldsymbol{S}_{\boldsymbol{\theta}} + \epsilon)^{\frac{1}{2}}} \nabla_{\boldsymbol{\theta}} E \tag{6.20}$$

where ϵ is a small positive value, typically 10^{-8}, to avoid division by 0. RMSprop can work with both full-batch or stochastic gradient descent. RMSprop is an excellent go-to optimisation algorithm to try.

Adam

Adaptive momentum estimation (Adam, Kingma and Ba 2014) combines momentum

and RMSprop. Initialise $\boldsymbol{v_\theta}$ to $\mathbf{0}$. In each iteration, compute $\frac{\partial E}{\partial \boldsymbol{\theta}}$ using the current mini-batch, then calculate:

$$\boldsymbol{v_\theta} = \beta_1 \boldsymbol{v_\theta} + (1 - \beta_1)\nabla_{\boldsymbol{\theta}} E \tag{6.21}$$

$$\boldsymbol{v}_{\boldsymbol{\theta}}^{corr} = \frac{\boldsymbol{V_\theta}}{1 - \beta_1^t} \tag{6.22}$$

$$\boldsymbol{S_\theta} = \beta_2 \boldsymbol{S_\theta} + (1 - \beta_2)(\nabla_{\boldsymbol{\theta}} E)^2 \tag{6.23}$$

$$\boldsymbol{S}_{\boldsymbol{\theta}}^{corr} = \frac{\boldsymbol{S_\theta}}{1 - \beta_2^t} \tag{6.24}$$

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \alpha \frac{\boldsymbol{v}_{\boldsymbol{\theta}}^{corr}}{(\boldsymbol{S}_{\boldsymbol{\theta}}^{corr} + \epsilon)^{\frac{1}{2}}} \tag{6.25}$$

Typically bias correction is implemented in Adam, unlike in momentum and RMSprop. The default choice for β_1 (the first moment) and β_2 (second moment) are 0.9 and 0.999, respectively. α still needs to be tuned. Adam is also an excellent go-to algorithm to try.

Batch Normalisation

Another approach to mitigate poor conditioning of Hessian and the vanishing and exploding gradient problems is to look at the neural networks layer by layer. If we fix all other layers, layer l can only "see" the activations coming out of layer $l-1$, which means that the Hessian with respect to the parameters of layer l is determined by $\boldsymbol{A}^{[l-1]}$. Thus a hidden layer cannot tell whether the input is from the training data or a hidden layer. We can normalise the hidden layers just as we can normalise the input data. Batch normalisation (Ioffe and Szegedy 2015) works by addressing the so-called covariance shift. In machine learning, we usually normalise the input features to a standard distribution $\mathcal{N}(0, 1)$. Covariance shift refers to the phenomenon that as training progresses, the hidden layer activations gradually deviate from the standard distribution, causing the Hessian to be poorly

conditioned. The authors also commented that batch normalisation might make the deep layers more robust to small perturbations in the shallower layers. The update rule of batch normalisation is as follows:

Within a mini-batch, calculate

$$\mu = \frac{1}{n}\sum_{i=1}^{n^{[l]}} z_i \tag{6.26}$$

$$\sigma^2 = \frac{1}{n}\sum_{i}^{n^{[l]}}(z_i - \mu)^2 \tag{6.27}$$

$$z_i^{norm} = \frac{z_i - \mu}{(\sigma^2 + \epsilon)^{\frac{1}{2}}} \tag{6.28}$$

$$z_i^* = \gamma_i z_i^{norm} + \beta_i \tag{6.29}$$

$$a_i = g(z_i^*) \tag{6.30}$$

where i indices each neuron in the layer. $\boldsymbol{\beta}_i$, $\boldsymbol{\gamma}_i$ are learnable parameters with each element corresponding to each neuron of the layer. So the parameters of the network are $\boldsymbol{W}^{[l]} \in \mathbb{R}^{n^{[l]} \times n^{[l-1]}}$, $\boldsymbol{\gamma}^{[l]} \in \mathbb{R}^{n^{[l]}}$, $\boldsymbol{\beta}^{[l]} \in \mathbb{R}^{n^{[l]}}$. $\boldsymbol{b}^{[l]}$ is "absorbed" by $\boldsymbol{\beta}^{[l]}$, so no longer needed. ($\boldsymbol{\beta}$ is not to be confused with the hyperparameters of momentum). $\boldsymbol{\gamma}^{[l]}$ and $\boldsymbol{\beta}^{[l]}$ have the same dimension as $\boldsymbol{b}^{[l]}$. $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ are updated similarly as $\boldsymbol{W}$ using backprop.

Batch normalisation is one of the most exciting recent innovations in deep learning. As the Hessian becomes better conditioned, the learning rate can be increased, thus dramatically accelerate training, especially when adaptive learning rate algorithms such as RMSprop and Adam are used, where the impact of improved Hessian conditioning will reflect in the increased magnitude of update steps. Also, by normalising $\boldsymbol{A}$ or $\boldsymbol{Z}$, batch normalisation can prevent $\nabla_{\boldsymbol{W}} E$ to be extremely large or small (equation 6.14 and 6.15), ameliorating exploding and vanishing gradient problems. There is some debate as to whether to normalise $\boldsymbol{Z}$ or $\boldsymbol{A}$. Ng recommends

normalising $\boldsymbol{Z}$, but the original paper normalised $\boldsymbol{A}$, and we also find normalising $\boldsymbol{A}$ works slightly better for Layer-Wise-Convex networks in the experiments.

Batch normalisation with small batch size (32, 64, or 128) also has a slight regularisation effect, as the mean and variance are calculated from mini-batches, hence contain noise. Batch normalisation can sometimes render dropout unnecessary, although batch normalisation is not recommended to be used as the only regularisation method (I. Goodfellow, Bengio, and Courville 2016). If the batch size is 1, for example, when testing one case at a time, the mean and variance can no longer be evaluated using equations 6.26 and 6.27. An exponentially weighted running average of mean and variance obtained during training may be used instead.

Second-Order Methods

Optimisation algorithms using only the first-order derivative is called first-order methods, and algorithms using the second-order derivative is called second-order methods, presented below:

Newton's Method

Newton's method is the most commonly-used second-order method (I. J. Goodfellow, Warde-Farley, et al. 2013). It is derived from the second order Taylor expansion of the loss at any point $\boldsymbol{\theta}_0$:

$$E(\boldsymbol{\theta}) = E(\boldsymbol{\theta}_0) + (\boldsymbol{\theta} - \boldsymbol{\theta}_0)^\intercal \nabla_{\boldsymbol{\theta}} E(\boldsymbol{\theta}_0) + \frac{1}{2}(\boldsymbol{\theta} - \boldsymbol{\theta}_0)^\intercal \boldsymbol{H}(\boldsymbol{\theta} - \boldsymbol{\theta}_0) + O((\boldsymbol{\theta} - \boldsymbol{\theta_0})^3) \quad (6.31)$$

where $\boldsymbol{H}$ is the Hessian evaluated at $\boldsymbol{\theta}_0$. Ignoring $O((\boldsymbol{\theta} - \boldsymbol{\theta_0})^3)$, at a critical point $\boldsymbol{\theta}^*$, $\nabla_{\boldsymbol{\theta}} E(\boldsymbol{\theta}^*) = 0$, we obtain the update rule for Newton's method:

$$\boldsymbol{\theta}^* = \boldsymbol{\theta}_0 - \boldsymbol{H}^{-1} \nabla_{\boldsymbol{\theta}} E(\boldsymbol{\theta}_0) \quad (6.32)$$

If the loss is a quadratic function, Newton's method can jump to the global minimum in one step. If the loss is locally convex, then Newton method can be applied iteratively and converges faster than gradient descent. However, if the Hessian is not locally positive definite, as is often encountered in deep learning,

Newton's method can update in the wrong direction and be attracted to saddle points and maxima. The Levenberg-Marquardt algorithm (Levenberg 1944; Marquardt 1963) improves upon Newton's method by introducing a hyperparameter α which is added to the diagonal of the Hessian, but it works only when the negative eigenvalues do not have large magnitude, in which case α would need to be so large that the Hessian is dominated by the diagonal and behaves similar to gradient descent but at a low convergence rate (the rate of convergence is roughly linear to $|\frac{1}{\lambda_{max}}|$, and in this case, roughly $|\frac{1}{\alpha}|$).

Newton's method also requires inverting $\boldsymbol{H}$, which has cubic complexity of the number of parameters. Thus Newton's method is only applicable to networks with few parameters.

Conjugate Gradient Descent

One way to utilise $\boldsymbol{H}$ without inversion is conjugate gradient descent. It is developed from analysing the drawbacks of gradient descent with line search. Line search is the method of jumping to the minimum of the direction corresponding to the negative gradient in each iteration. The next iteration will start at the critical point of the previous gradient-direction, thus in steepest descent with line search, the two consecutive updates are along orthogonal directions. It is easy to see that this certainly is not the shortest path towards a minimum. Conjugate gradient descent makes training follow the "conjugate directions" in two consecutive steps, defined as $\boldsymbol{d}_t^\intercal \boldsymbol{H} \boldsymbol{d}_{t-1} = 0$, where $\boldsymbol{d}_t$ and $\boldsymbol{d}_{t-1}$ are the directions to descent in step t and step $t-1$, respectively. It can be shown that conjugate gradient descent will take at most $n_{\boldsymbol{\theta}}$ steps to converge to a minimum for quadratic surfaces, with $n_{\boldsymbol{\theta}}$ being the number of parameters.

Traditionally conjugate gradient descent was developed for convex loss functions and as a batch approach. Non-convex conjugate gradient descent has been developed with occasionally resets using steepest descent with line search, and I. Goodfellow, Bengio, and Courville 2016 commented that it is beneficial to initialise conjugate gradient descent with stochastic gradient descent, possibly to arrive at a location

where the surface is approximately quadratic.

BFGS

Algorithms that approximate Newton methods are called *quasi-Newton methods*, of which the Broyden-Fletcher-Goldfarb-Shanno (BFGS, (Head and Zerner 1985)) algorithm is the most prominent, which approximates $\boldsymbol{H}^{-1}$ by iterative low-rank updates of a matrix $\boldsymbol{M}$ of the same dimension. As the update is of lower-rank, the computational complexity is less than $O(n_{\boldsymbol{\theta}}^2)$, but it requires storage of the matrix $\boldsymbol{M}$, which requires $O(n_{\boldsymbol{\theta}}^2)$ memory. The limited memory BFGS (L-BFGS) initialises the $\boldsymbol{M}$ to be identity matrix for each step, and stores the vectors used to update $\boldsymbol{M}$ instead of $\boldsymbol{M}$ itself, which only requires $O(n_{\boldsymbol{\theta}})$ memory.

Parameter Initialisation

Finally, we discuss issues in parameter initialisation. As deep learning uses iterative numerical optimisation, and the loss surface is non-convex, the optimisation is sensitive to the initial parameter values. The initialisation can determine whether the training will converge at all. Weights and biases are usually initialised differently. Weights are usually initialised to be small random values, rather than 0, as the latter will not be able to learn anything due to weight space symmetry. Bias is allowed to be initialised to be 0, although it is advisable to be initialised to be small positive values in ReLU nets to allow gradients to propagate at the beginning of training.

There are several heuristics for weight initialisation: we take note that $\boldsymbol{Z}^{[l]}$ is a weighted sum of $\boldsymbol{A}^{[l-1]}$, so the larger the dimension (number of neurons) of the previous layer, the smaller we want the activation to be. One way is to initialise the weights of layer l to be a uniform distribution of $\mathcal{U}(-\sqrt{\frac{6}{n_{in}+n_{out}}}, \sqrt{\frac{6}{n_{in}+n_{out}}})$, where n_{in} and n_{out} are the number of input and output units, respectively (Glorot and Bengio 2010). This is known as the *Xavier initialisation*. Moreover, if the layer l activation function is ReLU, then initialising the weights to be $\mathcal{N}(0, \frac{2}{n_{in}})$ works better (K. He et al. 2015). This is known as *He Initialisation*.

6.2.3 Regularisation

Weight Norm Penalty

Just like traditional machine learning, we can add a regularisation term to discourage the network from learning exceedingly large parameters in order to reduce overfitting. Weight norm penalty is simply adding a regularisation term $\sum_{i=1}^{n_w} |w_i|_p$, where n_w is the number of weights in the network. L1 loss is the $p = 1$ case and encourages the network to be sparse, while L2 regulation, also called weight decay, in which $p = 2$, encourages weights to shrink to small magnitude. I. Goodfellow, Bengio, and Courville 2016 commented that training may be stuck at a local minimum corresponding to small weight norms, because weight norm penalty will also shrink the magnitude of the gradients (equation 6.14 and 6.15).

Dropout

Dropout (Nitish Srivastava et al. 2014) is one of the most popular regularisation techniques in deep learning thanks to its simplicity. In the layers with dropout, each neuron has the probability d of being multiplied by 0, in other words, being "turned off". d is called the dropout rate. The output of these layers are then divided by $1 - d$ to keep the expectation of the output unchanged. Dropout could be interpreted as an implicit ensemble of many sub-networks of the original network, thus reaping many benefits of ensemble methods. However, dropout can introduce much noise and make hyperparameter tuning difficult, as we would be less sure whether the loss reduction is due to an intervention or the noise in training. Dropout can also limit the model capacity, and when the training set is large, under-fitting rather than overfitting is the primary concern. Thus in large training sets, dropout's harm outweighs its benefit.

Early Stopping

Early stopping is perhaps the most popular regularisation technique in deep learning, thanks to its simplicity and saving of computational resources. It works by tracking the loss on the validation set then terminate training when the validation loss

stops improving for a preset number of epochs or is below a preset value. After the training is completed, the set of parameters with the minimum validation loss is used to make final predictions on the test set. I. Goodfellow, Bengio, and Courville 2016 proved that early stopping is equivalent to L2 weight decay for linear nets. A. Ng 2015 recommends the "orthogonalisation" principle in training neural nets: separating techniques that reduce the training loss (i.e. reduce under-fitting) and techniques that reduce the gap between the training and validation losses (i.e. reduce overfitting), and first apply techniques to lower the training loss to satisfactory values, then use techniques to reduce the gap between the training and validation losses. Early stopping impacts both overfitting and under-fitting thus does not fit into the orthogonalisation principle. Nonetheless, modern practitioners use early stopping almost universally.

Besides the techniques mentioned above, there are many other approaches to improve generalisation, including data augmentation, adversarial training, using small batches, weight sharing/tying, using narrow-and-deep networks instead of wide-and-shallow networks, and as mentioned before, batch normalisation with small batches also has regularisation effects. Usually, if a technique is broadly applicable to different application domains, such as dropout and early stopping, it is considered a regularisation technique, while approaches highly specific to an application domain, such as flipping images, are considered data preprocessing. I. Goodfellow, Bengio, and Courville 2016 recommends when the training loss is acceptably low and the gap between training and test losses are large, gathering more data is almost always the most desirable way to reduce overfitting. When collecting more training data is infeasible, invasive, risky, or costly, which is typical in medical applications, innovation in algorithmic regularisation is especially important.

6.2.4 Specialised Architectures

In this section, we introduce neural network architectures that are especially suitable to process input data with specific structures, namely convolutional neural networks (CNNs) and recurrent neural networks (RNNs). Note that although CNNs and

RNNs were initially proposed to process image data and sequential data, respectively, they have also been successfully applied to many other types of data, thus although the type of neural networks may hint what kind of data they are most adept at, they are not limited to those data structures. The underlying mathematical operations and computational costs, and the practitioners' familiarity with the architecture (for ease of hyperparameter tuning) should be the primary considerations when deciding which type of neural network to employ.

Convolutional Neural Networks

Convolutional neural networks (CNN) are networks with at least one layer of convolutional operation, illustrated in Figure 6.2, equations 6.33 and 6.34, and formally defined by equation 6.35. It is an example of weight sharing mechanism. The "small patch" (yellow) in Figure 6.2 is called a *kernel* or *filter*, which acts as a feature detector. The motivation for CNN is that we want to reuse the "feature detectors" at multiple locations of the input data. For example, we might want to detect eyes anywhere in the image. CNN is not restricted to applications in image processing. Instead, it applies to any data that has distributed features, such as the ECG time-series waveform. Another motivation is that we want to share the weights within the same layer in order to reduce the number of parameters, effectively reducing overfitting and lower computational cost.

$$z_{11} = w_{111}x_{111} + w_{121}x_{121} + ... + w_{333}x_{333} \tag{6.33}$$

$$z_{12} = w_{111}x_{121} + w_{121}x_{131} + w_{131}x_{141} + ... + w_{333}x_{343} \tag{6.34}$$

$$z_{i',j'} = b + \sum_{k=1}^{f_c}\sum_{i=1}^{f_h}\sum_{j=1}^{f_w} w_{i,j,k}x_{(i'-1)s+i,(j'-1)s+j,k} \tag{6.35}$$

Figure 6.2 and equations 6.33 and 6.34 show the convolution operation in deep learning, which is slightly different than what is defined in mathematics, where

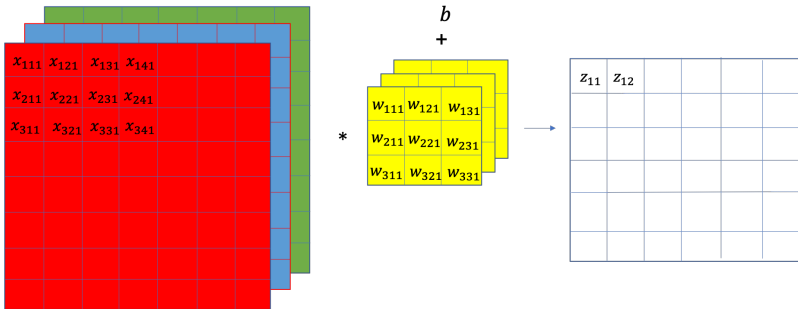


Figure 6.2: The convolution operation of a filter

typically a kernel flipping operation is involved. In deep learning, kernel flipping is usually not implemented. The convolution operation in deep learning is formally defined by equation 6.35, where b is the bias parameter, by convention one bias per filter; c is the number of input channels; f_h and f_w are the kernel height and width respectively; w is the kernel weights; x is the element in the input tensor; s is the stride. The input tensor and the filters must have the same number of channels. Let f_h, f_w, and f_c denote the height, width, and the number of channels of the filter, then the filter has $f_h \times f_w \times f_c$ weight parameters. The resulting tensor from the convolution operation $\boldsymbol{Z} = \{z_{i',j'}\}$ is called a *feature map*. If we have n_f filters, then we will have n_f feature maps. If we use the convention of having one bias parameter per filter, a convolutional layer with n_f filters will have $n_f \times f_w \times f_h \times f_c$ weights and n_f bias parameters. The filters in CNN is equivalent to the *neurons* in feed-forward neural networks.

The number of pixels shifted each time is the *stride* hyperparameter, denoted s, and Figure 6.2 and equations 6.33 and 6.34 show the case when $s = 1$. We can see that the resulting feature map's dimension is 5×5, which is smaller than the input tensor. If we perform this convolution operation for many layers, the resulting tensor will be smaller and smaller. We can "pad" arbitrary values, typically 0s, around the edges of the input tensor, and the number of rows or columns added is another hyperparameter, called *padding*. If there is no row nor column added, it is called "valid padding", and if the resulting feature map has the same width and height as the input tensor, it is called "same padding". The "same padding" is often used to preserve the dimensions of the layers to facilitate skip connections.

In summary, if the input tensor dimension to a convolutional layer is $n_h \times n_w \times n_c$, the kernel dimension is $f_h \times f_w \times f_c$, and there are n_f filters, and we use the convention of one bias per filter, and pad p rows or columns on all edges, with stride s, then by convention $f_c = n_c$, then the output shape of such a convolutional layer is $\lfloor 1 + \frac{n_h - f_h + 2p}{s} \rfloor \times \lfloor 1 + \frac{n_w - f_w + 2p}{s} \rfloor \times n_f$, and the number of parameters (weights and biases) are $n_f \times (f_w \times f_h \times f_c + 1)$.

The convolutional operation is linear and is often followed by a non-linear activation layer, such as ReLU. Another operation often applied in CNN is pooling, which calculates a value from every k input values, typically the max value or the mean value, in effect reduces the dimension of the resulting tensor. Pooling layers do not have parameters to learn. If the input tensor has n_c channels, the output of max-pooling also has n_c channels. The pooling is done on each channel independently. Average pooling is less often used than max-pooling, perhaps because the averaging operation is linear, which can be learned at the convolutional layer, while maxing operation cannot be learned from other layers. Maxpooling usually does not use any padding.

There is very little theoretical guidance on the choice of the CNN hyperparameters - kernel size f_h, f_w, stride s, number of filters in each layer, the number of layers, etc. The following notable CNN architectures have played essential roles in the renaissance of deep learning, and mostly drew attention when they won the ImageNet Large Scale Visual Recognition Competition (ILSVRC) held each year. As will be introduced shortly, they all made some attempts to choose the hyperparameters in a principled way.

Notable Architectures

The first CNN, LeNet-5, was proposed by LeCun et al. 1998 to read handwritten digits. LeNet-5 started using repeating structures comprised of one or more convolutional layers, followed by a pooling layer. These repeating structures were then followed by a flatten layer to concatenate the last output tensor into one long vector, then connect to several densely connected layers for classification. LeNet-5

also popularised the heuristic of reducing f_h and f_w and increasing f_c as the layers go deeper. The convolution-pooling blocks served as feature extraction layers, and the fully-connected layers, typically having a decreasing number of neurons, reduced dimensions gradually, and the final layer served as the classifier. LeNet-5 was trained on grey-scale images and had 7 layers and about 60,000 parameters. Modern CNNs typically have millions of parameters.

AlexNet was proposed by and named after Alexander Krizhevsky (Krizhevsky, Sutskever, and G. E. Hinton 2012) and was the first neural network to win ILSVRC, which has a profound impact on deep learning history as it convinced the computer vision community of the power of deep learning. AlexNet has a similar architecture as LeNet-5 but is a much larger network, with 8 layers and over 62 million parameters. AlexNet was trained on RGB images.

AlexNet has many arbitrary choices of the hyperparameters, especially the kernel size and the stride. Google Inception (Szegedy et al. 2015) forsook the choice of kernel size, and whether or not to use max-pooling, instead, it stacks the outputs of 64 1×1, 128 3×3, 32 5×5 convolutional layers, and one maxpooling layer to form an "inception module", then stack the inception modules to form the whole inception network. Inception has 22 layers and over 6 million parameters. It won ILSVRC in 2013. The computational cost of inception is very high. It used 1×1 convolution (M. Lin, Q. Chen, and Yan 2013) to reduce the computational cost. Inception network has a few side branches with softmax outputs to make sure that hidden layers are indeed learning useful features for the final classification. Inception is also named "GoogLeNet" to pay homage to Yann LeCun and LeNet-5.[3]

Simonyan and Zisserman 2014 took the "principled" hyperparameter selection to another level to build VGG-16. They used an increasing number of neurons as the layers go deeper, resulting in a total of 16 layers and 138 million parameters. The relatively rational choice of hyperparameters makes it attractive to the developers. VGG-16 won ILSVRC in 2014.

[3]The name "inception" comes from the internet meme "we need to go deeper" from the movie Inception (2010).

Residual connections (K. He et al. 2016) are also called skip connections. It is one way to address the vanishing gradient problem in training deep networks and works by copying the activations of a faraway layer to the current layer, and the addition is performed originally before activation and after the affine transformation (equation 6.36, where the residual connection connects layer l and layer $l - \delta$), although there are many variations. K. He et al. 2016 developed the "ResNet" featuring residual connections and won ILSVRC in 2015. ResNet has 152 layers and 60 million parameters.

$$\boldsymbol{A}^{[l]} = g(\boldsymbol{W}^{[l]}\boldsymbol{A}^{[l]} + \boldsymbol{b}^{[l]} + \boldsymbol{A}^{[l-\delta]}) \tag{6.36}$$

We can see that the development of the state-of-the-art CNNs has the trend of increasing depth, but the number of parameters does not necessarily increase.

1-D CNN

The convolution operation can also be performed on sequential data, such as ECG time-series waveform, which can be single-lead or multi-lead, and the ECG leads correspond to the RGB channels of images. The only difference is that $n_h = f_h = 1$. Note that 1-D CNN does not treat multi-channel sequential data as an image. In other words, using 1-D CNN on multi-channel sequential data is not equivalent to stacking the channels together and form a 2-D "image" then feed into a 2-D CNN, as the former approach would require the kernels of the first convolutional layer to have precisely n_c channels, while the latter approach allows for free choice of the kernel size along the n_h dimension as long as $f_h \leq n_c$, while $f_c = 1$.

Recurrent Neural Networks

Recurrent neural networks (RNN), illustrated in Figure 6.3, is designed to use the same model parameters to process long sequences of data, by reusing the weight matrices over many time-steps. RNN is also a weight sharing mechanism, and instead of sharing the weights within the same layer, as CNN does, RNN shares

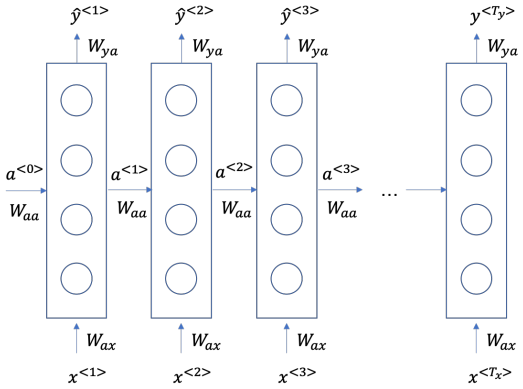


Figure 6.3: Recurrent neural network unrolled through time

weights over depth. If we use $\boldsymbol{X}^{<t>}$ and $\boldsymbol{Y}^{<t>}$to denote the value of $\boldsymbol{X}$ and $\boldsymbol{Y}$ at time step t, then the network can be expressed as

$$\boldsymbol{A}^{<t>} = g_1(\boldsymbol{W}_{aa}\boldsymbol{A}^{<t-1>} + \boldsymbol{W}_{ax}\boldsymbol{X}^{<t>} + \boldsymbol{b}_a) = g_1(\boldsymbol{W}_a[\boldsymbol{A}^{<t-1>}; \boldsymbol{X}^{<t>}] + \boldsymbol{b}_a) \quad (6.37)$$

$$\hat{\boldsymbol{Y}}^{<t>} = g_2(\boldsymbol{W}_y\boldsymbol{A}^{<t>} + \boldsymbol{b}_y) \quad (6.38)$$

where $\boldsymbol{A}^{<0>}$ is initialised as $\mathbf{0}$. $[\boldsymbol{A}; \boldsymbol{B}]$ means vertically stacking matrices $\boldsymbol{A}$ and $\boldsymbol{B}$, and $[\boldsymbol{A}, \boldsymbol{B}]$ means horizontally stacking matrices $\boldsymbol{A}$ and $\boldsymbol{B}$. $\boldsymbol{W}_a = [\boldsymbol{W}_{aa}, \boldsymbol{W}_{ax}]$. Backpropagation in RNN along the time axis is called backprop through time. The activation function g_1 in RNN is usually tanh, and less commonly ReLU. The output activation g_2 is usually sigmoid.

