THE APPLICATIONS OF DEEP LEARNING IN AUTOMATIC ECG CLASSIFICATION

SHENSHENG XU

PhD

The Hong Kong Polytechnic University

2020

The Hong Kong Polytechnic University
Department of Electronic and Information Engineering

The Applications of Deep Learning in Automatic ECG Classification

Shensheng XU

A thesis submitted in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy

May 2019

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgment has been made in the text.

_____ (Signed)

Shensheng XU (Name of student)

Abstract

Heart arrhythmias or arrhythmias refer to the irregular heartbeats of patients. Not all arrhythmias are serious or life threatening but some types (e.g., atrial fibrillation, ventricular escape and ventricular fibrillation) may be a sign of heart diseases and could cause sudden cardiac death if prompt treatments are not received. Usually medical doctors use different types of electrocardiography (ECG) (e.g., 12-lead ECG and Holter monitors) to check for a variety of heart conditions and identify arrhythmias through analyzing the ECG. With the growing popularity of wearable technology, a large amount of ECG data are required to be analyzed. Therefore, automatic heartbeat classification from ECG signals is an essential step toward arrhythmias detection in medical practice. This thesis explores the applications of deep learning in automatic heartbeat classification, especially for the detection of occasional arrhythmias during long-term continuous cardiac monitoring.

A lot of research efforts have been spent on the classification of heartbeats based on the University of California, Irvine, (UCI) cardiac arrhythmia dataset. Among them, support vector machines (SVMs) and shallow neural networks (NNs) are the most popular classification methods. However, most of the previous studies reported the performance of either the SVMs or the ANNs without in-depth comparisons between these two methods. Also, a large number of handcrafted features have been provided by the UCI dataset, and some may be more relevant to arrhythmias than the others. This thesis is to enhance the performance of heartbeat classification by selecting relevant features from ECG signals, applying dimension reduction on the feature vectors, and applying deep neural networks (DNNs) for classification. A holistic comparison among DNNs, SVMs, and shallow NNs will be provided. Experimental results based on the UCI dataset suggest that DNNs outperform both SVMs and shallow NNs, provided that relevant features have been selected.

To obtain better ECG representation for heartbeat classification, this thesis proposes deep learning methods with signal alignment that facilitate the end-to-end classification of raw ECG signals into heartbeat types, i.e., normal beat or different types of arrhythmias. Time-domain sample points are extracted from raw ECG signals, and consecutive vectors are extracted from a sliding time-window covering these sample points. Each of these vectors comprises the consecutive sample points of a complete heartbeat cycle, which includes not only the QRS complex but also the P and T waves. Unlike existing heartbeat classification methods in which medical doctors extract handcrafted features from raw ECG signals, the proposed end-to-end method leverages a DNN for both feature extraction and classification based on aligned heartbeats. This strategy not only obviates the need to handcraft the features but also produces optimized ECG representation for heartbeat classification. Evaluations on the MIT-BIH arrhythmia database show that at the same specificity, the proposed patient-independent classifier can detect supraventricular- and ventricular-ectopic beats at a sensitivity that is at least 10% higher than current state-of-the-art methods. More importantly, there is a wide range of operating points in which both the sensitivity and specificity of the proposed classifier are higher than those achieved by state-of-the-art classifiers. The proposed classifier can also perform comparable to patient-specific classifiers, but at the same time enjoys the advantage of patient independency.

To address the significant variability in waveforms and characteristics of ECG signals among different patients, termed as inter-patient variations, this thesis proposes adapting a patient-independent DNN using the information in the patient-dependent identity vectors (i-vectors). The adapted networks, namely i-vector adapted patient-specific DNNs (iAP-DNNs), are tuned towards the ECG characteristics of individual patients. For each patient, his/her ECG waveforms are compressed into an i-vector

using a factor analysis model. Then, this i-vector is concatenated to the middle hidden layer of the patient-independent DNN. Stochastic gradient descent is then applied to fine-tune the whole network to form a patient-specific classifier. As a result, the adaptation makes use of not only the raw ECG waveforms from the specific patient but also the compact representation of his/her ECG characteristics through the i-vector. Analysis on the hidden-layer activations show that by leveraging the information in the i-vectors, the iAP-DNNs are more capable of discriminating normal heartbeats against arrhythmic heartbeats than the networks that use the patient-specific ECG only for the adaptation. Experimental results based on the MIT-BIH arrhythmia database suggest that the iAP-DNNs perform better than existing patient-specific classifiers in terms of various performance measures. In particular, the sensitivity and specificity of the existing methods are all under the receiver operating characteristic curves of iAP-DNNs.

AUTHOR'S PUBLICATIONS

Journal Papers

1. **S. S. Xu**, M. W. Mak, and C. C. Cheung, "Towards End-to-End ECG Classification with Raw Signal Extraction and Deep Neural Networks", *IEEE Journal of Biomedical and Health Informatics*, vol. 23, no. 4, pp. 1574-1584, Jul. 2019.

2. **S. S. Xu**, M. W. Mak, and C. C. Cheung, "I-Vector Based Patient Adaptation of Deep Neural Networks for Automatic Heartbeat Classification", *IEEE Journal of Biomedical and Health Informatics*, doi: 10.1109/JBHI.2019.2919732.

Conference Papers

1. **S. S. Xu**, M. W. Mak, and C. C. Cheung, "Patient-Specific Heartbeat Classification Based on I-Vector Adapted Deep Neural Networks", in *IEEE International Conference on Bioinformatics and Biomedicine*, Dec. 2018, pp. 784-787.

2. **S. S. Xu**, M. W. Mak, and C. C. Cheung, "Deep Neural Networks versus Support Vector Machines for ECG Arrhythmia Classification", in *IEEE International Conference on Multimedia and Expo Workshops*, Jul. 2017, pp. 127–132.

ACKNOWLEDGMENTS

Foremost, I would like to express my sincere gratitude to my chief supervisor Dr. M. W. Mak for the continuous support of my Ph.D study, for his patience, vision, motivation, and vast knowledge (e.g., speaker recognition, machine learning, and bioinformatics). He has taught me the methodology to carry out the research and to present the research works as clearly as possible. It is my fortune and honor to study under his guidance. Without his assistance and suggestion, it is impossible for me to complete the study in HKPolyU.

My sincere thanks also goes to Dr. C. C. Cheung. I appreciate his constructive advice when I got stuck in my research. He has been giving me guidance, assistance and encouragement for many years. He is not only my supervisor but also a friend.

Besides my supervisors, I would like to say thanks to my friends and research colleagues, Youzhi TU, Lu YI and Weiwei LIN for the academic discussions and for all the fun we have had during the study period.

Finally, it is my pleasure to acknowledge the general office of the department of Electronic and Information Engineering and the research office of HKPolyU for their generous support over the past three years.

Last but not least, I am very much thankful to my parents, my wife, my daughter and all my family. Their endless love and understanding support me in completing the doctoral study.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

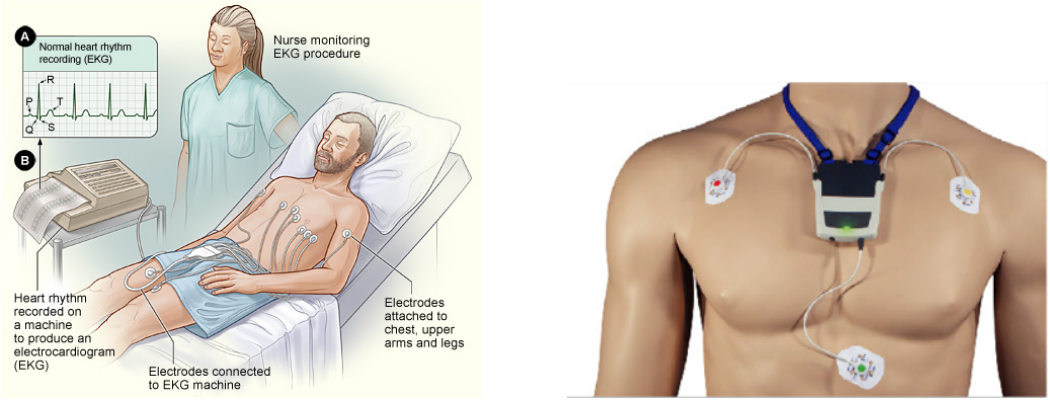
INTRODUCTION

1.1 Heart Arrhythmias

Heart arrhythmias refer to the condition in which a patient's heart beats irregularly. Most types of arrhythmias have no symptoms and are not serious. However, arrhythmias may cause symptoms of heart diseases, including lightheadedness, passing out, shortness of breath and chest pain. Some types of arrhythmias such as atrial fibrillation, ventricular escape and ventricular fibrillation may cause strokes and cardiac arrest that are extremely dangerous and require immediate treatment [17].

1.2 ECG Measurement

Heart arrhythmias can be detected through electrocardiography (ECG), which is a process of recording the electrical activities of the heart. The conventional ECG uses a 12-lead configuration in which a number of electrodes are placed on a patient's limbs and on the surface of the chest. The 12-lead ECG is made up of the three standard limb leads (I, II and III), the augmented limb leads (aVR, aVL and aVF) and the six precordial leads (V1, V2, V3, V4, V5 and V6). During measurement of 12-lead ECG, the patient is asked to lie quietly on a bed so that high quality 12-lead ECG signals can be recorded, but this arrangement is impractical for long-term monitoring. Unfortunately, some intermittent arrhythmias can only be detected by long-term monitoring because they can be easily missed in ordinary recording



(a) 12-lead ECG configuration (b) 2-lead ECG configuration

Figure 1.1: Two types of ECG configurations

sessions. To overcome this issue, a 2-lead configuration is routinely used in Holter monitoring [18] and is widely accepted as a practical means of long-term continuous heart monitoring. Figure 1.1 shows the two types of ECG configurations.[1]

1.3 ECG Waveforms and Intervals

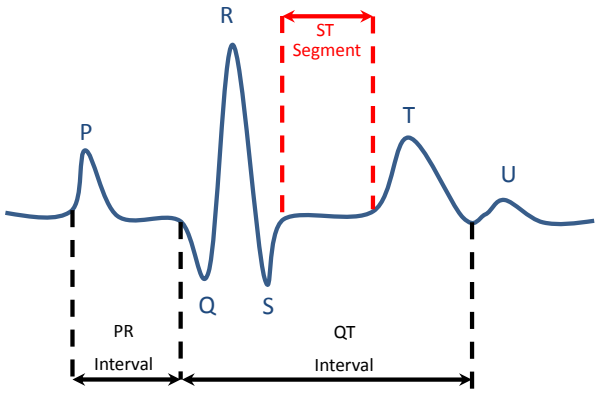


Figure 1.2: Typical ECG in a normal sinus rhythm

[1]https://commons.wikimedia.org/wiki/File:Ekg_NIH.jpg; https://www.promed.ie/product/custo-med-ecg-holter-monitor

Figure 1.2 shows a typical ECG in a sinus rhythm. A sinus rhythm (heartbeat) is a normal regular rhythm of the heart; it is set by the sinus node which is the natural pacemaker of the heart. As illustrated in Figure 1.2, a normal rhythm produces a P wave, a QRS complex, a T wave, and a U wave [19]. The P wave represents atrial depolarization, the QRS complex represents ventricular depolarization, the T wave represents ventricular repolarization, and the U wave represents papillary muscle repolarization. Among them, the U wave is not typically seen and its absence is generally ignored.

A normal rhythm can be divided into different segments and intervals, such as ST segment, PR interval and QT interval. The ST segment connects the QRS complex and the T wave. It represents the period when the ventricles are depolarized. The PR interval is measured from the beginning of the P wave to the beginning of the QRS complex. It reflects the time the impulse takes to reach the ventricles from the sinus node. The QT interval is measured from the beginning of the QRS complex to the end of the T wave. It represents electrical depolarization and repolarization of the ventricles. The above durations are closely related to the condition of cardiac conduction system, and thus they are meaningful indexes for doctors to diagnose heart arrhythmia. Therefore, being able to identify the dangerous types of heart arrhythmia from ECG signals is an important skill of medical professionals.

1.4 Automation

In general, an ECG recording session lasts several minutes, and medical doctors examine the ECG waveforms beat-by-beat to diagnose whether heart arrhythmias exist or not. With the increasing use of personal portable devices to acquire ECG data, a large number of ECG recordings can be collected. However, it is impossible to read and analyze all of these data manually by medical professionals. As a result, the development of automatic techniques for identifying abnormal conditions from daily

recorded ECG data is of fundamentally importance. Moreover, timely first-aid procedures can be applied if such abnormal conditions can be detected automatically by health monitoring equipment. In this regard, it is better to use machines to classify heartbeats automatically so as to assist clinicians in diagnosing arrhythmias.

1.5 Thesis Organization

This thesis is organized as follows:

In Chapter 2, we introduce two popular ECG datasets and the metrics for evaluating the performance of ECG classifiers. We also give a literature review on automatic ECG classification, including patient-independent and patient-specific ECG classification.

In Chapter 3, we review the commonly used machine learning techniques, focussing on the feedforward fully-connected neural networks. The network structure, backpropagation (BP) fine-tuning and restricted Boltzmann machines (RBMs) will be discussed.

In Chapter 4, we apply feature selection and dimension reduction methods to handcrafted features for producing better representation of heartbeats. A performance comparison among different machine learning techniques will be provided.

In Chapter 5, we propose a patient-independent heartbeat classifier for detecting arrhythmias types during long-term heart monitoring.

In Chapter 6, we propose a patient-specific heartbeat classifier to address the inter-patient variation in ECG signals.

In Chapter 7, we draw the conclusions and suggest some directions for future work.

Chapter 2

ECG DATA AND RELATED WORK

2.1 Sources of ECG Data

Usually, ECG data are available only in hospitals or specialized research centers, and the collection and usage may cause privacy issues. Thus, collecting ECG data is very expensive. Because of this, there are only a few public-domain ECG datasets. To evaluate the performance of the feature pre-processing methods and classification algorithms in ECG classification, the UCI cardiac arrhythmia dataset [20] and the MIT-BIH arrhythmia dataset [21] were used in this work. The UCI data was collected by using the conventional 12-lead ECG configuration. The dataset was used for investigating which sets of feature are appropriate for which classification methods (Chapter 3). Note that we do not need to apply feature extraction to this dataset because handcrafted features[1] have already been provided. In contrast, the MIT-BIH dataset contains raw ECG signals. The proposed end-to-end heartbeat classification (Chapter 5) and the i-vector adapted deep neural networks (Chapter 6) are designed to detect some types of arrhythmias from the raw ECG waveforms during continuous heart monitoring. Thus, we used the 2-lead ECG configuration and also used the MIT-BIH dataset for performance evaluation because it comprises a standard set of Holter recordings for evaluating arrhythmia detectors.

[1]The term "handcrafted features" is frequently used in the machine learning community to refer to features that are handcrafted by human experts of the field based on their knowledge and past experience.

Table 2.1: Class distribution in the UCI cardiac arrhythmia dataset

Class Code	Class	Number of Instances
01	Normal	245
02	Ischemic changes	44
03	Old Anterior Myocardial Infarction	15
04	Old Inferior Myocardial Infarction	15
05	Sinus tachycardy	13
06	Sinus bradycardy	25
07	Ventricular Premature Contraction (PVC)	3
08	Supraventricular Premature Contraction	2
09	Left bundle branch block	9
10	Right bundle branch block	50
11	1. degree AtrioVentricular block	0
12	2. degree AV block	0
13	3. degree AV block	0
14	Left ventricule hypertrophy	4
15	Atrial Fibrillation or Flutter	5
16	Others	22

2.1.1 UCI Cardiac Arrhythmia Dataset

The UCI cardiac arrhythmia [20] dataset comprises 452 labelled heartbeats divided into 16 different heartbeat types (or classes). Each heartbeat has 279 handcrafted features, such as QRS duration, Q-T interval, P-R interval, T interval and so on. One of the 16 classes is named "Normal", which contains 245 normal heartbeats. The remaining 15 classes represent different kinds of heart arrhythmia, which comprise 207 abnormal heartbeats. The class distribution is shown in Table 2.1.

Because features corresponding to the 452 heartbeats have been provided, it is not necessary to perform heartbeat segmentation and feature extraction. However, pre-processing of feature vectors such as z-norm is still necessary to ensure that individual features in the feature vectors have the same range. There are only two kinds of features in the dataset: Nominal (such as "Sex" and "Existence of ragged R wave")

and Continuous (such as "QRS druation" and "Q-T interval"). All of the nominal features have two categories ("M" and "F" for "sex", "Yes" and "No" for "Existence of ragged R wave"), and we used "0" and "1" to represent them. For continuous features, z-norm was applied.

2.1.2 MIT-BIH Arrhythmia Database

The MIT-BIH arrhythmia database [21] contains 48 half-hour excerpts of two-channel ambulatory ECG recordings. The database was obtained by the BIH Arrhythmia Laboratory between 1975 and 1979. It involves 47 subjects (25 men aged between 32 and 89, and 22 women aged between 23 and 89). Each record contains a continuous recording of ECG signals from a single subject, except for Records 201 and 202 in which the data were obtained from the same male subject. All records were labelled beat-by-beat by two or more cardiologists independently. The total number of labelled heartbeats is 108,655. These heartbeats are divided into 15 different types (see Table 2.2). Compared with the UCI arrhythmia dataset, the total number of heartbeats in the MIT-BIH arrhythmia database is 200 times bigger. Moreover, the records in this database contain raw ECG signals. Thus, it can also be used for testing heartbeat segmentation and feature extraction algorithms mentioned in Section 3.4, which are very important in ECG classification. Therefore, the MIT-BIH arrhythmia database is one of the most popular data source for studying ECG classification.

Table 2.2 shows the 15 types of heart arrhythmia in the MIT-BIH arrhythmia database. According to the American National Standard (ANSI/AAMI EC57:1998) [22] prepared by the Association for the Advancement of Medical Instrumentation, these heartbeat types can be combined into five classes as shown in Table 2.3. These classes include normal beat (N), ventricular ectopic beat (V), supraventricular ectopic beat (S), fusion of a normal and a ventricular ectopic beat (F) and unknown beat type (Q).

Signals in MIT-BIH arrhythmia database were digitized at 360 samples per sec-

Table 2.2: Heartbeat types in the MIT-BIH arrhythmia database

Class Code	Class	Number of Instances
N	Normal	75,054
L	Left bundle branch block beat	8,074
R	Right bundle branch block beat	7,259
A	Atrial premature beat	2,544
a	Aberrated atrial premature beat	150
J	Nodal (junctional) premature beat	83
S	Supraventricular premature beat	2
V	Premature ventricular contraction	7,129
F	Fusion of ventricular and normal beat	803
e	Atrial escape beat	16
j	Nodal (junctional) escape beat	229
E	Ventricular escape beat	106
P	Paced beat	7,028
f	Fusion of paced and normal beat	982
Q	Unclassifiable beat	33

ond per channel with 11-bit resolution over a ±5mV range. Note that the samples represent the real measured voltage ranging between −5mV to +5mV, which is from 0 to 2,047 inclusive, with a value of 1,024 corresponding to zero volt.

In most of the recordings in the MIT-BIH database, the upper signal is a modified limb lead II (MLII), and the lower one is a modified lead V1.[2] In our experiments, only the upper signal was used for ECG classification because normal QRS complexes are usually prominent in it.

2.2 Automatic ECG Classification

To assist doctors in identifying heart arrhythmia, computer scientists have applied machine learning techniques to automatically discover patterns in ECG data that

[2]We adopted the terminology from MIT-BIH and used upper and lower signals to refer to the two channels of ECG recordings.

Table 2.3: Mapping the MIT-BIH arrhythmia types into five heartbeat classes recommended by AAMI

	AAMI Class				
	N	S	V	F	Q
MIT-BIH Class Code (see Table 2.2)	NOR, LBBB, RBBB, AE, NE	AP, aAP, NP, SP	PVC, VE	fVN	P, fPN, U
No. of Instances	90,042	2,779	7,007	802	15

are related to heart arrhythmias. Kohli *et al.* [23] used a one-versus-rest SVM as the classifier to predict heart arrhythmia and achieved good performance on the UCI benchmark dataset [24] (the best classification accuracy on their test data is over 70%). In [25], Khare *et al.* proposed a hybrid approach combining rank correlation [26] and principal component analysis (PCA) [27] for feature extraction and SVMs for classification. They demonstrated that the hybrid approach achieves much better performance than the predictor proposed by Kohli *et al.* [23] on the same dataset. However, the hyper-parameters of the heart arrhythmia classifiers in these works were optimized based on the test data. As a result, the claimed accuracy in these studies may be over-estimated. In [28], ANNs were applied to the same heart arrhythmia dataset. The authors showed that the best performance of the ANNs is close to that of the SVMs. Unfortunately, they did not specify the network structures and parameter settings in their paper, causing difficulty in comparing the capability of ANNs and SVMs in predicting heart arrhythmias.

In the recent past, there have been much efforts [1, 3–15] in classifying heartbeats automatically. Many of these studies [1, 3–12] adopted a beat-by-beat analysis strategy and used the MIT-BIH arrhythmia database [21] for performance evaluation. Moreover, they followed the standard prepared by the Association for the

Advancement of Medical Instrumentation (ANSI/AAMI EC57:1998) [22] for testing and reporting performance.

2.2.1 Evaluation Schemes

As mentioned in [3], two evaluation schemes, namely "class-oriented" and "subject-oriented", are commonly used for ECG classification. Using the class-oriented evaluation scheme, the performance of the classifiers in [10, 11, 13–15] may be overestimated because signals in the training and test sets could belong to the same patient. The "well trained" classifier may fail to predict the ECG signals from an unseen individual. The scheme is not applicable in practice because of the significant variation in ECG characteristics among different subjects. Using the subject-oriented evaluation scheme, the data in [1, 3–9, 12] were divided into the training set and the testing set based on ECG recordings. This means that the ECG signals in the training and test sets were definitely not from the same patient. The classifiers developed through this scheme are more realistic.

The subject-oriented evaluation scheme leads to two types of classifiers—patient-independent classifiers (e.g., [1, 3, 4]) and patient-specific classifiers (e.g., [5–9, 12]). In general, patient-specific classifiers perform much better than patient-independent classifiers because the formers are trained on a small set of annotated data from the respective patients. In contrast, the cost of patient-independent classifiers is much lower because no patient-specific data or expert intervention is required. Note that the proposed end-to-end method adopts the subject-oriented evaluation scheme, and a patient-independent classifier is built for beat-by-beat classification of ECG signals.

2.2.2 Patient-Independent ECG Classification

Chazal *et al.* [1] utilized morphological and dynamic features to represent heartbeats and then classified them into five classes. The classifier is based on linear discrimi-

nants, and its parameters are determined by maximum-likelihood estimation. In [3], Ye *et al.* applied wavelet transform and independent component analysis (ICA) to extract morphological features from segmented heartbeats. Heartbeat intervals were also used as dynamic features. The features were applied to an SVM for classifying heartbeats into five classes.

In [4], a new feature extraction method (sparse decomposition over a Gabor dictionary) is proposed to represent various classes of heartbeats. Four kinds of features (i.e., time delay, frequency, width parameter and square of expansion coefficients) are extracted from each of the significant atoms of the dictionary and concatenated to constitute a feature vector. The feature vectors are classified into five classes using some typical classification models. Among the different proposed methods, the performance of the particle swarm optimization (PSO) optimized least-square twin SVM model achieves the best performance.