Sometimes not only the information before the query is informative but also the information following the query, such as in the blank-filling task of "They are taking the _____ to Isengard." This gives rise to bidirectional RNN (Schuster and Paliwal 1997).

Because of the weight sharing mechanism, RNN typically has few parameters, but reusing the weight matrices over depth makes RNN especially susceptible to the vanishing and exploding gradient problem. As a result, it is difficult to train RNN over long sequences. Similar to skip connections in CNN, researchers come up with addition operations to help gradient to propagate to distant time-steps. Moreover,

unlike the number of skipped layers being fixed hyperparameters in the ResNet, researchers make the network learn the appropriate time-steps to skip, using "gates". At each time-step, the "state" of the network has a probability of being forgotten, kept the same, or updated. This idea gives rise to gated architectures, with Gated Recurrent Units (GRU, Cho et al. 2014) and Long Short-Term Memory (LSTM, Hochreiter and Schmidhuber 1997) as the most prominent examples. Despite these advances, an RNN still struggles to retrieve information from very distant sequences. The attention mechanism (Ashish Vaswani et al. 2017) makes significant contribution in this respect. It was developed as an encoder-decoder RNN network. The encoder RNN is typically a bidirectional LSTM, and the decoder network is another RNN. The encoder and decoder networks are linked by learnable parameters which represent how much weight the decoder RNN should place on the different time-steps of the encoder RNN outputs.

RNNs can also be stacked together, although the computational cost of a 3-layer RNN is already intensive, so deep RNN is relatively rare.

6.3 Layer-Wise Convex Networks

6.3.1 Motivation

The Layer-Wise convex network (LCN) theorem is motivated by the aim to design neural networks rationally and to make the most out of the training set. A feed-forward neural network is essentially a computational graph where each layer can only "see" the layers directly connected to it, and has no way to tell whether its upstream layer is an input layer or a hidden layer. This "layer-unawareness" idea is similar to what is acknowledged in the development of batch normalisation and is central to the LCN theorem. LCN approaches machine learning from function approximation and information theory perspectives, detailed below:

6.3.2 Derivation

Suppose we have a training set of $\boldsymbol{X} \in \mathbb{R}^{D \times m}$ and training labels $\boldsymbol{Y} \in \mathbb{R}^{m}$, and there exists a deterministic data generating process $f : \boldsymbol{X} \mapsto \boldsymbol{Y}$. We aim to approximate

the data generating process f using a neural network. The universal approximation theorem (Hornik, Stinchcombe, and White 1989; Cybenko 1989) states that a feed-forward neural networks with linear output and at least one sufficiently-wide hidden activation layer with a broad class of activation functions, including sigmoidal and piece-wise linear functions (Leshno et al. 1993, can approximate any continuous function and its derivative (Hornik, Stinchcombe, and White 1990)) defined on a closed and bounded subset of $\mathbb{R}^n$ to arbitrary precision. But how wide should the hidden layer be?

According to universal approximation theorem, there exists a set of neural network parameters $\boldsymbol{\theta}$ such that

$$|f - f(\boldsymbol{\theta})| < \epsilon \tag{6.39}$$

$\forall \epsilon > 0$. As the neural network computes a chain of functions, if we can find $\boldsymbol{\theta}$, then $\forall \epsilon > 0$ and $l \in [0, L]$, it must satisfies the following equations:

$$|g^{[l]}(\boldsymbol{\theta}^{[l]} \tilde{\boldsymbol{A}}^{[l-1]}) - \tilde{\boldsymbol{A}}^{[l]}| < \epsilon \tag{6.40}$$

$$\boldsymbol{A}^{[0]} = \boldsymbol{X} \tag{6.41}$$

$$|\boldsymbol{A}^{[L]} - \boldsymbol{Y}| < \epsilon \tag{6.42}$$

where $\tilde{\boldsymbol{A}}^{[l]} \in \mathbb{R}^{(n^{[l]}+1)\times m}$ and it differs from $\boldsymbol{A}^{[l]}$ as it has one dummy row of 1s to include $\boldsymbol{b}$ into $\boldsymbol{\theta}$. In other words, $\tilde{\boldsymbol{A}} = [\boldsymbol{1}; \boldsymbol{A}]$

To estimate $\boldsymbol{\theta}$: Recall an over-determined system of linear equations $\boldsymbol{Ax} = \boldsymbol{y}$ has a unique set of solutions that minimises the Euclidean distance $|\boldsymbol{Ax} - \boldsymbol{y}|_2$. Can this property be extended to nonlinear equations? The answer is yes, as long as the nonlinear activation $g^{[l]}$ is strictly monotonic and its reverse function is Lipschitz continuous. A real function h is said to be Lipschitz continuous if one can find a positive real constant K such that

$$|h(x_1) - h(x_2)| \leq K|x_1 - x_2| \tag{6.43}$$

for any real x_1 and x_2 on the domain of h. Any function with bounded gradient on its domain is Lipschitz continuous. As the inverse function of strictly monotonic function is defined and unique, we can write the equivalent form of inequality 6.40 and take reverse function of both sides:

$$g^{-1[l]}(\tilde{\boldsymbol{A}}^{[l]} - \epsilon) < \boldsymbol{\theta}^{[l]}\tilde{\boldsymbol{A}}^{[l-1]} < g^{-1[l]}(\tilde{\boldsymbol{A}}^{[l]} + \epsilon) \tag{6.44}$$

Using Lipschitz continuity of $g^{-1[l]}$, we can find a positive real constant K such that

$$g^{-1[l]}(\tilde{\boldsymbol{A}}^{[l]}) - K\epsilon \leq g^{-1[l]}(\tilde{\boldsymbol{A}}^{[l]} - \epsilon) < \boldsymbol{\theta}^{[l]}\tilde{\boldsymbol{A}}^{[l-1]} < g^{-1[l]}(\tilde{\boldsymbol{A}}^{[l]} + \epsilon) \leq g^{-1[l]}(\tilde{\boldsymbol{A}}^{[l]]}) + K\epsilon \tag{6.45}$$

$\forall\epsilon$, which implies

$$|\boldsymbol{\theta}^{[l]}\tilde{\boldsymbol{A}}^{[l-1]} - g^{-1}(\tilde{\boldsymbol{A}}^{[l]})| < K\epsilon \tag{6.46}$$

$$\lim_{\epsilon \to 0} \boldsymbol{\theta}^{[l]}\tilde{\boldsymbol{A}}^{[l-1]} = g^{-1}(\tilde{\boldsymbol{A}}^{[l]}) \tag{6.47}$$

We have conveniently transformed the nonlinear inequations 6.40 into a set of linear equations (equation 6.47), and all we need to do is to make sure equation 6.47 is over-determined, i.e. we have more equations than the number of variables, as we have m training examples, each contributing to one equation, thus the sufficient and necessary condition for equation 6.47 to have a unique solution that minimises the Euclidean distance $|\boldsymbol{\theta}\tilde{\boldsymbol{A}}^{[\boldsymbol{L-1}]} - g^{-1}(\boldsymbol{A}^{[\boldsymbol{L}]})|_2$ is $n_{\boldsymbol{\theta}} \leq m$, and it is easy to see that when $n_{\boldsymbol{\theta}} = m$ we can find the unique solution to make the Euclidean distance arbitrarily close to 0. The formal theory is given in the next section.

6.3.3 The Layer-Wise Convex Theorem

Theorem 1 *For an L-layer feed-forward neural network, the sufficient conditions for there to exist a unique set of parameters* $\boldsymbol{W}^{[l]}$ *and* $\boldsymbol{b}^{[l]}$ *that minimises the Euclidean distance* $|\boldsymbol{A}^{[l]} - g^{[l]}(\boldsymbol{W}^{[l]}\boldsymbol{A}^{[l-1]} + \boldsymbol{b}^{[l]})|_2, \forall l \in [1, L]$ *are:*

- $n_W^{[l]} + n_b^{[l]} \leq m$, $\forall l \in [0, L]$, *where* m *is the number of training examples, and* $n_W^{[l]}$ *and* $n_b^{[l]}$ *are the number of weights and biases in layer* l*, respectively.*
- *The network does not have skip connections;*
- *All activation functions of the network are strictly monotonic, but different layers may have different monotonicity. For example, some layers can be strictly increasing, while other layers can be strictly decreasing.*
- *All reverse functions of the activation functions are Lipschitz continuous.*

Definition 6.3.1 *Layer-Wise Convex Network: Any network fulfilling theorem 1 is called a Layer-Wise Convex Network (LCN).*

Intuitively, the LCN theorem states that if a network fulfills the above conditions, then there exists a unique set of $\boldsymbol{W}^{[l]}$ and $\boldsymbol{b}^{[l]}$ that minimises the distance between $\boldsymbol{A}^{[l]}$ and $g^{[l]}(\boldsymbol{W}^{[l]}\boldsymbol{A}^{[l]} + b^{[l]})$. One may wonder: isn't the distance suppose to be 0 all the time, as defined by equations 6.1 and 6.2? The key difference is that in each backpropagation iteration, LCN views the activations as fixed and already have the appropriate values corresponding to the "optimal" model that can approximate the data generating process with minimum possible error from the information available in the training set, and our goal is to "reverse-engineer" the appropriate values for $\boldsymbol{W}$ and $\boldsymbol{b}$. During forward pass, the values of $\boldsymbol{A}$ are updated with the new $\boldsymbol{W}$ and $\boldsymbol{b}$ using equations 6.1 and 6.2, and our hypothesis is that in this way the network will converge to the "optimal model". In some sense, the LCN reverses the role of $\boldsymbol{A}$ and $\boldsymbol{\theta}$, and regards $\boldsymbol{A}$ being the ones that are initialised at the beginning of training (which is equivalent to initialising the $\boldsymbol{W}$ and $\boldsymbol{b}$), and training should in theory start with backward pass rather than the forward pass.

However, so far we still use conventional optimizers such as Adam, mainly due to limitations in programming skills to build robust customised optimizer for LCN. As will be shown in the later sections, optimizing LCN with Adam works very well, which validates our hypothesis that training starting from backward pass or forward pass are computationally equivalent, but may have different interpretations depending on the perspectives to view the network.

The name "Layer-Wise convex network" comes from a related legacy hypothesis which states that for a network $\hat{y}(\boldsymbol{\theta})$ fulfilling conditions in theorem 1, any convex loss $E(\hat{y}(\boldsymbol{\theta}), y)$ with respect to $\hat{y}$ is also convex with respect to the hidden layer parameters, provided all parameters of the other layers are fixed. This is later proven not true for networks allowing negative activations, such as leaky ReLU nets, and we would not want to restrict the network to have only positive activations, thus we proposed the current version of the LCN theorem. It is easy to see that minimising the Euclidean distance is equivalent to minimising the mean squared error (MSE), which is not only convex but quadratic, thus a more accurate name should be "Over-Determined Layer-Wise Quadratic Networks (OLQN)", but it is a mouthful and does not have a nice ring like "Layer-Wise Convex Network (LCN)", thus we continue naming it LCN theorem.

6.3.4 The Timescale Hyperparameter for Periodic Sequential Inputs

In our pilot experiments, we found that for signals with clear periodicity, informing the model with the timescale of the period can be very helpful. The estimation of the period need not be precise. For example, in the ECG data, we only need to let the model know that the input data period is in the order of seconds, so the model is designed to create a prediction roughly every second. As one can see, the timescale information given to the model is very rough, as the average heart rate is 70-100 beats per minute, meaning the period is, in fact, less than 1s, and irregular heartbeats may be even more off the 1s estimation. Still, as will be demonstrated shortly, the model can learn well with this simple information.

We call this rough estimation of the input data period the *timescale* hyperparameter and denote it as τ. The number of max-pooling layers is determined by the timescale hyperparameter τ, sampling frequency f_s, and pooling size p according to the equation 6.48. For example, if the input is 500Hz ECG time-series, and we set the timescale $\tau = 1s$, and use default $p = 2$, then we can calculate the number of max-pooling layers to be $\lceil 1s \log_2(500Hz) \rceil = 9$.

$$n_{maxpool} = \lceil \log_p(f_s \tau) \rceil \tag{6.48}$$

If the input signal is not apparently periodic, then one only needs to set $f_s\tau = D$, i.e., assume the entire input time-series represents one period, and the model will output only one prediction for the entire signal.

6.3.5 Building Layer-Wise Convex Networks: a Worked Example

Let us look at a concrete example of applying LCN theorem to design model architecture for the CKB dataset. The CKB problem can be cast into a four-class classification problem. The balanced dataset has 7,472 examples. If we separate it into training, validation, and test sets at 8.1:0.9:1 ratio, then we have 6,056 training examples, 672 validation examples, and 744 test examples. Each training example is 12-lead, 10s, 500Hz ECG time-series, which means the input dimension D of each training example is $5,000 \times 12 = 60,000$. According to the LCN theorem, the number of parameters per layer should not exceed 6,065. Because $D > m$, if we use a feed-forward network, the first layer will have at least D parameters, thus we must use weight-sharing mechanisms, and CNN is a natural choice.

Because we are analysing time-series data, 1-D CNN is a natural choice. In 1-D CNN, one of n_w and n_h equals 1, and n_c equals the number of input channels. In this thesis we use the convention $n_h = 1$, f_h is also constrained to be 1. We use the letter k to denote f_w.

To simplify the design process, we use repeating structures and make sure all layers have the same output shape until the output layer. The repeating structure

not only reduces the number of hyperparameters, but also is the least susceptible to vanishing and exploding gradient problems (Hanin 2018). It is also easy to see that between the last convolutional layer and the output layer we should not add fully connected layers, because in order not to exceed the upper bound, the dimension of densely-connected layers has to be very small, which means that it will become "bottlenecks" of the flow of information. Therefore we should only use convolutional, pooling (for dimension reduction because of $5,000 \times 12 \times 4 + 4 > 6,056$), and softmax output layers. If we use CNN layers with kernel size k, stride s, padding p, and the number of filters n_f, the output shape of such convolutional layer is $(\lfloor \frac{input\ \ dimension - k + 2p + 1}{s} \rfloor, n_f)$, and the number of parameters of this convolutional layer is $n_f(kn_f + 1)$ (assuming we are stacking several convolutional layers together). Since stride $s > 1$ will result in dimension reduction, and empirically, it is not performing as well as max-pooling, we keep $s = 1$. To keep output shape identical to the input shape, we use "same" padding, then we calculate k and n_f by equation 6.50.

$$k = n_f = \operatorname{argmax} n_f(n_f^2 + 1) \tag{6.49}$$

subject to

$$n_f(n_f^2 + 1) \leq m \tag{6.50}$$

We constrain $k = n_f$ to avoid k being unreasonably large for long signals with few channels.

Since the CKB problem is a four-class classification problem, the output layer will be a four-unit softmax layer. Finally, to determine the number of max-pooling layers, we recommend using as small a pooling kernel as possible , so we can build as deep networks as possible. The smallest pooling size is 2. Since the CKB input data contains highly periodic ECG, with the duration of a heartbeat roughly once a second, we therefore set the timescale hyperparameter $\tau = 1s$, and make the model produce one prediction roughly every second. The number of max-pooling layers is thus $\lceil \log_p(f_s\tau) \rceil = \lceil \log_2(500Hz \times 1s) \rceil = 9$.

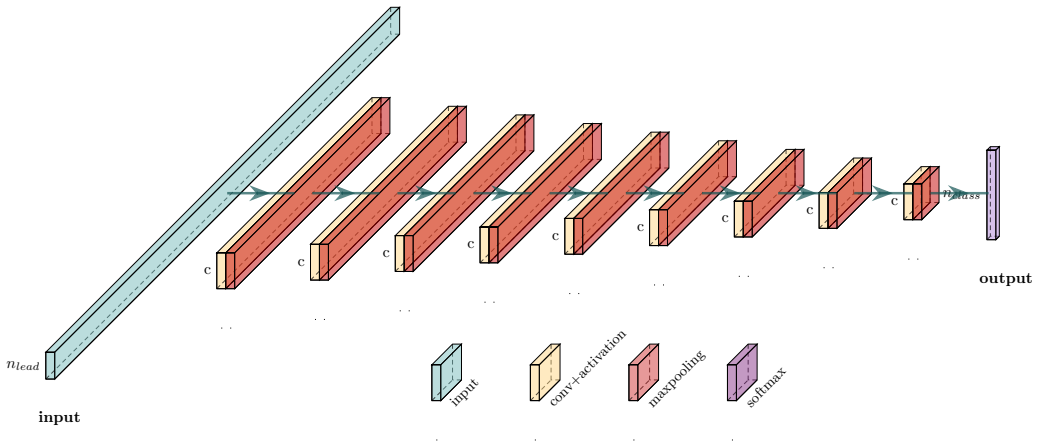


Figure 6.4: Baseline model architecture. The number of max-pooling layers is calculated by equation 6.48. Before each max-pooling layer, the baseline model has one convolutional layer and one activation layer, which can be ReLU or leaky ReLU. When adding skip connections, the post-convolution (before activation) tensor is added to every $n_{maxpool} - 1$ post-convolution tensor (see figures 6.5). When necessary, the batch normalisation layers are added after the input layer, and after every activation layer.

Now we have the baseline model (figure 6.4), and to improve results, we only need to stack convolutional layers between max-pooling layers. The number of convolutional layers stacked between 2 maxpooling layers is a hyperparameter called n_{repeat}. Unfortunately, there are no guidelines to calculate the optimal depth, but the principle is that adding layers should not harm performance, although the training may become more difficult. In the next section, we introduce a heuristic algorithm that is inspired by the principle of reinforcement learning, called the AutoNet.

6.4 The AutoNet Algorithm

The AutoNet algorithm is designed to generate a Layer-Wise convex network given a dataset automatically. The algorithm is outlined in algorithms 1 and 2 and described as follows:

6.4.1 Step One: Generate the Baseline Model

The LCN model for ECG classification has only five hyperparameters: $n_{repeat} \in \mathbb{N}$, $n_{maxpool} \in \mathbb{N}$, $n_f \in \mathbb{N}$, $skip \in \mathbb{B}$ (Boolean domain), and $bn \in \mathbb{B}$, which can all

Algorithm 1: Build LCN. See Figure 6.5 for the positions of convolutional, activation, batch normalisation, and maxpooling layers.

Input: m, $n_{channel}$, n_{class}, n_{repeat}, *skip*, *bn*, $n_{maxpool}$
Output: model

$n_f = \text{argmax}_{n_f}\, n_f(n_f^2+1)$ subject to $n_f(n_f^2+1) \leq m$
add the input layer
if *bn* **then**
  add a batch normalisation layer
end
add a convolutional layer, kernel size = n_f, $n_{filters} = n_f$
if *bn* **then**
  add a batch normalisation layer
end
add a maxpooling layer, pooling size= 2
for _ *in range* $n_{maxpool}$*-1* **do**
  for _ *in range* n_{repeat} **do**
    add a convolutional layer, kernel size = n_f, $n_{filter} = n_f$
    if *skip* **then**
      connect the before-activation output of every $n_{maxpool} - 1$ convolutional layers by addition
    end
    add an activation (ReLU or leaky ReLU) layer
    if *bn* **then**
      add a batchnorm layer
    end
  end
  add a maxpooling layer
end
add a time distributed softmax layer

be determined from the training set or by the AutoNet algorithm. n_f is the number of filters of each convolutional layer, calculated according to equations 6.49 and 6.50. The number of max-pooling is determined according to equation 6.48. The output layer is a time-distributed softmax layer for classification and classifies the entire signal by majority voting. *skip* and *bn* are the "switches" representing whether the network adds skip connections and batch normalisation, respectively, and are determined by the AutoNet algorithm according to the steps outlined in the next section:

Algorithm 2: Grow the model using AutoNet. This algorithm calls algorithm 1 to build each LCN, then train the model until early stopping criteria is met. It tracks the minimum training loss and the minimum validation loss during training and compare them against the policy.

Input: m, $n_{channel}$, n_{class}, n_{repeat}, $skip$, bn, $n_{maxpool}$, $\boldsymbol{X}$, $\boldsymbol{Y}$, model_averaging, $fold = 10$
Output: best model

batch size = 32, patience = 8, bn = False, $skip$ = False
build baseline model using algorithm 1
train model
while *train loss or validation loss declines* **do**
    n_{repeat}+=1
    build LCN suing algorithm 1 train the resulting model
end
$skip$ =True
while *min train loss or min validation loss declines* **do**
    $n_{repeat} += 1$
    build LCN using algorithm 1 train the resulting model
end
bn =True
while *min train loss or min validation loss declines* **do**
    $n_{repeat} += 1$
    build LCN using algorithm 1 train the resulting model
end
best_model = model with min validation loss
if *model_average* **then**
    train the best model $fold$ times
    use mean class probability of the $fold$ models to predict
end

6.4.2 Step Two: Grow the Model

- Start with the baseline model, without batch normalisation, nor skip connection, i.e. $bn = FALSE, skip = FALSE$. Batch size = 32. The early stopping criterion is no reduction in validation loss for eight epochs. In the baseline model, $n_{repeat} = 1$.

- Increase n_{repeat} by one each time, until *neither the training loss nor the validation loss* decreases, then turn on skip connection and connect every $n_{maxpool} - 1$ layer by adding the post-convolution-before-activation tensors with the output tensor of $n_{maxpool} - 1$ convolutional layers later (figure 6.5).

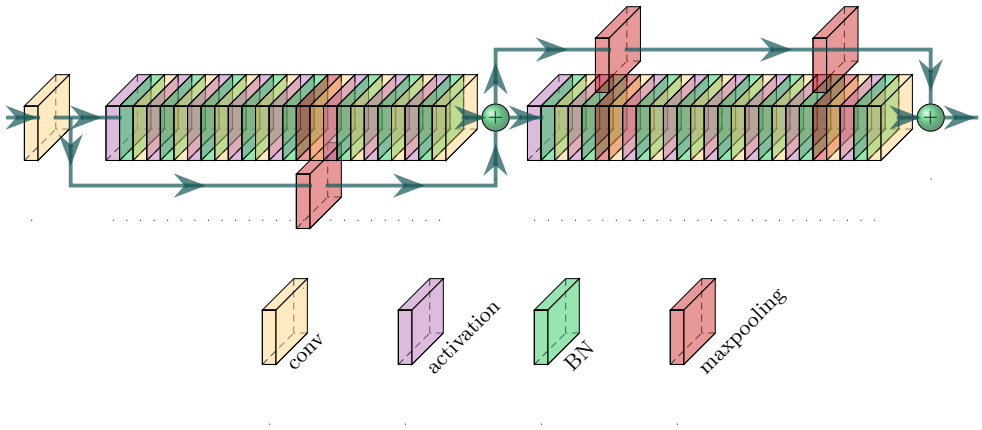


Figure 6.5: The positions of convolutional, activation, batch normalisation, max-pooling layers, and the skip connection. The illustrated network has convolution-activation-BN repeating structure, with $n_{maxpool} = 9$, $n_{repeat} = 5$. A max-pooling layer is added after every n_{repeat} (5 in this example) batch normalisation layers. The element-wise addition is applied to the output tensor of every $n_{maxpool} - 1$ (8 in this example) convolutional layers. For example, the output tensor of the first convolutional layer is element-wisely added to the output tensor of the 9th convolutional layer, and the resulting tensor is the input to the following activation layer and is also used in the element-wise addition with the output tensor of the 17th convolutional layer.

- Increase n_{repeat} by one each time, until *neither the training loss nor the validation loss* decreases, then add batch normalisation after every activation and after the input layer.

- Increase n_{repeat} by one each time until *neither the training loss nor the validation loss* decreases. The model which yielded minimum validation loss is selected to be the "best" model.

6.4.3 Step Three: Model Averaging

Train the identified "best" model K times. At test time, calculate the average probability predictions provided by the K models, then classify the test case to the class with the highest mean probability, i.e.

$$\hat{i} = \text{argmax} \frac{1}{K} \sum_{j=1}^{K} p_{ij} \tag{6.51}$$

where p_{ij} is the class i probability predicted by the jth model, this step can be omitted if one is not reporting the final results and only wishes to prototype quickly.

6.5 Benchmark with the State-of-the-Art Model

In the following sections of this chapter, we benchmark LCNs generated by the AutoNet algorithm with the ResNet-based Hannun-Rajpurkar model (Hannun et al. 2019; Rajpurkar et al. 2017) which has been demonstrated to exceed average cardiologist performance in classifying 12 rhythm classes on 91,232 recordings from 53,549 patients and is well regarded as the state-of-the-art end-to-end deep learning model for ECG classification. We compare the results of Hannun-Rajpurkar model and the LCN models on three datasets: ICBEB, PhysioNet, and CKB, the description of which are given in Chapter 4.

6.5.1 Computational Environment

All experiments were done on Ubuntu 18.04, CPU with 32G RAM, single Nvidia GeForce GTX 1080 GPU, with Python version 2.7.15, and Tensorflow version 1.8.0.

6.5.2 Two LCN Variants

The LCN theory is derived from the assumption that the activation functions are strictly monotonic. While we hypothesise that the LCN theory can be extended to non-strict monotonic activation functions such as ReLU, the strictness of monotonicity may make a difference. Thus we study two variants of LCN: ReLU-LCN and Leaky-LCN. As the names suggest, the hidden layer activations of ReLU-LCN are all ReLU, while the hidden layer activations of Leaky-LCN are all leaky ReLU with $\alpha = 0.3$ (equation 6.52). It is easy to see that the selected leaky ReLU function is strictly monotonically increasing, while the ReLU function is non-decreasing but not strictly monotonic.

$$y = \begin{cases} x & \text{if } x > 0 \\ \alpha x & \text{if } x \leq 0 \end{cases} \tag{6.52}$$

6.5.3 Model Training

All LCN models were trained using Adam with default hyperparameters ($\beta_1 = 0.9, \beta_2 = 0.999$) and the default learning rate (0.001). The Hannun-Rajpurkar model, as a bench-marking approach, was trained using the authors' original implementation (`https://github.com/awni/ecg`) to ensure identical implementation. In brief, Hannun-Rajpurkar model used Adam with learning rate scheduler that decreases learning rate after no improvement on the validation loss for two epochs. All hyperparameters were kept the same as in their codes and as described in Hannun et al. 2019.

All models were trained using early stopping with patience 8 epochs, for a maximum of 100 epochs, which is the same as in Hannun's codes and in Hannun et al. 2019.

6.5.4 Power Analysis

To detect statistical significance, a power analysis was conducted for the two-tail paired, t-test at effect size 0.8, $\alpha = 0.05$, power = 0.8, and the required sample size was found to be 14.30. Therefore we conducted five repeats for each of the ICBEB, PhysioNet, and CKB experiments, producing a total of 15 experiments. In each repeat, all models were trained and tested on the same training, validation, and test sets. Note that the paired t-test only assumes the differences of the means, rather than the samples themselves, follow a Gaussian distribution, and does not assume equal variance of the samples (Jaynes 2003). Therefore the 15 experiments created by five repeats on three different datasets are appropriate for the two-tail paired t-test, if the differences of the means pass normality tests.

6.5.5 ICBEB

Train-Validation-Test Split

We did not have access to the hidden test set, therefore we randomly took 50 samples from each class from the publicly available training set ($n = 6,877$) to build a balanced test set ($n = 450$) of the same size and class distribution as the

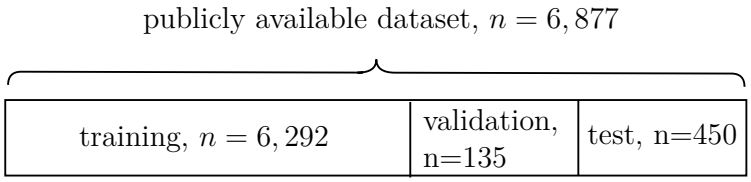


Figure 6.6: Train-validation-test split for ICBEB

ICBEB Challenge, another 15 samples from each of the 9 classes to form a balanced validation set of $15 \times 9 = 135$ samples for early stopping and check-pointing, and the rest is the training set for gradient calculation (figure 6.6).

The above approach was repeated five times to generate five repeats of the experiments. In each repeat, all models share the same training, validation, and test sets.

Sample Weighting

The samples in the training set (not including the validation samples) were weighted by the inverse of their class ratio in the training set. For example, if there are n_i class i samples in the training set, then each class i sample receives $\frac{\sum_i n_i}{n_i}$ weight during training.

Signal Padding

Since the pooling size is fixed in both LCN models and the Hannun-Rajpurkar model during training, the model requires the input signal to have the same length. Ideally, the target length should be the maximum signal length in the training set, i.e. 61s. However, due to memory constraints, we could only feed in 37s signals. Thus the target length for ICBEB is 37s. If the original signal was shorter than the target length, 0s are padded to the end of the signal; if the signal is longer than the target length, the end of the signal was truncated. At test time, no padding is needed as the model generates a label every 512 time steps (1.024s).

Model Generation

In each repeat, the AutoNet algorithm identified the "best" ReLU-LCN model and the "best" Leaky-LCN model separately. The hyperparameter n_f is calculated

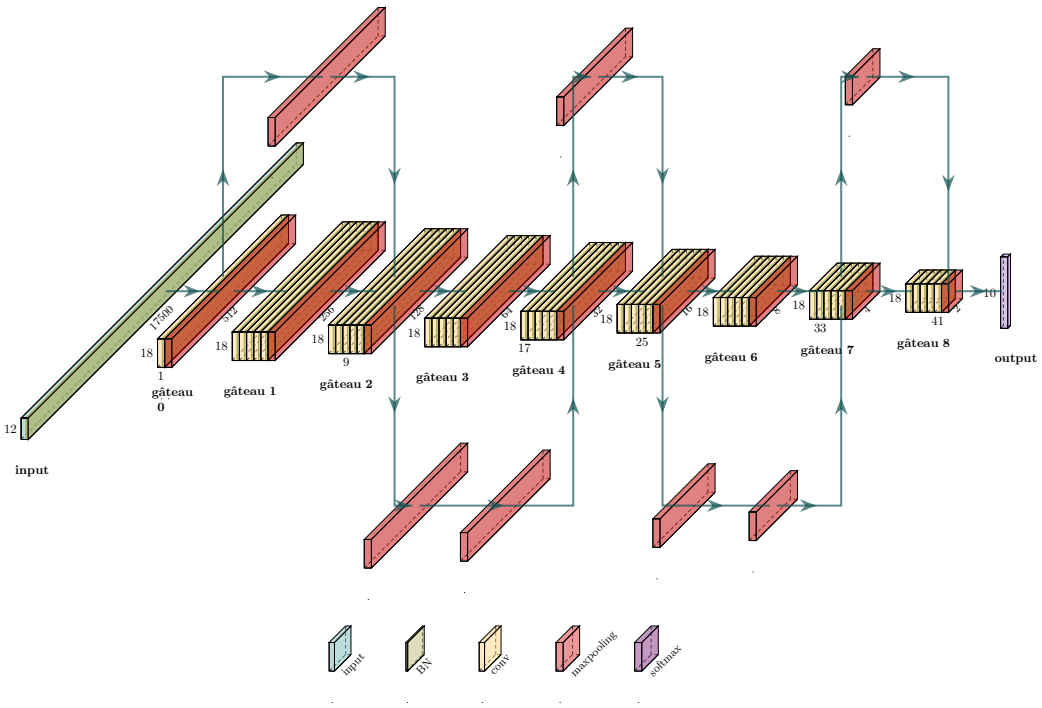


Figure 6.7: The automatically generated ReLU-LCN architecture for ICBEB. n_{repeat} = 5, $n_{maxpool} = 9$, meaning there are a total of 9 max-pooling layers, and there are five convolutional layers stacked between every two max-pooling layers. The activation can be ReLU or leaky ReLU, which follows every convolutional layer, not shown in the Figure to declutter the diagram. Batch normalisation (green) is added after the input layer (blue) and after each convolutional layer (yellow). The after-convolution tensor is added to every 8 subsequent after-convolutional tensors, which are labelled in the figure. See Figure 6.5 for magnified connection structure. The output layer is a time-distributed 10-unit softmax layer, one unit for each of the nine classes and one unit to indicate noise/zero paddings.

according to equations 6.49 and 6.50 with $m = 6,292$, thus $n_f = 20$. $n_{maxpool}$ is calculated according to equation 6.48 with $f_s = 500Hz$, $\tau = 1s$, $p = 2$, to be 9.

It took AutoNet 1h 25min (5,095s) on average to identify the best ReLU-LCN model and 1h 55min (6,936s) to identify the best Leaky-LCN model. For ReLU-LCN, three out of five repeats converged at $n_{repeat} = 5$ with both skip connection and batch normalisation (figure 6.7), one experiment converged at $n_{repeat} = 6$, with both skip connections and batch normalisation, one experiment converged at $n_{repeat} = 4$, with both skip connection and batch normalisation (Table B.1); for Leaky-LCN, four out of five repeats converged at $n_{repeat} = 5$, with both skip connections and batch normalisation, while the other repeat converged at $n_{repeat} = 7$, with both

Table 6.1: The architecture and training characteristics of ReLU-LCN, Leaky-LCN, and the Hannun-Rajpurkar models on ICBEB. conv: convolutional layer; BN: batch normalisation; TDS: time distributed softmax.

	ReLU-LCN	**Leaky-LCN**	**Hannun-Rajpurkar**
Train size	6,427	6,427	6,427
Test size	450	450	450
Batch size	32	32	32
Signal padding (s)	35	35	35
N parametric layers	84 (41 conv, 42 BN, 1 TDS)	84 (41 conv, 42 BN, 1 TDS)	67 (33 conv, 33 BN, 1 TDS)
N parameters (%)*	239,596 (2.3)	239,596 (2.3)	10,473,322(100)
Speed (s/epoch)	36	41	91
Total epoch	27	30	21
Runtime (s, %)*	955 (50.0)	1,248 (65.3)	1,911 (100)

* % relative to the Hannun-Rajpurkar model.

Table 6.2: Mean and standard deviation of the test F_1 on five experiments by ReLU-LCN, Leaky-LCN, and Hannun-Rajpurkar models on PhysioNet. The highest F_1 of each category is in bold font. No model averaging was performed.

	Training size	**ReLU-LCN**	**Leaky-LCN**	**Hannun-Rajpurkar**
N	868	64.1±3.8	64.8±6.0	**69.8±4.4**
AF	1,048	84.2±3.3	**85.4±1.4**	84.7±3.7
I-AVB	654	84.2±1.9	85.2±3.1	**86.0±3.7**
LBBB	1,57	**89.1±1.7**	88.7±2.4	88.0±2.0
RBBB	1,645	76.5±3.4	**78.4±4.6**	76.0±4.1
PAC	506	64.8±12.6	**67.5±4.3**	61.4±9.7
PVC	622	81.4±4.7	**83.1±2.7**	80.1±5.6
STD	775	68.1±6.9	76.2±5.1	**78.9±4.7**
STE	152	68.1±3.9	**69.2±2.8**	58.3±7.7
9-class F_1		75.6±3.6	**77.6±2.0**	75.9±2.9
F_{AF}		84.2±3.3	**85.4±1.4**	84.7±3.7
F_{Block}		83.3±2.1	**84.1±2.1**	83.0±2.3
F_{PC}		72.0±9.3	**75.0±3.1**	70.7±7.1
F_{ST}		68.1±4.5	**72.5±3.0**	69.9±4.0

skip connection and batch normalisation.

Results

The model architecture and training characteristics of ReLU-LCN, Leaky-LCN, and the Hannun-Rajpurkar model are shown in Table 6.1. The number of parametric layers are taken from the most frequently found architecture among the 5 experiments, and the speed (s/epoch) and total epochs are the average value over the five experiments. The runtime is calculated by equation 6.53. The identified "best" architectures were identical for ReLU-LCN and Leaky-LCN, both have only 2.3% parameters compared to the Hannun-Rajpurkar model. Both ReLU-LCN and Leaky-LCN converged at deeper architectures than Hannun-Rajpurkar model, which agrees with our hypothesis that the parsimony of LCN encourages the model to grow deeper.

$$runtime = \frac{1}{5}\sum_{i=1}^{5} total\ epoch \times speed \tag{6.53}$$

Both LCN models computed each epoch faster than Hannun-Rajpurkar model, although the latter converged in fewer epochs (table 6.1). Both LCN models need much less average runtime than the Hannun-Rajpurkar model. The training speed not only depends on the architecture but also on the input signal length and the batch size (the longer the signal, the smaller the batch size, the slower it is to train). Thus the runtime comparison between the LCN models and the Hannun-Rajpurkar model is less dramatic than the parameter comparison. On average, Leaky-LCN needed more runtime as it tended to find deeper models than ReLU-LCN (Table B.1).

Table 6.2 shows the test F_1 of the three models. We can see that Leaky-LCN has the highest mean in most cases, while ReLU-LCN is comparable to Hannun-Rajpurkar in most cases. For sub-abnormal groups and the 9-class F_1, which the Challenge used as the evaluation criteria, Leaky-LCN performed universally better than the other two models. Surprisingly, all three models performed best in the LBBB class, despite that LBBB is the second smallest class in the training set. It may be explained by the fact that LBBB has clear clinical ECG diagnosis criterion (Chapter 4). The model performances did not seem to correlate highly with the training size: STE has the similar number of training examples as LBBB but is

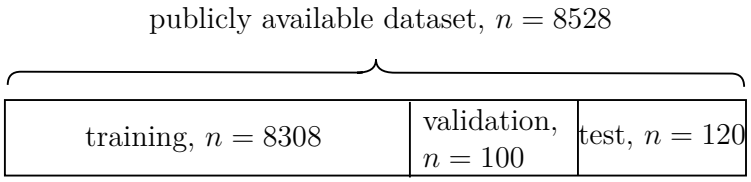


Figure 6.8: Train-validation-test split for PhysioNet

poorly classified. It suggests certain medical conditions are inherently difficult for CNN based architectures to classify from ECG, which agrees with the clinical knowledge that some conditions do not have definite ECG characteristics.

To compare with the performance of the winning team, we took the ReLU-LCN model found in the first experiment and preformed 10-fold model averaging. Our model obtained 0.854 9-class F_1 which outperformed the winning team ($F_1 = 0.837$). We chose to average ReLU-LCN model instead of the Leaky-LCN model because there is no statistical difference between the F_1 scores of the two models, but the latter has significantly higher runtime cost (see 6.5.9 for more details).

Note that these results were higher than the winning team despite being trained on fewer data. The winning team by Chen et al. used 6,877 training examples, also tested on 450 test cases (exclusive from the 6,877 training cases), and padded the signals to 144s, while ReLU-LCN was trained on 6,427 recordings, and the signals are padded to only 35s. Although the winning team's exact architecture is unknown, their model is based on bidirectional GRU (a type of RNN), which is known to be slow to train; their input signal length is about 4 times of the input to the ReLU-LCN; and they needed to average over 130 models, while ReLU-LCN only needed to average over 10 models to obtain the above results. These all suggest that Chen et al.'s model is likely to have a higher runtime cost.

6.5.6 PhysioNet

Train-validation-test Split

We randomly selected 30 samples (roughly 10% of the smallest class) from each class to build a balanced test set ($n = 120$), and another 25 samples (roughly 9% of the smallest class) from each class to build a balanced validation set, and the rest

of the dataset is the training set. The train-validation-test split process is shown in Figure 6.8. The above approach was repeated five times to generate five sets of training, validation, and test sets, and shared among all models in each repeat.

Sample Weighting

The samples were weighted using the same procedure as described in section 6.5.5.

Signal Padding

All signals were padded to the maximum length in the training set, i.e. 61s similarly as described in section 6.5.5.

Model Generation

In each repeat, the AutoNet algorithm identified the "best" ReLU-LCN model and the "best" Leaky-LCN model separately. The hyperparameter n_f is calculated according to equations 6.49 and 6.50 with $m = 8,308$, thus $n_f = 20$. $n_{maxpool}$ is calculated according to equation 6.48 with $f_s = 300Hz$, $\tau = 1s$, $p = 2$, to be 8. It took AutoNet 52 min (3,203s) on average to identify the best ReLU-LCN model and 1h 30min (5,413s) to identify the best Leaky-LCN model.

For ReLU-LCN, 2 out of 5 repeats converged at $n_{repeat} = 2$ without skip connection nor batch normalisation (table B.2); 1 experiment converged at $n_{repeat} = 2$, with only skip connection and without batch normalisation; 1 experiment converged at $n_{repeat} = 3$, with both skip connections and batch normalisation; and the other repeat converged at $n_{repeat} = 4$ with only skip connection and without batch normalisation. For Leaky-LCN, 4 out 5 repeats converged at $n_{repeat} = 4$, with both skip connections and batch normalisation (figure 6.9), and the other repeat converged at $n_{repeat} = 5$, with only skip connection and without batch normalisation.

Results

The model architecture and training characteristics of the three models are shown in Table 6.3. The LCN models have no more than 2.2% of the parameters than those of the Hannun-Rajpurkar model. The same conclusions regarding runtime, total

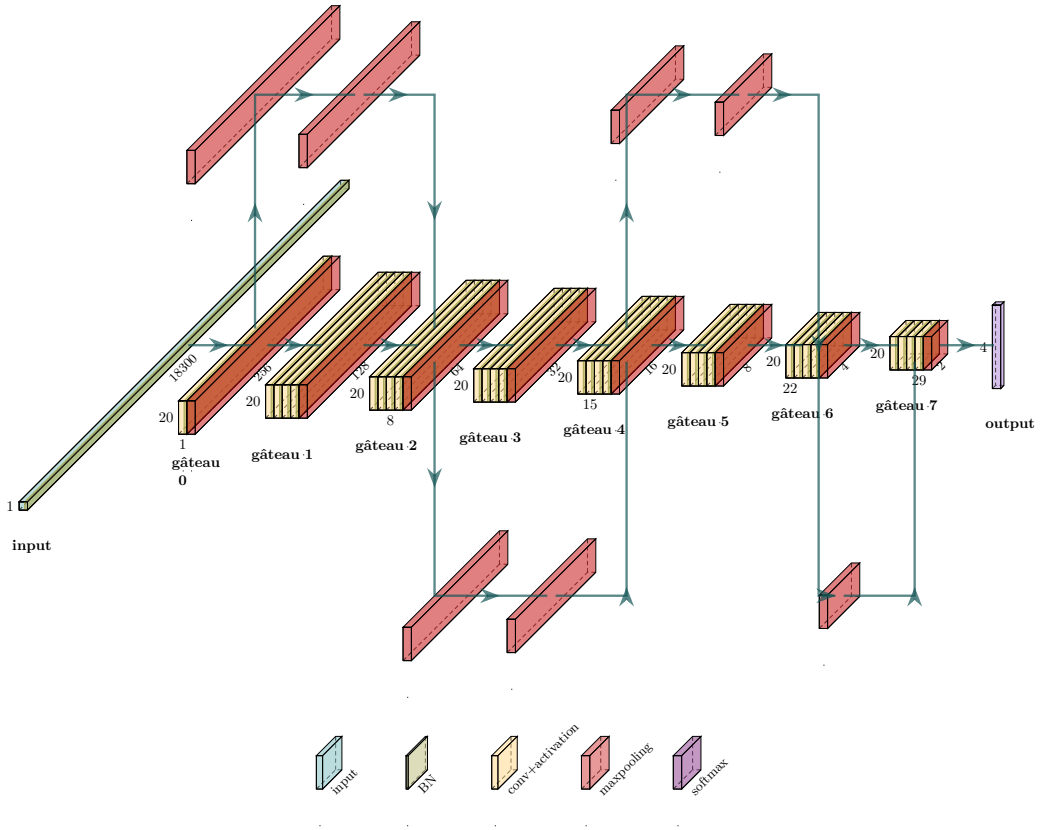


Figure 6.9: The most commonly found Leaky-LCN architecture for PhysioNet. $n_{repeat} = 4$, $n_{maxpool} = 8$, $c = k = 20$. A batch normalisation layer (green) is added after the input layer (green) and after every convolutional layer (yellow). The output is a 4-unit time distributed softmax layer (purple). The network provides one prediction roughly every second (256-time steps, 300Hz). The after-convolution tensor is added to every 7 subsequent after-convolution tensors.

epochs, and training speed as in ICBEB hold in PhysioNet experiments, suggesting the LCNs behave consistently on different datasets.

Table 6.4 shows the test F_1 of the three models. We can see ReLU-LCN is better at identifying atrial fibrillation and noise, while the Leaky-LCN model gave the best normal and "other rhythms" classification among the three models. Similarly, all three models are not biased towards large classes, suggesting the sample weighting mechanism is effective. The three-class average F_1 (F_{13}) is lower than what is reported in Chapter 4, which is likely because in this thesis we only trained the models on part of the training data and tested on a balanced test

Table 6.3: The architecture and training characteristics of ReLU-LCN, Leaky-LCN, and the Hannun-Rajpurkar model on PhysioNet. conv: convolutional layer; BN: batch normalisation; TDS: time distributed softmax.

	ReLU-LCN	**Leaky-LCN**	**Hannun-Rajpurkar**
Training size	8,308	8,308	8,308
Test size	120	120	120
Batch size	32	32	32
Signal padding (s)	61	61	61
N parametric layers	16 (15 Conv, 1 TDS)	60 (29 conv, 30 BN, 1 TDS)	67 (33 conv, 33 BN, 1 TDS)
N parameters (%)*	112,784 (1.1)	226,226 (2.2)	104,661,48 (100)
Training speed (s/epoch)	20.6	43.2	121
Total epoch	30	28	21
Runtime (s,%)	611 (23.6)	1,207 (46.6)	2,589 (100)

* % relative to the Hannun-Rajpurkar model.

Table 6.4: The mean and standard deviation of the test F_1 in five experiments by ReLU-LCN, Leaky-LCN, and Hannun-Rajpurkar models on PhysioNet. The highest F_1 of each category is in bold font. No model averaging was performed.

	Training size	**ReLU-LCN**	**Leaky-LCN**	**Hannun-Rajpurkar**
AF	708	**88.8±2.8**	80.4±2.3	87.9±4.2
Normal	5,020	80.3±3.6	**86.4±4.3**	77.0±2.0
Other rhythms	2,426	72.3±7.7	**79.5±3.7**	74.6±3.8
Noise	254	**87.9±4.3**	72.4±4.6	74.7±6.1
F_{14}		82.3±3.1	**83.3±5.2**	78.5±3.3
F_{13}		**80.5±3.6**	79.5±1.5	79.8±2.6

set, while the cited studies were trained using all publicly available training data and tested on an imbalanced test set.

6.5.7 CKB

Train-Validation-Test Split

Due to memory constraints, we could not train on all the recordings. Therefore we constructed the largest balanced set of normal, "arrhythmia", "ischaemia", and "hypertrophy" classes by randomly sampling 1,868 (the size of the smallest class) recordings from each of the four classes. The resulting set is then stratified at

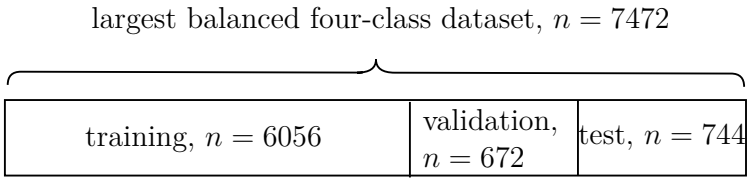


Figure 6.10: Train-validation-test split for CKB.

8.1:0.9:1 ratio into training, validation, and test sets, respectively (figure 6.10). The sampling and split is repeated five times to generate five sets of the training, validation, and test sets for five repeats of the experiment. In each repeat, the training, validation, and test sets are shared among all models.

Sample Weighting

Since all classes are balanced in the training set in the CKB experiments, there is no need for sample weighting.

Signal Padding

All signals in CKB have the same duration (10s, 500Hz), thus there is no need for signal padding.

Model Generation

In each repeat, the AutoNet algorithm identifies the "best" ReLU-LCN model and the "best" Leaky-LCN model separately. The hyperparameter n_f is calculated according to equations 6.49 and 6.50 with $m = 6,056$, thus $n_f = 18$. $n_{maxpool}$ is calculated according to equation 6.48 with $f_s = 500Hz$, $\tau = 1s$, $p = 2$, to be 9.

It took AutoNet 7 min (427s) on average to identify the best ReLU-LCN model and 11 min (693s) to identify the best Leaky-LCN model. For ReLU-LCN, all five repeats converged at $n_{repeat} = 1$ without skip connection nor batch normalisation (figure 6.11); for Leaky-LCN, three out of five repeats converged at $n_{repeat} = 1$, without skip connection nor batch normalisation, while the other 2 repeats converged at $n_{repeat} = 2$, with only skip connection and without batch normalisation (Table B.3).

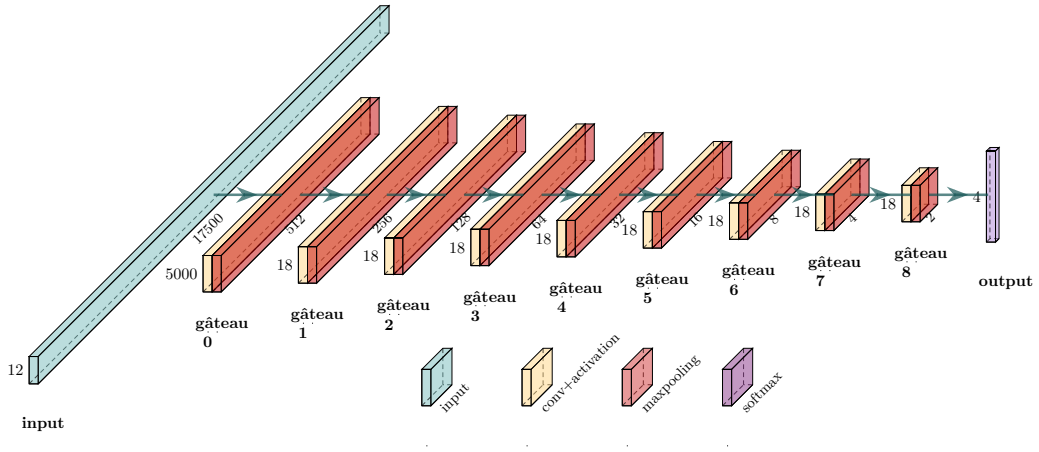


Figure 6.11: The automatically generated model architecture for CKB. $n_{repeat} = 3$, $n_{maxpool} = 9$, $n_f = k = 18$. No batch normalisation nor skip connection was needed. The output (purple) is a 4-unit time distributed softmax layer. The model provides one prediction roughly every second (512-time steps, 500Hz).

Table 6.5: The architecture and training characteristics of ReLU-LCN, Leaky-LCN, and the Hannun-Rajpurkar model on CKB. The architecture and training characteristics of ReLU-LCN, Leaky-LCN, and the Hannun-Rajpurkar model on CKB. conv: convolutional layer; BN: batch normalisation; TDS: time distributed softmax.

	ReLU-LCN	**Leaky-LCN**	**Hannun-Rajpurkar**
Training size	6,728	6,728	6,728
Test size	744	744	744
Batch size	32	32	32
Signal padding (s)	10	10	10
N parametric layers	10 (9 conv, 1 TDS)	10 (9 conv, 1 TDS)	67 (33 conv, 33BN, 1 TDS)
N parameters (%)*	50,782 (0.5)	50,7872 (0.5)	10,471,780 (100)
Speed (s/epoch)	4	5	34
Total epoch	24	20	13
Runtime (s, %)*	95 (21.5)	97 (22.0)	442 (100)

* % relative to the Hannun-Rajpurkar model.