2.2.3 Patient-Specific ECG Classification

Jiang *et al.* [5] proposed using Hermite transform coefficients to approximate the QRS complexes of heartbeats. The coefficients and R-R intervals were used as heartbeat features for classification by an evolvable block-based neural network (BbNN) [30]. In the training stage, both common (totally 142 beats from 20 patients) and patient-specific data (5-minute ECG from each patient) were used for evolving the patient-specific BbNNs. The results suggest that high accuracies can be achieved by using personalized ECG classification. However, a large number of parameters or thresholds needed to be set empirically in this approach.

In [6], wavelet transform and principal component analysis (PCA) were applied to extract morphological features. The low dimensional morphological feature vectors were combined with temporal features to form the final feature vectors. A multi-dimensional particle swarm optimization (MD PSO) method was proposed, which optimizes neural network based classifiers according to 245 common training beats and

a variable number of patient-specific beats. Overall, this method achieves performance that is comparable with the BbNN-based personalized ECG classifier in [5].

In [7], the raw data of each beat were downsampled to 64 or 128 time-points centered on the R-peak, and FFT representations were used as the input to a patient-specific 1-D convolutional neural network (CNN). Each CNN was trained by using 245 representative beats that are common to all patients and five minutes of patient-specific beats. Results show that the CNNs outperform any existing arrhythmia classifiers under the same evaluation protocol.

Ye *et al.* [12] utilized wavelet transform and independent component analysis (ICA) to extract morphological features from segmented heartbeats. Unlike other patient-specific classifiers, the classifiers in [12] can be trained on unlabeled patient-specific data, meaning that no manual intervention is required during training. Specifically, a general classifier was trained on the data extracted from the patients who are similar to the target patient. Then, a patient-specific classifier was trained on a small amount of patient-specific ECG with high-confident labels hypothesized by a multi-view model. The final result was obtained by combining the two classifiers probabilistically. Results shown that the customized models together with automatic adaptation can improve classification performance.

In [8], the beats were transformed into dual-beat coupling matrices, which are used as 2-D inputs to a CNN classifier. The matrices captured both beat morphology and beat-to-beat correlation in ECG. A heartbeat selection procedure was also proposed to select the most representative beats. For each patient, a classifier was trained based on these representative beats and the patient-specific ECG. Results demonstrated that the 2-D CNN-based classifiers were superior to several state-of-the-art detectors.

In [9], a generic convolutional neural network (GCNN) was trained based on the ECG of a general population. The GCNN was then fine-tuned to form a tuned dedicated CNN (TDCNN) using patient-specific ECG. Raw ECG signals were used as the input of the CNN classifiers and the heartbeat segmentation procedure was

the same as [7]. To explore the influence of the amount of training samples on the performance of TDCNN, 2-, 3-, 4- and 5-minute patient-specific ECG were used to adapt the GCNN. The results show that more training samples help the TDCNN to achieve higher classification accuracy and the performance was comparable with the existing patient-specific classifiers.

2.3 Performance Evaluation Metrics

The classification performance on each heartbeat class was measured by using four standard metrics, namely, classification accuracy (Acc), sensitivity (Sen), specificity (Spe) and positive predictive value (Ppv), which are calculated based on the number of true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN), as follows. Accuracy is the fraction of the total number of instances that is correctly identified, i.e.,

$$\text{Acc} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}. \tag{2.1}$$

Sensitivity is the proportion of positives that are correctly identified, i.e.,

$$\text{Sen} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{2.2}$$

Specificity is the proportion of negatives that are correctly identified, i.e.,

$$\text{Spe} = \frac{\text{TN}}{\text{TN} + \text{FP}}. \tag{2.3}$$

Positive predictive value is the fraction of the positive predictions that are actually positive, i.e.,

$$\text{Ppv} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{2.4}$$

Details on how to interpret these four metrics can be found in [31–33].

Matthews correlation coefficient (MCC) [34] is a better measure for imbalanced datasets (datasets with imbalanced number of classes). It returns a value between -1 and $+1$. A coefficient of "$+1$" represents a perfect prediction, a "0" means it is not better than random prediction and a "-1" indicates total disagreement between prediction and observation. According to [35], denote $\mathbf{M} \in \Re^{C\times C}$ as the confusion matrix of the prediction result, where C is the number of classes. Then $M_{i,j}$ represents the number of instances that actually belong to class i but are predicted as class j, where $1 \leq i, j \leq C$. We further denote

$$p_c = M_{c,c}, \quad q_c = \sum_{i=1,i\neq c}^{C} \sum_{j=1,j\neq c}^{C} M_{i,j}, \quad r_c = \sum_{i=1,i\neq c}^{C} M_{i,c}, \quad s_c = \sum_{j=1,j\neq c}^{C} M_{c,j}, \tag{2.5}$$

where c ($1 \leq c \leq C$) is the index of a particular class. For class c, p_c is the number of true positives, q_c is the number of true negatives, r_c is the number of false positives, and s_c is the number of false negatives. The Matthews correlation coefficient of class c (MCC_c) and the overall MCC (OMCC) are defined respectively as:

$$\text{MCC}_c = \frac{p_c q_c - r_c s_c}{\sqrt{(p_c + s_c)(p_c + r_c)(q_c + s_c)(q_c + r_c)}}, \tag{2.6}$$

$$\text{OMCC} = \frac{\hat{p}\hat{q} - \hat{r}\hat{s}}{\sqrt{(\hat{p} + \hat{s})(\hat{p} + \hat{r})(\hat{q} + \hat{s})(\hat{q} + \hat{r})}} \tag{2.7}$$

where $\hat{p} = \sum_{c=1}^{C} p_c$, $\hat{q} = \sum_{c=1}^{C} q_c$, $\hat{r} = \sum_{c=1}^{C} r_c$ and $\hat{s} = \sum_{c=1}^{C} s_c$.

Receiver operating characteristics (ROCs) [36] were used to show the tradeoff between the performance measures (i.e., Sen vs. Spe) of a binary classifier when the decision threshold varies. Because the threshold typically has a wide range, ROC curves can provide more comprehensive information on performance.

In addition, the p-value can be used for statistical tests. It is assigned to determine whether the results produced by two systems are statistically significant. Given the target and non-target scores of two systems, the p-value can be calculated based on

McNemar's test. However, these scores are intermediate results and usually they are not mentioned in the papers. Thus, it is very difficult to get such scores in existing systems unless we fully design and implement their systems. That is why other researchers [1, 3–9, 12] also did not provide the result of p-value.

Table 2.4: Summary of the studies [1,3,4,6,7,10–15] and the proposed method. (a) Studies that do not follow the ANSI/AAMI EC57 standard. (b) Studies that follow this standard. All studies in (a) use the class-oriented evaluation scheme. Therefore, their classifiers are neither patient-specific nor patient-independent.

Ref.	Classes	Database	Analysis Strategy	Evaluation Scheme	Features	Classifier	Classifier Tpye
Osowski [10]	13	MITDB	Beat-by-beat	Class-oriented	Hermite	SVM	N/A
Jun [11]	2	MITDB	Beat-by-beat	Class-oriented	Morphological, RR-interval	DNN	N/A
Acharya [13]	4	MITDB, CUDB [29], AFDB [29]	Segments	Class-oriented	Downsampling, Wavelet, Sample points	CNN	N/A
Tan [14]	2	PhysioNet [29]	Segments	Class-oriented	Upsampling, Wavelet, Sample points	CNN-LSTM	N/A
Acharya [15]	2	PTB [29]	Segments	Class-oriented	Raw ECG	CNN	N/A

(a)

Ref.	Classes	Database	Analysis Strategy	Evaluation Scheme	Features	Classifier	Classifier Tpye
Chazal [1]	5	MITDB	Beat-by-beat	Subject-oriented	Morphological, RR-interval	LD	Patient-independent
Ye [3]	5	MITDB	Beat-by-beat	Subject-oriented	Wavelet, ICA, RR-interval	SVM	Patient-independent
Raj [4]	5	MITDB	Beat-by-beat	Subject-oriented	Sparse	Least-square Twin SVM	Patient-independent
Jiang [5]	5	MITDB	Beat-by-beat	Subject-oriented	Hermite transform, coefficients	BbNN	Patient-specific with expert intervention
Ince [6]	5	MITDB	Beat-by-beat	Subject -oriented	Wavelet, PCA, RR-interval	ANN	Patient-specific with expert intervention
Kiranyaz [7]	5	MITDB	Beat-by-beat	Subject-oriented	Downsampling, FFT	CNN	Patient-specific with expert intervention
Ye [12]	5	MITDB	Beat-by-beat	Subject-oriented	Wavelet, ICA, RR-interval	Multi-view Learning	Patient-specific without expert intervention
Zhai [8]	5	MITDB	Beat-by-beat	Subject-oriented	Dual heartbeat, coupling matrix	CNN	Patient-specific with expert intervention
Li [9]	5	MITDB	Beat-by-beat	Subject-oriented	Raw ECG	TDCNN	Patient-specific with expert intervention

(b)

Chapter 3

MACHINE LEARNING TECHNIQUES

3.1 Principal Component Analysis

Principal component analysis (PCA) is a linear transformation method that reduces the dimensionality of data while retaining most of the variation in the transformed data [27]. Using PCA, data in the input space are projected onto a subspace in which the spread of data is maximum. Given a dataset consisting of D-dimensional samples, the eigenvectors and their corresponding eigenvalues can be computed from the covariance matrix (with dimension $D \times D$) of the samples. Then, the eigenvectors are sorted according to the descending order of the eigenvalues and the K eigenvectors with the largest eigenvalues are selected to form a matrix $\boldsymbol{W}$ of size $D \times K$. After that, this matrix $\boldsymbol{W}$ is used to transform the samples onto the new subspace. Through this operation, the dimension of the original dataset is reduced from D to K.

The number of principal components (the value of K) should be less than or equal to the number of original variables (i.e., $K \leq D$). It can be determined by the proportion of variance explained. Specifically, given that $\lambda_1, \lambda_2, \ldots, \lambda_D$ are the eigenvalues (in descending order) of the covariance matrix, the percentage of total variance retained can be calculated by using the following equation:

$$\eta = \frac{\sum_{j=1}^{K} \lambda_j}{\sum_{j=1}^{D} \lambda_j}. \tag{3.1}$$

Usually, the first K eigenvectors are expected to capture at least 90% of the total variances.

3.2 Support Vector Machines

Support Vector Machines (SVMs) [37] are one of the popular classification methods because of their good performance in many applications. The objective of a linear SVM is to find a separating hyper-plane that maximizes the margin of two classes.

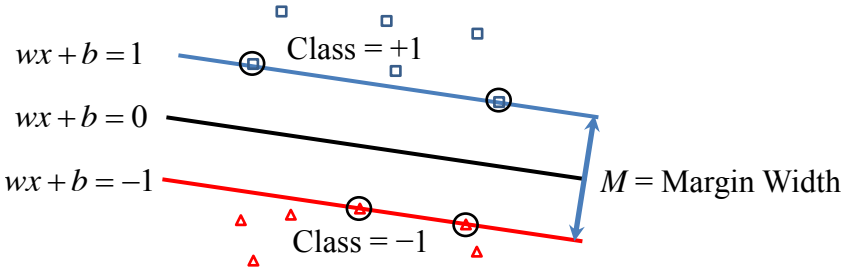


Figure 3.1: Linear SVM (markers with a circle are support vectors)

For linearly separable problems (see Figure 3.1), the margin M is determined by $\mathbf{w}$ and b, and the constrained optimization problem is:

$$\text{Minimize } \frac{1}{2}\|\mathbf{w}\|^2, \quad \text{subject to } y_i(\mathbf{x}_i \cdot \mathbf{w} + b \geq 1), \forall i = 1, \ldots, N, \tag{3.2}$$

where $y_i \in \{-1, +1\}$. This constraint optimization problem can be solved by writing it as a Lagrangian function in which a set of Lagrange multipliers are introduced. For nonlinearly separable problem, the above minimization problem becomes:

$$\text{Minimize } \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i \xi_i, \quad \text{subject to } y_i(\mathbf{x}_i \cdot \mathbf{w} + b \geq 1 - \xi_i), \forall i = 1, \ldots, N, \tag{3.3}$$

where ξ_i represents the degree of violation of the constraint in Eq. 3.2 and C is a user-defined penalty parameter to specify the severity of such violation.

The hyper-plane decision function is defined as:

$$f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b = \sum_{i=1}^{N} \alpha_i y_i (\mathbf{x}_i \cdot \mathbf{x}) + b, \tag{3.4}$$

where $\alpha_i \geq 0$ is the Lagrange multipliers corresponding to the i-th support vector $\mathbf{x}_i$.

For nonlinear SVMs, the decision function becomes:

$$f(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i y_i \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}) + b, \tag{3.5}$$

where $\phi(\cdot)$ is a nonlinear map. Using the nonlinear map $\phi(\mathbf{x})$, the original samples $\mathbf{x}_i$ in the input space are mapped to a higher dimensional feature space in which $\phi(\mathbf{x}_i)$ become linearly separable. Note that, the dimension of $\phi(\mathbf{x})$ may be very high and could be infinite in some cases, meaning that this function may not be implementable. The dot product in Eq. 3.5 can be replaced by a kernel function:

$$\phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}) = K(\mathbf{x}_i, \mathbf{x}), \tag{3.6}$$

which can be efficiently implemented. The kernel function may be different for different problems. The followings are some common kernel functions used in SVMs:

$$\text{Linear kernel} : K(\mathbf{x}_i, \mathbf{x}) = \mathbf{x}_i \cdot \mathbf{x}, \tag{3.7}$$

$$\text{Polynomial kernel} : K(\mathbf{x}_i, \mathbf{x}) = \left(1 + \frac{\mathbf{x}_i \cdot \mathbf{x}}{\sigma^2}\right)^d, \ d > 0 \tag{3.8}$$

$$\text{Radial basis function (RBF) kernel} : K(\mathbf{x}_i, \mathbf{x}) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}\|^2}{2\sigma^2}\right), \tag{3.9}$$

$$\text{Sigmoid kernel} : K(\mathbf{x}_i, \mathbf{x}) = \frac{1}{1 + e^{-\frac{\mathbf{x}_i \cdot \mathbf{x} + b}{\sigma^2}}}, \tag{3.10}$$

where d, σ and b are parameters of the kernel functions. The RBF kernel usually works well in practice and it is relatively easy to calibrate compared with other kernels.

SVMs are efficient for small datasets and they scale relatively well to high dimensional data. However, for large datasets, the training process will be slow.

3.3 Artificial Neural Networks

Artificial neural networks (ANNs) [38] are biologically inspired networks that consist of processing neurons. The neurons are ordered into layers and connected with each other and are capable of receiving and sending signals. The strength of the connections are represented by network weights. The main contribution of ANNs is their ability to capture hidden information from known data, and the process of establishing such ability is called learning. ANNs have been very successful in different domains [39–41]. Note that, in this dissertation, the term ANNs refer to feed-forward networks with a shallow architecture (i.e., one or two hidden layers). Figure 3.2 shows a fully connected ANN that consists of an input layer, a hidden layer and an output layer.

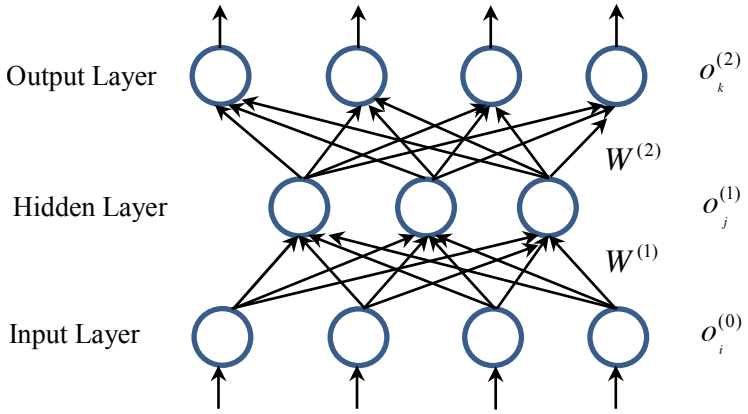


Figure 3.2: A fully connected ANN with one hidden layer

3.3.1 Training by Backpropagation

Backpropagation (BP) [42], which is based on the notion of gradient descent, is the most popular algorithm for training neural networks because of its simplicity and low

computational complexity. The gradient descent is an iterative minimization method. Denote by $\mathbf{W}$ as a set of random weights (in the form of a weight vector) and $E(\mathbf{W})$ as an error function of $\mathbf{W}$. The objective is to iteratively minimize $E(\mathbf{W})$ with respect to $\mathbf{W}$:

$$\mathbf{W}^{t+1} = \mathbf{W}^t - \eta \nabla E(\mathbf{W}^t), \tag{3.11}$$

where t is the iteration index and η is a small positive learning rate.

The BP algorithm mainly consists of two parts: forward pass and backward pass. In the sequel, the description of the forward and backward passes are based on the network structure shown in Figure 3.2.

1. Forward Pass: Compute the outputs $o_k^{(2)}$ and $o_j^{(1)}$ at the output layer and hidden layer respectively by using the following equations:

$$o_k^{(2)} = f(z_k^{(2)}) \tag{3.12}$$

and

$$o_j^{(1)} = f(z_j^{(1)}), \tag{3.13}$$

where $f(\cdot) = \frac{1}{1+e^{-z}}$ is a sigmoid function and $z_k^{(2)} = \sum_j W_{kj}^{(2)} o_j^{(1)}$. In Eq. 3.13, $z_j^{(1)} = \sum_i W_{ji}^{(1)} o_i^{(0)} = \sum_i W_{ji}^{(1)} x_i$, where x_i is the i-th element of the input vector $\mathbf{x}$.

For regression problems, the objective is to minimize the sum of the squared error between the target output t_k and the actual output y_k:

$$E = \frac{1}{2} \sum_k (y_k - t_k)^2. \tag{3.14}$$

2. Backward Pass: Compute the error gradient with respect to the weights:

$$\frac{\partial E}{\partial W_{kj}^{(2)}} = \frac{\partial E}{\partial z_k^{(2)}} \cdot \frac{\partial z_k^{(2)}}{\partial W_{kj}^{(2)}} = \delta_k^{(2)} o_j^{(1)}, \tag{3.15}$$

where $\delta_k^{(2)} = \frac{\partial E}{\partial z_k^{(2)}} = (o_k^{(2)} - t_k) o_k^{(2)} (1 - o_k^{(2)})$, and

$$\frac{\partial E}{\partial W_{ji}^{(1)}} = \frac{\partial E}{\partial z_j^{(1)}} \cdot \frac{\partial z_j^{(1)}}{\partial W_{ji}^{(1)}} = \delta_j^{(1)} x_i, \tag{3.16}$$

where $\delta_j^{(1)} = \frac{\partial E}{\partial z_j^{(1)}} = \sum_k \frac{\partial E}{\partial z_k^{(2)}} \cdot \frac{\partial z_k^{(2)}}{\partial z_j^{(1)}} = \sum_k \delta_k^{(2)} f'(z_j^{(1)}) = o_j^{(1)}(1 - o_j^{(1)}) \sum_k \delta_k^{(2)}$.

Thus, according to Eq. 3.11, we have the weight update equation for output-layer weights:

$$W_{kj}^{(2)} \leftarrow W_{kj}^{(2)} - \mu \frac{\partial E}{\partial W_{kj}^{(2)}} \tag{3.17}$$

and hidden-layer weights:

$$W_{ji}^{(1)} \leftarrow W_{ji}^{(1)} - \mu \frac{\partial E}{\partial W_{ji}^{(1)}}. \tag{3.18}$$

3.3.2 Deep Neural Networks

Because of the architectural depth of the brain, neural network researchers have attempted to train deep (multi-layer) neural networks (DNNs) for decades. They expect DNNs are able to achieve better representations than ANNs so that they should achieve better performance in some complicated applications. However, training DNNs always leads to poor generalization [43] and thus, before 2006, many practical applications still used ANNs on top of handcrafted features, which require a considerable amount of engineering skill and domain expertise. Empirically, DNNs were generally found to be not better and sometimes even worse than ANNs.

To address the above limitations, around 2006, Hinton and Salakhutdinov [44] developed an effective strategy for training DNNs: a local unsupervised criterion to

pre-train each layer in turn, followed by applying gradient descent on the supervised objective. The two-step approach leads to much better solutions.

The weights of the DNNs are initialized by this unsupervised pre-training instead of random weight initialization. In the pre-training step, each layer is considered as a restricted Boltzmann machine and its weights are found by a method called contrastive divergence (see the next section). It was a breakthrough and marked as the beginning of exploding success of deep learning. Some researchers believed that the resurgence of artificial intelligence is primarily due to the recent advances in deep learning and DNNs [45, 46].

3.4 Restricted Boltzmann Machines

A restricted Boltzmann machine (RBM) [47] is an undirected probabilistic graph model as shown in Figure 4.1. It consists of n visible units $\mathbf{v} = (v_1, \ldots, v_n)$ and m hidden units $\mathbf{h} = (h_1, \ldots, h_m)$. Assuming that both $\mathbf{v}$ and $\mathbf{h}$ are binary random variables, we have $v_i \in \{0, 1\}$ and $h_j \in \{0, 1\}$. The energy function is defined as:

$$E(\mathbf{v}, \mathbf{h}|\theta) = -\sum_{i=1}^{n} a_i v_i - \sum_{j=1}^{m} b_j h_j - \sum_{i=1}^{n}\sum_{j=1}^{m} v_i W_{ij} h_j, \tag{3.19}$$

where $\theta = \{W_{ij}, a_i, b_j\}$, W_{ij} is a real valued weight, a_i and b_j are real valued bias terms associated with the i-th visible unit and the j-th hidden unit, respectively.

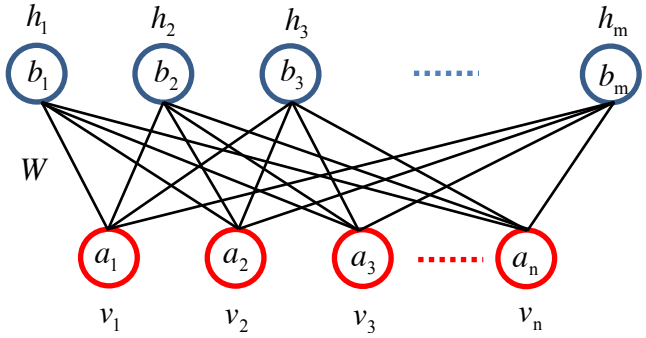


Figure 3.3: The undirected graph of an RBM

The network assigns a probability to every possible pair of a visible and a hidden vector via the energy function. Thus, the joint probability distribution of $(\mathbf{v}, \mathbf{h})$ is:

$$P(\mathbf{v}, \mathbf{h}|\theta) = \frac{e^{-E(\mathbf{v},\mathbf{h}|\theta)}}{Z(\theta)}, \tag{3.20}$$

where $Z(\theta) = \sum_{\mathbf{v},\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h}|\theta)}$ is the partition function. The marginal distribution of $\mathbf{v}$ is given by:

$$p(\mathbf{v}|\theta) = \frac{1}{Z(\theta)} \sum_{\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h}|\theta)}. \tag{3.21}$$

For most applications, it is difficult to evaluate $Z(\theta)$ exactly because the computational complexity is very high.