Results

The model architecture and training characteristics of the three models are shown in the Table 6.5. Both LCN models converged at nine convolutional layers without the need for batch normalisation, with only 0.5% parameters and needed fives times less runtime as the Hannun-Rajpurkar model.

Table 6.6: Mean and standard deviation of the F_1 on five experiments by ReLU-LCN, Leaky-LCN, and Hannun-Rajpurkar models on CKB. The highest F_1 of each category is in bold font. No model averaging was performed. A: "arrhythmia", H: "hypertrophy", I: "ischaemia", N: normal.

	Training size	**ReLU-LCN**	**Leaky-LCN**	**Hannun-Rajpurkar**
A	1,681	**74.0±1.4**	71.7±3.7	63.7±10.1
H	1,681	**85.2±1.5**	82.5±1.0	75.2±16.8
I	1,681	72.4±2.6	**73.2±2.0**	66.9±2.2
N	1,681	**77.2±2.9**	75.6±2.7	69.5±3.3
4-class F_1		**77.2±1.6**	75.8±1.9	68.9±4.6

Table 6.2 shows the test set classification F_1 of the three models. LCN models outperformed the Hannun-Rajpurkar model universally, with 8-16% improvement on performance depending on the category and model. ReLU-LCN performed best in most categories, except "ischaemia", but the difference with Leaky-LCN and ReLU-LCN is insignificant. In this dataset, both training and test sets are balanced, so the difference given by the same model comes solely from the nature of the medical condition. "Arrhythmia" and "ischaemia" were more difficult for all three models, while "hypertrophy" was the easiest. This agrees with the result in ICBEB (section 6.5.5) where LBBB was the best classified.

This is a classic case that a large model, even if well-regularised, may not outperform a smaller model. In fact, as demonstrated in all three datasets in this chapter, the smaller but carefully designed network can perform from slightly better to markedly better than a larger network. Moreover, we have demonstrated that the hyperparameters of such "careful" design of networks can indeed be mathematically derived.

Recall in Chapter 5 the best performing traditional machine learning model - stochastic gradient boosting - yielded 0.773 classification accuracy, using 84 handcrafted features. Of course, SGB needs feature extraction and cannot handle raw ECG inputs, unlike the deep learning model. SGB is also a boosting method, which is more comparable to LCN with model averaging. We conducted a 10-fold model averaging on the first experiment using the identified "best" ReLU-LCN and obtained 0.812 for both F_1 and accuracy.

Table 6.7: F_1 of 15 experiments using the three models. In each experiment, the training and test sets are shared among all models. In PhysioNet, the shown results are 4-class average F_1. The highest F_1 of each experiment is shown in bold font.

Dataset	Experiment	ReLU-LCN	Leaky-LCN	Hannun-Rajpurkar
ICBEB	1	**81.8**	81.5	77.5
	2	70.7	**76.8**	75.9
	3	76.8	75.6	**79.6**
	4	74.5	**77.1**	70.8
	5	74.3	**77.0**	75.7
PhysioNet	1	**82.5**	78.5	80.9
	2	**87.4**	80.1	73.7
	3	77.8	**81.5**	76.1
	4	82.4	**84.3**	82.9
	5	**81.5**	77.7	79.0
CKB	1	77.2	**78.0**	73.2
	2	**76.4**	75.1	61.7
	3	74.7	**77.6**	72.1
	4	**78.7**	75.5	65.3
	5	**78.9**	72.7	72.0

6.5.8 Statistical Analysis

To test the applicability of a paired t-test on the F_1 of 15 experiments (Table 6.7), we performed Shapiro-Wilk test for normality (Shapiro and Wilk 1965) on the *differences* between the F_1 scores obtained by the Hannun-Rajpurkar model and the ReLU-LCN model on 15 experiments (5 repeats on each of the three datasets), and found p-value $= 0.158 > 0.05$. Similarly, we tested the normality of the differences between Leaky-LCN and Hannun-Rajpurkar and found p-value $= 0.832 > 0.05$. Both passed the normality test [4], meaning both differences do not deviate significantly from a Gaussian distribution, thus appropriate for two-sided paired t-test [5].

We then did pair-wise two-tail paired t-test on the F_1 scores of the three models, and found p-value $= 0.023 < 0.05$ between ReLU-LCN and Hannun-Rajpurkar, and p-value $= 0.012 < 0.05$ between Leaky-LCN and Hannun-Rajpurkar, and p-value

[4] The null hypothesis of Shapiro-Wilk test of normality is that the samples come from a Gaussian distribution, thus p-value > the chosen significance level ($\alpha = 0.05$) fails to reject the null hypothesis, thus passing the Shapiro-Wilk test.

[5] As long as the sample difference does not deviate significantly from a Gaussian, it is appropriate to use paired t-tests (Jaynes 2003)

Table 6.8: The PC ratio, calculated as $\frac{runtime(s)}{F_1} \times 10000$. The higher the value is, the better. The highest value of each experiment is in bold font.

Dataset	Experiment	ReLU-LCN	Leaky-LCN	Hannun-Rajpurkar
ICBEB	1	**7.1**	3.8	5.0
	2	**7.7**	7.4	4.0
	3	8.3	**9.7**	4.2
	4	**8.2**	6.3	3.9
	5	**8.6**	6.2	3.5
PhysioNet	1	**10.6**	8.4	3.6
	2	**9.3**	5.5	3.7
	3	**17.9**	6.2	3.7
	4	**19.8**	6.4	3.5
	5	**16.3**	6.6	1.9
CKB	1	**96.5**	64.2	17.6
	2	**76.4**	70.1	18.4
	3	**69.2**	50.1	16.3
	4	85.5	**90.7**	20.2
	5	82.2	**105.9**	14.3

$= 0.667 > 0.05$ between ReLU-LCN and Leaky-LCN. We conclude that there is a significant difference between ReLU-LCN and Hannun-Rajpurkar models, and between Leaky-ReLU and Hannun-Rajpurkar models, but no significant difference in F_1 scores were found between ReLU-LCN and Leaky-LCN. However, we cannot conclude from the above results that there are significant differences among the three models, as that would require repeated measurement analysis of variance (ANOVA), the assumption of which is that samples, i.e. the 15 F_1 scores, come from a single Gaussian distribution for each model. However, the 15 F_1 scores of each model failed the Shapiro-Wilk test for normality, thus not suitable for ANOVA.

6.5.9 Performance-to-Computational Cost (PC) Ratio

We propose an intuitive metric to evaluate the computational efficiency of deep learning models, called the Performance-to-Computational Cost (PC) ratio, to help with the decision making as to which model to try and how to improve performance from a study design perspective. The PC ratio is defined below:

$$PC\ ratio = K \times \frac{(performance\ metric)^p}{(computational\ cost)^q} \tag{6.54}$$

where K is a scaling constant to scale the PC ratio to a convenient range. The higher the PC ratio, the better. The performance metric and the computational cost can be anything appropriate for the practitioner as long as it is consistent across all models and datasets. p and q are constants reflecting the practitioners' emphasis on performance or computational cost. For example, here, we use $p = q = 1$, representing an equal preference for the performance and the computational cost. Practitioners more concerned with the performance may use $p = 2$, $q = 1$, for example. Using runtime cost (s) as the metric for computational cost, and F_1 as the performance metric, and $K = 10,000$, we can calculate the value for ReLU-LCN, Leaky-LCN, and Hannun-Rajpurkar model as in Table 6.8.

The PC ratio can compare not only different models on the same dataset but also compare different datasets using the same model. Take ReLU-LCN as an example, we can see that the PC ratios of CKB are much higher than the other two datasets, suggesting CKB is relatively easy to achieve good performance with low computational cost, perhaps due to high signal quality and a large number of training examples per class. However, in Table 6.6 the actual F_1 in CKB is no higher than those of the other two datasets (tables 6.2 and 6.4), suggesting improving upon CKB performance from the model perspective is difficult given the current dataset, perhaps due to the short signal duration (10s) compared to ICBEB (35s) and PhysioNet (61s). This gives us insights as to which direction to pursue if we want to improve performance further: to improve the model, or to collect more data from the same study participants, or to recruit more study participants. A high PC ratio, such as in CKB, may suggest the number of training examples is abundant, while a low PC ratio, such as in ICBEB, may suggest the curse of dimensionality, or in other words, the number of training examples per class is insufficient to train a model that can take advantage of the high dimensional feature vector of each training example.

6.6 Discussion and Conclusion

Each dataset has unique difficulties: ICBEB has the most numerous classes and least number of training examples per class; PhysioNet has the highest noise ratio, and has only single lead; CKB has the shortest signal duration. Comparing the test F_1 across three datasets (Table 6.7), it is encouraging to see that the lowest performance was in fact from CKB, as it implies that the bottleneck of performance lies with the amount of information contained in each training example. This suggests that LCN can indeed make the most out of the training set. It is also encouraging to see that LCN can perform well even if there are few training examples per class, which is often the limiting factor for deep learning. Also, the simple sample weighting method effectively addressed the class skewness, and the LCN models have almost no bias towards the large classes. Table 6.7 shows that given the same experiment, it is almost always one of the LCN models that yielded the best performance. Although Hannun-Rajpurkar model seemed to be the least well-performing model in this chapter, we shall not forget that it has been proven to exceed average human cardiologists on 12 rhythm classes of 91,232 recordings from 53,549 participants (Hannun et al. 2019). LCN models outperformed the Hannun-Rajpurkar model slightly in ICBEB and PhysioNet, and markedly in CKB. The results suggest the model complexity of the Hannun-Rajpurkar model may be appropriate for ICBEB and PhysioNet but too high for CKB, which leads us to hypothesise that the model complexity of AutoNet generated LCNs may be very close to the optimal model complexity given the dataset, and their test loss is close to the Bayesian loss. From this perspective, LCN may be used to estimate the real complexity of the problem.

In each train-validation-test split, the training set is different. Thus the AutoNet may converge at different architectures. Leaky-LCN seemed to have higher consistency than ReLU-LCN (tables B.1, B.2, and B.3). One modification to the AutoNet algorithm to encourage model consistency would be to train each architecture more than once and use the mean validation and training losses to decide on the next step, but it will require more computation.

We have proposed the PC ratio as a simple measure of computational efficiency, and we can see that ReLU-LCN has much higher PC ratio than the other two models. Thus we recommend ReLU-LCN. Also, the PC ratio of each dataset may be a measure of the difficulty of the classification task.

The current state-of-the-art neural network development is trial and error. And the randomness inherent in neural network training due to random weight initialisation, stochastic gradient estimation, and other sources of randomness makes model development especially challenging, as we do not know if the change in the performance is due to an intervention (such as adding layers and changing hyperparameters) or due to the randomness in training. Traditionally, researchers would train the model on the same set of hyperparameters for several times before concluding the helpfulness or the harmfulness of an intervention. This is undesirable when the model becomes very large, and training once would take days to months. The AutoNet algorithm addresses this problem in 3 ways:

- It monitors both training and validation losses to decide on the next step.
- It avoided drop out entirely and did not add batch normalisation until the last step when growing the model, as both dropout and batch normalisation add much noise to the training process.
- By starting from a small model and grow the model to be just the right size for the problem, the algorithm avoids wasting computational resource in solving simple problems with huge models.

The earlier version of AutoNet algorithm included dropout, but we found that LCN did not work well with dropout. Restricting the number of parameters per layer is a strong regularisation in itself, and dropout would result in the model not able to utilise the training set information fully. One improvement may be replacing equation 6.50 with $n_f(1-d)[n_f^2(1-d)+1] \leq m$, where $d \in [0,1]$ is the dropout ratio, and the network might be able to learn a more robust model, but the training would be noisier, and we might need to run each step in the model growth

phase more than once and make decisions on the mean training and validation loss, which will significantly increase computational cost of the AutoNet algorithm. The current version of AutoNet-LCN without dropout already performs comparably, if not better, than a large architecture with dropout, thus the potential benefit introduced by dropout may not worth the increased computational cost.

The ease of optimising LCN may suggest the LCN having many nice properties. Compared with traditional networks where the layers are overparameterised and regularised, LCNs may be much easier to train. LCNs as deep as 16 layers can be successfully trained without any skip connections nor batch normalisation. The hidden layers are over-determined and have identical dimensions, which may make the Hessian well-conditioned.

Although the final loss is not guaranteed to be convex with respect to the hidden layer weights if the network is allowed to have negative hidden activations, such as in Leaky-LCN, the LCN hidden layers are effectively over-determined systems of monotonic equations. Over-determined systems of monotonic equations have a unique solution that minimises the Euclidean distance, which is equivalent to minimising the mean squared error (MSE), which is not only convex but quadratic. Theoretically, we should use a loss which has MSE terms from each layer. In this study, we used conventional cross-entropy loss as an approximation, and it has been proven to work very well. Future work will include designing experiments to study the properties of the loss surface of LCN and experiment with alternative loss functions. LCN may also enable optimising the cross-entropy loss and the quadratic loss layer-by-layer in alternating steps using second order methods, such as Newton's method, as it would only require less than $O(m^3)$ complexity, with m being the number of training examples, which can be very desirable for small datasets.

In this study, we used Adam with all default hyperparameters as the optimiser, without even tuning the learning rate. Our view is similar to I. J. Goodfellow, Warde-Farley, et al. 2013: It is better to design architectures to facilitate optimisation, than designing powerful optimisation algorithms. Our principle is to use as many default hyperparameters as possible, including the learning rate, of a robust optimisation

algorithm, such as Adam, and innovate in model architectures so that tuning optimisation hyperparameters is unnecessary.

One of the major contributions of LCN is a novel paradigm to determine the hyperparameters of CNN. Central to the LCN theorem is the choice of n_f and k. In the current version of LCN, the kernel size k is set to be equal to n_f. Theoretically, k should be independently optimised to maximise the total number of parameters in each layer, subject to $n_f(n_f k + 1) \leq m$. However, for long single-lead signals, such as those in PhysioNet, k would end up being unreasonably large (for example $k > 300$). Thus we kept k to be the same as n_f. This also implicitly expresses our view that the parameters in the kernels and the parameters in the channel dimension are not fundamentally different.

The calculated k and n_f are very unconventional choices compared to what is often used in the literature. In CNN literature, k is typically a small odd number, such as 3, 5, 7, and n_f is typically powers of 2, such as 32, 64, 128, 256. There is no particular reason for these choices except that CNN originated from computer vision research, and odd-numbered k may help learn symmetrical features from images. We forsook this convention entirely and have demonstrated that instead of heuristically tuning the numerous CNN hyperparameters or performing hyperparameter search at the price of high computational cost, we can build efficient neural networks by keeping most of the hyperparameters fixed and rationally calculate the rest of them.

The resulting LCN typically has no more than 2% of the parameters compared to the state of the art model, which is very encouraging as this means at least $O(n_{\boldsymbol{\theta}})$ saving in memory and computational complexity. LCN may also make second-order algorithms feasible, as many second-order methods need $O(n_{\boldsymbol{\theta}}^2)$ (conjugate gradient descent, BFGS) or $O(n_{\boldsymbol{\theta}}^3)$ (Newton method) complexity. If we optimise the parameters layer-by-layer, the computational complexity can be further reduced to be less than $O(m^2)$, where m is the number of training examples. The hypothesised Layer-Wise quadratic property suggests the second-order methods such as Newton's method may be very applicable. Future work may include designing experiments to study the behaviour of convex optimisation in LCN networks. The 50-200 times

fewer parameters may enable the algorithm to run on devices where it is otherwise impossible to run deep learning models.

Although LCN approaches machine learning from the deterministic function approximation perspective, the philosophy behind LCN is similar to the Bayesian approach: we should determine the model complexity from the size of the training set rather than the hypothesised complexity of the problem. Traditional CNN design is "neuron-oriented", which means most of the design considerations and innovations (e.g. dropout and batch normalisation) apply to the neurons, while LCN focuses on the parameters, which is also similar to the Bayesian view.

While developing the AutoNet algorithm, We found the following techniques very helpful in boosting the model performance:

- Handle class imbalance by weighting the training samples by the inverse of the class ratio in the training set. The key is to have a balanced validation set for model check-pointing, even if the final test set is not balanced;
- Time-distributed softmax output for periodic time series signals;
- The batch size is also essential, even without batch normalisation. Although a large batch is faster to train, it is also prone to overfitting. Therefore the AutoNet algorithm keeps batch size to be 32 regardless of the training size (as long as the training set has at least 32 samples);
- Model averaging.

One caveat in our study is that all three datasets have a few thousand training examples. Therefore the hyperparameters n_f calculated for different datasets were similar. Whether the AutoNet algorithm and the LCN theorem will have consistent performance on datasets with very different training sizes remains to be validated.

Another limitation is that in this chapter we used only F1 to evaluate the results. Although F1 is a good choice for evaluating machine learning models in classification tasks, in clinical setting, sensitivity and specificity are more important metrics. Future work will include sensitivity and specificity along with F1.

From the theoretical perspective of the neural network width and depth, this chapter illustrates the surprising effect of rational choice of model sizes based on the training set. Although practitioners generally use a small model when the training data is scarce and a large model when the training data is abundant, it is rare for deep learning practitioners to design the model architecture based on the exact number of training examples. We looked at deep learning architecture design from an unusual perspective: function approximation and equation solving. Although this perspective is not entirely new, the conventional approach is to try to reduce the loss rather than creating conditions to "force" the loss surface to be almost convex or even quadratic. We reverse engineered the conditions to make the "optimal solution" easy to be discovered in training, by mathematically determine the architecture hyperparameters based on the characteristics of the training set.

7 The Heart Age

7.1 Introduction

In the previous chapters, we have classified ECG using Mortara labels as the gold standard. However, the Mortara labels differ from human expert generated labels as the Mortara device follows rule-based algorithms, which means in theory, that a neural network can recover such algorithms with arbitrary precision. The lack of human expert labelling is a common problem in machine learning. In this chapter, we examine a novel paradigm in which we use neural networks to predict alternative labels from unstructured data. We define alternative labels as labels that are accurate and easy to acquire in addition to being relevant to clinical problems. The goal is not to predict alternative labels, but to explore the potential knowledge gainable from the learning process. In the present study, the participants' age and blood pressure are appropriate alternative labels. We provide a proof of concept for this approach by predicting the age of participants from their raw ECG waveforms. Our hypothesis is the AutoNet-LCN will under-predict age of healthy participants, but over-predict the age of participants with cardiovascular diseases. It should also provide a test for AutoNet-LCN's performance in regression tasks.

7.2 Training-Validation-Test Split

We used the 10-s 12-lead ECG waveforms from 24,959 participants, as described in Chapter 4. We trained on 90% of the normal participants, and tested on the remaining normal participants, in addition to all participants with any "arrhythmia", "ischaemia", "hypertrophy" or "other" abnormalities. We constructed the "abnormal" class by aggregating all "arrhythmia", "ischaemia", and "hypertrophy" classes, and used it as an additional test set. The models were also conducted separately in males and females. The numbers of participants in gender-specific and gender-agnostic models are shown in table 7.1.

Table 7.1: The number of participants in the training, validation, and test sets for the female, male, and gender-agnostic models.

		Females	Males	Gender-Agnostic
Normal	train	5,892	2,713	8,605
	validation	655	302	957
	test	727	334	1,061
Arrhythmia	test	1,093	957	2,050
"Ischaemia"	test	1,159	656	1,815
Hypertrophy	test	1,652	1,998	3,650
Other	test	3,882	2,363	6,245
Abnormal	test	3,904	3,611	7,515

7.3 Methods

7.3.1 Computational Environment

All experiments were performed using Google Cloud Ubuntu 16.04 instance with 32v CPU (120G RAM), 2 Nvidia Tesla T4 GPUs, Python version 2.7.15, and Tensorflow version 1.8.0.

7.3.2 Model Creation

After the baseline model was built, the model was grown as described in Chapter 6. Appendix D shows the model evolution for the gender-agnostic, female, and male models. Details of the converged models are shown in figures 7.1, 7.2, and 7.3.

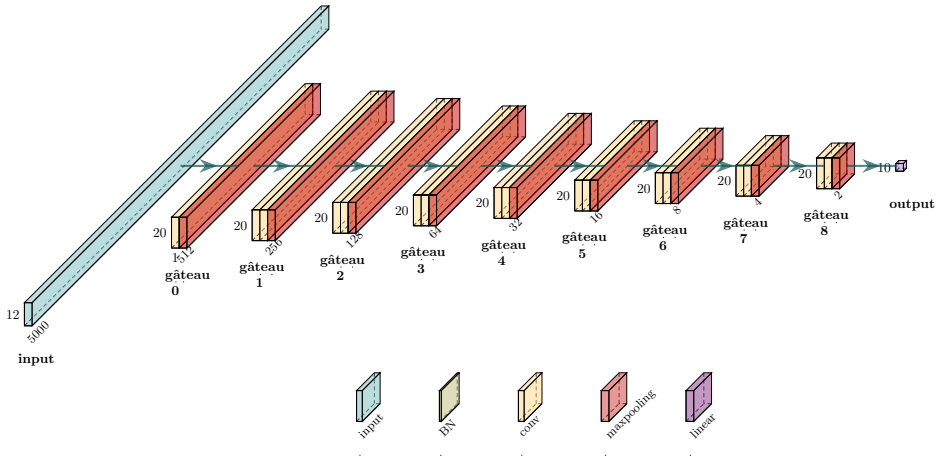


Figure 7.1: The automatically generated ReLU-LCN architecture for the gender-agnostic model. $n_{repeat} = 2$, $n_{maxpool} = 9$, meaning there were a total of 9 max-pooling layers, and there were two convolutional layers stacked between every two max-pooling layers. The activation was ReLU, which followed every convolutional layer, albeit not shown in the figure for simplicity. Batch normalization (green) and skip connection were not needed. The output layer was a time-distributed single-unit linear layer used to make the predictions.

7.3.3 Model Averaging

We performed a 10-fold model averaging using the same approach as described in Chapter 6. The best model was trained for ten times, and the predictions were averaged to make the final prediction, i.e.:

$$\hat{y}_j^{(average)} = \frac{1}{10} \sum_{i=1}^{10} \hat{y}_j^{(i)} \tag{7.1}$$

where $\hat{y}_i$ is the prediction given by the ith model to the jth participant.

7.4 Results

7.4.1 Computational Cost

It took AutoNet 5,151s to find the best female model and 1,381s to find the best male model, and 4,951s to find the best gender-agnostic model.

Table 7.2: Summary statistics of the gender-agnostic model. MAE unit: years

Class	Normal	"Arrhythmia"	"Ischaemia"	"Hypertrophy"	Other	Abnormal
Test size	1,061	2,050	1,815	3,650	6,245	7,515
MAE	5.7	7.5	6.6	6.5	6.3	6.8
Trivial MAE	7.9	9.0	8.4	8.4	8.3	8.6
MSE	51.4	84.7	66.6	65.4	60.5	71.0
Trivial MSE	92.0	115.7	101.5	103.4	100.7	107.4
R^2	44.1%	26.8%	34.4%	36.7%	40.0%	33.9%
Trivial R^2	0	0	0	0	0	0
$\hat{\mu}$	56.3	60.3	59.6	60.3	58.2	60.1
μ	57.2	63.2	60.5	61.2	58.9	61.6
$\hat{\sigma}$	5.5	5.7	5.6	6.7	6.0	6.2
σ	9.6	10.8	10.1	10.2	10.0	10.4
$\hat{y}_{min}$	45.3	45.4	46.2	45.5	45.2	45.4
y_{min}	38	37	39	39	34	37
$\hat{y}_{max}$	76.7	81.2	76.2	93.4	80.6	93.4
y_{max}	82	88	83	86	88	88

7.4.2 Results for the Gender-Agnostic Model

To benchmark, we used a simple model to predict every test case as the mean of the test set, i.e.

$$\hat{y}_i^{(trivial)} = \frac{1}{N}\sum_{i=1}^{N} y_i \tag{7.2}$$

where y_i is the chronological age of the sample, i indices each example, and N is the total number of cases in the set. Mean absolute error (MAE) and mean squared error (MSE) are calculated using equations 7.3 and 7.4.

$$MAE = \frac{1}{N}\sum_{i=1}^{N} |\hat{y}_i - y_i| \tag{7.3}$$

$$MSE = \frac{1}{N}\sum_{i=1}^{N} (\hat{y}_i - y_i)^2 \tag{7.4}$$

R^2 is the coefficient of determination, which measures the fraction of variance explained by a model in a dataset. It is defined as equation 7.5.

$$R^2 = 1 - \frac{\sum_i (\hat{y}_i - y_i)^2}{\sum_i (y_i - \mu)^2} \tag{7.5}$$

It is easy to see that the simple model's R^2 is 0, since $\hat{y}_i = \mu$, $R^2 = 1 - \frac{\sum_i (\mu - y_i)^2}{\sum_i (y_i - \mu)^2} = 0$.

$\hat{\mu}$ and μ represent the predicted and the mean chronological age in the set, and similarly $\hat{\sigma}$ and σ represent the standard deviations of the predicted age and the chronological age in the set, respectively; $\hat{y}_{min}$ and $\hat{y}_{max}$ represent the predicted age range, while y_{min} and y_{max} represent the range of the chronological age. All results were reported on the test set.

We can see that the AutoNet-LCN's MAE was the lowest in the normal class and highest in the "arrhythmia" class. The "other" class had the second-lowest MAE, which is consistent with our conclusion in Chapter 5 that the "other" class chiefly involves "sub-healthy" participants without overt symptoms of CVD. A similar observation can be found in the AutoNet-LCN's MSE. In all classes, the results of the AutoNet-LCN MAE and MSE models were much lower than those of the simple model. The AutoNet-LCN's R^2 explained 44.1% of the variance of the normal class and 40.0% of the variance of the "other" class, while the R^2 for all other classes were much lower.

The predicted mean age ($\hat{\mu}$) was slightly lower than that of the chronological age (μ) in all classes, and the predicted standard deviation ($\hat{\sigma}$) was much lower than that of the chronological age (σ). This is mainly because the model was trained on the normal class, in which high age was rare. Consequently, the model learned a narrower distribution than the chronological age distribution, and is centred around $\hat{\mu}$. Despite this, the model predicted lower mean and standard deviation in the normal test set than any non-normal test sets. The minimum predicted age was 45.2 in the "other" class, and the maximum predicted age was 76.2 in the "ischaemia" class, but the predicted minimum and maximum age were also the lowest in normal individuals.

7.4.3 Female Model Results

We also performed gender-specific modelling by training and testing on female or male participants only. In the female model, the AutoNet-LCN MAEs were lower

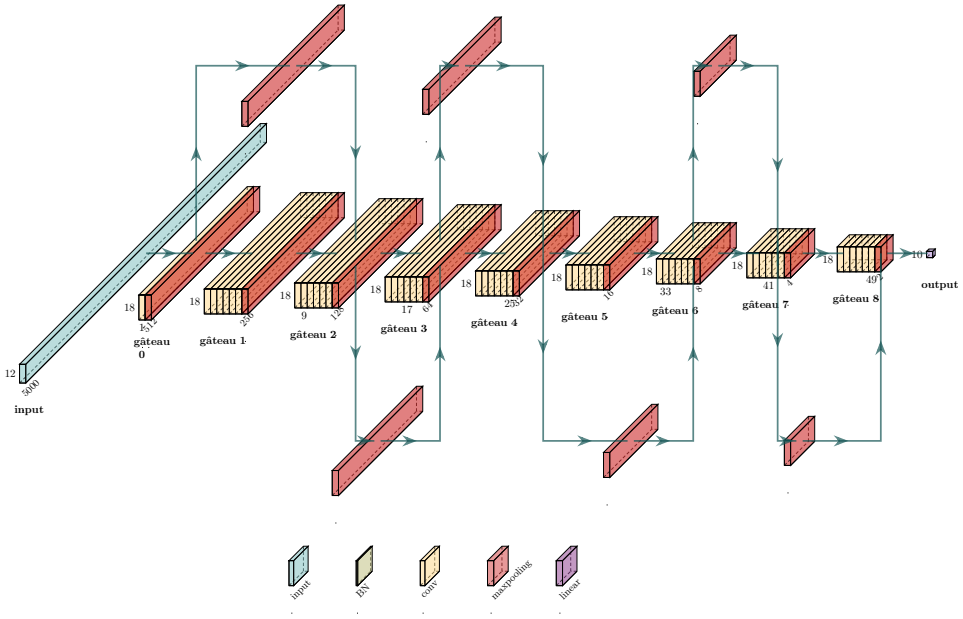


Figure 7.2: The automatically generated ReLU-LCN architecture for the female model. $n_{repeat} = 6$, $n_{maxpool} = 9$, meaning there were a total of 9 max-pooling layers, and there were 6 convolutional layers stacked between every two max-pooling layers. The activation was ReLU, which followed every convolutional layer, not shown in the figure for simplicity. Batch normalisation (green) was not needed. The after-convolution tensors were added to every eighth after-convolutional tensors, which were labelled in the figure. The output layer was a time-distributed single-unit linear layer to make the prediction.

Table 7.3: Summary statistics of the female model. MAE unit: years.

Class	Normal	"Arrhythmia"	"Ischaemia"	"Hypertrophy"	Other	Abnormal
Test size	727	1,093	1,159	1,652	3,882	3,904
MAE	5.6	7.8	6.5	7.3	6.1	7.8
Trivial MAE	8.0	8.7	8.1	8.2	8.1	8.3
MSE	50.6	94.5	66.4	83.9	59.7	81.7
Trivial MSE	92.6	110.0	96.9	98.7	97.2	102.4
R^2	45.4%	14.1%	31.5%	14.9%	38.6%	20.2%
Trivial R^2	0	0	0	0	0	0
$\hat{\mu}$	56.8	59.4	59.9	64.0	58.9	61.5
μ	57.0	62.8	60.1	61.8	58.8	61.6
$\hat{\sigma}$	6.1	5.9	6.5	7.7	6.5	7.2
σ	9.6	10.5	9.8	9.9	9.9	10.1
$\hat{y}_{min}$	41.1	43.7	43.0	38.9	42.6	38.9
y_{min}	39	39	39	40	36	39
$\hat{y}_{max}$	78.2	81.1	84.2	100.5	85.6	100.5
y_{max}	82	84	83	86	88	86

than the trivial MAE in all classes, and the normal class had the lowest MAE. A similar trend can be observed for MSE. The R^2 increased from 44.1% to 45.4% in the female normal test set compared to the gender-agnostic test set. The R^2 for "arrhythmia" and "hypertrophy" were markedly lower than the normal and the "other" classes, with abnormal aggregate class R^2 only 20.2%. The female model demonstrated a stronger trend for the model being able to explain a higher proportion of variance in the normal class than the "arrhythmia" and "hypertrophy" classes, which in turn indicated that "arrhythmia", "ischaemia", and "hypertrophy" classes have distinct characteristics in their ECGs from the normal classes, which are best captured by the "heart age."

The predicted means in all classes were very close to the mean of the chronological age (μ), except for the "arrhythmia" and "hypertrophy" classes. Interestingly, the predicted mean age ($\hat{\mu}$) was over two years higher than the chronological mean in the "hypertrophy" class. In comparison, the predicted mean age ($\hat{\mu}$) in "arrhythmia" was 3.4 years lower than the mean chronological age (μ), suggesting that the model under-predicted the "arrhythmia" class but over-predicted the "hypertrophy" class.

The predicted standard deviation ($\hat{\sigma}$) was lower than the standard deviation of the chronological age (σ) in all classes, and the "arrhythmia" class had the lowest predicted standard deviation ($\hat{\sigma}$), which was surprising as one would expect the normal class to have the lowest predicted standard deviation ($\hat{\sigma}$). However, the normal class indeed had a lower predicted standard deviation ($\hat{\sigma}$) than all other classes except for "arrhythmia".

The predicted range was again narrower than the chronological age range in the normal and the "arrhythmia" classes. In contrast, the predicted minimum age was very close to the minimal chronological age (y_{min}) in the "hypertrophy" and the abnormal aggregate classes, which is encouraging considering the results of the gender-agnostic model where the predicted minimum was higher than the minimum chronological age (y_{min}) in all classes, suggesting the female model can learn a more versatile distribution than the gender-agnostic model.

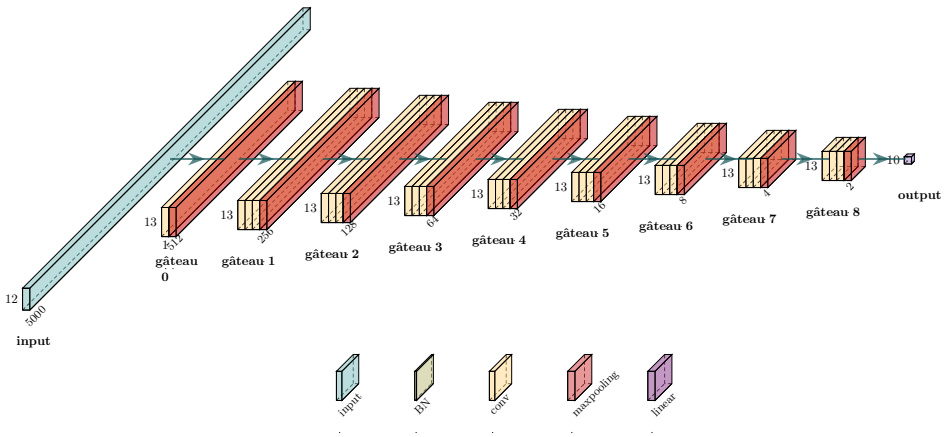


Figure 7.3: The automatically generated ReLU-LCN architects for the male model. $n_{repeat} = 3, n_{maxpool} = 9$, meaning there were a total of 9 max-pooling layers, and there were three convolutional layers stacked between every two max-pooling layers. The activation was ReLU, which followed every convolutional layer, not shown in the figure to declutter the diagram. Batch normalization (green) and skip connections were not needed. The output layer was a time-distributed single-unit linear layer to make the prediction.

The predicted maximum age ($\hat{y}_{max}$) was higher than the maximum chronological age (y_{max}) in "ischaemia", "hypertrophy", and abnormal classes, especially in the "hypertrophy" and the abnormal classes, where the predicted maximum age can be as high as 100.5 years old.

7.4.4 Male Model Results

The male model had fewer training examples ($n = 3713$). Thus, the trend in the female model was less evident in the male model. The AutoNet-LCN MAE and MSE were the smallest in the normal class, which are consistent with the conclusions from the female model, but both were higher than their female counterparts. The AutoNet-LCN R^2 were the highest in the normal classes but were lower than its female counterpart. The predicted mean was very close to the chronological mean in the normal, "other", and the abnormal aggregate classes, while the "arrhythmia" class was again under-predicted, and "ischaemia" and "hypertrophy" classes were again over-predicted. The predicted standard deviation ($\hat{\sigma}$) was the lowest in the normal and much lower than the standard deviation of the chronological age

Table 7.4: Summary statistics of the male model. MAE unit: years.

Class	Normal	"Arrhythmia"	"Ischaemia"	"Hypertrophy"	Other	Abnormal
Test size	334	957	656	1,998	2,363	3,611
MAE	6.2	7.6	7.2	6.9	6.9	7.1
Trivial MAE	7.7	9.4	8.8	8.6	8.6	8.9
MSE	59.3	85.6	77.3	72.4	72.1	76.7
Trivial MSE	90.7	121.9	108.9	106.6	106.4	112.8
R^2	34.7%	29.8%	29.1%	32.1%	32.2%	31.9%
Trivial R^2	0	0	0	0	0	0
$\hat{\mu}$	57.7	61.8	62.2	61.1	59.7	61.5
μ	57.5	63.7	61.2	60.6	59.2	61.6
$\hat{\sigma}$	5.0	6.3	6.5	6.9	6.1	6.7
σ	9.5	11.0	10.4	10.3	10.3	10.6
$\hat{y}_{min}$	46.6	46.8	47.2	47.8	45.9	46.8
y_{min}	38	37	40	39	34	37
$\hat{y}_{max}$	72.8	89.2	82.6	90.2	88.0	90.2
y_{max}	81	88	83	84	85	88

(σ), and also lower than their female counterpart, suggesting a smaller number of training examples resulted in the male model learning a more "rigid" distribution than the female and gender-agnostic models.

The predicted range was narrower than the chronological age range in all classes, while the predicted maximum age was higher than the maximum chronological age (y_{max}) in the "arrhythmia", "hypertrophy", "other", and the abnormal aggregate classes. However, we did not observe as extremely high predicted age as in the gender-agnostic model and the female model, which may be due to the the smaller male training set.

7.4.5 Over-predicted and Under-predicted Ratios

Figures 7.4 and 7.5 show the ratios of over-predicted and under-predicted participants in each test class. Over-prediction was defined as $\hat{y} < y - 2$, and under-prediction was defined as $\hat{y} > y + 2$, and $\hat{y} - 2 \leq y \leq \hat{y} + 2$ were considered correctly predicted. In essence, we ignored the prediction errors of 2 years or less.

It is evident that in the female normal test set, the model tended to over-predict, in contrast to our hypothesis. In "ischaemia", "hypertrophy" and "other" classes,

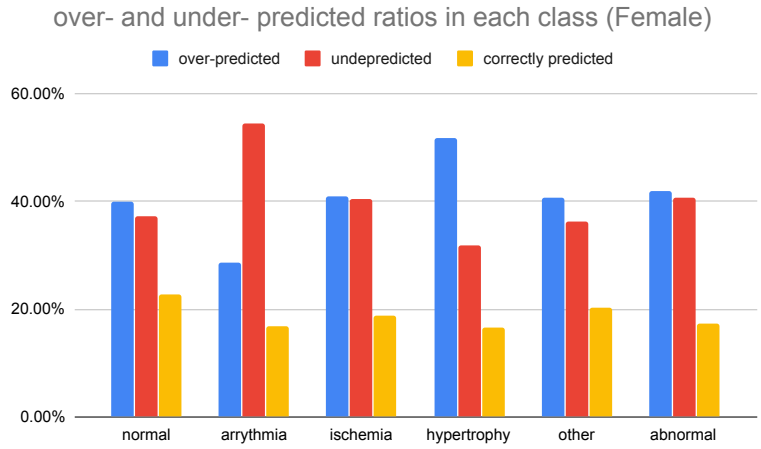


Figure 7.4: Over- and under-predicted ratios in each class (female). The corresponding numbers are in appendix E.

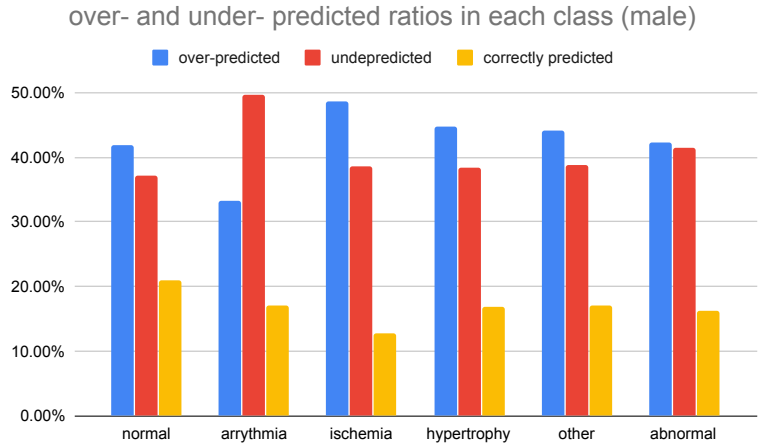


Figure 7.5: Over- and under-predicted ratios in each class (male). The corresponding numbers are in appendix E.

the model tended to over-predict, which is consistent with our hypothesis. It is surprising to find that in the "arrhythmia" class, the model tended to under-predict rather than over-predict, and by a wide margin. This may suggest either the model is not suitable to apply to "arrhythmia" patients, or the absolute prediction errors, rather than the signed errors, may suggest underlying CVD abnormalities.

Similar results were observed in the male sets, with the contrast of "ischaemia" over-and under-prediction being more extreme than their female counterparts. This

may suggest that the male "ischaemia" participants in the CKB dataset have more recognizable ECG abnormalities than their female counterparts.

We examined the top 10 over-predicted and under-predicted female and male cases in each test set, and these results are presented below:

7.4.6 Top 10 Over- and Under-predicted Cases

We show the participants' Mortara descriptions, and the "reasons", if any, provided by the Mortara device to support its descriptions given the ECG waveforms. The top 10 cases were ranked by the prediction error, and the light-blue shaded cases were flagged "abnormal ECG" by the Mortara device. We complement the Mortara descriptions and reasons with the participants' systolic blood pressure (SBP) and diastolic blood pressure (DBP) which neither the Mortara algorithm nor the AutoNet-LCN had access to. The last row of each table shows the averages of the numerical variables.

Unsurprisingly, the over-predicted cases were in the middle-age group, while the chronological ages of the under-predicted cases were over 70 years. We can see that both top under-predicted and over-predicted cases have many ECG abnormalities. "Arrhythmia" and "other" cases appeared in the under-predicted, while "hypertrophy" cases dominated the over-predicted group. This, in turn, rejects our hypothesis that under-predicted individuals were "healthier" than their peers. The absolute prediction error may be an indicator of heart health, with the under-prediction implying arrhythmic abnormalities and positive errors implying hypertrophic abnormalities. Interestingly, the over-predicted cases also had higher mean levels of SBP and DBP than the under-predicted cases. The mean SBP of over-predicted cases were higher than the normal value (120 mm Hg) while the mean DBP of the under-predicted cases is lower than the normal value (80 mmHg), suggesting both under-prediction and over-prediction imply CVD abnormalities, but in different ways.

Table 7.5: Top 10 under-predicted cases in the female model. A: "arrhythmia"; DQW: data quality warning; H: "hypertrophy"; I: "ischaemia"; LVH: left ventricular hypertrophy; MI: myocardial infraction; N: normal; O: "other" class; RBBB: right bundle branch block; SR: sinus rhythm

y	$\hat{y}$	Class	Descriptions	Reasons	SBP	DBP
79	49	I	SR with occasional ventricular premature complexes, probable inferior MI, probably old, abnormal ECG	35 ms Q wave in II/aVF	126	69
82	53	N	SR, normal ECG		164.5	69
68	39	H	SR, RBBB, left anterior fascicular block, voltage criteria for LVH, abnormal ECG	120+ ms QRS duration, upright V1, 40+ ms S in I/aVL/V4/V5/V6, QRS axis $\leq$ -45, QR in I, RS in II, meets criteria in one of: R(aVL), S(V1), R(V5), R(V5/V6)+S(V1)	190	87
82	53	A	SR, RBBB, abnormal ECG	120+ ms QRS duration, upright V1, 40+ ms S in I/aVL/V4/V5/V6	156.5	76.5
83	55	A	SR with frequent ventricular premature complexes, abnormal rhythm ECG		143	61
83	55	I	sinus bradycardia with occasional ventricular premature complexes, moderate T-wave abnormality, consider inferior ischaemia, abnormal ECG	-0.1+ mV T wave in II/aVF	156	70.5
80	52	I	SR, anteroseptal MI, of indeterminate age, abnormal ECG	40+ ms Q wave in V1-V4	116.5	65
82	54	O	SR, marked left axis deviation, abnormal ECG, DQW	QRS axis < -30	102.5	61.5
80	53	A	sinus tachycardia, abnormal rhythm ECG		159.5	75
79	52	I	sinus rhythm with prolonged PR interval, RBBB, left posterior fascicular block, inferior MI, probably old, anterolateral MI, probably old, abnormal ECG	120+ ms QRS duration, upright V1, 40+ ms S in I/aVL/V4/V5/V6, QRS axis > 109, inferior Q, 40+ ms Q wave and/or St/T abnormality in II/aVF, 40+ ms Q wave in I/aVL/V3-V6	148	59
79.8	46.2				127.3	61.9

Table 7.6: Top 10 over-predicted cases in the female model. A: "arrhythmia"; H: "hypertrophy"; LVH: left ventricular hypertrophy; SR: sinus rhythm.

y	$\hat{y}$	Class	Descriptions	Reasons	SBP	DBP
65	93	H	atrial fibrillation, voltage criteria for LVH, ST deviation and marked T-wave abnormality, consider anterolateral ischaemia, ST deviation and moderate T-wave abnormality, consider inferior ischaemia, abnormal ECG	meets criteria one of: R(aVL), S(V1), R(V5), R(V5/V6)+S(V1), -0.5+ mV T wave in I/aVL/V3-V6, -0.1+ mV T wave in II/aVF	130	77
44	72	H	SR, voltage criteria for LVH, nonspecific ST & T-wave abnormality, abnormal ECG	meets criteria in one of: R(aVL), S(V1), R(V5), R(V5/V6)+S(V1)	156.5	95
48	77	H	SR, possible right ventricular conduction delay, left ventricular hypertrophy, and ST-T change, abnormal ECG	voltage criteria plus ST/T abnormally	128	78
52	82	H	SR, moderate voltage criteria for LVH, consider normal variant, moderate ST depression, abnormal QRS-T angle, abnormal ECG	meets criteria in one of: R(aVL), S(V1), R(V5), R(V5/V6)+S(V1), 0.05+ mV ST depression, QRS-T axis difference > 60	163.5	75.5
67	97	H	SR with short PR interval, left ventricular hypertrophy and ST-T, abnormal ECG	voltage criteria plus ST/T abnormality	122	70.5
58	89	H	SR, left ventricular hypertrophy and ST-T change, abnormal ECG	voltage criteria plut St/T abnormality	141.5	72
51	83	H	ectopic atrial rhythm, left ventricular hypertrophy and ST-T change, abnormal ECG	voltage criteria plus ST/T abnormality	148	87.5
41	75	A	sinus rhythm, with short PR interval with occasional ventricular premature complexes, borderline ECG		111.5	80.5
44	80	H	SR,m voltage criteria for LVH, abnormal ECG	meets criteria in one of: R(aVL), S(V1), R(V5), R(V5/V6)+S(V1)	187	87
42	82	H	SR wit short PR interval, left ventricular hypertrophy and ST-T change, probably inferior myocardial infarction, of indeterminate age with posterior extension [prominent R wave in V1/V2], abnormal ECG	voltage criteria plus ST/T abnormality, 35 ms Q wave in II/aVF	151	79.5
51.2	83				143.9	80.3

Table 7.7: Top 10 under-predicted cases in the male model. A: "arrhythmia"; H: "hypertrophy"; I: "ischaemia"; LVH: left ventricular hypertrophy; N: normal; O: "other" class; SR: sinus rhythm.

y	$\hat{y}$	Class	Descriptions	reasons	SBP	DBP
84	52	H	SR, voltage criteria for LVH	meets criteria in one of: R(aVL), S(V1), R(V5), R(V5/V6)+S(V1)	126.5	72.5
80	54	N	SR, normal ECG		121.5	58.5
78	52	I	SR, inferior myocardial infarction, of indeterminate age, abnormal ECG	40+ ms Q wave and/or ST/T abnormality in II/aVF	110	60.5
76	51	A	SR, early repolarisation, borderline ECG	ST elevation with normally inflected T wave	114	61
81	56	O	SR, low QRS voltage in extremity leads, pattern consistent with pulmonary disease, abnormal ECG	QRS deflection < 0.5 mV in limb leads	121.5	68
82	57	A	SR, early repolarisation, tall T-waves, suggesting hyperkalemia, abnormal ECG	ST elevation with normally inflected T wave	190.5	88
82	58	O	SR, marked left axis deviation, moderate intraventricular conduction delay, abnormal ECG	QRS axis $<$ -30, 110+ ms QRS duration	142.5	70.5
79	55	I	SR, moderate T-wave abnormality, consider inferior ischaemia, abnormal ECG	-0.1+ mV T wave in II/aVF	152.5	62
76	52	A	sinus tachycardia, moderate intraventricular conduction delay, abnormal rhythm ECG	110+ ms QRS duration	143.5	76.5
77	53	A	sinus tachycardia, nonspecific T-wave abnormality, abnormal rhythm ECG		116.5	56.5
79.5	54				133.9	67.4

Table 7.8: Top 10 over-predicted cases in the male model. H: "hypertrophy"; MI: myocardial infarction; I: "ischaemia"; LVH: left ventricular hypertrophy; O: other; SR: sinus rhythm.

y	$\hat{y}$	Class	Descriptions	Reasons	SBP	DBP
54	79	I	SR, left anterior fascicular block, moderate T-wave abnormality, consider lateral ischaemia, abnormal ECG	QRS axis $\leq$ -45, QR in I, RS in II, -0.1+ mV T wave in I/aVL/V5/V6	106	65
48	73	H	SR, borderline right axis deviation, moderate voltage criteria for LVH, consider normal variant, borderline ECG	QRS axis $>$ 90, meets criteria in one of: R(aVL), S(V1), R(V5), R(V5/V6)+S(V1)	121	73
41	67	I	SR, possible anterior MI, of indeterminate age, abnormal ECG	30 ms Q wave in V3/V4, or R $<$ 0.2 mV in V4	106	71
55	81	H	atrial fibrillation, voltage criteria for LVH, nonspecific T-wave abnormality, abnormal ECG	meets criteria in one of: R(aVL), S(V1), R(V5), R(V5/V6)+S(V1)	108.5	69
46	72	O	SR, nonspecific ST & T-wave abnormality, borderline ECG		171.5	111.5
59	86	H	atrial fibrillation, possible left ventricular hypertrophy, marked T-wave abnormality, consider lateral ischaemia, moderate T-wave abnormality, consider inferior ischaemia, abnormal ECG	voltage criteria plus LAE or QRS widening, -0.5+ mV T wave in I/aVL/V5/V6, -0.1+ mV T wave in II/aVF	128	58
48	75	H	SR, left ventricular hypertrophy and ST-T change, abnormal ECG	voltage criteria plus ST/T abnormality	146.5	91
43	70	H	SR, left ventricular hypertrophy and ST-T change, abnormal ECG	voltage criteria pus ST/T abnormality	222.5	146
43	71	H	sinus tachycardia, voltage criteria for LVH, junctional ST depression, consider normal variant, abnormal ECG	meets criteria in one of: R(aVL), S(V1), R(V5), R(V5/V6)+S(V1), 0.1+ mV junctional depression	151	87.5
44	75	O	SR, arm leads reversed, atypical ECG	inverted P and QRS in I	147.5	99
48.1	74.9				140.9	87.1

7.5 Discussion and Conclusion

Previous studies have reported heart age calculators (Bonner, Jansen, Newell, Irwig, Teixeira-Pinto, et al. 2015; Lopez-Gonzalez et al. 2015; Lowensteyn et al. 1998). For example, NHS uses JBS3 risk calculator, which takes account of established CVD risk factors, including blood pressure, smoking, cholesterol, and diabetes [1]. The NHS heart age is estimated from the lifetime risk of CVD, relative to people of the same age, sex, and ethnicity who have "optimal" risk factor levels (for example, non-smoker, systolic blood pressure $<$ 120mm Hg) (Patel et al. 2016). The NHS heart age test gives a higher heart age if any of the risk factors were not in the optimal range, and 78% of the 2 million users obtained higher heart age than their chronological age [2] and thereby encouraged to visit their GP.

The intuitive interpretation of the heart age is that higher heart age than their chronological age implies a higher CVD risk, while a lower heart age implies a healthier heart compared with people of their chronological age group. The NHS heart age test states that "having a heart age older than your chronological age means that you are at a higher risk of having a heart attack or stroke." while not providing any interpretation when the heart age is lower than the chronological age. Although heart age calculators are intuitive to use and raise CVD risk awareness in the population (Wells et al. 2010), their results can be misleading to the public and lead to over-diagnosis (Bonner, McKinn, et al. 2019). The relation between the heart age and the CVD risk is not well calibrated nor well validated (Bonner, McKinn, et al. 2019). Systematic reviews and randomised trials have concluded that there is insufficient evidence to recommend heart age in clinical practice (French et al. 2017; Bize et al. 2012; Waldron et al. 2011; Kulendrarajah, Grey, and Nunan 2020; Bonner, Jansen, Newell, Irwig, Teixeira-Pinto, et al. 2015; Svendsen et al. 2020; Krogsbøll et al. 2012; Bonner, Jansen, Newell, Irwig, Glasziou, et al. 2014).