Because there is no intra-layer connections in an RBM, the conditional probability that a hidden unit is on is independent of other hidden units. Thus, the conditional probabilities of individual units are:

$$P(h_j = 1|\mathbf{v}, \theta) = \sigma(b_j + \sum_i v_i W_{ij}), \tag{3.22}$$

$$P(v_i = 1|\mathbf{h}, \theta) = \sigma(a_i + \sum_j W_{ij} h_j), \tag{3.23}$$

where $\sigma(x) = \frac{1}{1+\exp(-x)}$.

The learning algorithm of RBMs is based on gradient ascent on the log-likelihood (i.e., gradient-based maximization of the likelihood). According to Eq. 3.21, the log-likelihood of a training vector $\mathbf{v}$ is:

$$\log p(\mathbf{v}|\theta) = \log \frac{1}{Z(\theta)} \sum_{\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h}|\theta)} = \log \sum_{\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h}|\theta)} - \log \sum_{\mathbf{v},\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h}|\theta)}, \quad (3.24)$$

and the derivative (or gradient) of the likelihood is:

$$\begin{aligned} \frac{\partial}{\partial\theta}(\log p(\mathbf{v}|\theta)) &= \frac{\partial}{\partial\theta}(\log \sum_{\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h}|\theta)}) - \frac{\partial}{\partial\theta}(\log \sum_{\mathbf{v},\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h}|\theta)}) \\ &= -\sum_{\mathbf{h}} p(h|\mathbf{v}) \frac{\partial E(v,h)}{\partial\theta} + \sum_{\mathbf{v},\mathbf{h}} p(\mathbf{v},\mathbf{h}) \frac{\partial E(v,h)}{\partial\theta}. \end{aligned} \quad (3.25)$$

The derivative of the log likelihood of a training vector with respect to the weight W_{ij} is given by [44]:

$$\frac{\partial \log p(\mathbf{v}|\theta)}{\partial W_{ij}} = \langle v_i h_j \rangle_{\text{data}} - \langle v_i h_j \rangle_{\text{model}}, \quad (3.26)$$

where $\langle v_i h_j \rangle_{\text{distribution}}$ denotes an expectation under a distribution. The stochastic gradient descent is performed, then we have:

$$\Delta W_{ij} = \eta(\langle v_i h_j \rangle_{\text{data}} - \langle v_i h_j \rangle_{\text{model}}), \quad (3.27)$$

where η is the learning rate.

In Eq. 3.27, the expectation $\langle v_i h_j \rangle_{\text{data}}$ can be easily obtained from training data v_i's and h_j are sampled from Eq. 3.22 given v_i. However, the calculation of an unbiased sample under the distribution defined by the model $\langle v_i h_j \rangle_{\text{model}}$ is inefficient [48, 49].

In 2002, Hinton [50] proposed the contrastive divergence (CD) algorithm to approximate $\langle v_i h_j \rangle_{\text{model}}$. Given the observation $\mathbf{v}^{(0)}$, a binary vector $\mathbf{h}^{(0)}$ in the hidden layer is obtained by sampling the distribution in Eq. 3.22, i.e., $\mathbf{h}^{(0)} \sim p(\mathbf{h}|\mathbf{v}^{(0)}, \theta)$.

Then, using Eq. 3.23, the "reconstruction" $\mathbf{v}^{(1)}$ of the visible layer is computed, i.e., $\mathbf{v}^{(1)} \sim p(\mathbf{v}|\mathbf{h}^{(0)}, \theta)$. Using $\mathbf{v}^{(1)}$, the binary vector in the hidden layer is then obtained through sampling Eq. 3.22, i.e., $\mathbf{h}^{(1)} \sim p(\mathbf{h}|\mathbf{v}^{(1)}, \theta)$. The weight update rule becomes:

$$\begin{aligned} \Delta W_{ij} &= \eta(\langle v_i h_j \rangle_{\text{data}} - \langle v_i h_j \rangle_{\text{reconstruction}}) \\ &= \eta(\langle v_i^{(0)} h_j^{(0)} \rangle - \langle v_i^{(1)} h_j^{(1)} \rangle). \end{aligned} \tag{3.28}$$

With the CD algorithm, the learning of RBM is much faster. In [44], Hinton and salakhutdinov proposed a strategy to construct a DNN by stacking layers of RBMs, and the pre-training process of DNNs consists of learning a stack of RBMs. Figure 3.4 shows how to use the stacked RBMs to create a DNN. In Figure 3.4(a), the weight sets (W_1, W_2 and W_3) can be obtained after training three RBMs. In Figure 3.4(b), the stacked RBMs are used to build a DNN with three hidden layers. W_1, W_2 and W_3 are used as the initial weights of the DNN, which are then fine-tuned by the backpropagation algorithm.

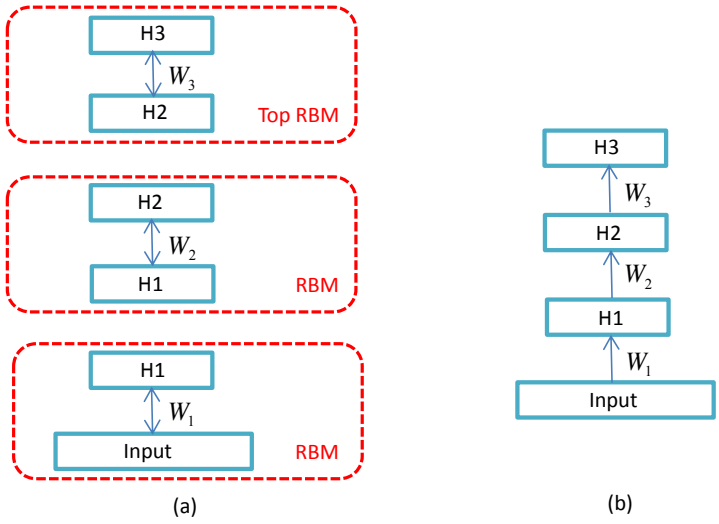


Figure 3.4: Using stacked RBMs to create a DNN

Chapter 4

SVMS VERSUS DNNS FOR ECG CLASSIFICATION

4.1 Introduction

Previous work based on the UCI dataset either optimized the hyper-parameters of the feature extractors and classifiers using test data (e.g., [23,25]) or provided a single random split of the benchmark dataset into a training set and a test set (e.g., [23,25, 28]). These experimental settings make comparison of methods difficult. In this thesis, we perform 10-fold cross validations on the dataset and repeat the cross-validation a number of times, each with a different random split of the dataset. Therefore, unlike previous work, this dissertation reports not only the classification accuracy but also its range in these repeated cross-validation runs. This thesis also investigates various feature pre-processing methods, including Fisher discriminant ratio (FDR) [51] and PCA, and various classification methods, including SVMs and deep neural networks (DNNs). More importantly, we investigate which feature pre-processing methods are appropriate for which classification methods. Performance evaluations on the UCI benchmark dataset suggests that feature selection together with deep neural networks achieve the best performance.

4.2 Methodology

4.2.1 Preprocessing: Missing Entries

It is not uncommon for biological data to contain missing values and heart arrhythmia data derived from ECG signals are of no exception. For example, in the UCI

benchmark dataset, there are 408 missing entries, which account for about 0.33% of the total number of entries. In this work, we filled these missing entries with the average value of the corresponding features.

4.2.2 Preprocessing: Feature Selection

Another characteristic of heart arrhythmia data is the high dimensionality of the feature vectors. For example, in the UCI dataset, the dimension is 279 but the number of feature vectors is only 452. To address this problem, we used Fisher discriminant ratio (FDR) [51] to select relevant features and PCA to reduce the dimension of feature vectors.

FDR is a simple and effective measure of features for classification problems. For two-class problems, FDR of the j-th feature is defined as:

$$FDR(j) = \frac{\left[\mu_j^{(1)} - \mu_j^{(2)}\right]^2}{\left[\sigma_j^{(1)}\right]^2 + \left[\sigma_j^{(2)}\right]^2}, \tag{4.1}$$

where $\mu_j^{(1)}$, $\mu_j^{(2)}$, $\sigma_j^{(1)}$ and $\sigma_j^{(2)}$ represent the class-conditional means and standard derivations of the j-th feature, respectively. In Eq. 4.1, the superscript represents the class labels. For multi-class problems, we may estimate the average FDR values across all class pairs.

A high FDR implies that the corresponding feature produces large separation between different classes. Therefore, its classification capability is stronger, and it should be selected for classification. In practice, the FDR of individual features can be computed independently and ranked in descending order. We retained the features with FDR scores larger than a predefined threshold (0.001 in this work). FDR can remove all insignificant features from the data set. Performance evaluations show that dropping some irrelevant features by FDR helps the training of SVMs and boost the performance of DNNs.

4.2.3 Heartbeat Classification by SVMs

To apply SVMs for K-class classification, we constructed K one-versus-rest RBF-SVM [27, 52], one for each class. Specifically, the k-th SVM is trained to discriminate between the feature vectors of the k-th class and those of the other classes. During recognition, given an unknown vector $\mathbf{x}$, its class label is predicted according to the maximal output:

$$l(\mathbf{x}) = \arg\max_{k \in \{1,\ldots,K\}} h^k(\mathbf{x}), \tag{4.2}$$

where

$$h^k(\mathbf{x}) = \sum_{i \in SV_k} \alpha_i^k y_i^k K(\mathbf{x}, \mathbf{x}_i) + b^k \tag{4.3}$$

is the output of the k-th SVM. In Eq. 4.3, SV_k is the set of support vector indexes corresponding to the k-th SVM, $y_i^k \in \{-1, +1\}$ are the target output of the k-th SVM, α_i^k's are the Lagrange multipliers, b^k's are bias terms, and $K(\cdot, \cdot)$ is a kernel function. In this work, the radial basis function (RBF) kernel was used.

4.2.4 Heartbeat Classification by DNNs

To apply DNNs for K-class classification, we trained a DNN with several hidden layers comprising sigmoid nonlinearity and a softmax output layer comprising K outputs nodes. We applied the greedy layer-wise unsupervised training [53] to pre-train the hidden layers. Then, we fine-tuned the whole network with backpropagation. The pre-training step is very important for arrhythmia classification because the number of training vectors is typically small for this task. Details of the design will be described in Section 5.2.5. (The same network design is also used in Chapter 5.)

4.3 Experiments and Results

4.3.1 Evaluation Protocol

The UCI cardiac arrhythmia dataset is typically used for evaluating two types of ECG classification: binary and multi-class. For the former, the classifiers aim to discriminate the 245 "normal" heartbeats against the 207 "abnormal" heartbeats. For the latter, the classifiers aim to classify the 452 heart beats into six types, including the normal. Table 4.1 shows the class labels and number of samples in each class for multi-class classification. Section 4.4.3 and Section 4.4.4 report results on binary classification, whereas Section 4.4.5 reports results on multi-class classification.

Table 4.1: The class labels and number of samples in each class for multi-class classification

Class Code	Class Label	Number of Instances
01	Normal	237
02	Ischemic changes	36
04	Old Inferior Myocardial Infarction	14
06	Sinus bradycardy	24
10	Right bundle branch block	48
16	Others	18

To rigorously estimate the accuracy of different classifiers, 10-fold cross validation was performed on the dataset. For each configuration of feature pre-processing and classification, the corresponding 10-fold cross-validation was repeated 10 times, each with a random reshuffling of the samples in the dataset. Then, the average accuracy and the range of accuracy were obtained from the results of the 10 repetitions.

The SVMs are based on Mathwork's Matlab library in Bioinformatics Toolbox and the DNNs are based on G. E. Hinton's Matlab code [54].

Table 4.2: The top-10 features selected by FDR

Rank	Feature ID	FDR Score	Feature Information
1	199	0.237	QRSTA from channel AVR
2	5	0.230	Average QRS (msec.)
3	167	0.204	Amplitude of T from channel DI
4	169	0.200	QRSTA from channel DI
5	197	0.183	Amplitude of T wave from channel AVR
6	277	0.173	Amplitude of T wave from channel V6
7	91	0.155	Average width of R wave from channel V1
8	279	0.139	QRSTA from channel V6
9	179	0.125	QRSTA from channel DII
10	93	0.122	Number of intrinsic deflections from channel V1

4.3.2 Selected Features

Table 4.2 shows the top-10 features selected by FDR, i.e., features with the top-10 FDR scores. These features can be divided into five types since some of them just obtained from different channels. The five types include QRSTA, QRS duration, Amplitude of T, Average width of R, and the number of intrinsic deflections. These are the features that were found important by medical professionals [55]. Therefore, our feature selection method agrees well with the diagnostic criteria of medical doctors.

4.3.3 Performance of SVM Classifiers

For the RBF-SVMs, the hyper-parameters (RBF width σ and penalty factor C) were further optimized based on the training data in each fold. Specifically, for each fold of the 10-fold cross-validation, we applied an inner 5-fold cross validation on the training split to optimize the hyper-parameters of the RBF-SVMs. The optimal RBF-SVMs were than tested on the remaining data in the test split. In other words, we further partitioned the training split of each fold into 5 portions in the inner 5-fold cross validation. While different folds of the cross validation have different set of

Table 4.3: The average accuracy (across ten 10-fold cross-validations) of SVM classifiers with different feature pre-processing methods

Feature Pre-processing	Feature Dimension	Classification Acc. (average)
Nil (All features)	279	77.77%
FDR	236	78.23%
PCA	89	76.97%

parameters, our experience had been that setting the RBF width and penalty factor to values from 1/16 to 16 gives good performance.

Table 4.3 shows the performance of the SVM classifiers with different feature pre-processing methods. For FDR, the cut-off threshold for feature selection is 0.001, which results in 236 selected features. For PCA, we kept 95% of the variance after projection, which results in projected vectors with 89 dimensions. The results show that FDR is the best pre-processing method for SVMs and PCA degrades the performance. This is understandable because SVMs are known to be able to handle high dimensional data and PCA will inevitably remove some useful information. On the other hand, feature selection is able to keep the relevant features.

4.3.4 Performance of DNN Classifiers

Figure 4.1 show the effect of applying pre-training on a DNN with three hidden layers. For the network without pre-training, the backpropagation algorithm was applied to a DNN whose weights were initialized with small random values. On the other hand, 5 epochs of contrastive divergence (CD-1) [47] were applied to pre-train the network when pre-training was applied. The result clearly shows that pre-training can help the backpropagation algorithm to find a better solution.

Figure 4.2 shows the effect of increasing the number hidden nodes (in all hidden layers) on the classification accuracy. It shows that peak performance (80.64%) is achieved when the number of hidden nodes is 25, with the second best (80.04%)

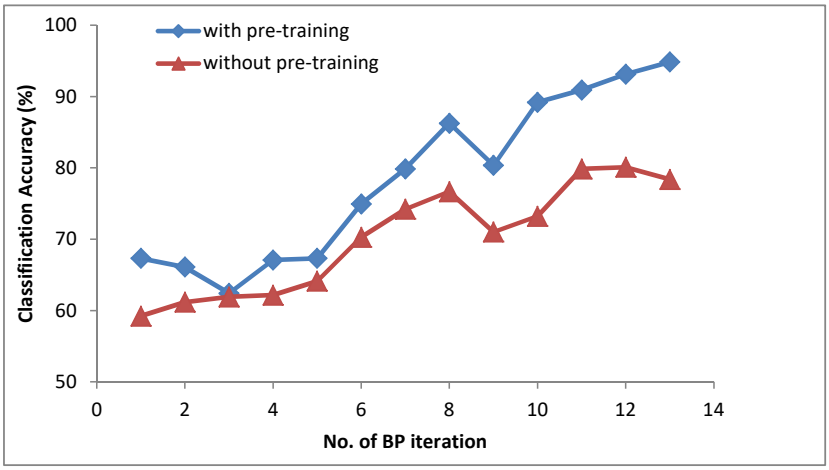


Figure 4.1: Classification accuracy of the DNN with or without pretraining

occurs at 20 nodes. Therefore, we used 25 hidden nodes per layer in the rest of the experimnets on the UCI dataset.

In order to optimize the network structure, we fixed the number of hidden nodes per layer to 25 and tried different numbers of hidden layers. According to Table 4.4, the performance becomes worse if the number of hidden layers is more than four because of the small number of training samples in this dataset.

Table 4.5 shows the performance of DNNs with different feature pre-processing methods. From the table, DNNs with FDR outperform DNNs with PCA and DNNs without any feature pre-processing (i.e., using the full features). The results also show that PCA does not work well with DNNs.

Figure 4.3 shows the range and rough distributions of the classification accuracies across the 10 runs of 10-fold cross-validation for different feature pre-processing methods combined with different classification methods. In this figure, the central mark inside each box indicates the median accuracy, and the bottom and top edges of each box indicate the 25th and 75th percentiles, respectively. The horizontal dashes represent the lowest and highest accuracies. The results in Figure 4.3 clearly show

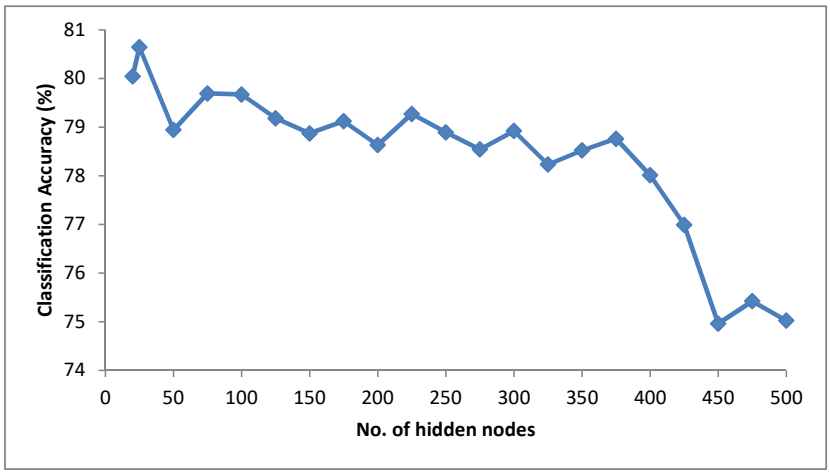


Figure 4.2: The effect of increasing the hidden nodes on the DNN

that FDR can improve the performance of DNN and SVM. However, PCA degrades their performance. Moreover, the performance of DNN is better than SVM, except when PCA is applied.

A reason for the poor performance of PCA is that it is a linear transformation method that reduces the dimensionality of data while retaining most of the variance. Therefore, PCA is not suitable when the data lie on a nonlinear manifold of the feature space. Table 4.3, Table 4.5 and Figure 4.3 suggest that PCA is not an appropriate pre-processing method for this dataset, regardless of the classification methods used. Intuitively, when the data dimension is high and the amount of training data is small (the so-called small sample-size problem), PCA should be able to reduce the dimension so that the overfitting problem can be avoided. However, our results suggest that PCA is not necessary and that overfitting does not occur in our DNNs even for such a small dataset. This is mainly because we pre-trained [44, 53] our DNNs before applying backpropagation with early stopping (20 epoches). The pre-training step provides the necessary regularization to the networks [56] and the early stopping strategy avoids overfitting (see Figure 4.4).

Table 4.4: Performance comparisons of DNNs with different numbers of hidden layers

Feature Pre-Processing	Feature Dimension	Network Structure	Acc. (average)
Nil (All Features)	279	[25]	78.11%
		[25 25]	79.00%
		[25 25 25]	79.18%
		[25 25 25 25]	79.29%
		[25 25 25 25 25]	78.25%
FDR	236	[25]	78.85%
		[25 25]	79.23%
		[25 25 25]	80.64%
		[25 25 25 25]	79.91%
		[25 25 25 25 25]	79.54%
PCA	89	[25]	74.77%
		[25 25]	74.89%
		[25 25 25]	73.65%
		[25 25 25 25]	73.50%
		[25 25 25 25 25]	71.11%

4.3.5 Comparing with Other Studies

Because there is no standard protocol for this dataset, different studies used different evaluation protocols, causing difficulty in comparing performance across studies. For examples, in [25], 30% of the data were used for training and the remaining 70% were used for testing, whereas in [28], various percentages of splitting were tried and the best result was obtained from the split where 90% of the data were used for training and the remaining 10% were used for testing. Also, these studies optimized the hyper-parameters (such as the number of hidden nodes and parameters of RBF kernels) of the classifiers based on the test set, which may give over-optimistic performance. Nevertheless, we attempt to compare our classifiers with [28] and [23] whose evaluation protocols are closest to ours.

Two-class Case: As [28] reported the best performance of its ANN, for fair comparisons, we compare its accuracy with the highest achievable accuracy of our DNNs.

Table 4.5: The average accuracy (across ten 10-fold cross-validations) of DNN classifiers with different feature pre-processing methods

Feature Pre-processing	Feature Dimension	Classification Acc. (average)
Nil (All features)	279	79.18%
FDR	236	80.64%
PCA	89	73.65%

Table 4.6: The best accuracy (across ten 10-fold cross-validations) achieved by the DNN classifiers with different feature pre-processing methods

Feature Pre-Processing with ANNs/DNNs	Feature Dimension	Classification Acc. (best)
ANNs only [28]	279	82.22%
DNNs only	279	81.42%
FDR with DNNs	236	82.96%
PCA with DNNs	89	75.22%

The results are shown in Table 4.6, which show that the performance of DNNs is comparable with that of the ANN in [28]. When relevant features have been selected, the DNN slightly outperforms the ANN in [28].

Multi-class Case: we have also compared the performance of our heart arrhythmia classifiers with those in [16] under the multi-class scenarios. We generally followed the evaluation protocol and data preparation procedures in [16] to make performance comparisons meaningful. Specifically, we followed [16] to remove the features whose values are all zeros across all samples and to remove the samples that contain missing values. After this data preparation step, 377 samples remain. These samples are distributed into 6 classes as shown in Table 4.1. By dropping Classes 04, 06 and 16, which contain a small number of samples only, the number of the classes is reduced from six to three. Similar to [16], we selected half of the samples for training and remaining half for testing. However, unlike [16], we repeated the division of data 100

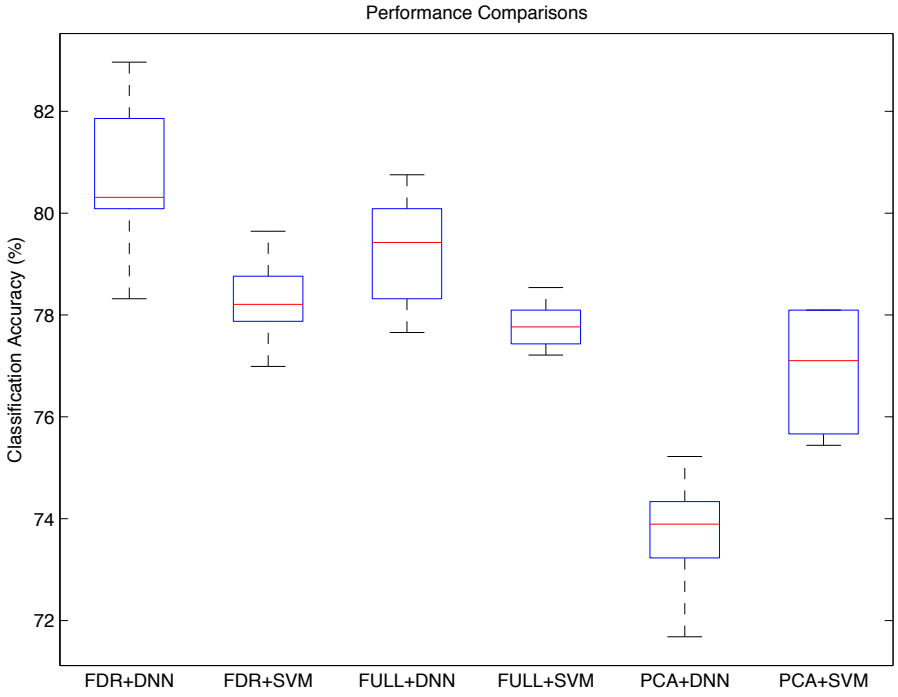


Figure 4.3: The distribution of classification accuracy of different algorithms

times, each with different training and test sets, to obtain the average accuracy.

In [16], PCA was applied to reduce the dimension of feature vectors. In this work, we not only applied PCA to reduce dimension but also used FDR to select relevant features. Although FDR is originally designed for binary classification problems, it can be easily adopted to the multi-class scenarios by noting that each SVM in the one-versus-rest SVM classifier is a binary classifier. Therefore, for a K-class problem, there will be K sets of FDR-selected features, one set for each SVM. While this strategy works very well for one-versus-rest SVM classifiers, it is not applicable to the DNN classifiers. Therefore, we did not use DNNs for comparison.

Table 4.7 shows the performance of the SVM classifiers and the best arrhythmia classifier in [16] under the 6-class and 3-class scenarios. Note that in Table 4.1, Classes 04, 06 and 16 contain a small number of samples only. By dropping these classes,

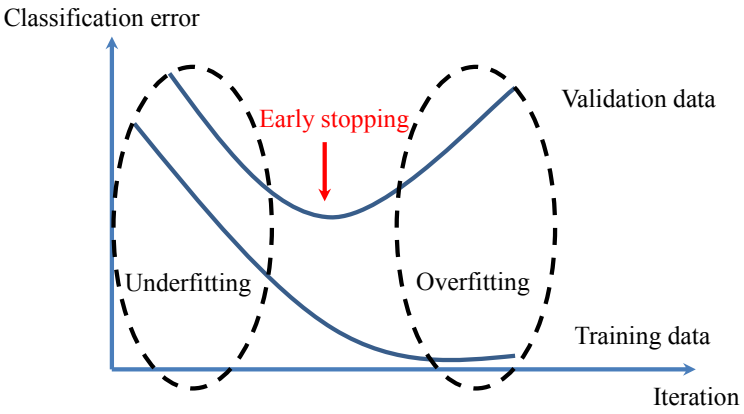


Figure 4.4: The strategy of early stopping

we reduce the 6-class problem to a 3-class one. Two conclusions can be drawn from Table 4.7. First, FDR not only reduces the feature dimension but also helps the SVM classifier to achieve better performance. Second, our classifier outperforms the best classifier in [16].

Table 4.7: Performance of the best SVM classifier in [16] and the SVM classifiers in this study

Feature Pre-processing	Feature Dimension	Classification Acc. (best)
Nil [16]	166	75.0%
Nil	245	77.77%
FDR	236	78.23%
PCA	80	76.97%

(a) 6-class Case

Feature Pre-processing	Feature Dimension	Classification Acc. (best)
Nil [16]	166	78.13%
PCA [16]	70	83.71%
Nil	245	86.15%
FDR	236	86.26%
PCA	77	85.04%

(b) 3-class Case

Chapter 5

END-TO-END ECG CLASSIFICATION

5.1 Introduction

We propose an end-to-end method with a deep neural network (DNN) for both feature extraction and classification based on aligned heartbeats. This method obviates the need to handcraft the features and produces optimized ECG representation for heartbeat classification. Through the performance investigation using the MIT-BIH arrhythmia database, the proposed method performs better than current state-of-the-art methods.

5.2 Methodology

In this section, we first explain the motivation to build an end-to-end ECG classifier, and then provide a system overview of the classifier. Next, we describe the deep neural network inside the classifier, and then explain the heartbeat segmentation and alignment procedures in the classifier. Finally, we summarize the advantages of the proposed method.

5.2.1 Motivation

In most previous works [1, 3, 6, 7, 10–12], handcrafted feature vectors were extracted from the QRS complex of heartbeats because this region is thought to contain most ECG pulse information. However, previous studies [57–59] have shown that the P and T waves also contain important information relevant to heart arrhythmias. In light of this observation, we proposed to use raw ECG waveforms as the input of a deep

neural network (DNN) classifier, which we refer to as end-to-end ECG classification. The advantage of using raw ECG waveforms is that the QRS complex and P and T waves can be included in the extracted heartbeats so that better representations can be obtained for classification.

While both of our proposed DNNs and the CNNs in [7, 13–15] use raw ECG signals as input, our raw signal extraction method has two advantages over them. First, instead of simply cropping equal numbers of time points from the left and right of an R-peak as in [15], we align the heartbeats to ensure that the input to the DNN contains the QRS complex, the P wave and the T wave. Second, to fix the input dimension, the method in [7, 13, 14] upsamples or downsamples the raw ECG signals to certain time-points per beat, which may cause information loss. In contrast, our alignment method allows the DNN to fully utilize the information in the ECG signals by keeping more time-points (417 in this work, which will be described in Section 5.2.3) per beat.

5.2.2 System Overview

This work proposes an end-to-end ECG classification system shown in Figure 5.1. The system receives raw ECG signals at one end and produces beat-by-beat classification decisions at the other end. In the figure, preprocessing refers to the process of extracting heartbeats from continuous ECG signals, which involves heartbeat segmentation and alignment. The DNN in Figure 5.1 is used for both feature extraction and classification, which are achieved by the lower part and the upper part of the network, respectively. The design of the DNN is discussed in the Section 5.2.5.

To extract fixed-length feature vectors from raw ECG signals, two steps must be performed: (1) heartbeat segmentation and (2) heartbeat alignment. These two steps will be described in the next subsections.

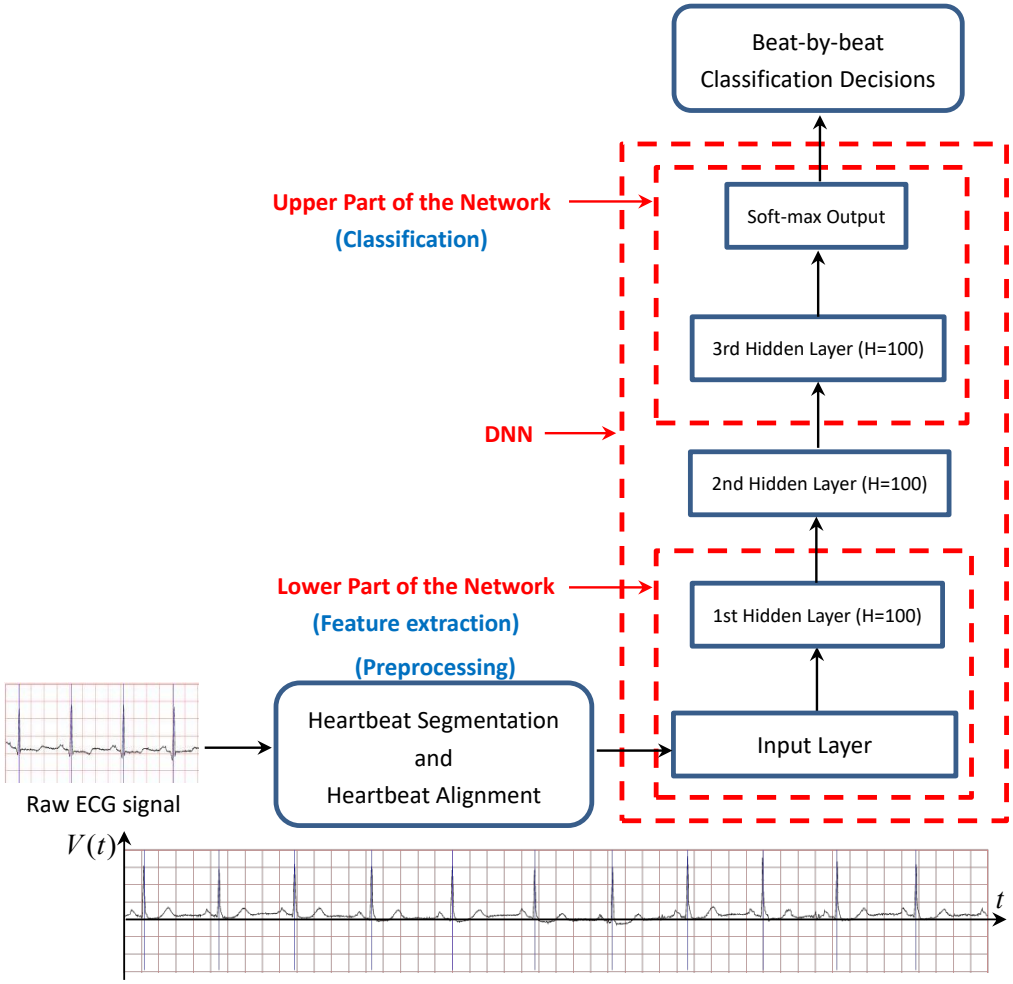


Figure 5.1: End-to-End heartbeat classification system.

5.2.3 Preprocessing: Heartbeat Segmentation

The bottom of Figure 5.1 shows a continuous ECG signal in the MIT-BIH arrhythmia database. To extract a complete heartbeat from the ECG signal, we need to define what a complete heartbeat is and then perform heartbeat segmentation. Since the R peak usually occurs around the middle of a heartbeat, we can use it as an anchor point for locating a complete heartbeat. The positions of R peaks can be accurately determined (over 99%) by using the Pan-Tompkins algorithm [60]. We assume that

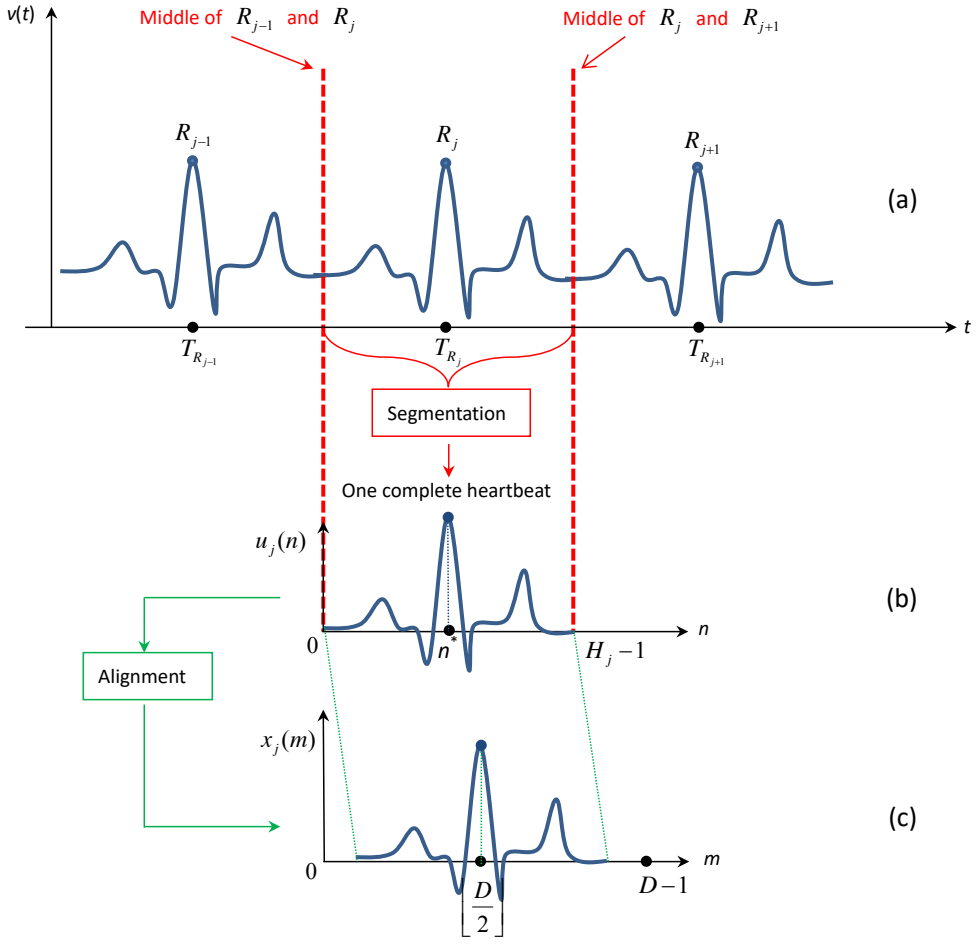


Figure 5.2: Hypothetical example illustrating the heartbeat segmentation and alignment processes. In (c), $\lfloor a \rfloor$ means the integer (floor) of a.

the R peak is located at the center of its corresponding heartbeat, and thus the boundary of a complete heartbeat is assumed to lie on the middle of two successive R peaks. Based on this assumption, a complete heartbeat comprises the sample points between the two middle points of three consecutive R peaks. Figure 5.2(a) shows an example of a complete heartbeat and its relationship with its preceding and succeeding heartbeats. In Figure 5.2(a), t indexes the sample points of an ECG signal, $v(t)$ is the voltage (in mV) of the ECG signal at time index t, R_j is the j-th R peak, and T_{R_j} is the time index of R_j.

After heartbeat segmentation, we obtain the j-th complete heartbeat $\mathcal{H}_j$, which is an integer set containing sample points between $\left\lfloor \frac{1}{2}(T_{R_{j-1}} + T_{R_j}) \right\rfloor$ and $\left\lfloor \frac{1}{2}(T_{R_j} + T_{R_{j+1}}) \right\rfloor$, where $\lfloor a \rfloor$ means the integer (floor) of a. As illustrated in Figure 5.2(b), the elements in $\mathcal{H}_j$ are indexed by $n = 0, \ldots, H_j - 1$, where H_j is the number of sample points in the complete heartbeat. More precisely, we have

$$H_j = \left\lfloor \frac{1}{2}(T_{R_j} + T_{R_{j+1}}) \right\rfloor - \left\lfloor \frac{1}{2}(T_{R_{j-1}} + T_{R_j}) \right\rfloor + 1. \tag{5.1}$$

We may use a vector $\mathbf{u}_j$ to represent $\mathcal{H}_j$ as follows:

$$\mathbf{u}_j = [u_j(0), \ldots, u_j(n^*), \ldots, u_j(H_j - 1)]^\mathsf{T}, \tag{5.2}$$

where $n^* = T_{R_j} - \left\lfloor \frac{1}{2}(T_{R_{j-1}} + T_{R_j}) \right\rfloor$ is the time index corresponding to the peak in $\mathbf{u}_j$.

However, $\mathbf{u}_j$ still cannot be directly used for training a DNN because the number of sample points is not a constant (the duration of each complete heartbeat is not the same). A fixed number of samples (D) needs to be set for each heartbeat. Thus, we measured the durations of all segmented heartbeats and found a value that is larger than 95% of all durations. In our experiments, D was found to be 417 and this value was applied to all of the completed heartbeats.

5.2.4 Preprocessing: Heartbeat Alignment

Because we use the R peak as the anchor point of a heartbeat in the heartbeat segmentation process, it is necessary to align it to the midpoint of the D consecutive time points of each heartbeat. Figure 5.2(b) and Figure 5.2(c) show the alignment process. We extract samples from $\mathbf{u}_j$ in Eq. 5.2 to produce a feature vector

$$\mathbf{x}_j = [x_j(0), \ldots, x_j(D-1)]^\mathsf{T} \tag{5.3}$$

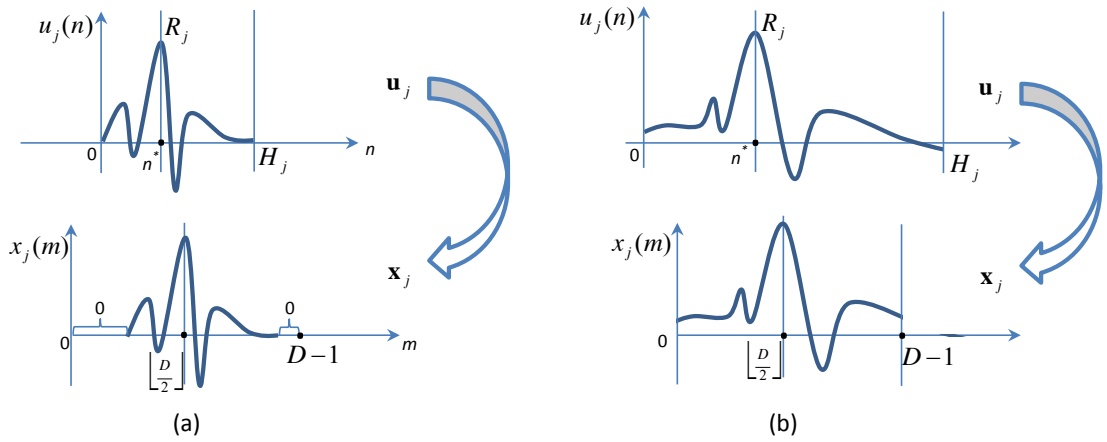


Figure 5.3: Creating feature vector $\mathbf{x}_j$ from $\mathbf{u}_j$ by aligning sample $u_j(n^*)$ to the midpoint of $\mathbf{x}_j$. (a) Example of zero-padding, $H_j < D$. (b) Example of truncation, $H_j > D$.

such that the $\lfloor \frac{D}{2} \rfloor$-th element in $\mathbf{x}_j$ is aligned to n^*-th element in Eq. 5.2. Note that this procedure requires zero padding and sample truncation for most heartbeats. Specifically, when $H_j > D$, we may need to truncate some of the samples in the head or tail or both the head and tail of $\mathbf{u}_j$. However, when $H_j < D$, we may need to pad zeros to the head or tail or both the head and tail of $\mathbf{u}_j$. In some rare cases, both zero padding and sample truncation need to be performed. Figure 5.3 shows some examples of the alignment process. Given Eq. 5.2 and Eq. 5.3, the alignment process can be implemented as follows:

$$x_j(m) = \begin{cases} 0, & \text{if} \quad m < \lfloor \frac{D}{2} \rfloor - n^* \\ & \text{or} \quad m > \lfloor \frac{D}{2} \rfloor + (H_j - n^*) \\ u_j(m - \lfloor \frac{D}{2} \rfloor + n^*), & \text{otherwise} \end{cases} \tag{5.4}$$

where $m = 0, 1, \ldots, D-1$.

After heartbeat segmentation and alignment, the set of feature vectors in a dataset is denoted as

$$\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_j, \ldots, \mathbf{x}_N\}, \tag{5.5}$$

where $\mathbf{x}_1$ and $\mathbf{x}_N$ correspond to the second and the second last beats in a record, and N is the number of complete heartbeats.[1]

The process of heartbeat alignment is vital to the high performance of the end-to-end DNN (see results in Section 5.6). Because the DNN receives time-domain ECG signals as input, its internal structure represents not only the pulse shapes of heartbeats but also their relative positions along the time axis. Without the R-peak alignment, the R peak in Figure 5.2(c) could be in many possible locations, causing high variability in the feature vectors. By aligning the R peak to the mid-point of $x_j(m)$ in Figure 5.2(c), we essentially make the DNN invariant to the phase shift of the ECG signals.

5.2.5 Design of Deep Neural Networks

To apply DNNs for K-class classification, we can construct a DNN with $L-1$ hidden layers and a softmax output layer with K output nodes. Specifically, denote $a_k^{(L)}$ as the activation of the k-th neuron in the softmax layer, where $k = 1, \ldots, K$, the softmax function gives the outputs:

$$y_k = \frac{\exp\left\{a_k^{(L)}\right\}}{\sum_{j=1}^{K} \exp\left\{a_j^{(L)}\right\}}, \qquad k = 1, \ldots, K. \tag{5.6}$$

With the softmax function, the outputs can be considered as the posterior probabilities of individual classes given an input vector $\mathbf{x}$, i.e., $y_k = P(\text{Class} = k|\mathbf{x})$. The activation $a_k^{(L)}$ is the linear weighted sum of the hidden nodes' output at the $(L-1)$-th hidden layer.

The weights in the hidden layers can be pre-trained by a greedy layer-wise unsupervised training process [53] in which each hidden layer is considered as a restricted

[1] A GitHub page (https://github.com/seanssx) has been created for other researchers to download the implementation the procedure.

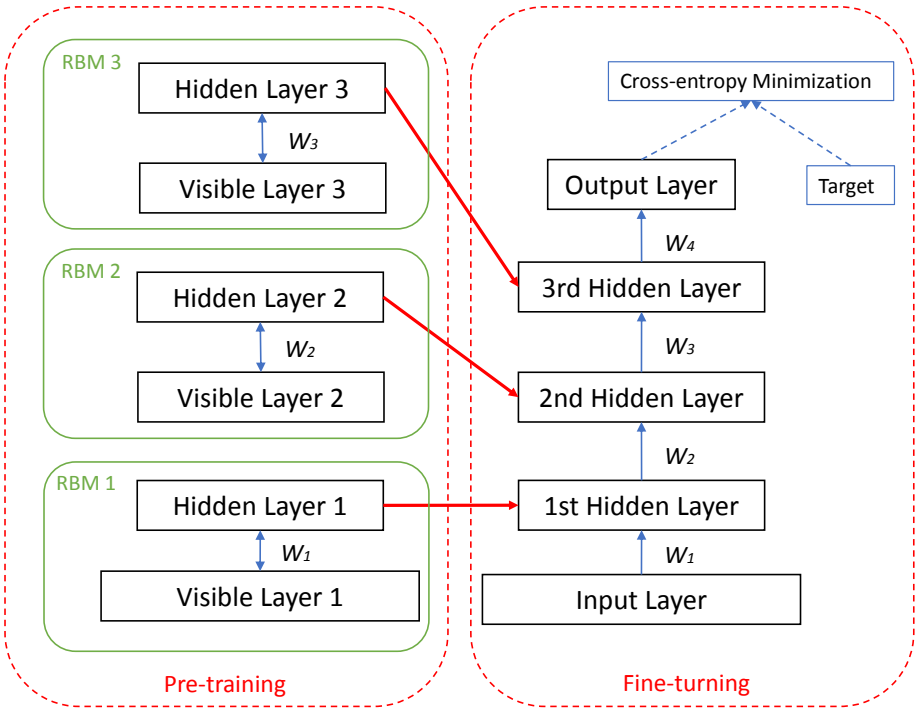


Figure 5.4: DNN with stacked RBMs.

Boltzmann machine (RBM) [44, 46] whose weights are optimized by the contrastive divergence algorithm [48]. Alternatively, the weights can be initialized by the Xavier initializer [61]. Then, the backpropagation algorithm is used to fine-tune the whole network by minimizing the cross-entropy error between the target outputs and the actual outputs:

$$E_{\text{ce}} = -\sum_{n}\sum_{k=1}^{K} t_{n,k} \log y_{n,k}, \tag{5.7}$$

where $y_{n,k}$ is the actual output of node k, n indexes the training vectors in a mini-batch, and $t_{n,k} \in \{0, 1\}$ are the target outputs which follow the one-hot encoding scheme.