[1] https://www.nhs.uk/conditions/nhs-health-check/check-your-heart-age-tool/

[2] https://www.gov.uk/government/news/heart-age-test-gives-early-warning-of-heart-attack-and-stroke. Accessed 30 April 2020

However, all previous studies assessed the effect of the heart age as a pre-consultation screening tool in clinical practice, where the patient's medical history and co-morbidity are not taken into account, resulting in implausible estimates of heart age that discredited the results (for example, older heart age in very fit people, or younger heart age in people who are obese)(Bonner, Jansen, Newell, Irwig, Glasziou, et al. 2014). The heart age has been found to motivate lifestyle changes in clinical practice (Bonner, McKinn, et al. 2019). Our models calculate the heart age from the standard 12-lead ECG, which can only be obtained in a clinical setting. We do not recommend our model be used as a heart age calculator as a pre-consultation screening tool, but potentially as a risk score associated with the ECG waveforms for clinical risk assessment.

The results of the present study are consistent with previous studies: Attia et al. 2019 trained a CNN on 10-s 12-lead standard ECG on 499,727 participants and reported MAE 5.9 and R^2 0.7. They also concluded that the absolute error, rather than the signed error, may be useful as a heart health score. They did not differentiate normality nor gender in their training and test sets. We performed a detailed analysis of different CVD disease groups separately in men and women and found lower absolute errors in the healthy population than in the "arrhythmia", "ischaemia", and "hypertrophy" populations. However, the findings contradict our hypothesis that under-prediction would imply being healthier. In contrast, under-prediction indicated a higher risk of arrhythmic abnormalities. Our models were also automatically generated by the AutoNet-LCN algorithms and involved no human effort in determining the model architecture.

The analyses presented in this chapter evaluated heart age using ECG waveforms. Future work may include using the SBP and DBP as alternative labels and possibly predict the blood pressure either alone or in combination with age, for example, by using three linear output units. Predicting blood pressure may be informative for cardiovascular medicine, as the relationship between ECG waveforms and blood pressure have not been well established.

8 Conclusions and Future Work

8.1 Summary of Results

8.1.1 SGB-F82 Model for 4-class Classification using Extracted Features

In Chapter 5, we demonstrated that machine learning models could classify ECG with high accuracy without any knowledge of the diagnosis criteria; all they need is relevant features. We examined 11 machine learning models (Logistic Regression, Linear Discriminant Analysis, Naive Bayes, support vector machine, CART, KNN, stochastic gradient boosting, bagging, random forest, AdaBoost, and extra trees) representative of all major machine learning model families except neural networks, and found stochastic gradient boosting performed best. The 77.3% four-class classification accuracy achieved by SGB-F82 is encouraging. We extracted the amplitudes of the P, Q, R, S, T waves, and the baseline levels, from each of the 12 leads, constructing a total of 72 new features. The addition of these 72 new features lent significant improvement over the features provided by the Mortara device. The top features identified by the SGB-F84 model were very different from the ones commonly used in clinic. Lead I did not appear in the top 10 features at all, which may suggest many studies using only single lead, typically lead I or II, even when the 12-lead ECG is available, have sub-optimal performances. Our

findings suggest using lead V5 instead of lead I when the single-lead analysis is inevitable due to resource constraints.

8.1.2 End-to-end Deep Learning ECG Classification using AutoNet-LCN

In Chapter 6, we proposed a novel theorem called the Layer-wise Convex Network (LCN) and a heuristic neural architecture search algorithm - the AutoNet - to directly analyse the raw ECG waveforms, without beat segmentation, denoising, nor feature extraction. The AutoNet can generate LCNs end-to-end based on the characteristics of the datasets and the machine learning task. The AutoNet generated LCNs were benchmarked with the state-of-the-art model and evaluated on three ECG classification datasets (PhysioNet 2017 Challenge, ICBEB 2018 Challenge, and China Kadoorie Biobank ECG dataset), and have outperformed the state-of-the-art model on all three datasets by a wide margin (9-16% improvement in terms of F1 score), with 1-2% of the parameters and no more than 2 hours of architecture search time, in comparison to weeks to months of trial-and-error by human researchers in the conventional deep learning model development process. It is especially encouraging considering the state-of-the-art model has already been demonstrated to exceed the average cardiologist ability to classify "arrhythmias".

Also, we proposed the PC ratio as an intuitive measure of the computational efficiency and the difficulty of the machine learning task given a dataset. Given the same model, the higher the PC ratio, the easier the machine learning task is, which may be thanks to the high signal quality or abundant training examples in the dataset; given the same dataset, the higher the PC ratio is, the more computationally efficient the model is. PC ratio may help researchers focus effort on gathering more data from the same study participants, recruiting more study participants, or improving the model.

8.1.3 Predicting the "Heart Age" from Raw ECG Waveforms

In Chapter 7, to address the issue that the training targets in CKB are provided by the deterministic rule-based heuristics called the Minnesota Code, which in theory can be approximated to arbitrary precision by a large enough neural network with enough training examples and training time, we proposed a novel paradigm: learning using alternative labels. We defined alternative labels as clinically relevant, easy to acquire, and relatively accurate labels, such as people's age, sex, and blood pressure. We used AutoNet-LCN to automatically build heart age predictors by training the neural networks on the healthy population, and test our hypothesis that the model would predict lower-than-chronological age for a healthy individual, and higher-than-chronological age for individuals with CVD abnormalities. Our findings are surprising that under-prediction does not indicate the participant is healthier than their peers, but has more tendency towards "arrhythmia", while over-prediction suggests a tendency towards "hypertrophy" and "ischaemia". Under prediction also correlates with low blood pressure and over-prediction correlates hypertension. While our models need further calibration and validation with follow-up longitudinal clinical outcomes, the findings may suggest some relation between ECG-derived age, blood pressure, and CVD conditions in ways that are not yet clear in medical research, which may merit further investigation.

8.2 Future Work

8.2.1 Validate Machine Learning Probabilistic Risk Scores with Clinical Diagnosis

Although in Chapters 5 and 6, the results were reported as classification accuracy or F1 scores, the actual outputs of the machine learning models are probabilities. Also, "arrhythmia", "ischaemia", and "hypertrophy" are not mutually exclusive. Thus future work may involve validating the probabilistic outputs of the machine learning models with clinical diagnostics and prognosis in follow-up studies. The machine learning probabilistic risk scores may be especially relevant to the participants in

the "other" class, who did not meet the criteria of "arrhythmia", "ischaemia", or "hypertrophy". It would be interesting to examine whether the machine learning models can forecast the CVD onset by following up this group of participants.

A retrospective analysis may be conducted taking advantage of the occurrence time of the diseases in the health insurance data available in the CKB database, to illustrate how machine learning prediction improves as distant events from ECG acquisition are removed, and further study the performance of machine learning models as a function of time.

8.2.2 Epidemiology Analysis with Other Risk Factors

The 4-class probabilities provided by our machine learning models may be interpreted as risk scores, which can be further integrated with the heart age, chronological age, gender, body mass index (BMI), blood pressure, and genetic genome-wide association study (GWAS) data available in the CKB to develop a comprehensive deep learning heart disease diagnosis system, potentially for more disease types than we have initially considered, such as cancer and stroke. The latter will involve consideration of probabilistic models that permit the fusion of categorical data, and which may include sparse techniques for handling the largely-incomplete records in the dataset. The AutoNet LCN may be especially useful to fuse different types of features as it does not require human design of the networks or feature engineering.

8.2.3 Calibration of the Heart Age

In Chapter 7 our models have surprising findings that over-prediction correlates with hypertension and "hypertrophy", while under-prediction correlates with low blood pressure and "arrhythmia". Further investigation is needed to explain this phenomenon. Also, the heart age model needs further statistical analysis of the significance of the findings, and validation with longitudinal clinical outcomes to see whether the predicted heart age has predictive value.

8.2.4 Theoretical Study of the LCN Theorem

Although we have explained the rationale of the LCN theorem in Chapter 6, more rigorous proof and study of its mathematical properties are needed to validate our hypothesis. For example, is the loss indeed convex with respect to the parameters? The idea behind LCN has similarities to (Bengio et al. 2006) in which the loss is indeed proven to be convex with respect to the parameters, and a global minimum can be proven to exist. Although the resulting networks have similarities, the way such networks are found are different. In this thesis, the LCNs are found by mathematically calculating the number of required parameters per layer, while in Bengio et al. 2006 the number of neurons per layer is found by inserting one neuron at a time and use a linear classifier to minimise a weighted sum of loss. Secondly, is the Hessian of the LCNs indeed better conditioned compared to alternative CNNs, such as the ResNet? How does the loss surface evolve during training and in the model growth step? Finally, in this thesis, we used conventional Adam optimizer to optimize LCN, while LCN theorem is motivated by regarding the parameters ($\boldsymbol{w}$ and $\boldsymbol{b}$) and the layer outputs ($\boldsymbol{a}$) as if their roles are reversed. Nevertheless, we hypothesize that the optimization of LCN is mathematically and computationally equivalent to conventional neural network optimization using backpropagation. This hypothesis also needs more rigorous proof.

The AutoNet algorithm (algo. 2) and the LCN-generation algorithm (algo. 1) are equivalent to the controller-generator systems described in the seminal paper by Zoph and Le 2016 in neural architecture search (NAS). Auto-Net LCN is essentially a NAS algorithm, although the focus of this thesis is its application on ECG classification. To fully establish the advantage of AutoNet-LCN, we need to apply it to standard machine learning benchmarks such as the ImageNet and study whether AutoNet-LCN can perform consistently well when the training size is very different from a few thousand.

Appendices

A

The Number of Participants in Each Class and Age Group

Table A.1: The number of participants in each class and age group in the CKB dataset (all participants). The numbers in the brackets are the percentage of the conditions in the relevant age group. A: "arrhythmia", I: "ischaemia", H: "hypertrophy", O: other.

Age	**N**	**A**	**I**	**H**	**O**	**Total**
<50	3,021	300	308	573	1,352	5,554
	(54.4%)	(5.4%)	(5.6%)	(10.3%)	(24.3%)	(100%)
50-59	3,546	582	585	987	2,006	7,706
	(46.0%)	(7.6%)	(7.6%)	(12.8%)	(26.0%)	(100%)
60-69	2,982	732	597	1,114	1,924	7,349
	(40.6%)	(10.0%)	(8.1%)	(15.2%)	(26.2%)	(100%)
70+	1,230	858	380	749	1,080	4,297
	(28.6%)	(20.0%)	(8.8%)	(17.4%)	(25.1%)	(100%)
Total	10,779	2,472	1,870	3,423	6,362	24,906
	(43.3%)	(9.9%)	(7.5%)	(13.7%)	(25.5%)	(100%)

Table A.2: The number of participants in each class and age group in the CKB dataset (female participants). The numbers in the brackets are the percentage of the conditions in the relevant age group. A: arrhythmia, I: "ischaemia", H: "hypertrophy", O: other.

Age	N	A	I	H	O	Total
<50	2,161	167	199	219	834	3,580
	(60.4%)	(4.7%)	(5.6%)	(6.1%)	(23.3%)	(100%)
50-59	446	308	387	442	1,289	4,872
	(50.2%)	(6.3%)	(7.9%)	(9.1%)	(26.5%)	(100%)
60-69	2,006	417	382	537	1,204	4,546
	(44.1%)	(9.2%)	(8.4%)	(11.8%)	(26.5%)	(100%)
70+	761	415	225	364	641	2,406
	(31.6%)	(17.3%)	(9.4%)	(15.1%)	(26.6%)	(100%)
Total	5,374	1,307	1,193	1,562	3,968	15,404
	(34.9%)	(8.5%)	(7.7%)	(10.1%)	(25.8%)	(100%)

Table A.3: The number of participants in each class and age group in the CKB dataset (male participants). The numbers in the brackets are the percentage of the conditions in the relevant age group. A: arrhythmia, I: "ischaemia", H: "hypertrophy", O: other.

Age	N	A	I	H	O	Total
<50	860	133	100	354	518	1,974
	(43.6%)	(6.7%)	(5.5%)	(17.9%)	(26.2%)	(100%)
50-59	1,100	274	198	545	717	2,834
	(38.8%)	(9.7%)	(7.0%)	(19.2%)	(25.3%)	(100%)
60-69	976	315	215	577	720	2,803
	(34.8%)	(11.2%)	(7.7%)	(20.6%)	(25.7%)	(100%)
70+	469	443	155	385	439	1,891
	(24.8%)	(23.4%)	(8.2%)	(8.2%)	(23.2%)	(100%)
Total	3,405	1,165	677	1,861	2,394	9,502
	(35.8%)	(12.3%)	(7.1%)	(19.6%)	(25.2%)	(100%)

B

Architectures Found in the Five Repeats on the Three Datasets

Table B.1: The hyperparameters of the models found on the five ICBEB experiments. The most common architectures are in bold font.

Repeat	ReLU-LCN			Leaky-LCN		
	n_{repeat}	*skip*	*bn*	n_{repeat}	*skip*	*bn*
1	**5**	+	+	7	+	+
2	6	+	+	**5**	+	+
3	4	+	+	**5**	+	+
4	**5**	+	+	**5**	+	+
5	**5**	+	+	**5**	+	+

Table B.2: The hyperparameters of the models found on the five PhysioNet experiments. The most common architectures are in bold font.

Repeat	ReLU-LCN			Leaky-LCN		
	n_{repeat}	*skip*	*bn*	n_{repeat}	*skip*	*bn*
1	3	+	+	**4**	+	+
2	4	+	-	5	+	-
3	2	+	-	**4**	+	+
4	**2**	**-**	**-**	**4**	+	+
5	**2**	**-**	**-**	**4**	+	+

Table B.3: The hyperparameters of the models found on the five CKB experiments. The most common architectures are in bold font.

Repeat	ReLU-LCN			Leaky-LCN		
	n_{repeat}	*skip*	*bn*	n_{repeat}	*skip*	*bn*
1	**1**	**-**	**-**	2	+	-
2	**1**	**-**	**-**	**1**	**-**	**-**
3	**1**	**-**	**-**	2	+	-
4	**1**	**-**	**-**	**1**	**-**	**-**
5	**1**	**-**	**-**	**1**	**-**	**-**

C
Runtime Costs

Table C.1: Runtime (s) of the 15 experiments using the three models. The lowest runtime of each experiment is shown in bold font.

Dataset	Experiment	ReLU-LCN	Leaky-LCN	Hannun-Rajpurkar
ICBEB	1	**1,152**	2,050	1,638
	2	**920**	1,025	1,911
	3	930	**819**	1,820
	4	**910**	1,131	2,002
	5	**864**	1,216	2,184
PhysioNet	1	**780**	966	2,178
	2	**936**	1,350	2,178
	3	**435**	1,230	2,178
	4	**416**	1,302	2,420
	5	**500**	1,189	3,993
CKB	1	**80**	114	442
	2	100	**88**	408
	3	**108**	144	476
	4	92	**72**	374
	5	96	**68**	510

The runtime is calculated by equation 6.53. The runtime cost in weight initialisation step, which typically took a few seconds, is omitted. ReLU-LCN runtime is significantly lower than Leaky-LCN, which is significantly lower than the Hannun-Rajpurkar model.

D

Evolution of the Heart Age Models

Table D.1: Gender-agnostic model evolution. Training: 8605, validation: 957, maximum parameters per layer: 8020, c = 20. n_{cnov}: number of convolutional layers. $n_{\boldsymbol{\theta}}$: total parameters. l_v: validation loss. l_t: training loss. l_v^*: minimal validation loss so far. l_t^*: minimal training loss so far. The shaded setting was the best model discovered by the AutoNet.

n_{conv}	Repeat	Skip	BN	$n_{\boldsymbol{\theta}}$	l_v	l_t	Decision
9	1	0	F	69,001	57.01	43.71	repeat+=1
17	2	0	F	133,161	56.48	50.62	$l_v < l_v^*$, repeat+=1
25	3	0	F	197,321	57.42	53.57	$l_v > l_v^*, l_t > l_t^*$, skip=8
25	3	8	F	197,321	56.95	53.05	$l_v > l_v^*, l_t > l_t^*$, BN=True
25	3	8	T	198,345	70.23	75.52	$l_v > l_v^*, l_t > l_t^*$, converge

Table D.2: Female model evolution. Training: 5892, validation: 655, maximum parameters per layer: 2210, c = 18. n_{cov}: number of convolutional layers. $n_{\boldsymbol{\theta}}$: total parameters. l_v: validation loss. l_t: training loss. l_v^*: minimal validation loss so far. l_t^*: minimal training loss so far. The shaded setting was the best model discovered by the AutoNet.

n_{conv}	Repeat	Skip	BN	$n_{\boldsymbol{\theta}}$	l_v	l_t	Decision
9	1	0	F	50,725	59.03	6.25	repeat+=1
17	2	0	F	97,525	53.91	5.99	$l_v < l_v^*, l_t < l_t^*$, repeat+=1
25	3	0	F	144,325	52.31	5.73	$l_v < l_v^*, l_t < l_t^*$, repeat+=1
33	4	0	F	191,125	51.69	5.76	$l_v < l_v^*, l_t < l_t^*$, repeat+=1
41	5	0	F	237,925	55.60	6.03	$l_v > l_v^*, l_t > l_t^*$, skip=8
41	5	8	F	237,925	53.74	5.60	$l_t < l_t^*$, repeat=repeat+=1
49	6	8	F	284,725	51.67	5.66	$l_v < l_v^*$, repeat=repeat+=1
57	7	8	F	331,525	52.95	5.80	$l_v > l_v^*, l_t > l_t^*$, BN=True
57	7	8	T	333,601	83.33	7.64	$l_v > l_v^*, l_t > l_t^*$, converge

Table D.3: Male model evolution. Training: 2713, validation: 302, maximum parameters per layer: 2210. c = 13. $n_{\boldsymbol{\theta}}$: total parameters. l_v: validation loss. l_t: training loss. l_v^*: minimal validation loss so far. l_t^*: minimal training loss so far. The shaded setting was the best model discovered by the AutoNet.

n_{cov}	Repeat	Skip	BN	$n_{\boldsymbol{\theta}}$	l_v	l_t	Decision
9	1	0	F	19,735	73.23	69.79	repeat+=1
17	2	0	F	37,415	71.48	66.59	$l_v < l_v^*$, repeat+=1
25	3	0	F	55,095	63.93	57.25	$l_v < l_v^*$, repeat+=1
33	4	0	F	72,775	92.99	100.57	$l_v > l_v^*, l_t > l_t^*$, skip=8
33	4	8	F	72,775	66.65	67.75	$l_v > l_v^*, l_t > l_t^*$, BN=True
33	4	8	T	73,657	80.23	80.17	$l_v > l_v^*, l_t > l_t^*$, converge

E

Over- and Under-predicted Ratios and Numbers

Table E.1: The over-predicted and under-predicted numbers and ratios in each class (female model)

Class	Test Size	Over-Predicted (%)	Under-Predicted (%)	Correctly Predicted (%)
Normal	727	291 (40.03)	270 (37.14)	166 (22.83)
Arrhythmia	1,093	313 (28.64)	595 (54.44)	185 (16.93)
Ischaemia	1,159	473 (40.81)	467 (40.29)	219 (18.90)
Hypertrophy	1,652	853 (51.63)	524 (31.72)	275 (16.65)
Other	3,882	1,627 (41.91)	1,449 (37.33)	806 (20.76)
Abnormal	3,904	1,639 (41.98)	1,586 (40.63)	679 (17.39)

Table E.2: The over-predicted and under-predicted numbers and ratios in each class (male model)

Class	Test Size	Over-Predicted (%)	Under-Predicted (%)	Correctly Predicted (%)
Normal	334	140 (41.92)	124 (37.13)	70 (20.96)
Arrhythmia	957	319 (33.33)	475 (49.63)	163 (17.03)
Ischaemia	656	319 (48.63)	253 (38.57)	84 (12.80)
Hypertrophy	1998	892 (44.64)	767 (38.39)	339 (16.97)
Other	2363	1044 (44.18)	916 (38.76)	403 (17.05)
Abnormal	3611	1530 (42.37)	1495 (41.40)	586 (16.23)

F

Mapping from the Mortara Labels to the Four Classes

ECG description	Label
ACUTE MI	“ischaemia”
..PEDIATRIC ECG INTERPRETATION	unclassified
ABNORMAL ECG	unclassified
ABNORMAL QRS-T ANGLE	unclassified
ABNORMAL RHYTHM ECG	“arrhythmia”
ANTERIOR MYOCARDIAL INFARCTION , OF INDETERMINATE AGE	“ischaemia”
ANTERIOR MYOCARDIAL INFARCTION , POSSIBLY ACUTE	“ischaemia”
ANTERIOR MYOCARDIAL INFARCTION , PROBABLY OLD	“ischaemia”
ANTERIOR MYOCARDIAL INFARCTION , PROBABLY RECENT	“ischaemia”
ANTEROLATERAL MYOCARDIAL INFARCTION , OF INDETERMINATE AGE	“ischaemia”
ANTEROLATERAL MYOCARDIAL INFARCTION , PROBABLY OLD	“ischaemia”
ANTEROLATERAL MYOCARDIAL INFARCTION , PROBABLY RECENT	“ischaemia”
ANTEROSEPTAL MYOCARDIAL INFARCTION , OF INDETERMINATE AGE	“ischaemia”
ANTEROSEPTAL MYOCARDIAL INFARCTION , POSSIBLY ACUTE	“ischaemia”
ANTEROSEPTAL MYOCARDIAL INFARCTION , PROBABLY RECENT	“ischaemia”
ARM LEADS REVERSED	unclassified
ATRIAL FIBRILLATION	“arrhythmia”
ATRIAL FIBRILLATION WITH ABERRANT CONDUCTION OR VENTRICULAR PREMATURE COMPLEXES	“arrhythmia”
ATRIAL FIBRILLATION WITH RAPID VENTRICULAR RESPONSE	“arrhythmia”
ATRIAL FIBRILLATION WITH RAPID VENTRICULAR RESPONSE WITH ABERRANT	“arrhythmia”
CONDUCTION OR VENTRICULAR PREMATURE COMPLEXES	“arrhythmia”
ATRIAL FIBRILLATION WITH SLOW VENTRICULAR RESPONSE	“arrhythmia”
ATRIAL FIBRILLATION WITH SLOW VENTRICULAR RESPONSE WITH ABERRANT CONDUCTION OR VENTRICULAR PREMATURE COMPLEXES	“arrhythmia”
ATRIAL FLUTTER/TACHYCARDIA	“arrhythmia”

ATRIAL FLUTTER/TACHYCARDIA WITH RAPID VENTRICULAR RESPONSE	"arrhythmia"
ATRIAL FLUTTER/TACHYCARDIA WITH SLOW VENTRICULAR RESPONSE WITH ABERRANT	"arrhythmia"
CONDUCTION OR VENTRICULAR PREMATURE COMPLEXES	"arrhythmia"
ATYPICAL ECG	unclassified
BORDERLINE ECG	unclassified
BORDERLINE LEFT AXIS DEVIATION	unclassified
BORDERLINE RIGHT AXIS DEVIATION	unclassified
DEXTROCARDIA	unclassified
EARLY REPOLARIZATION	"arrhythmia"
ECTOPIC ATRIAL BRADYCARDIA	"arrhythmia"
ECTOPIC ATRIAL RHYTHM	"arrhythmia"
ECTOPIC ATRIAL RHYTHM WITH FREQUENT VENTRICULAR PREMATURE COMPLEXES IN A BIGEMINAL PATTERN	"arrhythmia"
ECTOPIC ATRIAL RHYTHM WITH OCCASIONAL SUPRAVENTRICULAR PREMATURE COMPLEXES	"arrhythmia"
ECTOPIC ATRIAL RHYTHM WITH OCCASIONAL VENTRICULAR PREMATURE COMPLEXES	"arrhythmia"
ECTOPIC ATRIAL RHYTHM WITH PROLONGED PR INTERVAL	"arrhythmia"
ECTOPIC ATRIAL RHYTHM WITH SHORT PR INTERVAL	"arrhythmia"
ECTOPIC ATRIAL RHYTHM WITH SHORT PR INTERVAL WITH FREQUENT SUPRAVENTRICULAR PREMATURE COMPLEXES	"arrhythmia"
ECTOPIC ATRIAL RHYTHM WITH SHORT PR INTERVAL WITH FREQUENT VENTRICULAR PREMATURE COMPLEXES	"arrhythmia"
ECTOPIC ATRIAL TACHYCARDIA	"arrhythmia"
ECTOPIC ATRIAL TACHYCARDIA, POSSIBLE ATRIAL FLUTTER	"arrhythmia"
ELECTRONIC ATRIAL PACEMAKER	unclassified
ELECTRONIC VENTRICULAR PACEMAKER	unclassified
ELECTRONIC VENTRICULAR PACEMAKER – CONTOUR ANALYSIS BASED ON INTRINSIC RHYTHM	unclassified
INCOMPLETE RIGHT BUNDLE BRANCH BLOCK	"arrhythmia"
INDETERMINATE AXIS	unclassified
INFERIOR MYOCARDIAL INFARCTION , OF INDETERMINATE AGE	"ischaemia"
INFERIOR MYOCARDIAL INFARCTION , OF INDETERMINATE AGE WITH POSTERIOR EXTENSION [PROMINENT R WAVE IN V1/V2]	"ischaemia"
INFERIOR MYOCARDIAL INFARCTION , POSSIBLY ACUTE	"ischaemia"
INFERIOR MYOCARDIAL INFARCTION , PROBABLY OLD	"ischaemia"
INFERIOR MYOCARDIAL INFARCTION , PROBABLY OLD WITH POSTERIOR EXTENSION [PROMINENT R WAVE IN V1/V2]	"ischaemia"
INFERIOR MYOCARDIAL INFARCTION , PROBABLY RECENT	"ischaemia"
INTERMITTENT VENTRICULAR PREEXCITATION/WPW	"arrhythmia"
INTERPRETATION BASED ON A DEFAULT AGE OF 40 YEARS	unclassified
INTRAVENTRICULAR CONDUCTION DELAY	"arrhythmia"
JUNCTIONAL BRADYCARDIA	"arrhythmia"
JUNCTIONAL RHYTHM	"arrhythmia"
JUNCTIONAL RHYTHM WITH OCCASIONAL SUPRAVENTRICULAR PREMATURE COMPLEXES	"arrhythmia"
JUNCTIONAL RHYTHM WITH OCCASIONAL VENTRICULAR PREMATURE COMPLEXES	"arrhythmia"
JUNCTIONAL ST DEPRESSION, CONSIDER NORMAL VARIANT	unclassified
JUNCTIONAL TACHYCARDIA	"arrhythmia"
LATERAL MYOCARDIAL INFARCTION , OF INDETERMINATE AGE	"ischaemia"
LATERAL MYOCARDIAL INFARCTION , PROBABLY OLD	"ischaemia"
LATERAL MYOCARDIAL INFARCTION , PROBABLY RECENT	"ischaemia"