In this work, we used a DNN with stacked RBMs as shown in Figure 5.4. The RBM at the bottom layer has Gaussian visible nodes and Bernoulli hidden nodes. The remaining RBMs have Bernoulli distributions in both visible and hidden layers.

During fine-tuning, the pre-trained weights (W_1, W_2 and W_3) were used as the initial weights and the weights between the upper two layers (W_4) were initialized with small random numbers. In addition, 30% of the training set was used for computing the accuracy of the network after every epoch, so that early stopping can be applied to prevent overfitting. Note that the pre-training step can provide necessary regularization to the network [56] and the early stopping strategy provides guidance on how many iterations should be run before the model begins to over-fit the training data. We found that without the pre-training (i.e,. BP with random weights initialization), the classification accuracy dropped 20%. Moreover, the DNN failed to converge to a solution when the number of hidden layers was increased to five.

5.3 Experimental Setting

In this section, we first introduce some issues concerning our implementation (i.e., evaluation protocol). Then, since DNN performance is greatly affected by its network structure, we describe how we find the optimized network structure of the DNN used in the classifier.

5.3.1 Evaluation Protocol

In compliance with the AAMI recommended practice, four recordings containing paced beats were removed from the dataset. The remaining 44 records were split into two datasets (DS1 and DS2),[2] with each dataset containing approximately 50,000 beats from 22 recordings. Note that this way of splitting the data had also been used in [1], [3] and [4]. Following their evaluation protocols (the subject-oriented evaluation scheme), we applied 22-fold cross validation on DS1 in one experiment (Exp. 1) and

[2]DS1 contains data from ECG recordings 101, 106, 108, 109, 112, 114, 115, 116, 118, 119, 122, 124, 201, 203, 205, 207, 208, 209, 215, 220, 223 and 230; DS2 contains data from ECG recordings 100, 103, 105, 111, 113, 117, 121, 123, 200, 202, 210, 212, 213, 214, 219, 221, 222, 228, 231, 232, 233 and 234.

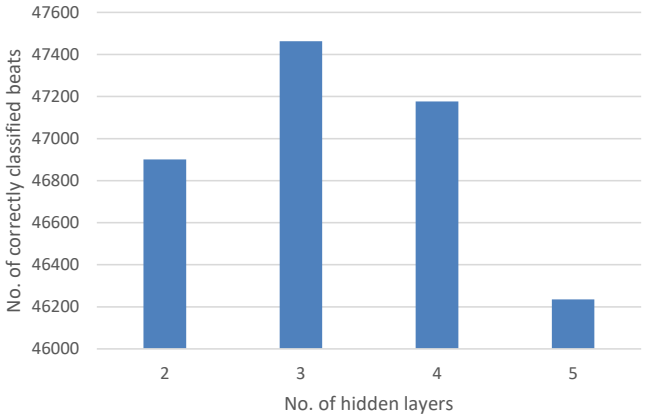
used DS1 as the training set and DS2 as the test set in another experiment (Exp. 2). Note that in Exp. 1, each record was used as test data in sequence and the other 21 records were used as training data. Such process was repeated 22 times so that each record had been used once as the test data. As a result, we may compare our results with previous studies.

As suggested by the ANSI/AAMI EC57 standard [22], we focused on evaluating the classification performance of the two majority arrhythmia classes (Classes S and V). Among the performance indicators for medical diagnoses, sensitivity (Sen) and specificity (Spe) are two important measures of the diagnostic accuracy of a test because a highly sensitive test can be useful for ruling out a disease if a person has a negative result, whereas a highly specific test can be useful for ruling in patients who have a certain disease. Some medical publications [31, 32] recommend clinicians to choose the most sensitive diagnostic test to rule out disease and the most specific diagnostic test to rule in disease. Therefore, in this work, the diagnostic performance on Class S and Class V was measured using Sen and Spe. Besides, because the overall accuracy measures the overall system performance over all classes, it was also used in this work.

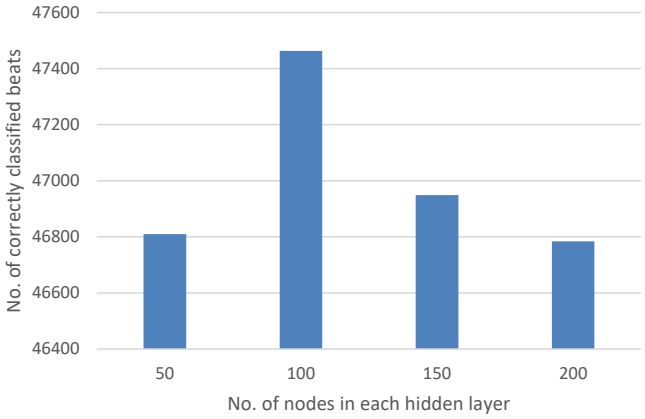
In this work, we evaluated the performance of the proposed method on the MIT-BIH arrhythmia database and followed the ANSI/AAMI EC57 standard to compare with those in [1,3,4,6,7,11,12]. Since the classifiers in [1,3,6] are patient-independent, the proposed method can be compared directly. We have also compared the performance of the proposed method with those in [4, 7, 12] though their classifiers are patient-dependent. Study [11] was chosen for performance comparison because it uses DNNs as classifiers.

5.3.2 Network Structure

Figure 5.5(a) shows the effect of increasing the number of hidden layers on the network's ability to classify heartbeats. 22-fold cross validation was applied to DS1 and



(a) Increase hidden layers



(b) Different number of hidden nodes

Figure 5.5: Network structure optimization

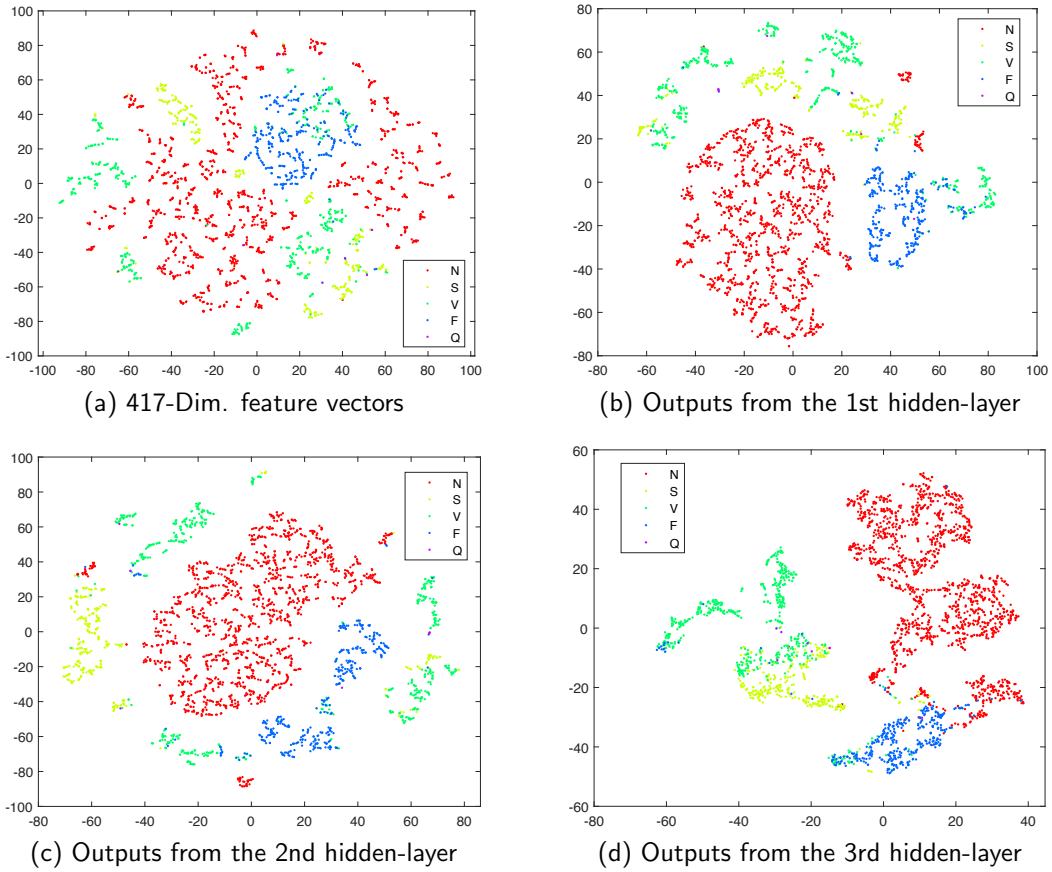
the total number of heartbeats used for evaluation was 50,977. The results show that the performance was the best when the number of hidden layer was 3. To further optimize the network structure, we fixed the number of hidden layers to 3 and varied the numbers of hidden nodes (e.g., 50, 100, 150 and 200) per layer. According to Figure 5.5(b), the best performance was obtained when the number of hidden nodes equaled to 100. Therefore, in subsequent experiments, the DNN classifier contained 3 hidden layers and each layer had 100 nodes.

5.4 Performance Investigation

This section first presents the feature extraction capability of our proposed method, then it describes and compares our method's performance with current start-of-the-art methods. Next, we compare our proposed method with other DNN classifiers including comparing our proposed method with patient-specific classifiers. Finally, we summarize our findings.

5.4.1 Hidden Node Representation

The t-distributed stochastic neighbor embedding (t-SNE) [62] is a nonlinear dimension reduction method for visualizing high-dimensional data on a two or three-dimensional space. Using the 417-dimensional vectors as input, we extracted the outputs from the first, second and third hidden layers of the DNN. We applied t-SNE on the 417-dimensional vectors and different hidden layers. The results are shown in Figure 5.6 for Classes N, S, V, F and Q. To allow a good visualization, the number of samples of Class N is reduced in the figures. No obvious clusters can be observed in the feature vectors (Figure 5.6(a)). When we progressively move up the hidden layers (Figs. 5.6(b)–(c)), the clustering property becomes apparent. However, in the first two hidden layers, each class still has multiple clusters, meaning that further nonlinear operations are required. In the third hidden layer (Figure 5.6(d)), the clusters are very obvious, and more importantly each class has fewer clusters and the clusters of different classes become more separated. This means that from the bottom to the top layers, the representation becomes more and more discriminative. From another perspective, the hidden layers progressively disentangle the class information from the ECG signals, making the representation of the final layer very discriminative. Unlike the conventional handcrafted features, the feature extraction process in the DNN is purely data-driven, without any expert knowledge.



(a) 417-Dim. feature vectors

(b) Outputs from the 1st hidden-layer

(c) Outputs from the 2nd hidden-layer

(d) Outputs from the 3rd hidden-layer

Figure 5.6: t-SNE plots of input feature vectors and hidden-layer outputs

5.4.2 Performance of End-to-End ECG Classification

We applied the aligned feature vectors $\mathbf{x}_i$'s as described in Section 5.2.4 to train a DNN. We set $D = 417$ for all vectors, i.e., the DNN has 417 inputs and 5 output nodes, each output node corresponds to one class in Table 2.3. We used sigmoid nonlinearity in the hidden layers. Stochastic mini-batch (batch size of 128) gradient descent was used in the backpropagation fine-tuning. The learning rate, momentum and maximum number of iterations were set to 0.001, 0.5 and 50, respectively. The DNN has three hidden layers with a structure 417–100–100–100–5. Four experiments were conducted to evaluate the end-to-end approach.

Table 5.1: Performance of the classifiers in [1] and our end-to-end classifier (Exp. 1)

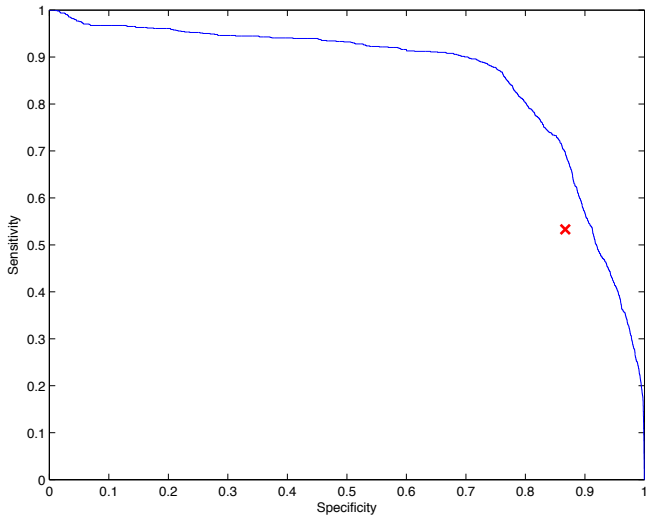
		Chazal *et al.* [1]	Proposed method
Overall accuracy		84.5%	93.1%
Class S (S *vs.* non-S)	Sen	53.3%	69.7%
	Spe	86.7%	86.7%
Class V (V *vs.* non-V)	Sen	67.7%	88.8%
	Spe	86.7%	86.7%

22-Fold Cross Validation

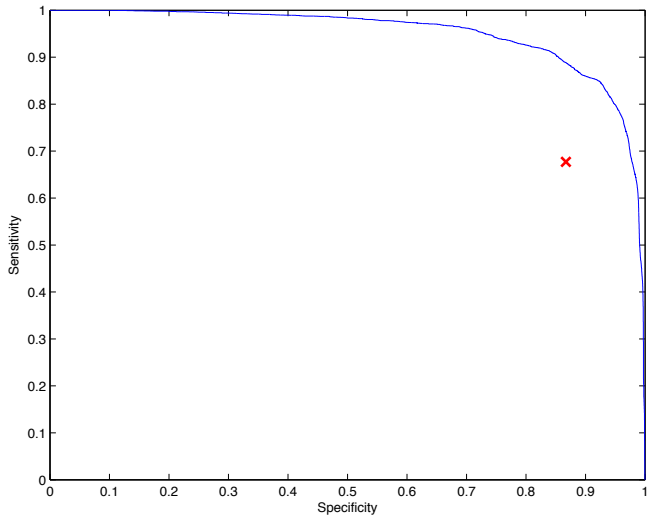
In the first experiment, 22-fold cross validation was applied to DS1. Table 5.1 compares the performance of [1] with that of the end-to-end DNN classifier. Note that the proportion of Classes F and Q in the dataset is very small (less than 1%). Thus, the classification performance on these two classes has insignificant contribution to the overall performance. On the other hand, the proportion of Classes S and V is much higher (about 10%) and these two classes contain the majority of arrhythmias. Therefore, we focused on these two classes. To improve the classification performance of Classes S and V, Chazal *et al.* [1] investigated different combinations of feature sets. For simplicity, their best results are shown in Table 5.1. As can be seen, the overall accuracy of the end-to-end DNN is much higher than that of [1]. In particular, at the same specificity, our DNN achieves a much higher sensitivity for both Class S and Class V.

The MCC of Classes N, S, V, F and Q achieved by the end-to-end classifier are 0.67, 0.26, 0.67, 0.01 and 0, respectively. To obtain a more balanced MCC performance of the end-to-end classifier, a constant (δ) was added to the output nodes corresponding to Classes S and F so that the classifier has a higher chance of correctly classifying the instances of Classes S and F. Through cross validation on DS1, we found that $\delta = 0.997$ can increase the MCC of Class F from 0.01 to 0.20 without significantly sacrificing the performance of the other classes. More precisely, when $\delta = 0.997$ was

added to the outputs of Classes S and F, the MCC of Classes N, S, V, F and Q become 0.59, 0.34, 0.51, 0.20 and 0, respectively.



(a) Class S *vs.* non S (AUC = 0.858)



(b) Class V *vs.* non V (AUC = 0.951)

Figure 5.7: ROC curves (Sen vs. Spe) of the end-to-end classifier in Exp. 1. Red markers correspond to the best performance in [1]. *AUC*: Area under the ROC curve [2].

Note that this work does not aim to optimize the solution to solve the data-imbalance problem [63], which is a branch of machine learning research. Therefore, we propose the above simple solution to handle this issue so that the classifier has a greater chance of correctly classifying the instances of the minority classes. The goal is to obtain more balanced MCC values for performance comparison across the five classes. Through the performance investigation, we find that the MCC performance changed to become acceptable. Thus, the above simple solution is sufficient in this work. Actually, we had oversampled the minority classes to deal with this data-imbalance issue. Specifically, we randomly duplicated samples in the minority classes to ensure that the number of instances of each class was balanced in each mini-batch. However, the results were poorer than our current approach.

Figure 5.7 shows the ROC curves of the end-to-end classifier for Class S and Class V. Also shown are the operating points (the red $\times$) of the best performing classifier in [1]. Figure 6.8 clearly shows that the sensitivity-specificity points in [1] are below the ROC curves of our DNN, suggesting that with a certain range of decision thresholds our DNN achieves better performance (in term of both sensitivity and specificity) than the classifier in [1].

In this experiment, the accuracy of Record 203 was very low (55.9%, the worst case). This record is special in MIT-BIH in that it has the following note [21]:

> "The PVCs are multiform. There are QRS morphology changes in the upper channel due to axis shifts. There is considerable noise in both channels, including muscle artefact and baseline shifts. This is a very difficult record, even for humans."

We suspect that the special characteristics of the heartbeats in this record are not well represented by the training data (in the other 21 records). To further investigate this issue, we randomly selected 10% of the instances (299 instances) in each class of Record 203, and then added them to the training set. Note that the original number of

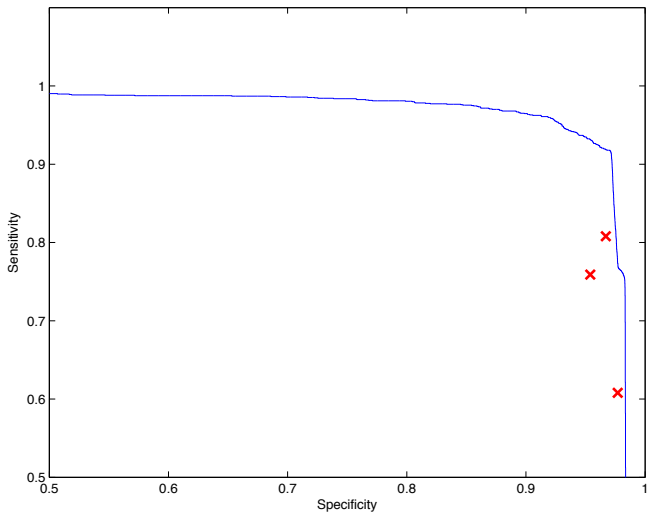
Table 5.2: Performance of the classifiers in [1, 3, 4] and our end-to-end classifier (Exp. 2)

		[1]	[3]	[4]	Proposed
Overall ACC		85.9%	86.4%	89.9%	94.7%
Class S (S *vs.* non-S)	Sen	75.9%	60.8%	80.8%	77.3%
	Spe	95.4%	97.7%	96.7%	97.7%
Class V (V *vs.* non-V)	Sen	77.7%	81.5%	82.2%	93.7%
	Spe	98.8%	96.4%	99.0%	98.8%

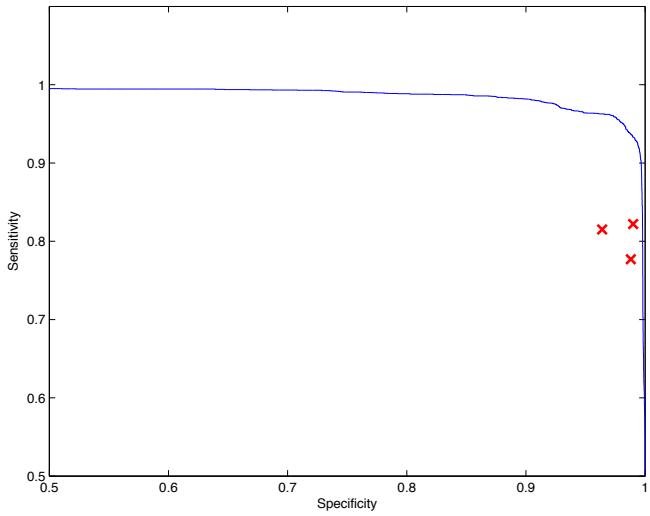
training instances was 47,999, which is much larger than 299. We did the experiment again, and the classification accuracy increased from 55.9% to 89.3%.

Test on 22 ECG recordings

In the second experiment, DS1 and DS2 were used as the training set and test set, respectively. Table 5.2 shows the performance of the end-to-end classifier and the best results in [1], [3] and [4]. Similar to the results in Exp. 1, the overall accuracy of our approach is much higher than that of [1], [3] and [4]. The end-to-end DNN not only achieves a much higher overall accuracy than that of [1], [3] and [4], it also yields a higher sensitivity and specificity for Class S and Class V. Figure 5.8 shows the ROC curves of the end-to-end classifier in this experiment. It shows that the best performance in [1], [3] and [4] are below the ROCs of our DNN, which suggests that the end-to-end approach is very promising.



(a) Class S *vs.* non S (AUC = 0.970)



(b) Class V *vs.* non V (AUC = 0.991)

Figure 5.8: ROC curves (Sen vs. Spe) of the end-to-end classifier in Exp. 2. Red crosses correspond to the best performance in $[1, 3, 4]$.

Table 5.3 shows the MCC performance of the classifiers in $[1, 3, 4]$ and our end-to-end classifier. OMCC in the table refers to overall MCC of the five classes. For

Table 5.3: MCC performance of the classifiers in [1,3,4] and our end-to-end classifier (Exp. 2)

Method	Class					OMCC
	N	S	V	F	Q	
Chazal *et al.* [1]	0.61	0.52	0.78	0.26	0	0.82
Ye *et al.* [3]	0.57	0.54	0.68	0.05	0	0.83
Raj *et al.* [4]	0.69	0.61	0.82	0.33	0	0.87
Proposed	0.69	0.67	0.91	0.22	0	0.88

Table 5.4: Performance of the classifiers in [11] and our end-to-end classifier

	Jun *et al.* [11]	Proposed method
Overall accuracy	99.41%	99.70%
Sen of class PVC	96.08%	97.68%
Spe of class PVC	Did not specify	99.89%

the end-to-end classifier, the MCC values were obtained by adding the constant ($\delta = 0.997$) found in Exp. 1 to the outputs of Classes S and F. More precisely, we applied cross validation on the training set (DS1) to find an appropriate value for boosting the outputs of Classes S and F to balance the MCC across the five classes. The results show that the MCC performance of the end-to-end classifier is much better than that in [1] and [3]. Good performance is not only found in Classes N, S, and V, but also in the overall. Compared with [4], our MCC performance is still better except for Class F.

Binary Classification

Jun *et al.* [11] proposed using a 6-hidden-layer DNN for PVC beat detection based on the MIT-BIH arrhythmia database. This is a two-class problem in which normal and PVC (NOR and PVC in Table 2.2) heartbeats were extracted for evaluation. In contrast to our raw signal extraction, six handcrafted features were used to represent a heartbeat including R-peak amplitude, RR interval, QRS duration, ventricular

activation time, Q-peak amplitude and S-peak amplitude. K-fold cross validation was used to evaluate performance and the performance is optimal when K equals 8. Note that, although 8-fold cross validation was applied in [11], the heartbeats in the cross validation training set and test set could belong to the same patient. However, our previous experiment (Exp. 1) is based on leave-one-subject/patient-out cross validation.