LEFT ANTERIOR FASCICULAR BLOCK	"arrhythmia"
LEFT ATRIAL ENLARGEMENT	"hypertrophy"
LEFT AXIS DEVIATION	unclassified
LEFT BUNDLE BRANCH BLOCK	"arrhythmia"
LEFT POSTERIOR FASCICULAR BLOCK	"arrhythmia"
LEFT VENTRICULAR "hypertrophy" AND ST-T CHANGE	"hypertrophy"
LOW QRS VOLTAGE	unclassified
LOW QRS VOLTAGE IN EXTREMITY LEADS	unclassified
LOW QRS VOLTAGE IN PRECORDIAL LEADS	unclassified
MARKED LEFT AXIS DEVIATION	unclassified
MARKED RIGHT AXIS DEVIATION	unclassified
MARKED ST DEPRESSION, CONSIDER SUBENDOCARDIAL INJURY	"ischaemia"
MARKED ST ELEVATION, CONSIDER ANTERIOR INJURY	"ischaemia"
MARKED ST ELEVATION, CONSIDER ANTEROLATERAL INJURY	"ischaemia"
MARKED ST ELEVATION, CONSIDER INFERIOR INJURY	"ischaemia"
MARKED ST ELEVATION, CONSIDER LATERAL INJURY	"ischaemia"
MARKED ST ELEVATION, CONSIDER SEPTAL INJURY	"ischaemia"
MARKED T-WAVE ABNORMALITY, CONSIDER ANTERIOR "ischaemia"	"ischaemia"
MARKED T-WAVE ABNORMALITY, CONSIDER ANTEROLATERAL "ischaemia"	"ischaemia"
MARKED T-WAVE ABNORMALITY, CONSIDER INFERIOR "ischaemia"	"ischaemia"
MARKED T-WAVE ABNORMALITY, CONSIDER LATERAL "ischaemia"	"ischaemia"
MINIMAL ST DEPRESSION	unclassified
MINIMAL VOLTAGE CRITERIA FOR LVH, CONSIDER NORMAL VARIANT	unclassified
MODERATE INTRAVENTRICULAR CONDUCTION DELAY	"arrhythmia"
MODERATE ST DEPRESSION	"ischaemia"
MODERATE T-WAVE ABNORMALITY, CONSIDER ANTERIOR "ischaemia"	"ischaemia"
MODERATE T-WAVE ABNORMALITY, CONSIDER ANTEROLATERAL "ischaemia"	"ischaemia"
MODERATE T-WAVE ABNORMALITY, CONSIDER INFERIOR "ischaemia"	"ischaemia"
MODERATE T-WAVE ABNORMALITY, CONSIDER LATERAL "ischaemia"	"ischaemia"
MODERATE VOLTAGE CRITERIA FOR LVH, CONSIDER NORMAL VARIANT	unclassified
NO FURTHER INTERPRETATION POSSIBLE	unclassified
NONSPECIFIC ST & T-WAVE ABNORMALITY	"ischaemia"
NONSPECIFIC ST ELEVATION	"ischaemia"
NONSPECIFIC T-WAVE ABNORMALITY	"ischaemia"
NORMAL ECG	normal
PATTERN CONSISTENT WITH PULMONARY DISEASE	unclassified
POSSIBLE ANTERIOR MYOCARDIAL INFARCTION , OF INDETERMINATE AGE	"ischaemia"
POSSIBLE ANTERIOR MYOCARDIAL INFARCTION , PROBABLY OLD	"ischaemia"
POSSIBLE ANTEROLATERAL MYOCARDIAL INFARCTION , OF INDETERMINATE AGE	"ischaemia"
POSSIBLE ANTEROSEPTAL MYOCARDIAL INFARCTION , OF INDETERMINATE AGE	"ischaemia"
POSSIBLE ANTEROSEPTAL MYOCARDIAL INFARCTION , POSSIBLY ACUTE	"ischaemia"
POSSIBLE INFERIOR MYOCARDIAL INFARCTION , OF INDETERMINATE AGE	"ischaemia"
POSSIBLE INFERIOR MYOCARDIAL INFARCTION , OF INDETERMINATE AGE WITH POSTERIOR EXTENSION [PROMINENT R WAVE IN V1/V2]	"ischaemia"

POSSIBLE INFERIOR MYOCARDIAL INFARCTION , PROBABLY OLD	“ischaemia”
POSSIBLE INFERIOR MYOCARDIAL INFARCTION , PROBABLY OLD WITH POSTERIOR EXTENSION [PROMINENT R WAVE IN V1/V2]	“ischaemia”
POSSIBLE LATERAL MYOCARDIAL INFARCTION , OF INDETERMINATE AGE	“ischaemia”
POSSIBLE LATERAL MYOCARDIAL INFARCTION , PROBABLY OLD	“ischaemia”
POSSIBLE LEFT ATRIAL ENLARGEMENT	“hypertrophy”
POSSIBLE LEFT VENTRICULAR “hypertrophy”	“hypertrophy”
POSSIBLE RIGHT ATRIAL ENLARGEMENT	“hypertrophy”
POSSIBLE RIGHT VENTRICULAR CONDUCTION DELAY	“arrhythmia”
POSSIBLE RIGHT VENTRICULAR “hypertrophy”	“hypertrophy”
POSSIBLE SEPTAL MYOCARDIAL INFARCTION , OF INDETERMINATE AGE	“ischaemia”
POSSIBLE SEPTAL MYOCARDIAL INFARCTION , POSSIBLY ACUTE	“ischaemia”
POSSIBLE SEPTAL MYOCARDIAL INFARCTION , PROBABLY OLD	“ischaemia”
PROBABLE ANTERIOR MYOCARDIAL INFARCTION , OF INDETERMINATE AGE	“ischaemia”
PROBABLE ANTEROLATERAL MYOCARDIAL INFARCTION , OF INDETERMINATE AGE	“ischaemia”
PROBABLE ANTEROSEPTAL MYOCARDIAL INFARCTION , PROBABLY RECENT	“ischaemia”
PROBABLE INFERIOR MYOCARDIAL INFARCTION , OF INDETERMINATE AGE	“ischaemia”
PROBABLE INFERIOR MYOCARDIAL INFARCTION , OF INDETERMINATE AGE WITH POSTERIOR EXTENSION [PROMINENT R WAVE IN V1/V2]	“ischaemia”
PROBABLE INFERIOR MYOCARDIAL INFARCTION , PROBABLY OLD	“ischaemia”
PROBABLE INFERIOR MYOCARDIAL INFARCTION , PROBABLY OLD WITH POSTERIOR EXTENSION [PROMINENT R WAVE IN V1/V2]	“ischaemia”
PROBABLE LATERAL MYOCARDIAL INFARCTION , OF INDETERMINATE AGE	“ischaemia”
PROBABLE LATERAL MYOCARDIAL INFARCTION , PROBABLY OLD	“ischaemia”
PROBABLE RIGHT VENTRICULAR “hypertrophy”	“hypertrophy”
PROBABLE SEPTAL MYOCARDIAL INFARCTION , OF INDETERMINATE AGE	“ischaemia”
PROLONGED QT INTERVAL	unclassified
RIGHT ATRIAL ENLARGEMENT	“hypertrophy”
RIGHT BUNDLE BRANCH BLOCK	arrythmia
RIGHT BUNDLE BRANCH BLOCK AND POSSIBLE RIGHT VENTRICULAR “hypertrophy”	“hypertrophy”
RIGHT VENTRICULAR “hypertrophy”	hypertropjy
RIGHT VENTRICULAR “hypertrophy” AND ST-T CHANGE	“hypertrophy”
S1-S2-S3 PATTERN, CONSISTENT WITH PULMONARY DISEASE, RVH, OR NORMAL VARIANT	
unclassified	
SEPTAL MYOCARDIAL INFARCTION , OF INDETERMINATE AGE	“ischaemia”
SEPTAL MYOCARDIAL INFARCTION , POSSIBLY ACUTE	“ischaemia”
SEPTAL MYOCARDIAL INFARCTION , PROBABLY OLD	“ischaemia”
SEPTAL MYOCARDIAL INFARCTION , PROBABLY RECENT	“ischaemia”
SINUS BRADYCARDIA	unclassified
SINUS BRADYCARDIA WITH 2ND DEGREE AV BLOCK, MOBITZ TYPE II	arrythmia
SINUS BRADYCARDIA WITH FREQUENT SUPRAVENTRICULAR PREMATURE COMPLEXES	arrythmia

SINUS BRADYCARDIA WITH MARKED RHYTHM IRREGULARITY, POSSIBLE NON-CONDUCTED PAC, SA BLOCK, AV BLOCK, OR SINUS PAUSE	arrythmia
SINUS BRADYCARDIA WITH MARKED SINUS "arrhythmia"	arrythmia
SINUS BRADYCARDIA WITH OCCASIONAL ECTOPIC PREMATURE COMPLEXES	arrythmia
SINUS BRADYCARDIA WITH OCCASIONAL SUPRAVENTRICULAR PREMATURE COMPLEXES	arrythmia
SINUS BRADYCARDIA WITH OCCASIONAL SUPRAVENTRICULAR PREMATURE COMPLEXES IN A BIGEMINAL PATTERN	arrythmia
SINUS BRADYCARDIA WITH OCCASIONAL VENTRICULAR PREMATURE COMPLEXES	arrythmia
SINUS BRADYCARDIA WITH OCCASIONAL VENTRICULAR PREMATURE COMPLEXES WITH FREQUENT SUPRAVENTRICULAR PREMATURE COMPLEXES	arrythmia
SINUS BRADYCARDIA WITH PROLONGED PR INTERVAL	arrythmia
SINUS BRADYCARDIA WITH PROLONGED PR INTERVAL WITH OCCASIONAL SUPRAVENTRICULAR PREMATURE COMPLEXES	arrythmia
SINUS BRADYCARDIA WITH SHORT PR INTERVAL	arrythmia
SINUS BRADYCARDIA WITH SINUS "arrhythmia"	arrythmia
SINUS BRADYCARDIA WITH SINUS "arrhythmia" WITH PROLONGED PR INTERVAL	arrythmia
SINUS BRADYCARDIA WITH SINUS "arrhythmia" WITH SHORT PR INTERVAL	arrythmia
SINUS RHYTHM	unclassified
SINUS RHYTHM WITH 2ND DEGREE AV BLOCK, MOBITZ TYPE I (WENCKEBACH)	arrythmia
SINUS RHYTHM WITH 2ND DEGREE AV BLOCK, MOBITZ TYPE II	arrythmia
SINUS RHYTHM WITH FREQUENT ECTOPIC PREMATURE COMPLEXES	arrythmia
SINUS RHYTHM WITH FREQUENT ECTOPIC PREMATURE COMPLEXES IN A BIGEMINAL PATTERN	arrythmia
SINUS RHYTHM WITH FREQUENT SUPRAVENTRICULAR PREMATURE COMPLEXES	arrythmia
SINUS RHYTHM WITH FREQUENT SUPRAVENTRICULAR PREMATURE COMPLEXES IN A BIGEMINAL PATTERN	arrythmia
SINUS RHYTHM WITH FREQUENT VENTRICULAR PREMATURE COMPLEXES	arrythmia
SINUS RHYTHM WITH FREQUENT VENTRICULAR PREMATURE COMPLEXES IN A BIGEMINAL PATTERN	arrythmia
SINUS RHYTHM WITH HIGH GRADE AV BLOCK	arrythmia
SINUS RHYTHM WITH MARKED RHYTHM IRREGULARITY, POSSIBLE NON-CONDUCTED PAC, SA BLOCK, AV BLOCK, OR SINUS PAUSE	arrythmia
SINUS RHYTHM WITH MARKED SINUS "arrhythmia"	arrythmia
SINUS RHYTHM WITH MARKED SINUS "arrhythmia" WITH PROLONGED PR INTERVAL	arrythmia
SINUS RHYTHM WITH MARKED SINUS "arrhythmia" WITH SHORT PR INTERVAL	arrythmia
SINUS RHYTHM WITH OCCASIONAL ECTOPIC PREMATURE COMPLEXES	arrythmia
SINUS RHYTHM WITH OCCASIONAL SUPRAVENTRICULAR PREMATURE COMPLEXES	arrythmia
SINUS RHYTHM WITH OCCASIONAL VENTRICULAR PREMATURE COMPLEXES	arrythmia

SINUS RHYTHM WITH OCCASIONAL VENTRICULAR PREMATURE COMPLEXES WITH FREQUENT SUPRAVENTRICULAR PREMATURE COMPLEXES	arrythmia
SINUS RHYTHM WITH OCCASIONAL VENTRICULAR PREMATURE COMPLEXES WITH MARKED RHYTHM IRREGULARITY, POSSIBLE NON-CONDUCTED PAC, SA BLOCK, AV BLOCK, OR SINUS PAUSE	arrythmia
SINUS RHYTHM WITH OCCASIONAL VENTRICULAR PREMATURE COMPLEXES WITH OCCASIONAL SUPRAVENTRICULAR PREMATURE COMPLEXES	arrythnia
SINUS RHYTHM WITH PROLONGED PR INTERVAL	unclassified
SINUS RHYTHM WITH PROLONGED PR INTERVAL WITH FREQUENT SUPRAVENTRICULAR PREMATURE COMPLEXES	“arrhythmia”
SINUS RHYTHM WITH PROLONGED PR INTERVAL WITH FREQUENT VENTRICULAR PREMATURE COMPLEXES	“arrhythmia”
SINUS RHYTHM WITH PROLONGED PR INTERVAL WITH FREQUENT VENTRICULAR PREMATURE COMPLEXES IN A BIGEMINAL PATTERN	“arrhythmia”
SINUS RHYTHM WITH PROLONGED PR INTERVAL WITH OCCASIONAL SUPRAVENTRICULAR PREMATURE COMPLEXES	“arrhythmia”
SINUS RHYTHM WITH PROLONGED PR INTERVAL WITH OCCASIONAL VENTRICULAR PREMATURE COMPLEXES	“arrhythmia”
SINUS RHYTHM WITH PROLONGED PR INTERVAL WITH OCCASIONAL VENTRICULAR PREMATURE COMPLEXES WITH OCCASIONAL SUPRAVENTRICULAR PREMATURE COMPLEXES	“arrhythmia”
SINUS RHYTHM WITH SHORT PR INTERVAL	unclassified
SINUS RHYTHM WITH SHORT PR INTERVAL WITH FREQUENT SUPRAVENTRICULAR PREMATURE COMPLEXES	“arrhythmia”
SINUS RHYTHM WITH SHORT PR INTERVAL WITH FREQUENT VENTRICULAR PREMATURE COMPLEXES	“arrhythmia”
SINUS RHYTHM WITH SHORT PR INTERVAL WITH OCCASIONAL ECTOPIC PREMATURE COMPLEXES	“arrhythmia”
SINUS RHYTHM WITH SHORT PR INTERVAL WITH OCCASIONAL SUPRAVENTRICULAR PREMATURE COMPLEXES	“arrhythmia”
SINUS RHYTHM WITH SHORT PR INTERVAL WITH OCCASIONAL VENTRICULAR PREMATURE COMPLEXES	“arrhythmia”
SINUS RHYTHM WITH SHORT PR INTERVAL WITH OCCASIONAL VENTRICULAR PREMATURE COMPLEXES WITH OCCASIONAL SUPRAVENTRICULAR PREMATURE COMPLEXES	“arrhythmia”
SINUS RHYTHM WITH SINUS “arrhythmia”	“arrhythmia”
SINUS RHYTHM WITH SINUS “arrhythmia” WITH PROLONGED PR INTERVAL	“arrhythmia”
SINUS RHYTHM WITH SINUS “arrhythmia” WITH SHORT PR INTERVAL	“arrhythmia”
SINUS TACHYCARDIA	unclassified
SINUS TACHYCARDIA WITH 2ND DEGREE AV BLOCK, MOBITZ TYPE II	“arrhythmia”
SINUS TACHYCARDIA WITH FREQUENT ECTOPIC PREMATURE COMPLEXES	“arrhythmia”
SINUS TACHYCARDIA WITH FREQUENT SUPRAVENTRICULAR PREMATURE COMPLEXES	“arrhythmia”
SINUS TACHYCARDIA WITH FREQUENT VENTRICULAR PREMATURE COMPLEXES	“arrhythmia”
SINUS TACHYCARDIA WITH OCCASIONAL ECTOPIC PREMATURE COMPLEXES	“arrhythmia”
SINUS TACHYCARDIA WITH OCCASIONAL SUPRAVENTRICULAR PREMATURE COMPLEXES	“arrhythmia”

SINUS TACHYCARDIA WITH OCCASIONAL VENTRICULAR PREMATURE COMPLEXES	"arrhythmia"
SINUS TACHYCARDIA WITH OCCASIONAL VENTRICULAR PREMATURE COMPLEXES WITH OCCASIONAL SUPRAVENTRICULAR PREMATURE COMPLEXES	"arrhythmia"
SINUS TACHYCARDIA WITH PROLONGED PR INTERVAL	unclassified
SINUS TACHYCARDIA WITH PROLONGED PR INTERVAL WITH OCCASIONAL SUPRAVENTRICULAR PREMATURE COMPLEXES	"arrhythmia"
SINUS TACHYCARDIA WITH SHORT PR INTERVAL	"arrhythmia"
SINUS TACHYCARDIA WITH SHORT PR INTERVAL WITH FREQUENT SUPRAVENTRICULAR PREMATURE COMPLEXES	"arrhythmia"
SINUS TACHYCARDIA WITH SHORT PR INTERVAL WITH OCCASIONAL SUPRAVENTRICULAR PREMATURE COMPLEXES	"arrhythmia"
ST DEPRESSION, CONSIDER SUBENDOCARDIAL INJURY	"ischaemia"
ST DEVIATION AND MARKED T-WAVE ABNORMALITY, CONSIDER ANTERIOR "ischaemia"	"ischaemia"
ST DEVIATION AND MARKED T-WAVE ABNORMALITY, CONSIDER ANTEROLATERAL "ischaemia"	"ischaemia"
ST DEVIATION AND MARKED T-WAVE ABNORMALITY, CONSIDER LATERAL "ischaemia"	"ischaemia"
ST DEVIATION AND MODERATE T-WAVE ABNORMALITY, CONSIDER ANTERIOR "ischaemia"	"ischaemia"
ST DEVIATION AND MODERATE T-WAVE ABNORMALITY, CONSIDER ANTEROLATERAL "ischaemia"	"ischaemia"
ST DEVIATION AND MODERATE T-WAVE ABNORMALITY, CONSIDER INFERIOR "ischaemia"	"ischaemia"
ST DEVIATION AND MODERATE T-WAVE ABNORMALITY, CONSIDER LATERAL "ischaemia"	"ischaemia"
ST ELEVATION CONSISTENT WITH INJURY, PERICARDITIS, OR EARLY REPOLARIZATION	unclassified
ST ELEVATION, CONSIDER ANTERIOR INJURY	"ischaemia"
ST ELEVATION, CONSIDER ANTEROSEPTAL INJURY	"ischaemia"
ST ELEVATION, CONSIDER INFERIOR INJURY	"ischaemia"
ST ELEVATION, CONSIDER LATERAL INJURY	"ischaemia"
ST ELEVATION, CONSIDER SEPTAL INJURY	"ischaemia"
ST ELEVATION, PROBABLY EARLY REPOLARIZATION	"arrhythmia"
SUPRAVENTRICULAR BRADYCARDIA	"arrhythmia"
SUPRAVENTRICULAR RHYTHM	"arrhythmia"
SUPRAVENTRICULAR TACHYCARDIA	"arrhythmia"
TALL T-WAVES, SUGGESTS HYPERKALEMIA	unclassified
TYPE 2 BRUGADA PATTERN (NON-DIAGNOSTIC)	unclassified
TYPE 3 BRUGADA PATTERN (NON-DIAGNOSTIC)	unclassified
UNCERTAIN IRREGULAR RHYTHM	"ischaemia"
UNCERTAIN REGULAR RHYTHM	unclassified
UNCONFIRMED REPORT	unclassified
VENTRICULAR PREEXCITATION/WPW	"arrhythmia"
VOLTAGE CRITERIA FOR LVH	"hypertrophy"
WARNING: DATA QUALITY MAY AFFECT INTERPRETATION	unclassified

Table F.1: Mapping from Mortara labels to normal, ""arrhythmia"", "ischaemia", and "hypertrophy" classes

References

Arsanjani, Reza, Damini Dey, et al. (2015). "Prediction of revascularization after myocardial perfusion SPECT by machine learning in a large population". In: *Journal of Nuclear Cardiology* 22.5, pp. 877–884.

Association, American Diabetes et al. (2007). "Reduction in weight and cardiovascular disease risk factors in individuals with type 2 diabetes: one-year results of the look AHEAD trial". In: *Diabetes care* 30.6, pp. 1374–1383.

Dekker, Jacqueline M et al. (2005). "Metabolic syndrome and 10-year cardiovascular disease risk in the Hoorn Study". In: *Circulation* 112.5, pp. 666–673.

Tracy, Russell P et al. (1997). "Lifetime smoking exposure affects the association of C-reactive protein with cardiovascular disease risk factors and subclinical disease in healthy elderly subjects". In: *Arteriosclerosis, thrombosis, and vascular biology* 17.10, pp. 2167–2176.

Colombet, Isabelle et al. (2000). "Models to predict cardiovascular risk: comparison of CART, multilayer perceptron and logistic regression." In: *Proceedings of the AMIA Symposium*. American Medical Informatics Association, p. 156.

Gamberger, Dragan, Nada Lavrač, and Goran Krstačić (2003). "Active subgroup mining: a case study in coronary heart disease risk group detection". In: *Artificial Intelligence in Medicine* 28.1, pp. 27–57.

Kannel, William B, Tavia Gordon, and Dorsey Offutt (1969). "Left ventricular hypertrophy by electrocardiogram: prevalence, incidence, and mortality in the Framingham study". In: *Annals of internal medicine* 71.1, pp. 89–105.

Kannel, William B, Keaven Anderson, et al. (1987). "Nonspecific electrocardiographic abnormality as a predictor of coronary heart disease: the Framingham Study". In: *American heart journal* 113.2, pp. 370–376.

Savonitto, Stefano et al. (1999). "Prognostic value of the admission electrocardiogram in acute coronary syndromes". In: *Jama* 281.8, pp. 707–713.

Prineas, Ronald J, Richard S Crow, and Zhu-Ming Zhang (2009). *The Minnesota code manual of electrocardiographic findings*. Springer Science & Business Media.

Syed, Zeeshan et al. (2011). "Computationally generated cardiac biomarkers for risk stratification after acute coronary syndrome". In: *Science translational medicine* 3.102, 102ra95–102ra95.

Vapnik, Vladimir (2013). *The nature of statistical learning theory*. Springer science & business media.

Consortium, International Schizophrenia (2009). "Common polygenic variation contributes to risk of schizophrenia that overlaps with bipolar disorder". In: *Nature* 460.7256, p. 748.

Kubo, Michiaki et al. (2007). "A nonsynonymous SNP in PRKCH (protein kinase C η) increases the risk of cerebral infarction". In: *Nature genetics* 39.2, p. 212.

Wasan, PS et al. (2013). "Application of statistics and machine learning for risk stratification of heritable cardiac arrhythmias". In: *Expert Systems with Applications* 40.7, pp. 2476–2486.

Hill, Jonathan M et al. (2003). "Circulating endothelial progenitor cells, vascular function, and cardiovascular risk". In: *New England Journal of Medicine* 348.7, pp. 593–600.

McKinney, Brett A et al. (2006). "Machine learning for detecting gene-gene interactions". In: *Applied bioinformatics* 5.2, pp. 77–88.

Oresko, Joseph J et al. (2010). "A wearable smartphone-based platform for real-time cardiovascular disease detection via electrocardiogram processing". In: *IEEE Transactions on Information Technology in Biomedicine* 14.3, pp. 734–740.

Rajkumar, Asha and G Sophia Reena (2010). "Diagnosis of heart disease using datamining algorithm". In: *Global journal of computer science and technology* 10.10, pp. 38–43.

Katritsis, Demosthenes G et al. (2013). *Clinical cardiology: current practice guidelines*. Oxford University Press.

Knuiman, Matthew W and HT Vu (1997). "Prediction of coronary heart disease mortality in Busselton, Western Australia: an evaluation of the Framingham, national health epidemiologic follow up study, and WHO ERICA risk scores." In: *Journal of Epidemiology & Community Health* 51.5, pp. 515–519.

Lapuerta, Pablo, Stanley P Azen, and Laurie LaBree (1995). "Use of neural networks in predicting the risk of coronary artery disease". In: *Computers and Biomedical Research* 28.1, pp. 38–52.

Das, Resul, Ibrahim Turkoglu, and Abdulkadir Sengur (2009). "Effective diagnosis of heart disease through neural networks ensembles". In: *Expert systems with applications* 36.4, pp. 7675–7680.

Berikol, Göksu Bozdereli, Oktay Yildiz, and İ Türkay Özcan (2016). "Diagnosis of acute coronary syndrome with a support vector machine". In: *Journal of medical systems* 40.4, p. 84.

Alickovic, Emina and Abdulhamit Subasi (2015). "Effect of multiscale PCA de-noising in ECG beat classification for diagnosis of cardiovascular diseases". In: *Circuits, Systems, and Signal Processing* 34.2, pp. 513–533.

Mitra, Malay and RK Samanta (2013). "Cardiac arrhythmia classification using neural networks with selected features". In: *Procedia Technology* 10, pp. 76–84.

Homaeinezhad, Mohammad R et al. (2012). "ECG arrhythmia recognition via a neuro-SVM–KNN hybrid classifier with virtual QRS image-based geometrical features". In: *Expert Systems with Applications* 39.2, pp. 2047–2058.

Übeyli, Elif Derya (2008a). "Support vector machines for detection of electrocardiographic changes in partial epileptic patients". In: *Engineering Applications of Artificial Intelligence* 21.8, pp. 1196–1203.