In our experiment, $81,379$ heartbeats were retrieved from the dataset, including $74,478$ normal heartbeats and $6,901$ PVC heartbeats. To make a fair comparison, we also performed 8-fold cross validations. The DNN has the same structure (417–100–100–100–2) as before except for the number of output nodes. Table 5.4 shows the best performance of the classifier in [11] and our end-to-end DNNs. Although the overall accuracy in [11] is high (99.41%), ours (99.70%) is 0.29% higher. Moreover, at very high specificity (99.89%), the sensitivity of the proposed method for Class PVC is still higher than in [11]. Compared with the five-class classification in the previous subsection, this two-class problem is much easier. Not only is the overall accuracy close to 100%, but good performance of detecting PVC beats can also be obtained.

Patient-Independent vs. Patient-Specific ECG Classification Systems

Table 5.5 shows how the patient-specific ECG classification systems in [6,7,12] and our patient-independent end-to-end ECG classification system performed. We followed the experimental protocols in [6], [7] and [12]. For the patient-specific classifiers with expert intervention [6, 7] (Mode 1), to be as fair as possible, we used the first 5 minutes of ECG records (Record No.: 200–234) of 24 patients for training our patient-independent classifier. To evaluate the performance of the classifiers on "seen" patients, we used the remaining 25 minutes of ECG signals of these 24 patients for testing. Note that we used 5 minutes of ECG signals of 24 patients to train a patient-independent classifier. For the patient-specific classifiers without expert intervention [12] (Mode 2), to evaluate the performance of the classifiers on "unseen" patients, we

Table 5.5: Performance comparisons between the patient-specific classification systems in [6, 7, 12] and our patient-independent classification system on seen patients and unseen patients.

		Test on the remaining 25 min. ECG of 24 seen patients			Test on 22 unseen patients	
		Patient-specific classifiers with expert intervention (Mode 1)		Proposed method	Patient-specific classifiers without expert intervention (Mode 2)	Proposed method
		[6]	[7]		[12]	
Class S	Sen	62.1%	64.6%	66.2%	61.4%	61.4%
	Spe	98.5%	98.6%	98.6%	99.8%	98.3%
Class V	Sen	83.4%	95.0%	90.5%	91.8%	91.8%
	Spe	98.1%	98.1%	98.1%	99.9%	99.5%

trained a patient-independent classifier based on the ECG records of 22 patients in DS1, and tested the classifier on the other 22 patients in DS2. Table 5.5 shows that despite patient independency, our patient-independent classifiers achieve comparable performance with the patient-specific classifiers in [6,7,12], as evident in the fifth and seventh columns in the table. Bear in mind that any patient-specific classifier requires some patient-specific data or an expensive annotation process for each new patient, therefore our patient-independent classifier definitely has advantages.

5.5 Advantages and Limitations of End-to-End ECG Classification

The following are advantages of the proposed end-to-end ECG classification method:

1. By using raw-signal extraction and DNNs, the classification performance of our end-to-end system was found to be much better than existing patient-independent systems in terms of sensitivity-vs-specificity ROC and Mathews correlation coefficients; besides that, without expert intervention, its performance is still comparable to patient-specific systems.

2. The end-to-end DNN can perform feature extraction and classification at the

same time. Traditional feature extraction methods are limited by the professional knowledge of medical doctors. The end-to-end DNN can overcome such limitation by using aligned raw ECG waveforms as input so that better representations can be obtained for classification.

Note that the classification performance of the proposed method may not be much better than patient-specific classifiers because patient-specific classifiers have patient-specific data, which may be helpful in machine learning. However, the proposed algorithm is a patient-independent classifier which is universal, and it does not need patient-specific data and expert intervention for new patients.

Chapter 6

I-VECTOR ADAPTED PATIENT-SPECIFIC DNNS

6.1 Introduction

To address the patient-dependent variability in the ECG signals, we have developed a deep neural network (DNN) based heartbeat classifier [64] that is adaptive to the ECG characteristics of individual patients. The adaptation is achieved by using the i-vector representation [65] of patient-specific ECG as auxiliary information to adjust the weights in the DNN. This chapter is an extension of our earlier work in [64]. It provides additional analyses to explain why the i-vectors can help adapt the patient-independent DNN. In particular, new experiments have been performed to investigate the best layer for injecting the i-vectors. Visualizations of the network activities during the course of adaptation are provided to demonstrate the effectiveness of i-vector adaptation. Through these investigations, we are able to explain why this i-vector adaptation can lead to patient-specific classifiers that outperform other state-of-the-art patient-specific classifiers.

In general, the amount of ECG data from the general population is much larger than that from individual patients for adapting the classifiers. Therefore, the adapted patient-specific classifiers may be biased towards the patterns in the general population. To overcome this issue, in [5–8], the patient-specific classifiers were trained based on common and patient-specific beats. Specifically, the common heartbeats were randomly sampled from the corresponding classes of the general population in [5–7] while an automatic selection method was proposed to select the most representative beats in different classes in [8]. After all, the number of selected common beats was limited

to a few hundred only. In [9,12], instead of using the ECG of the entire population, a subset was selected for training the general classifier. However, reducing the amount of data from the general population is not a desirable way to address the issue because it throws away lots of useful information in the ECG data. Also, the common training data are useful when the patient-specific beats contain a few arrhythmia patterns only [21].

In our adaptation method, all of the ECG data from the general population are used for training a patient-independent DNN as shown in Figure 6.1(a). Then, for each patient, an i-vector is extracted from his/her 5-minute ECG data. As shown in Figure 6.1(b), to form a patient-specific classifier, the i-vector is used as another input to the middle layer of the patient-independent DNN and the whole network is fine-tuned by backpropagation. The patient-independent and patient-specific DNNs represent general population knowledge and specific personal knowledge, respectively [12]. The advantage of the method is that it can leverage all of the ECG data in the general population but still be able to adapt to the ECG characteristics of individual patients through the patient-specific ECG and the patients' i-vectors.

6.2 Methodology

This section first explain why i-vectors can be used for representing patient-specific information and outline the i-vector extraction process. Then, we present the proposed iAP-DNNs, specifically, showing the architecture of a patient-independent DNN and describing how to migrate it to a patient-specific DNN. In the patient-specific DNN, we not only make use of patient-specific data but also i-vectors of the patient for patient adaption. Thus, we also introduce the procedure to extract an i-vector from a particular patient and describe how to embed the i-vector into the adaption. Finally, we discuss the advantages and limitations of the iAP-DNNs.

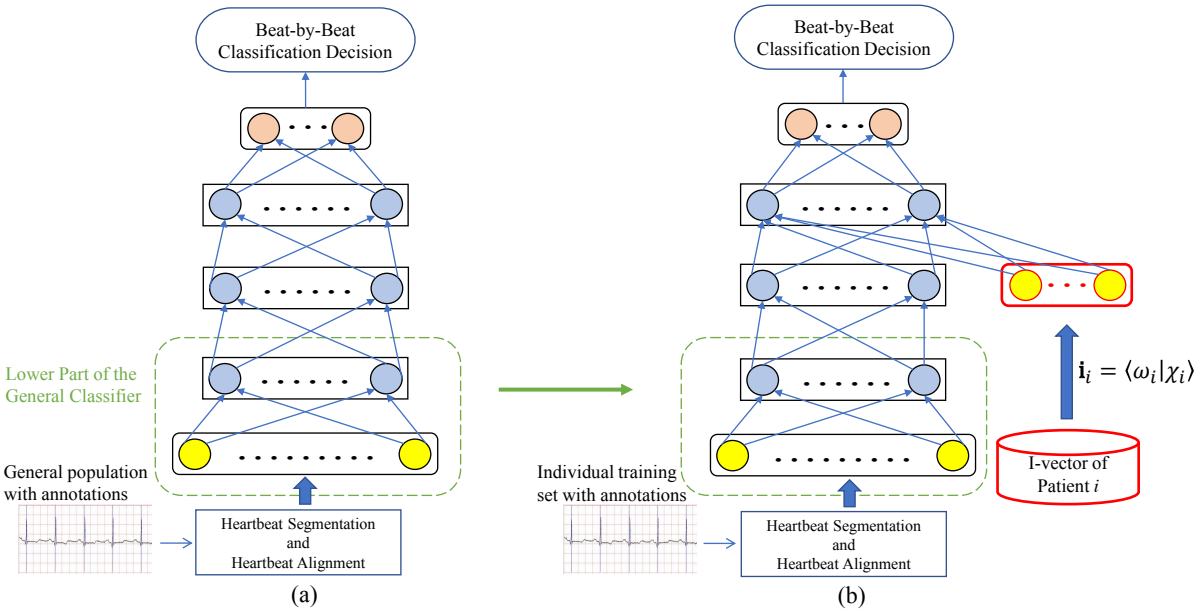


Figure 6.1: I-vector adapted patient-specific DNNs (iAP-DNNs). (a) General classier. (b) Patient-specific classifier.

6.2.1 Motivation: I-vector an ECG Representation

Figure 6.2 demonstrates why i-vectors are good for representing patient-dependent information, which makes them ideal for adapting ECG classifiers. In the figure, each marker corresponds to one patient and each point of the same marker corresponds to an i-vector extracted from an ECG record of that patient. Totally, there are five patients, each has five ECG records. For ease of visualization, the i-vectors were projected onto an embedding space created by the t-SNE (t-distributed stochastic neighbor embedding) software [62]. T-SNE is a nonlinear dimension reduction method for visualizing high-dimensional data on a two- or three-dimensional space. Apparently, the i-vectors of the same patient are close to each other, i.e., forming patient-specific clusters in the t-SNE space. This clustering phenomenon suggests that the i-vectors can capture patient-specific information, which is very useful for adapting ECG classifiers.

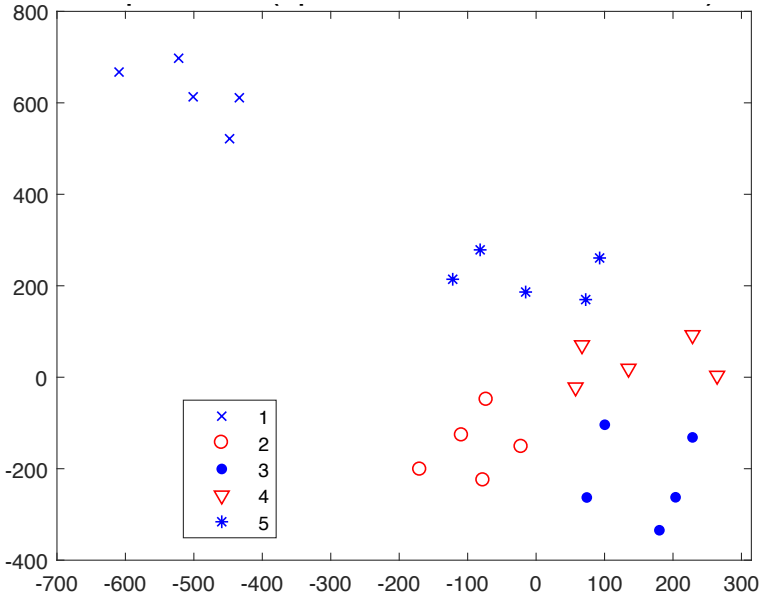


Figure 6.2: The i-vectors of five patients projected onto a 2-D t-SNE embedded space. Each patient is represented by one marker and each point represents an i-vector. Patient-dependent clusters are apparent.

Note that it is possible to learn an alternative representation instead of the i-vectors to capture patient-specific information. Recently, DNN embeddings have been widely used to learn low-dimensional representations in speaker recognition [66, 67]. It is also found that the DNN embeddings can make better use of large-scale training data than the i-vectors. However, the available ECG data are quite limited compared with the speech data. The overfitting problem may easily occur during the training of the DNN embeddings, resulting in poor performance of ECG classification.

6.2.2 I-vector Extraction

The idea of i-vectors is based on the factor analysis method that compresses speaker and channel information into a low-dimensional subspace [68]. Inspired by the success of i-vectors in representing speaker information, we applied i-vectors to represent patient-specific information in ECG signals.

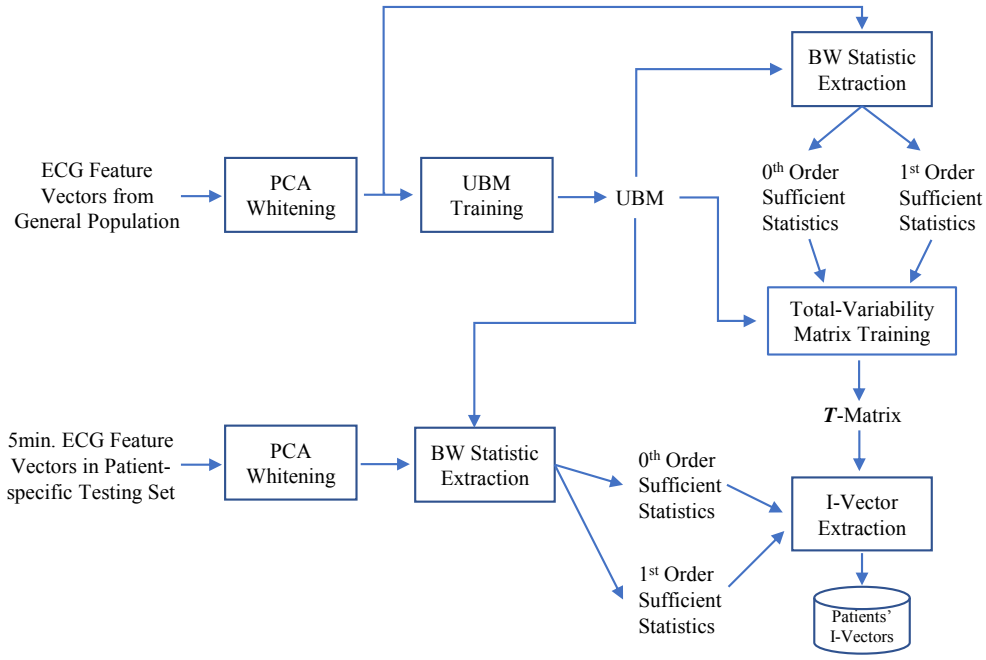


Figure 6.3: Training of i-vector extractor and i-vector extraction process.

Figure 6.3 illustrates the procedure of training an i-vector extractor given a set of ECG data from a general population; it also shows the process of extracting an i-vector from an ECG record. First, PCA whitening is applied to reduce the correlation among the time-points in the ECG vectors [69]. Then, the whitened ECG vectors from the general population are used to train a Gaussian mixture model, which we referred to as the universal background model (UBM). The ECG data are then aligned with the UBM to compute the 0th- and 1st-order sufficient statistics (Baum-Welch statistics), from which a total variability matrix (T-matrix) is trained. To extract an i-vector, the same processing pipeline is applied (see the lower branch of Figure 6.3) to an ECG record to compute the sufficient statistics. Given the T-matrix and the sufficient statistics, an i-vector representing the whole ECG record can be obtained. In the sequel, we outline the formulae for training an i-vector extractor and the i-vector extraction process. For detailed derivations, readers may refer to [70].

Given the i-th ECG record from a general population, we extract the D-dimensional heartbeat vectors $\mathcal{X}_i = \{\mathbf{x}_{i1}, \ldots, \mathbf{x}_{iT_i}\}$ from the record, where T_i is the number of complete heartbeats in the record.[1] We assume that the ECG vectors from this record are generated by a C-mixture GMM with parameters $\Lambda_i = \{\pi_c, \boldsymbol{\mu}_{ic}, \boldsymbol{\Sigma}_c\}_{c=1}^{C}$, i.e.,

$$p(\mathbf{x}_{it}) = \sum_{c=1}^{C} \pi_c^{(b)} \mathcal{N}(\mathbf{x}_{it} | \boldsymbol{\mu}_{ic}, \boldsymbol{\Sigma}_c^{(b)}), \quad t = 1, \ldots, T_i. \tag{6.1}$$

In Eq. 6.1, we assume that $\pi_c^{(b)}$ and $\boldsymbol{\Sigma}_c^{(b)}$ are tied across all ECG records and are equal to the mixture weights and covariance matrices of the UBM, respectively.

In the i-vector framework [65], the mean vectors $\{\boldsymbol{\mu}_{ic}\}_{c=1}^{C}$ are stacked to form a GMM-supervector [68] $\boldsymbol{\mu}_i = [\boldsymbol{\mu}_{i1}^{\mathsf{T}} \quad \ldots \quad \boldsymbol{\mu}_{iC}^{\mathsf{T}}]^{\mathsf{T}}$, which is assumed to be generated by the following factor analysis model [71]:

$$\boldsymbol{\mu}_i = \boldsymbol{\mu}^{(b)} + \boldsymbol{T}\boldsymbol{w}_i, \tag{6.2}$$

where $\boldsymbol{\mu}^{(b)}$ is obtained by stacking the mean vectors of the UBM, $\boldsymbol{T}$ is a $CD \times R$ low-rank total variability matrix modeling all sort of variability in the ECG vectors, and $\boldsymbol{w}_i \in \Re^R$ comprises the latent (total) factors. Eq. 6.2 suggests that the generated supervectors $\boldsymbol{\mu}_i$'s have mean $\boldsymbol{\mu}^{(b)}$ and covariance matrix $\boldsymbol{T}\boldsymbol{T}^{\mathsf{T}}$. Eq. 6.2 can also be written in a component-wise form:

$$\boldsymbol{\mu}_{ic} = \boldsymbol{\mu}_c^{(b)} + \boldsymbol{T}_c \boldsymbol{w}_i, \quad c = 1, \ldots, C \tag{6.3}$$

where $\boldsymbol{\mu}_{ic} \in \Re^D$ is the c-th sub-vector of $\boldsymbol{\mu}_i$ (similarly for $\boldsymbol{\mu}_c^{(b)}$) and $\boldsymbol{T}_c$ is a $D \times R$ sub-matrix of $\boldsymbol{T}$.

In the i-vector framework, every ECG record is assumed to be obtained from a different patient. As a result, the ECG vectors of Record i aligning to mixture c

[1] See Section 5.2.3 for the definition of complete heartbeats.

have mean $\boldsymbol{\mu}_{ic}$ and covariance matrix $\boldsymbol{\Sigma}_c^{(b)}$. This matrix measures the deviation of the ECG vectors associated with the c-th mixture from $\boldsymbol{\mu}_{ic}$. In practice, $\boldsymbol{\mu}_c^{(b)}$ and $\boldsymbol{\Sigma}_c^{(b)}$ are the mean vectors and covariance matrices of the UBM. As a result, we only need to estimate the T-matrix $\boldsymbol{T}$ from a set of training ECG vectors.

Assume that there are P ECG recordings from the general population. The T-matrix can be estimated according to the expectation-maximization (EM) algorithm as follows [70]:

- E-step:

$$\langle \boldsymbol{w}_i | \mathcal{X}_i \rangle = \boldsymbol{L}_i^{-1} \sum_{c=1}^{C} \boldsymbol{T}_c^{\mathsf{T}} (\boldsymbol{\Sigma}_c^{(b)})^{-1} \tilde{\boldsymbol{f}}_{ic}, \tag{6.4}$$

$$\langle \boldsymbol{w}_i \boldsymbol{w}_i^{\mathsf{T}} | \mathcal{X}_i \rangle = \boldsymbol{L}_i^{-1} + \langle \boldsymbol{w}_i | \mathcal{X}_i \rangle \langle \boldsymbol{w}_i | \mathcal{X}_i \rangle^{\mathsf{T}}, \tag{6.5}$$

$$\boldsymbol{L}_i = \boldsymbol{I} + \sum_{c=1}^{C} \boldsymbol{N}_{ic} \boldsymbol{T}_c^{\mathsf{T}} (\boldsymbol{\Sigma}_c^{(b)})^{-1} \boldsymbol{T}_c; \tag{6.6}$$

- M-step:

$$\boldsymbol{T}_c = \left[\sum\nolimits_i \tilde{\boldsymbol{f}}_{ic} \langle \boldsymbol{w}_i | \mathcal{X}_i \rangle^{\mathsf{T}} \right] \left[\sum\nolimits_i \boldsymbol{N}_{ic} \langle \boldsymbol{w}_i \boldsymbol{w}_i^{\mathsf{T}} | \mathcal{X}_i \rangle \right]^{-1}, \tag{6.7}$$

where $i = 1, \ldots, P$, $\langle \cdot | \cdot \rangle$ is the conditional expectation and $\boldsymbol{T}_c$ is the c-th partition of $\boldsymbol{T}$. The 0th-order and the 1st-order Baum-Welch statistics in Eq. 6.4, Eq. 6.6 and Eq. 6.7 can be computed as follows:

$$\begin{aligned} \boldsymbol{N}_{ic} &= \sum\nolimits_t \gamma_c(\mathbf{x}_{it}), \\ \tilde{\boldsymbol{f}}_{ic} &= \sum\nolimits_t \gamma_c(\mathbf{x}_{it})(\mathbf{x}_{it} - \boldsymbol{\mu}_c^{(b)}), \end{aligned} \tag{6.8}$$

where $\gamma_c(\mathbf{x}_{it})$ is the posterior probability of mixture c.

The i-vector $\mathbf{i}_i \equiv \langle \boldsymbol{w}_i | \mathcal{X}_i \rangle$ representing the i-th ECG recording can be computed according to Eq. 6.4. Figure 6.4 details the procedure of extracting the i-vector $\mathbf{i}_i$ from the i-th ECG recording.

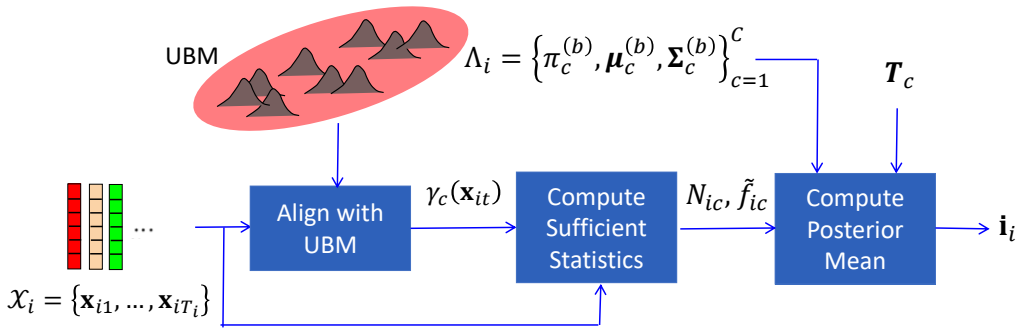


Figure 6.4: Procedure of extracting an i-vector from an ECG recording: after raw signal extraction, the input vectors are aligned with the UBM to compute the posteriors. Then, we can obtain the Baum-Welch statistics. With the trained T matrix, the i-vector can be calculated.

6.2.3 Patient-Independent DNN (General Classifier)

Figure 6.1(a) shows the architecture a patient-independent ECG classifier. It is essentially a DNN with fixed-length ECG waveforms as the input and heartbeat types as the output. The fixed-length waveforms can be obtained by the segmentation and alignment process described in Section 5.2.3, Section 5.2.4 and [69].