Özdemir, Ahmet Turan and Billur Barshan (2014). "Detecting falls with wearable sensors using machine learning techniques". In: *Sensors* 14.6, pp. 10691–10708.

Kim, Jinkwon et al. (2009). "Robust algorithm for arrhythmia classification in ECG using extreme learning machine". In: *Biomedical engineering online* 8.1, p. 31.

Li, Qiao, Cadathur Rajagopalan, and Gari D Clifford (2013). "Ventricular fibrillation and tachycardia classification using a machine learning approach". In: *IEEE Transactions on Biomedical Engineering* 61.6, pp. 1607–1613.

Karpagachelvi, S, M Arthanari, and M Sivakumar (2011). "Classification of ECG signals using extreme learning machine". In: *computer and information science* 4.1, p. 42.

Yu, Chenggang et al. (2006). "A method for automatic identification of reliable heart rates calculated from ECG and PPG waveforms". In: *Journal of the American Medical Informatics Association* 13.3, pp. 309–320.

Khandoker, Ahsan H, Jayavardhana Gubbi, and Marimuthu Palaniswami (2009). "Automated scoring of obstructive sleep apnea and hypopnea events using short-term electrocardiogram recordings". In: *IEEE Transactions on Information Technology in Biomedicine* 13.6, pp. 1057–1067.

Kampouraki, Argyro, George Manis, and Christophoros Nikou (2008). "Heartbeat time series classification with support vector machines". In: *IEEE transactions on information technology in biomedicine* 13.4, pp. 512–518.

Bsoul, Majdi, Hlaing Minn, and Lakshman Tamil (2010). "Apnea MedAssist: real-time sleep apnea monitor using single-lead ECG". In: *IEEE Transactions on Information Technology in Biomedicine* 15.3, pp. 416–427.

Monte-Moreno, Enric (2011). "Non-invasive estimate of blood glucose and blood pressure from a photoplethysmograph by means of machine learning techniques". In: *Artificial intelligence in medicine* 53.2, pp. 127–138.

Tantimongcolwat, Tanawut et al. (2008). "Identification of ischemic heart disease via machine learning analysis on magnetocardiograms". In: *Computers in biology and medicine* 38.7, pp. 817–825.

Sun, Yuwen and Allen C Cheng (2012). "Machine learning on-a-chip: A high-performance low-power reusable neuron architecture for artificial neural networks in ECG classifications". In: *Computers in biology and medicine* 42.7, pp. 751–757.

Zavar, M et al. (2011). "Evolutionary model selection in a wavelet-based support vector machine for automated seizure detection". In: *Expert Systems with Applications* 38.9, pp. 10751–10758.

Hsich, Eileen et al. (2011). "Identifying important risk factors for survival in patient with systolic heart failure using random survival forests". In: *Circulation: Cardiovascular Quality and Outcomes* 4.1, pp. 39–45.

Luz, Eduardo José da S et al. (2013). "ECG arrhythmia classification based on optimum-path forest". In: *Expert Systems with Applications* 40.9, pp. 3561–3573.

Arsanjani, Reza, Yuan Xu, et al. (2013). "Improved accuracy of myocardial perfusion SPECT for detection of coronary artery disease by machine learning in a large population". In: *Journal of Nuclear Cardiology* 20.4, pp. 553–562.

Pantelopoulos, Alexandros and Nikolaos G Bourbakis (2010). "Prognosis—a wearable health-monitoring system for people at risk: Methodology and modeling". In: *IEEE Transactions on Information Technology in Biomedicine* 14.3, pp. 613–621.

Kurz, David J et al. (2009). "Simple point-of-care risk stratification in acute coronary syndromes: the AMIS model". In: *Heart* 95.8, pp. 662–668.

El-Dahshan, El-Sayed A (2011). "Genetic algorithm and wavelet hybrid scheme for ECG signal denoising". In: *Telecommunication Systems* 46.3, pp. 209–215.

Übeyli, Elif Derya (2008b). "Eigenvector methods for automated detection of electrocardiographic changes in partial epileptic patients". In: *IEEE Transactions on Information Technology in Biomedicine* 13.4, pp. 478–485.

Vaswani, A et al. (2015). *Cardiology in a Heartbeat*. Scion Publishing Limited. URL: `https://books.google.co.uk/books?id=N5lWjgEACAAJ`.

Vaswani, Ashish et al. (2017). "Attention is all you need". In: *Advances in neural information processing systems*, pp. 5998–6008.

Chudáček, Václav et al. (2009). "Examining cross-database global training to evaluate five different methods for ventricular beat classification". In: *Physiological measurement* 30.7, p. 661.

Leite, Cicılia RM et al. (2010). "Classification of cardiac arrhythmias using competitive networks". In: *2010 Annual International Conference of the IEEE Engineering in Medicine and Biology*. IEEE, pp. 1386–1389.

Yaghouby, Farid et al. (2010). "Towards automatic detection of atrial fibrillation: A hybrid computational approach". In: *Computers in Biology and Medicine* 40.11-12, pp. 919–930.

Osowski, Stanislaw, Krzysztof Siwek, and Robert Siroic (2011). "Neural system for heartbeats recognition using genetically integrated ensemble of classifiers". In: *Computers in Biology and Medicine* 41.3, pp. 173–180.

Kostka, Pawel S and Ewaryst J Tkacz (2011). "Feature extraction in time-frequency signal analysis by means of matched wavelets as a feature generator". In: *2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. IEEE, pp. 4996–4999.

Haseena, Hassan Hamsa, Abraham T Mathew, and Joseph K Paul (2011). "Fuzzy clustered probabilistic and multi layered feed forward neural networks for electrocardiogram arrhythmia classification". In: *Journal of Medical Systems* 35.2, pp. 179–188.

Haseena, Hassan H, Paul K Joseph, and Abraham T Mathew (2011). "Classification of arrhythmia using hybrid networks". In: *Journal of medical systems* 35.6, pp. 1617–1630.

Javadi, Mehrdad et al. (2011). "Improving ECG classification accuracy using an ensemble of neural network modules". In: *PLoS one* 6.10, e24386.

Nejadgholi, Isar, Mohammad Hasan Moradi, and Fatemeh Abdolali (2011). "Using phase space reconstruction for patient independent heartbeat classification in comparison with some benchmark methods". In: *Computers in biology and medicine* 41.6, pp. 411–419.

Martis, Roshan Joy et al. (2012). "Automated screening of arrhythmia using wavelet based machine learning techniques". In: *Journal of medical systems* 36.2, pp. 677–688.

Benali, Radhwane, Fethi Bereksi Reguig, and Zinedine Hadj Slimane (2012). "Automatic classification of heartbeats using wavelet neural network". In: *Journal of medical systems* 36.2, pp. 883–892.

Chikh, Mohammed Amine, Mohammed Ammar, and Radja Marouf (2012). "A neuro-fuzzy identification of ECG beats". In: *Journal of medical systems* 36.2, pp. 903–914.

Chen, Ying-Hsiang and Sung-Nien Yu (2012). "Selection of effective features for ECG beat recognition based on nonlinear correlations". In: *Artificial intelligence in medicine* 54.1, pp. 43–52.

Chakroborty, Sandipan (2013). "Accurate Arrhythmia classification using auto-associative neural network". In: *2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE, pp. 4247–4250.

Liu, Shing-Hong, Da-Chuan Cheng, and Chih-Ming Lin (2013). "Arrhythmia identification with two-lead electrocardiograms using artificial neural networks and support vector machines for a portable ECG monitor system". In: *Sensors* 13.1, pp. 813–828.

Prasad, Hari et al. (2013). "Application of higher order spectra for accurate delineation of atrial arrhythmia". In: *2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE, pp. 57–60.

Javadi, Mehrdad (2013). "Combining neural networks and ANFIS classifiers for supervised examining of electrocardiogram beats". In: *Journal of medical engineering & technology* 37.8, pp. 484–497.

Yu, Sung-Nien and Fan-Tsen Liu (2014). "Subband higher-order statistics and cross-correlation for heartbeat type recognition based on two-lead electrocardiogram". In: *2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. IEEE, pp. 42–45.

Kiranyaz, Serkan, Turker Ince, Ridha Hamila, et al. (2015). "Convolutional neural networks for patient-specific ecg classification". In: *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE, pp. 2608–2611.

Poddar, Monappa Gundappa, Vinod Kumar, and Yash Paul Sharma (2015). "Automated diagnosis of coronary artery diseased patients by heart rate variability analysis using linear and non-linear methods". In: *Journal of medical engineering & technology* 39.6, pp. 331–341.

Yang, Jianli et al. (2015). "A novel method of diagnosing premature ventricular contraction based on sparse auto-encoder and softmax regression". In: *Bio-medical materials and engineering* 26.s1, S1549–S1558.

Kiranyaz, Serkan, Turker Ince, and Moncef Gabbouj (2015). "Real-time patient-specific ECG classification by 1-D convolutional neural networks". In: *IEEE Transactions on Biomedical Engineering* 63.3, pp. 664–675.

Altan, Gokhan, Yakup Kutlu, and Novruz Allahverdi (2016). "A new approach to early diagnosis of congestive heart failure disease by using Hilbert–Huang transform". In: *Computer methods and programs in biomedicine* 137, pp. 23–34.

Elhaj, Fatin A et al. (2016). "Arrhythmia recognition and classification using combined linear and nonlinear features of ECG signals". In: *Computer methods and programs in biomedicine* 127, pp. 52–63.

Li, Pengfei et al. (2016). "High-performance personalized heartbeat classification model for long-term ECG signal". In: *IEEE Transactions on Biomedical Engineering* 64.1, pp. 78–86.

Abdul-Kadir, Nurul Ashikin, Norlaili Mat Safri, and Mohd Afzan Othman (2016). "Dynamic ECG features for atrial fibrillation recognition". In: *Computer methods and programs in biomedicine* 136, pp. 143–150.

Kiranyaz, Serkan, Turker Ince, and Moncef Gabbouj (2016). "Real-time patient-specific ECG classification by 1-D convolutional neural networks". In: *IEEE Transactions on Biomedical Engineering* 63.3, pp. 664–675.

Sahoo, Santanu et al. (2017). "ECG beat classification using empirical mode decomposition and mixture of features". In: *Journal of medical engineering & technology* 41.8, pp. 652–661.

Acharya, U Rajendra et al. (2017). "A deep convolutional neural network model to classify heartbeats". In: *Computers in biology and medicine* 89, pp. 389–396.

Zhou, Fei-yan, Lin-peng Jin, and Jun Dong (2017). "Premature ventricular contraction detection combining deep neural networks and rules inference". In: *Artificial intelligence in medicine* 79, pp. 42–51.

Anwar, Syed Muhammad et al. (2018). "Arrhythmia Classification of ECG Signals Using Hybrid Features". In: *Computational and mathematical methods in medicine* 2018.

Beritelli, Francesco et al. (2018). "A novel training method to preserve generalization of RBPNN classifiers applied to ECG signals diagnosis". In: *Neural Networks* 108, pp. 331–338.

Yildirim, Özal (2018). "A novel wavelet sequence based on deep bidirectional LSTM network model for ECG signal classification". In: *Computers in biology and medicine* 96, pp. 189–202.

He, Ziyang et al. (2018). "LiteNet: Lightweight Neural Network for Detecting Arrhythmias at Resource-Constrained Mobile Devices". In: *Sensors* 18.4, p. 1229.

Sayantan, G, PT Kien, and KV Kadambari (2018). "Classification of ECG beats using deep belief network and active learning". In: *Medical & Biological Engineering & Computing*, pp. 1–12.

Oliveira, Bruno Rodrigues de et al. (2019). "Geometrical features for premature ventricular contraction recognition with analytic hierarchy process based machine learning algorithms selection". In: *Computer methods and programs in biomedicine* 169, pp. 59–69.

Xia, Yong et al. (2018). "Detecting atrial fibrillation by deep convolutional neural networks". In: *Computers in biology and medicine* 93, pp. 84–92.

Costantino, Giorgio et al. (2016). "Neural networks as a tool to predict syncope risk in the Emergency Department". In: *Ep Europace* 19.11, pp. 1891–1895.

Masetic, Zerina and Abdulhamit Subasi (2016). "Congestive heart failure detection using random forest classifier". In: *Computer methods and programs in biomedicine* 130, pp. 54–64.

Kora, Padmavathi (2017). "ECG based myocardial infarction detection using hybrid firefly algorithm". In: *Computer methods and programs in biomedicine* 152, pp. 141–148.

Liu, Wenhan et al. (2018). "Multiple-feature-branch convolutional neural network for myocardial infarction diagnosis using electrocardiogram". In: *Biomedical Signal Processing and Control* 45, pp. 22–32.

Jin, Lin-peng and Jun Dong (2016). "Ensemble deep learning for biomedical time series classification". In: *Computational intelligence and neuroscience* 2016.

Teijeiro, Tomás et al. (2018). "Abductive reasoning as a basis to reproduce expert criteria in ECG atrial fibrillation identification". In: *Physiological measurement* 39.8, p. 084006.

Sadr, Nadi et al. (2018). "A low-complexity algorithm for detection of atrial fibrillation using an ECG". In: *Physiological measurement* 39.6, p. 064003.

Kamaleswaran, Rishikesan, Ruhi Mahajan, and Oguz Akbilgic (2018). "A robust deep convolutional neural network for the classification of abnormal cardiac rhythm using single lead electrocardiograms of variable length". In: *Physiological measurement* 39.3, p. 035006.

Hannun, Awni Y et al. (2019). "Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network". In: *Nature medicine* 25.1, p. 65.

Ibaida, Ayman and Ibrahim Khalil (2010). "Distinguishing between ventricular tachycardia and ventricular fibrillation from compressed ECG signal in wireless Body Sensor Networks". In: *2010 Annual International Conference of the IEEE Engineering in Medicine and Biology*. IEEE, pp. 2013–2016.

Jovic, Alan and Nikola Bogunovic (2011). "Electrocardiogram analysis using a combination of statistical, geometric, and nonlinear heart rate variability features". In: *Artificial intelligence in medicine* 51.3, pp. 175–186.

Tejera, Eduardo et al. (2011). "Artificial neural network for normal, hypertensive, and preeclamptic pregnancy classification using maternal heart rate variability indexes". In: *The Journal of Maternal-Fetal & Neonatal Medicine* 24.9, pp. 1147–1151.

Park, Jinho, Witold Pedrycz, and Moongu Jeon (2012). "Ischemia episode detection in ECG using kernel density estimation, support vector machine and feature selection". In: *Biomedical engineering online* 11.1, p. 30.

Ebrahimzadeh, Elias, Mohammad Pooyan, and Ahmad Bijar (2014). "A novel approach to predict sudden cardiac death (SCD) using nonlinear and time-frequency analyses from HRV signals". In: *PloS one* 9.2, e81896.

Zhang, Lijuan et al. (2015). "Automatic recognition of cardiac arrhythmias based on the geometric patterns of Poincaré plots". In: *Physiological measurement* 36.2, p. 283.

He, Mi, Yubao Lu, et al. (2016). "Combining amplitude spectrum area with previous shock information using neural networks improves prediction performance of defibrillation outcome for subsequent shocks in out-of-hospital cardiac arrest patients". In: *PloS one* 11.2, e0149115.

He, Mi, Yushun Gong, et al. (2015). "Combining multiple ECG features does not improve prediction of defibrillation outcome compared to single features in a large population of out-of-hospital cardiac arrests". In: *Critical Care* 19.1, p. 425.

Yu, Ming et al. (2016). "A new method without reference channels used for ventricular fibrillation detection during cardiopulmonary resuscitation". In: *Australasian physical & engineering sciences in medicine* 39.2, pp. 391–401.

Immanuel, SA et al. (2016). "T-wave morphology can distinguish healthy controls from LQTS patients". In: *Physiological measurement* 37.9, p. 1456.

Isler, Yalcin (2016). "Discrimination of systolic and diastolic dysfunctions using multi-layer perceptron in heart rate variability analysis". In: *Computers in biology and medicine* 76, pp. 113–119.

Mjahad, Azeddine et al. (2017). "Ventricular Fibrillation and Tachycardia detection from surface ECG using time-frequency representation images as input dataset for machine learning". In: *Computer methods and programs in biomedicine* 141, pp. 119–127.

Rad, Ali Bahrami et al. (2017). "ECG-based classification of resuscitation cardiac rhythms for retrospective data analysis". In: *IEEE Transactions on Biomedical Engineering* 64.10, pp. 2411–2418.

Tan, Jen Hong et al. (2018). "Application of stacked convolutional and long short-term memory network for accurate identification of CAD ECG signals". In: *Computers in biology and medicine* 94, pp. 19–26.

Amezquita-Sanchez, Juan P et al. (2018). "A Novel Wavelet Transform-Homogeneity Model for Sudden Cardiac Death Prediction Using ECG Signals". In: *Journal of medical systems* 42.10, p. 176.

Moody, George B and Roger G Mark (2001). "The impact of the MIT-BIH arrhythmia database". In: *IEEE Engineering in Medicine and Biology Magazine* 20.3, pp. 45–50.

Goldberger, Ary L et al. (2000). "PhysioBank, PhysioToolkit, and PhysioNet: components of a new research resource for complex physiologic signals". In: *Circulation* 101.23, e215–e220.

Clifford, Gari D et al. (2017). "AF classification from a short single lead ECG recording: The Physionet Computing in Cardiology Challenge 2017". In: *Proceedings of Computing in Cardiology* 44, p. 1.

Bousseljot, R, D Kreiseler, and A Schnabel (1995). "Nutzung der EKG-Signaldatenbank CARDIODAT der PTB über das Internet". In: *Biomedizinische Technik/Biomedical Engineering* 40.s1, pp. 317–318.

Luo, Kan et al. (2017). "Patient-specific deep architectural model for ecg classification". In: *Journal of healthcare engineering* 2017.

Yao, Jun et al. (1998). "Pruning algorithm in wavelet neural network for ECG signal classification". In: *Engineering in Medicine and Biology Society, 1998. Proceedings of the 20th Annual International Conference of the IEEE*. Vol. 3. IEEE, pp. 1482–1485.

Isin, Ali and Selen Ozdalili (2017). "Cardiac arrhythmia detection using deep learning". In: *Procedia Computer Science* 120, pp. 268–275.

Xiao, Ran et al. (2018). "A Deep Learning Approach to Examine Ischemic ST Changes in Ambulatory ECG Recordings". In: *AMIA Summits on Translational Science Proceedings* 2017, p. 256.

Szegedy, Christian et al. (2015). "Going deeper with convolutions". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9.

Rajpurkar, Pranav et al. (2017). "Cardiologist-level arrhythmia detection with convolutional neural networks". In: *arXiv preprint arXiv:1707.01836*.

He, Kaiming et al. (2016). "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.

Chen, Zhengming et al. (2011). "China Kadoorie Biobank of 0.5 million people: survey methods, baseline characteristics and long-term follow-up". In: *International journal of epidemiology* 40.6, pp. 1652–1666.

Ramrakha, Punit and Jonathan Hill (2012). *Oxford handbook of cardiology*. OUP Oxford.

Scott, David W (2015). *Multivariate density estimation: theory, practice, and visualization*. John Wiley & Sons.

Zhou, Bin et al. (2017). "Worldwide trends in blood pressure from 1975 to 2015: a pooled analysis of 1479 population-based measurement studies with $19 \cdot 1$ million participants". In: *The Lancet* 389.10064, pp. 37–55.

Mitchell, Tom M et al. (1997). "Machine learning. 1997". In: *Burr Ridge, IL: McGraw Hill* 45.37, pp. 870–877.

Taylor, John Shawe and Nello Cristianini (2000). *Support Vector Machines and other kernel-based learning methods*.

Müller, Klaus-Robert et al. (2001). "An introduction to kernel-based learning algorithms". In: *IEEE transactions on neural networks* 12.2.

Schölkopf, Bernhard and Alexander J Smola (2002). *Learning with kernels. 2002*.

Herbrich, Ralf (2001). *Learning kernel classifiers: theory and algorithms*. MIT press.

Platt, John C, Nello Cristianini, and John Shawe-Taylor (2000). "Large margin DAGs for multiclass classification". In: *Advances in neural information processing systems*, pp. 547–553.

Tipping, Michael E (2001). "Sparse Bayesian learning and the relevance vector machine". In: *Journal of machine learning research* 1.Jun, pp. 211–244.

Breiman, Leo et al. (1984). "Classification and regression trees. Wadsworth Int". In: *Group* 37.15, pp. 237–251.

Quinlan, JR (1993). "C4. 5: Programs for machine learning. Morgan Kaufmann, San Francisco." In: *C4. 5: Programs for machine learning. Morgan Kaufmann, San Francisco.*

Quinlan, J. Ross (1986). "Induction of decision trees". In: *Machine learning* 1.1, pp. 81–106.

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman (2001). "The elements of statistical learning. Springer series in statistics". In: *:* Springer.

Geurts, Pierre, Damien Ernst, and Louis Wehenkel (2006). "Extremely randomized trees". In: *Machine learning* 63.1, pp. 3–42.

Friedman, Jerome H (2001). "Greedy function approximation: a gradient boosting machine". In: *Annals of statistics*, pp. 1189–1232.

Freund, Yoav, Robert E Schapire, et al. (1996). "Experiments with a new boosting algorithm". In: *icml*. Vol. 96. Citeseer, pp. 148–156.

Bishop, Christopher M (2006). "Pattern recognition and machine learning (information science and statistics) springer-verlag new york". In: *Inc. Secaucus, NJ, USA*.

Friedman, Jerome, Trevor Hastie, Robert Tibshirani, et al. (2000). "Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors)". In: *The annals of statistics* 28.2, pp. 337–407.

Friedman, Jerome H (2002). "Stochastic gradient boosting". In: *Computational statistics & data analysis* 38.4, pp. 367–378.

Sokolow, Maurice and Thomas P Lyon (1949). "The ventricular complex in left ventricular hypertrophy as obtained by unipolar precordial and limb leads". In: *American heart journal* 37.2, pp. 161–186.

Casale, Paul N et al. (1987). "Improved sex-specific criteria of left ventricular hypertrophy for clinical and computer interpretation of electrocardiograms: validation with autopsy findings." In: *Circulation* 75.3, pp. 565–572.

Romhilt, Donald W and E Harvey Estes Jr (1968). "A point-score system for the ECG diagnosis of left ventricular hypertrophy". In: *American heart journal* 75.6, pp. 752–758.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville (2016). *Deep learning*. MIT press.

Jarrett, Kevin et al. (2009). "What is the best multi-stage architecture for object recognition?" In: *2009 IEEE 12th international conference on computer vision*. IEEE, pp. 2146–2153.

Nair, Vinod and Geoffrey E Hinton (2010). "Rectified linear units improve restricted boltzmann machines". In: *Proceedings of the 27th international conference on machine learning (ICML-10)*, pp. 807–814.

Glorot, Xavier, Antoine Bordes, and Yoshua Bengio (2011). "Deep sparse rectifier neural networks". In: *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 315–323.

Maas, Andrew L, Awni Y Hannun, and Andrew Y Ng (2013). "Rectifier nonlinearities improve neural network acoustic models". In: *Proc. icml*. Vol. 30. 1, p. 3.

He, Kaiming et al. (2015). "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification". In: *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034.

Goodfellow, Ian J, David Warde-Farley, et al. (2013). "Maxout networks". In: *arXiv preprint arXiv:1302.4389*.

Marieb, Elaine Nicpon and Katja Hoehn (2007). *Human anatomy & physiology*. Pearson Education.

Hanin, Boris (2018). "Which neural net architectures give rise to exploding and vanishing gradients?" In: *Advances in Neural Information Processing Systems*, pp. 582–591.

Ng, A (2015). *Advice for applying machine learning [Internet]*.

Smith, Leslie N and Nicholay Topin (2017). "Exploring loss function topology with cyclical learning rates". In: *arXiv preprint arXiv:1702.04283*.

Smith, Leslie N (2017). "Cyclical learning rates for training neural networks". In: *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, pp. 464–472.

Goodfellow, Ian J, Oriol Vinyals, and Andrew M Saxe (2014). "Qualitatively characterizing neural network optimization problems". In: *arXiv preprint arXiv:1412.6544*.

Polyak, Boris T (1964). "Some methods of speeding up the convergence of iteration methods". In: *USSR Computational Mathematics and Mathematical Physics* 4.5, pp. 1–17.

Hinton, Geoffrey, Nitsh Srivastava, and Kevin Swersky (2012). "Neural networks for machine learning". In: *Coursera, video lectures* 264.

Kingma, Diederik P and Jimmy Ba (2014). "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980*.

Ioffe, Sergey and Christian Szegedy (2015). "Batch normalization: Accelerating deep network training by reducing internal covariate shift". In: *arXiv preprint arXiv:1502.03167*.

Levenberg, Kenneth (1944). "A method for the solution of certain non-linear problems in least squares". In: *Quarterly of applied mathematics* 2.2, pp. 164–168.

Marquardt, Donald W (1963). "An algorithm for least-squares estimation of nonlinear parameters". In: *Journal of the society for Industrial and Applied Mathematics* 11.2, pp. 431–441.

Head, John D and Michael C Zerner (1985). "A Broyden—Fletcher—Goldfarb—Shanno optimization procedure for molecular geometries". In: *Chemical physics letters* 122.3, pp. 264–270.

Glorot, Xavier and Yoshua Bengio (2010). "Understanding the difficulty of training deep feedforward neural networks". In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256.

Srivastava, Nitish et al. (2014). "Dropout: a simple way to prevent neural networks from overfitting". In: *The journal of machine learning research* 15.1, pp. 1929–1958.

LeCun, Yann et al. (1998). "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11, pp. 2278–2324.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2012). "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems*, pp. 1097–1105.

Lin, Min, Qiang Chen, and Shuicheng Yan (2013). "Network in network". In: *arXiv preprint arXiv:1312.4400*.

Simonyan, Karen and Andrew Zisserman (2014). "Very deep convolutional networks for large-scale image recognition". In: *arXiv preprint arXiv:1409.1556*.

Schuster, Mike and Kuldip K Paliwal (1997). "Bidirectional recurrent neural networks". In: *IEEE Transactions on Signal Processing* 45.11, pp. 2673–2681.

Cho, Kyunghyun et al. (2014). "On the properties of neural machine translation: Encoder-decoder approaches". In: *arXiv preprint arXiv:1409.1259*.