To apply DNNs for M-class classification, we can construct a DNN with $L-1$ hidden layers and a softmax output layer with M output nodes. Specifically, denote $a_m^{(L)}$ as the activation of the m-th neuron in the softmax layer, where $m = 1, \ldots, M$, the softmax function gives the outputs:

$$y_m = \frac{\exp\left\{a_m^{(L)}\right\}}{\sum_{j=1}^{M} \exp\left\{a_j^{(L)}\right\}}, \qquad m = 1, \ldots, M. \tag{6.9}$$

With the softmax function, the outputs can be considered as the posterior probabilities of individual classes given an input vector $\mathbf{x}$, i.e., $y_m \equiv P(\text{Class} = m|\mathbf{x})$. The activation $a_m^{(L)}$ is the linear weighted sum of the hidden nodes' output at the $(L-1)$-th hidden layer.

The patient-independent DNN is trained by the backpropagation algorithm using the ECG data of a number of patients in the general population.

6.2.4 Patient-Specific DNN

To create a patient-specific classifier, the weights in the lower part of the general classifier in Figure 6.1(a) are retained and the weights in the upper part are randomized. Then, for each patient, five minutes of his/her ECG data are presented to the input and an i-vector extracted from these 5-minute ECG data is injected into the middle layer of the patient-independent DNN, as shown in Figure 6.1(b). The whole network is then fine-tuned by backpropagation. The backpropagation algorithm will encourage the upper layers to represent patient-dependent ECG information at a more abstract level. This results in the output layer being tuned to the characteristics of the corresponding patient. The i-vector extracted from the training ECG of a patient is applied to adapt the patient-independent DNN to a patient-dependent DNN. The same i-vector will also be used as an auxiliary input to the adapted DNN (Figure 6.1(b)) during testing. This means that the identity of the patient is assumed to be known during testing. But this assumption is reasonable in clinical settings.

The i-vector is presented to the second hidden layer instead of the first hidden layer because it is well known that the feature representation becomes increasingly abstract when moving up the network [72]. For example, in DNN-based speech recognition, the bottom layers can capture low-level acoustic features that vary significantly across different speakers and the upper layers can capture high-level features that are less speaker dependent [73]. This suggests that the upper layer can implicitly normalize the features across speakers. By the same token, the upper layers of the DNN in Figure 6.1(a) will produce patient-invariant features, which is not good for patient-specific classification. This explains why it is necessary to use the patient-dependent i-vector to adapt the network. To check the correctness of the above justification, the patient's i-vector was injected into different hidden layers of the network and the

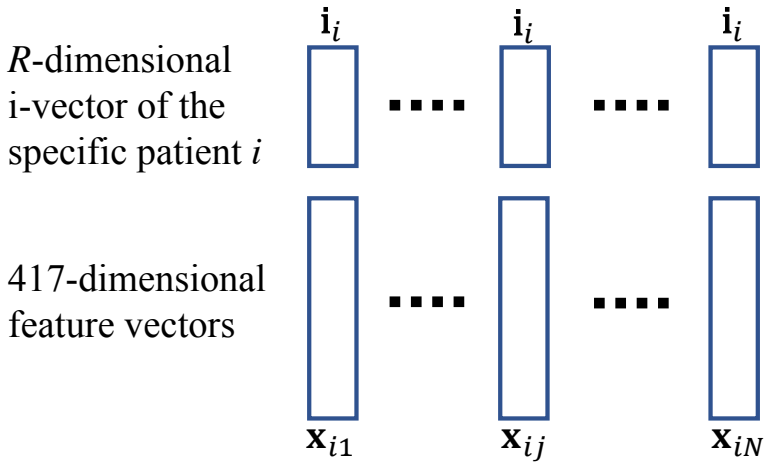


Figure 6.5: Repetition of an i-vector to match the number of ECG vectors for each patient. Vectors in top row will be injected into the middle layer of the DNN. Vectors in the bottom row are the input of the DNN.

results will be shown in Section 6.4.1.

Each patient has a number of heartbeat vectors. Specifically, for the r-th patient, his/her heartbeat vectors are denoted as $\mathcal{X}_r = \{\mathbf{x}_{r1}, \ldots, \mathbf{x}_{rT_r}\}$, where T_r is the number of heartbeats from this patient. On the other hand, each patient has one i-vector only, which is extracted from $\mathcal{X}_r$ using Eq. 6.4, i.e., $\mathbf{i}_r = \langle \boldsymbol{w}_r | \mathcal{X}_r \rangle$. The backpropagation algorithm, however, requires one input vector for every output vector. To overcome this imbalance in the number of input vectors, we repeated the same i-vector for each ECG vector, as shown in Figure 6.5.

Once the DNN has been adapted, it can be used for classifying the ECG of the corresponding patient in a beat-by-beat basis. Specifically, given a test ECG waveform of the patient, its heartbeats are segmented and aligned to form 417-dimensional heartbeat vectors [69]. The heartbeat vectors are presented to the input of the DNN. Meanwhile, the i-vector of this patient is retrieved from the i-vector repository (see Figure 6.1(b)). For each heartbeat vector, the i-vector is replicated and presented to the middle layer of the DNN. The outputs of the DNN are then averaged over all

of the heartbeat vectors to obtain the posterior probability of individual heartbeat classes.

6.3 Experimental Setting

This section first introduces the evaluation protocol in our experiments. Then, we describe some issues concerning the implementation (i.e., DNN structure and DNN training).

6.3.1 Evaluation Protocol

As suggested by the ANSI/AAMI EC57 standard [22], we focused on evaluating the classification performance of two majority arrhythmia classes (Class S and Class V). Besides, four ECG recordings (Record IDs 102, 104, 107 and 217), which contain paced beats, were excluded. As a result, a total of 44 recordings were used for performance evaluation.

We have conducted two experiments (Exp. 1 and Exp. 2) to compare the performance of the iAP-DNNs with six state-of-the-art patient-specific classifiers [5–9, 12]. For fair comparisons, we followed the experimental protocols described in these studies. The purposes of the data used in these two experiments are detailed as follows:

- **Exp. 1**: The first experiment aims to evaluate the performance of iAP-DNNs for classifying both Class S and Class V at the same time. To this end, we used 20 recordings (Record ID starting with Digit 1) for training the patient-independent DNN and another 24 recordings (Record ID starting with Digit 2) for adaptation and testing. This means that we have 24 test patients and 24 patient-specific DNNs, each was adapted (either fine-tuning only or fine-tuning plus i-vector adaptation) by using the initial 5 minutes of his/her ECG recording. The remaining 25 minutes in the 24 recordings were used for performance evaluation.

- **Exp. 2**: The second experiment aims to evaluate the performance of iAP-DNNs in detecting S beats and V beats separately. To this end, we used 14 recordings (Record IDs 200, 202, 210, 212, 213, 214, 219, 221, 222, 228, 231, 232, 233 and 234) for adaptation and testing of S-beat detection and 11 recordings (Record IDs 200, 202, 210, 213, 214, 219, 221, 228, 231, 233 and 234) for adaptation and testing of V-beat detection. As for the training, we used the remaining 30 recordings. Similar to Exp. 1, only the initial 5 minutes of these recordings were used for adaptation and the remaining were used for performance evaluation.

6.3.2 DNN Structure and DNN Training

The general classifier has three hidden layers with a structure 417–100–100–100–5. The Glorot uniform initializer [61] was used to initialize the weights of the patient-independent DNN and the upper layers of the patient-specific DNNs. We used the rectified linear unit (ReLU) in the hidden layers. The Adam optimizer [74] with default parameters was used for stochastic mini-batch (batch size of 128) gradient descent. Batch normalization and dropout were employed to train the DNNs. A dropout layer was added between the input and the first hidden layer, and the dropout rate was set to 20%. In addition, 30% of the training set was reserved for validating the performance of the network after every epoch, so that early stopping can be applied to prevent overfitting. The early stopping strategy provides guidance on how many iterations should be run before the model begins to overfit the training data. The maximum number of epochs used for both patient-independent training and patient-specific training was set to 50. To train the i-vector extractor, we investigated different numbers of mixture components in the UBM (e.g., 16 and 20) and different i-vector dimensions (e.g., 32, 64 and 128), and the optimal combination was found to be 20 and 64 for the number of mixtures and i-vector dimension, respectively. We used

Table 6.1: Performance of iAP-DNNs, with the i-vector being injected into different hidden layers of the network (Figure 6.1(b)). "Correctly Classified" represents the number of correctly classified beats.

AAMI class		Bottom H	Middle H	Top H
N	Correctly classified	1178	1165	1109
	Ground truth	1193		
S	Correctly classified	25	41	37
	Ground truth	126		
V	Correctly classified	0	167	0
	Ground truth	198		
F	Correctly classified	0	0	0
	Ground truth	2		
Q	Correctly classified	0	0	0
	Ground truth	0		
Accuracy(%)		79.2	**90.4**	75.4

Keras[2] on top of Tensorflow[3] to train, adapt and test the DNNs.

6.4 Performance Investigation

This section first investigates the best layer for injecting the i-vectors. Next, we demonstrate the effectiveness of i-vector adaptation. Finally, we compare the classification performance of iAP-DNNs with that of existing patient-specific classifiers.

6.4.1 Injecting I-vector into Different Hidden Layers

Table 6.1 provides the classification accuracies of the iAP-DNNs. The results show that the performance was the best when the i-vector was injected into the middle hidden layer. Therefore, our justification in Section 6.2.4 is supported and this settings was applied to subsequent experiments.

[2]https://keras.io/

[3]https://www.tensorflow.org/

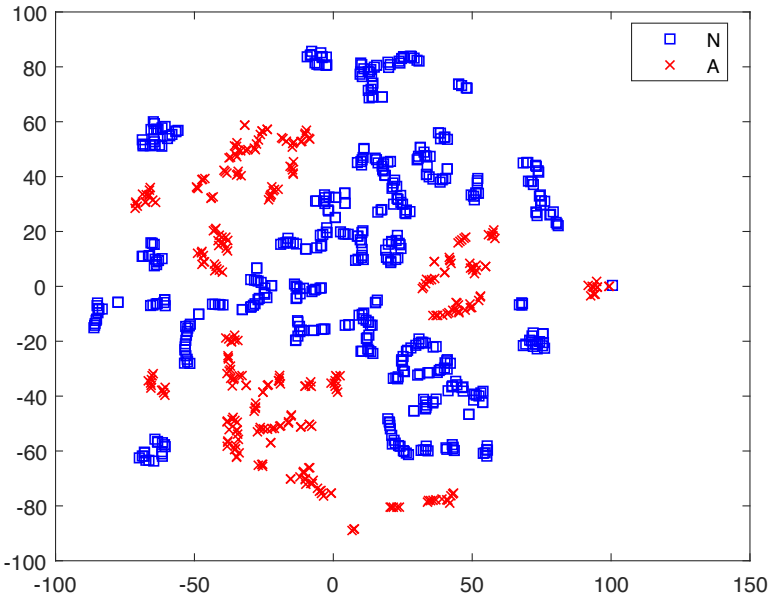


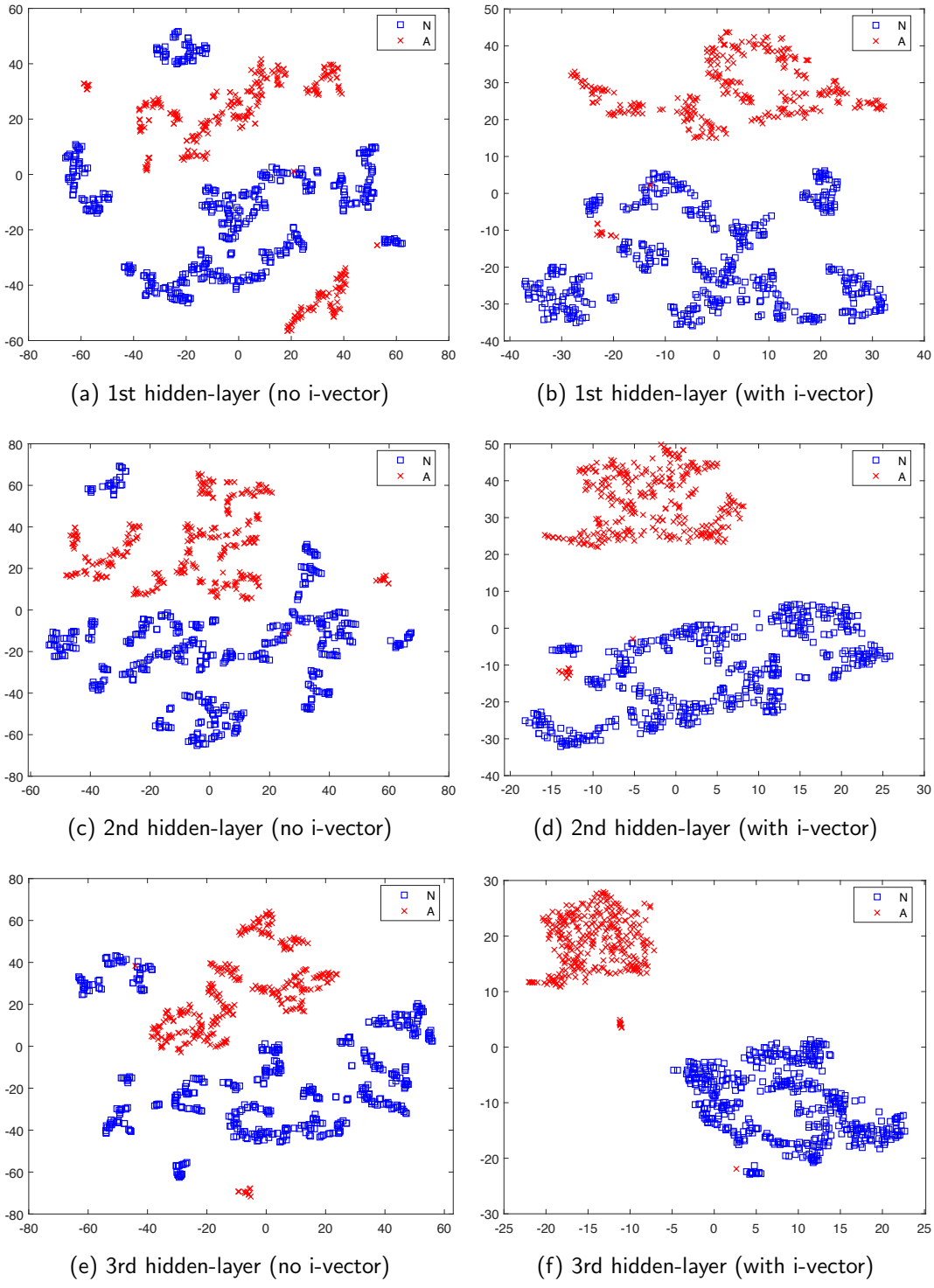
Figure 6.6: t-SNE plot of 417-dimensional feature vectors. Squares (in blue) and crosses (in red) refer to normal heartbeats (N) and arrhythmias (A) of a patient, respectively.

6.4.2 Effect of I-vector Adaptation

To show the effect of i-vector adaptation, we created a patient-specific DNN by applying backpropagation fine-tuning on the patient-independent DNN (Figure 6.1(a)) using 5 minutes of heartbeat vectors from a patient (e.g., Record ID 221). We also created a patient-specific iAP-DNN by applying backpropagation fine-tuning on the DNN in Figure 6.1(b), not only using the 5 minutes of heartbeat vectors but also an i-vector extracted from the 5-minute heartbeats. Then, we presented ten minutes of ECG, including the normal and arrhythmic heartbeats of this patient, to both DNNs. Note that the five minutes of ECG recordings comprise a majority of (but not necessarily all) ECG types of that patient. As different patients have different health conditions, the numbers of heartbeats for individual classes are also different.

The t-SNE plot of 417-dimensional feature vectors is shown in Figure 6.6, where $\square$ and $\times$ represent the normal (N) and arrhythmic (A) heartbeats, respectively. We

can see that there is no obvious clusters in Figure 6.6. We progressively moved up the hidden layers and projected the activations (before the ReLU) at the first, second and third hidden layers onto two-dimensional t-SNE spaces. The projected activations are shown in Figure 6.7. Obviously, without i-vector adaptation Figures. 6.7(a), (c) and (e), the projected vectors of both heartbeat types scatter in different regions of the t-SNE space and form multiple clusters, which makes classification more difficult. On the other hand, with i-vector adaptation (Figures. 6.7(b), (d) and (f)), the two heartbeat types are well separated, which makes classification by the softmax layer easy. Moreover, from Figures. 6.7(b), (d) and (f), we can see that each class has fewer clusters and the clusters of the two classes become more separate. This means that from the bottom to the top layers, the representation becomes more and more discriminative.



(a) 1st hidden-layer (no i-vector)

(b) 1st hidden-layer (with i-vector)

(c) 2nd hidden-layer (no i-vector)

(d) 2nd hidden-layer (with i-vector)

(e) 3rd hidden-layer (no i-vector)

(f) 3rd hidden-layer (with i-vector)

Figure 6.7: t-SNE plots of the neuron activations at different hidden layers: (a),(c) and (e) with patient's 5-minute ECG adaptation; (b), (d) and (f) with patient's 5-minute ECG and i-vector adaptation. It is clear that with i-vector adaptation, the number of clusters is smaller and the A and N classes are well separated in (b), (d) and (f).

Table 6.2: Confusion matrix of iAP-DNNs in Exp. 1. The values in parentheses correspond to fine-tuning the DNN without i-vector injection.

	N	S	V	F	Q
N	41600 (41630)	78 (77)	92 (56)	47 (29)	4 (29)
S	439 (523)	1829 (1749)	63 (50)	4 (12)	2 (3)
V	225 (305)	69 (349)	4473 (4097)	39 (47)	1 (9)
F	64 (86)	2 (1)	49 (43)	496 (481)	0 (0)
Q	5 (5)	0 (1)	2 (1)	1 (1)	0 (0)

6.4.3 Performance of iAP-DNNs

Experiment 1 (Exp. 1)

The first experiment was conducted to evaluate the proposed method based on 24 ECG recordings. Table 6.2 shows the confusion matrix of iAP-DNNs. We can see that the performance is better if patients' i-vectors were used for adaptation. Specifically, the numbers of true positives for Class S and Class V have been increased. Besides, the performance of the iAP-DNNs and that of [5–9] are shown in Table 6.3. Except for the Ppv of Class S and the Sen of Class V in [9], the overall performance of the proposed method for Class S and Class V is significantly better than that in [5–9] for all evaluation measures.

Using the confusion matrix in Table 6.2, the MCC performance of Classes N, S, V, F and Q can be calculated. Table 6.4 shows the performance comparison between the proposed iAP-DNNs and the existing patient-specific classifiers in [5–9]. Note that OMCC refers to overall MCC of the five classes. We can see that the MCC of the iAP-DNNs is much higher than the other three classifiers. The promising performance is not only found in the individual class, but also in the overall.

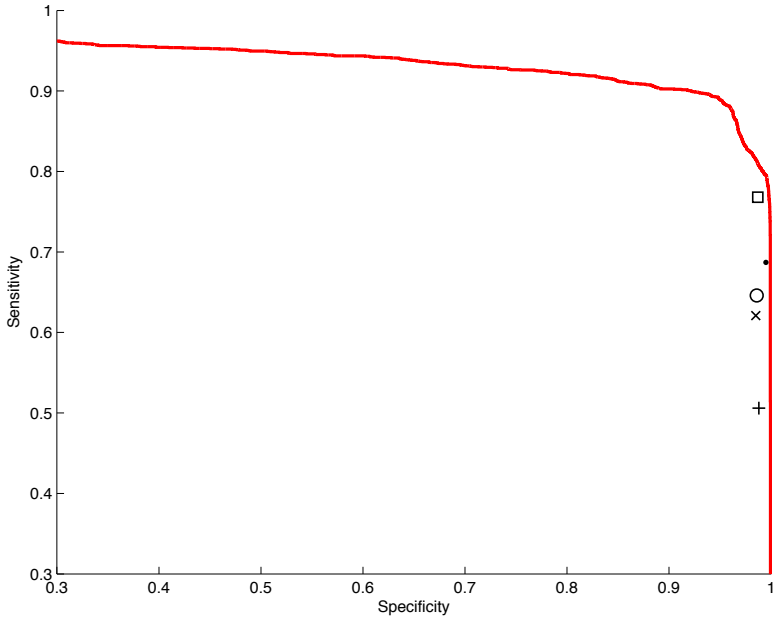
Table 6.3: Performance of the patient-specific classifiers in [5–9] and the proposed iAP-DNNs (Exp. 1)

Method		[5]	[6]	[7]	[8]	[9]	iAP-DNNs
Class S	Acc	96.6	96.1	96.4	97.5	98.3	**98.7**
	Sen	50.6	62.1	64.6	76.8	68.7	**78.3**
	Spe	98.8	98.5	98.6	98.7	**99.8**	**99.8**
	Ppv	67.9	56.7	62.1	74.0	**94.7**	92.5
Class V	Acc	98.1	97.6	98.6	98.6	98.8	**98.9**
	Sen	86.6	83.4	95.0	93.8	**95.5**	93.1
	Spe	99.3	98.1	98.1	99.2	99.1	**99.5**
	Ppv	93.3	87.4	89.5	92.4	92.2	**95.6**

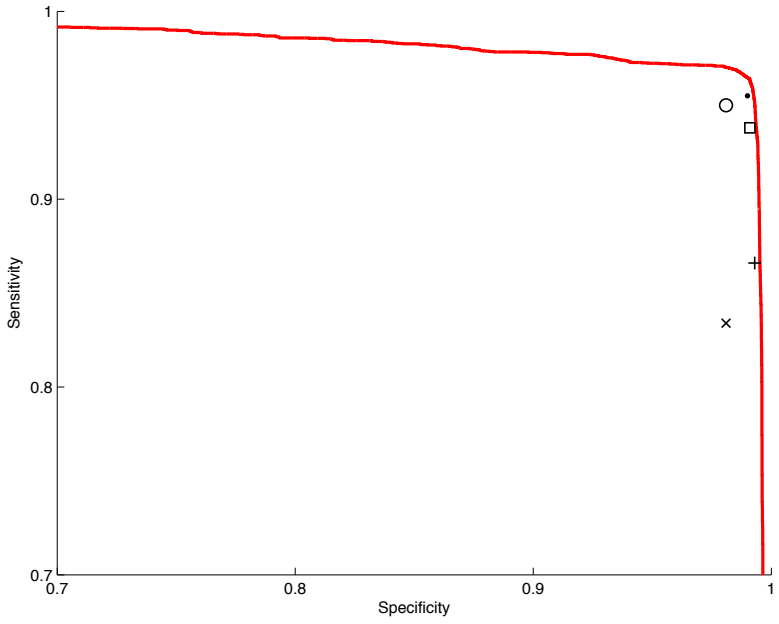
Table 6.4: Performance comparison in terms of MCC (Exp. 1)

Method		[5]	[6]	[7]	[8]	[9]	iAP-DNNs
Class	N	0.83	0.81	0.84	0.88	0.90	**0.93**
	S	0.57	0.57	0.62	0.74	0.80	**0.84**
	V	0.87	0.83	0.91	0.92	0.93	**0.94**
	F	0.55	0.67	0.78	0.70	0.78	**0.83**
	Q	0.00	0.00	0.00	0.00	0.00	**0.00**
OMCC		0.93	0.92	0.94	0.95	0.96	**0.97**

Figure 6.8 shows the ROC curves of the proposed method for Class S and Class V. In the ROC curves, perfect classification (Spe = 1.0 and Sen = 1.0) corresponds to the upper right corner of the graph. A sensitivity-specificity operating point is good if it is close to the upper-right corner. In Figure 6.8, the operating points of the best performing classifiers in [5–9] are also shown by the markers $+$, $\times$, $\circ$, $\square$, and $\bullet$, respectively. The figures clearly show that the sensitivity-specificity points in [5–9] are below the red curve. This means that, within a certain range of decision thresholds, the iAP-DNN achieves better performance in term of both sensitivity and specificity than the classifiers in [5–9].



(a) Class S *vs.* non S (AUC = 0.946)



(b) Class V *vs.* non V (AUC = 0.989)

Figure 6.8: ROC curves (Sen *vs.* Spe) of iAP-DNNs in Exp. 1. Black markers correspond to the best performance in [5–9]. *AUC*: Area under the ROC curve [2].

Table 6.5: Performance of the patient-specific classifiers in [5, 6, 8, 9, 12] and our iAP-DNNs (Exp. 2)

Method		[5]	[6]	[12] Method I	[12] Method II	[8]	[9]	iAP-DNNs
Class S	Acc	97.5	96.1	**99.1**	98.3	97.3	98.6	**99.1**
	Sen	74.9	81.8	76.5	61.4	**85.3**	77.2	78.4
	Spe	98.8	98.5	**99.9**	99.8	98.0	99.8	**99.9**
	Ppv	78.8	63.4	**99.1**	90.7	71.8	96.6	98.7
Class V	Acc	98.8	97.9	**99.7**	99.4	99.1	98.7	**99.7**
	Sen	94.3	90.3	97.1	91.8	96.4	97.2	**97.4**
	Spe	99.4	98.8	**99.9**	**99.9**	99.5	98.9	**99.9**
	Ppv	95.8	92.2	**98.5**	98.0	96.4	92.1	97.8

Experiment 2 (Exp. 2)

In the second experiment, for Class S and Class V, the evaluations were based on 14 and 11 test recordings, respectively. Table 6.5 shows the Acc, Sen, Spe and Ppv of the iAP-DNNs and that of [5, 6, 8, 9, 12]. Note that in Method I of [12], five minutes of labeled ECG signals of a patient was used to adapt the patient-specific classifier. In Method II, the hypothesized labels were used instead of the manual labeling process. In Table 6.5, for Class V, the Sen of the iAP-DNNs is the highest among all methods and a high Spe (99.9%) is achieved. For Class S, although the Sen of the iAP-DNNs is lower than that in [8], its Spe and Ppv are higher.

The performance of iAP-DNNs is similar to that of Method I in [12]. In [12], a subset was selected for training the general classifier based on the similarity among patients. The similarity is determined by calculating the dynamic time warping (DTW) distance, and the value of DTW threshold needs to be optimized by trial and error. However, in the proposed method, the ECG data of the general population can be used directly to train a general classifier before patient adaptation. Therefore, there is no need to throw away any ECG data from the general population nor do we need to optimize additional parameters. That is definitely an advantage.

6.5 Advantages and Limitations of iAP-DNNs

To deal with inter-patient variability in ECG signals, existing methods typically use three approaches: (1) pooling the patient-specific and patient-independent data together to train a patient-specific classifier [5–8], (2) combining the predictions made by a patient-independent classifier and a patient-specific classifier [12], and (3) fine-tuning a patient-independent classifier using patient-specific data [9]. The major problem of these approaches is that they fail to take advantage of the vast amount of ECG data from the general population. In particular, to prevent the limited amount of patient-dependent data from being overshadowed by the patient-independent data, only a small fraction of the patient-independent data will be used in the first and second approaches. While fine-tuning is a reasonable approach, the information learned from the general population could be easily lost or forgotten if the degree of fine-tuning is substantial.

The iAP-DNNs are designed to overcome the problems in the three approaches mentioned above. The key ideas are (1) to leverage the ECG data of a general population to create a patient-independent DNN and (2) to focus the adaptation on the upper layers of the DNN using patient-specific information to make it patient-dependent. To avoid being overshadowed by the data in the general population, the weights in the upper layers are re-initialized before adaptation begins. To avoid forgetting the learned information from the general population, the bottom layers of the network will only be adapted by a small amount of patient-specific data, i.e., the extent of adaptation in the lower layers will not be substantial. These strategies are superior to the data pooling approach in that it is not necessary to ensure a good balance between the patient-independent and patient-specific data. To gear the adaptation of the upper layers to specific patient, the i-vector that characterizes an individual patient is injected into the middle layer of the network. Results in Section 6.4.3 and Figure 6.6 suggest that this step has great impact on the DNN to

classify the ECG of individual patients.

Some recent studies [7, 9, 13, 15] applied convolutional neural networks (CNNs) to classify raw ECG signals into different arrhythmia types, primarily because of the intrinsic capability of CNNs in dealing with phase shift variability. In fact, it has been found in speech recognition research that applying max-pooling in time could produce representations that are less sensitive to phase shifts [75]. Our proposed method uses heartbeat segmentation and heartbeat alignment [69] to minimize the phase shift variation, which enables us to use DNN instead of CNN to classify the heartbeats. The question is "Which is a better way to deal with phase shifts: max-pooling or heartbeat alignment?". The answer lies in whether we can detect the R peaks accurately. If we can, heartbeat alignment is a better choice. On the other hand, if aligning the heartbeats is difficult, CNN is a better choice. For the MIT-BIH arrhythmia dataset, heartbeat alignment is a better choice because the R peaks in this dataset can be predicted at an accuracy of over 99% by using the Pan-Tompkins algorithm [60]. In fact, the results in Section 6.4.3 also suggest that heartbeat alignment together with the proposed DNN adaptation outperform state-of-the-art CNNs in this dataset.

Another advantage of heartbeat alignment is that DNNs are more amenable to adaptation by i-vectors than CNNs. This is because for ECG classification, the convolutional layers and max-pooling layers of a CNN have the concept of time, which are not compatible with the static information encoded in the i-vectors. Because the hidden layers in a DNN are static, injecting an i-vector into its hidden layers can be considered as shifting the activations of the hidden layers, where the shift accounts for the patient-specific information.

A limitation of iAP-DNNs is that the method requires some patient-specific ECG data that have been manually labelled by medical doctors to adapt the patient-independent DNN. As long as the amount of adaptation data is small, this requirement will not pose a serious burden on the medical doctors nor the patients. However, for those patients without the access to medical services, the method is not applicable or

they will need to fall back to using the patient-independent classifier.

Chapter 7

CONCLUSIONS AND FUTURE WORKS

7.1 Conclusions

In **Chapter 4**, we demonstrated how to identify heart arrhythmias by using handcraft features from ECG signals. SVMs and DNNs were applied to classify heart arrhythmia based on the UCI cardiac arrhythmia dataset. Feature pre-processing methods, such as FDR and PCA were investigated. Performance evaluations show that for classifying normal heartbeats against heart arrhythmias, the best combination of feature pre-processing and classification is FDR with DNNs. For multi-class classification, FDR can be easily adopted to one-vs-rest SVMs but not to the DNNs. We also demonstrated that pre-training of stacked RBMs is an essential step for training DNN classifiers, especially when the number of training samples is very limited.

In **Chapter 5**, we presented an end-to-end ECG classification system. One end of the system receives raw ECG signals and the other end gives beat-by-beat classification decisions. A new preprocessing method, which involves heartbeat segmentation and heartbeat alignment, was proposed to facilitate a deep neural network to form optimal representation of ECG signals and for the classification of heartbeat types.

Four experiments based on the MIT-BIH arrhythmia database were conducted. In the first experiment, 22-fold cross validations on a dataset comprising 50,977 heartbeats and five arrhythmia classes suggest that at the same specificity, the sensitivities of the end-to-end method for Class S and Class V are 16.4% and 21.1% higher (absolute) than those achieved by a conventional method. For all of the five classes, the proposed method achieves higher MCC and its ROC curves are above the operating

points reported in the literature. In the second experiment, the proposed end-to-end DNN was trained on 50,977 heartbeats from 22 patients and tested on 49,668 heartbeats from another 22 patients. Results demonstrate that this end-to-end DNN can capture useful information from the raw ECG signals, enabling it to outperform state-of-the-art arrhythmia classifiers (using either SVM or DNN) that rely on handcrafted ECG features. The third experiment shows the excellent performance (AUC = 0.999) of the proposed method in dealing with the binary ECG classification.

The fourth experiment compared our patient-independent end-to-end ECG classification system with patient-specific ECG classification systems. The results demonstrate that the patient-independent DNN-based classifier generalizes very well to new/unseen patients. The effect of the proposed raw signal extraction method (including segmentation and alignment of complete heartbeats) is remarkable. Thus, the end-to-end ECG classification approach not only outperforms the existing patient-independent classification system, but also performs as well as the patient-specific classification systems.

After using more data to train the patient-independent classifier and testing with more patients, the proposed end-to-end (input: raw ECG signals; output: beat-by-beat classification decisions) ECG classification system can be introduced as a tool to assist clinicians in diagnosing arrhythmias.

In **Chapter 6**, we proposed an adaptive patient-specific heartbeat classification model (i.e., iAP-DNNs) for diagnosing heart arrhythmias, which leverages the DNNs for both feature extraction and classification based on the raw ECG signals. A general classifier was first trained on the general population. Then, the weights in the lower part of the general classifier were retained and the weights in the upper part were randomized. To create a patient-specific classifier, not only patient-specific ECG but also patient-dependent i-vectors are used for adaptation. Two experiments based on the MIT-BIH arrthymia database have been conducted. The results show that the proposed iAP-DNNs achieve better performance than existing patient-specific

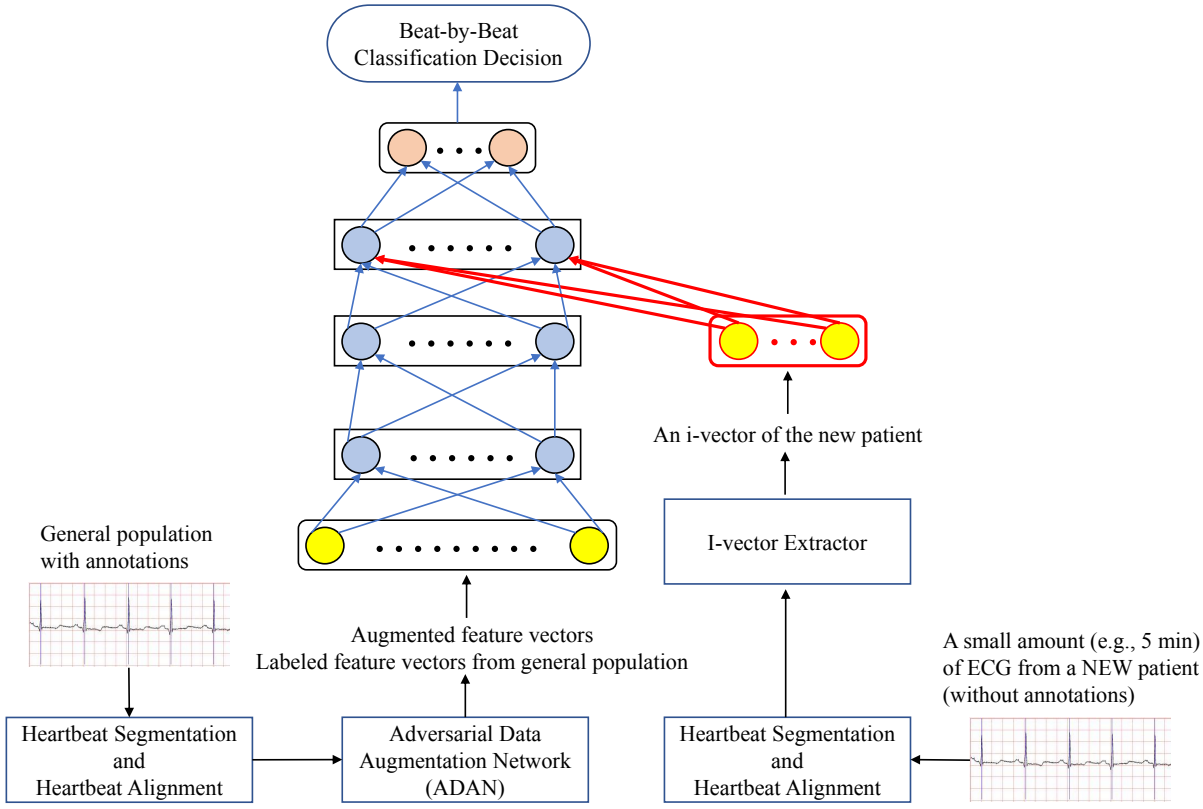


Figure 7.1: An unsupervised patient-adaptable DNN based on i-vector.

heartbeat classification systems.

To the best of our knowledge, this is the first study that uses i-vectors to characterize the ECG of individual patients and applies the i-vectors to adapt a DNN for patient-specific ECG classification. The key contribution is that by injecting the i-vectors into a middle layer of the DNN during backpropagation fine-tuning, we can make the upper layers of the DNN more patient-dependent. Without the i-vectors as an auxiliary input to the middle layer, it is much harder to ensure such patient dependence.

7.2 Future Work

The patient-independent classifier (e.g., the proposed end-to-end ECG classifier) is universal. When applying it to new/unseen patients, no patient-specific data is required. However, because of the inter-patient variations, such fixed classifier may not perform well. Patient-specific classifiers (e.g., the proposed iAP-DNNs) are proposed to address the issues of inter-patient variations. Most of the patient-specific ECG classification approaches rely on manual intervention to achieve good classification performance. However, manual intervention requires medical doctors to provide the ground truth labels of the patient-specific training data, i.e., beat-by-beat annotations on patient-dependent ECG signals. This labeling process may be costly.

To relax the limitations of the patient-independent and patient-specific classifiers, we can use an unsupervised patient adaptation approach as follows. An i-vector of a patient is injected to the middle layer of a DNN. Specifically, for each hidden node of the DNN receiving the injection, it adds up the values of all elements of the i-vector. In that case, the i-vector act as an additional bias term and the amount of bias is not trainable. The main input receives the labelled heartbeat vectors from the general population, and the auxiliary input receives the i-vector of the patient. The DNN is adapted/fine-tuned by backpropagation to form a patient-specific network. Since the i-vector contains the information of this patient, the upper layers of the DNN are expected to be patient-dependent.

Another issue is data-imbalance problem. In the MIT-BIH arrhythmia database, the numbers of samples in Classes S, V, F and Q are extremely small compared with the normal heartbeats (Class N). This data-imbalance issue will cause the trained DNN classifier to bias towards the majority class, leading to poor classification performance on the minority classes. The DNN may even predict all of the test data as Class N (the majority class). In the case of imbalanced training data, oversampling is a standard technique to avoid the classifier to bias towards the majority class. How-

ever, our preliminary experiments suggested that oversampling does not work well. Therefore, instead of duplicating the samples from the minority classes, adversarial data augmentation network (ADAN) can be used to enlarge the training data of the minority classes by generating fake samples, which is an elegant solution based on the idea of generative adversarial network (GAN) [76]. Figure 7.1 shows an unsupervised patient-adaptable ECG classifier based on i-vectors. It is referred as an automatic adaptation model because no manual labels are required during patient adaptation.

While the MIT-BIH arrhythmia dataset has been popular among the research community, it is also important to validate the accuracy using a larger dataset, e.g., the European ST-T Database [77]. This dataset consists of ninety two-hour ECG recordings with beats, rhythms, and signal quality annotation. We believe that the large amount of ECG data in this dataset is beneficial to the proposed method because it can leverage the data to train a better patient-independent classifier, which could lead to better patient-specific classifiers after i-vector adaptation.

BIBLIOGRAPHY

[1] P. D. Chazal, M. O'Dwyer, and R. B. Reilly, "Automatic classification of heartbeats using ECG morphology and heartbeat interval features," *IEEE Transactions on Biomedical Engineering*, vol. 51, no. 7, pp. 1196–1206, July 2004.

[2] T. Fawcett, "An introduction to roc analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, 2006.

[3] C. Ye, B. V. K. V. Kumar, and M. T. Coimbra, "Heartbeat classification using morphological and dynamic features of ECG signals," *IEEE Transactions on Biomedical Engineering*, vol. 59, no. 10, pp. 2930–2941, Oct. 2012.

[4] S. Raj and K. C. Ray, "Sparse representation of ECG signals for automated recognition of cardiac arrhythmias," *Expert Systems with Applications*, vol. 105, pp. 49–64, 2018.

[5] W. Jiang and S. G. Kong, "Block-based neural networks for personalized ECG signal classification," *IEEE Transactions on Neural Networks*, vol. 18, no. 6, pp. 1750–1761, Nov. 2007.

[6] T. Ince, S. Kiranyaz, and M. Gabbouj, "A generic and robust system for automated patient-specific classification of ECG signals," *IEEE Transactions on Biomedical Engineering*, vol. 56, no. 5, pp. 1415–1426, 2009.

[7] S. Kiranyaz, T. Ince, and M. Gabbouj, "Real-time patient-specific ECG classification by 1-D convolutional neural networks," *IEEE Transactions on Biomedical Engineering*, vol. 63, no. 3, pp. 664–675, 2016.

[8] X. Zhai and C. Tin, "Automated ECG classification using dual heartbeat coupling based on convolutional neural network," *IEEE Access*, vol. 6, pp. 27465–27472, June 2018.

[9] Y. Li, Y. Pang, J. Wang, and X. Li, "Patient-specific ECG classification by deeper CNN from generic to dedicated," *Neurocomputing*, vol. 314, no. 7, pp. 336–346, Nov. 2018.

[10] S. Osowski, L. T. Hoa, and T. Markiewic, "Support vector machine-based expert system for reliable heartbeat recognition," *IEEE Transactions on Biomedical Engineering*, vol. 51, no. 4, pp. 582–589, 2004.

[11] T. J. Jun, H. J. Park, and Y. H. Kim, "Premature ventricular contraction beat detection with deep neural networks," in *15th IEEE International Conference on Machine Learning and Applications*, 2016, pp. 859–864.

[12] C. Ye, B. V. K. V. Kumar, and M. T. Coimbra, "An automatic subject-adaptable heartbeat classifier based on multiview learning," *IEEE Journal of Biomedical and Health Informatics*, vol. 20, no. 6, pp. 1482–1492, Nov. 2016.

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

[14] J. H. Tan, Y. Hagiwara, W. Pang, I. Lim, O. S. Lih, M. Adam, R. S. Tan, M. Chen, and U. R. Acharya, "Application of stacked convolutional and long short-term memory network for accurate identification of CAD ECG signals," *Computers in Biology and Medicine*, vol. 94, pp. 19–26, 2018.

[15] U. R. Acharya, H. Fujita, O. S. Lih, Y. Hagiwara, J. H. Tan, and M. Adam, "Application of deep convolutional neural network for automated detection of myocardial infarction using ECG signals," *Information sciences*, vol. 415, pp. 190–198, 2017.

[16] N. Kohli and N. K. Verma, "Arrhythmia classification using SVM with selected features," *International Journal of Engineering, Science and Technology*, vol. 3, no. 8, pp. 122–131, 2011.

[17] A. B. de Luna, P. Coumel, and J. F. Leclercq, "Ambulatory sudden cardiac death: Mechanisms of production of fatal arrhythmia on the basis of data from 157 cases," *American Heart Journal*, vol. 117, no. 1, pp. 151–159, Jan. 1989.

[18] Holter Monitor, "Texas heart institute (cited at may 2010)," `http://www.texasheart.org/HIC/Topics/Diag/diholt.cfm`.

[19] American Medical Association, *15.3.1 Electrocardiographic Terms*, AMA Manual of Style.

[20] UCI Machine Learning Repository, "Cardiac arrhythmia database," `https://archive.ics.uci.edu/ml/datasets/arrhythmia`.

[21] G. B. Moody and R. G. Mark, "The impact of the MIT-BIH arrhythmia database," *IEEE Engineering in Medicine and Biology*, vol. 20, no. 3, pp. 45–50, May 2001.

[22] Assoc. Adv. Med. Instrument., "Testing and reporting performance results of cardiac rhythm and ST segment measurement algorithms," *ANSI/AAMI*, , no. EC57, 1998.

[23] N. Kohli, N. K. Verma, and A. Roy, "SVM based methods for arrhythmia classification in ECG," in *International Conference on Computer and Communication Technology (ICCCT)*, 2010, pp. 486–490.

[24] University of California, "UC Irvine machine learning repository," `http://archive.ics.uci.edu/ml`.

[25] S. Khare, A. Bhandari, S. Singh, and A. Arora, "ECG arrhythmia classification using Spearman rank correlation and support vector machine," in *Proceedings of the International Conference on Soft Computing for Problem Solving (SocProS)*, 2011, pp. 591–598.

[26] W. Wayne, "Spearman rank correlation coefficient," Applied Nonparametric Statistics. Boston: PWS-Kent, 2nd edition, 1990.

[27] C. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006.

[28] S. M. Jadhav, S. L. Nalbalwar, and A. A. Ghatol, "ECG arrhythmia classification using modular neural network model," in *IEEE EMBS Conference on Biomedical Engineering and Sciences (IECBES)*, 2010, pp. 62–66.

[29] A. L. Goldberger *et al.*, "Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals," *Circulation*, vol. 101, no. 23, pp. e215–e220, Jun. 2000.

[30] S. W. Moon and S. G. Kong, "Block-based neural networks," *IEEE Transactions on Neural Networks*, vol. 12, no. 2, pp. 307–317, Mar. 2001.

[31] E. J. Boyko, "Ruling out or ruling in disease with the most sensitiue or specific diagnostic test: Short cut or wrong turn?," *Medical Decision Making*, vol. 14, no. 2, pp. 175–179, 1994.

[32] A. Laupacis and N. Sekar, "Clinical prediction rules: a review and suggested modifications of methodological standards," *Jama*, vol. 277, no. 6, pp. 488–494, 1997.

[33] A. K. Akobeng, "Understanding diagnostic tests 1: sensitivity, specificity and predictive values," *Acta Paediatr.*, vol. 96, no. 3, pp. 338–341, Mar. 2007.

[34] B. W. Matthews, "Comparison of the predicted and observed secondary structure of t4 phage lysozyme," *Biochimica et Biophysica Acta (BBA)-Protein Structure*, vol. 405, no. 2, pp. 442–451, Oct. 1985.

[35] M. W. Mak, J. Guo, and S. Y. Kung, "PairProSVM: protein subcellular localization based on local pairwise profile alignment and SVM," *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, vol. 5, no. 3, pp. 416–422, 2008.

[36] J. A. Swets, *Signal detection theory and ROC analysis in psychology and diagnostics : collected papers*, Lawrence Erlbaum Associates, Mahwah, NJ, 1996.

[37] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, Sep. 1995.

[38] W.S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *Bulletin of Mathematical Biophysics*, vol. 5, no. 4, pp. 115–133, 1943.

[39] W. G. Baxt, "Application of artificial neural networks to clinical medicine," *The Lancet*, vol. 346, no. 8983, pp. 1135–1138, Oct. 1995.

[40] S. H. Liao and C. H. Wen, "Artificial neural networks classification and clustering of methodologies and applications - literature analysis from 1995 to 2005," *Expert Systems With Applications*, vol. 32, no. 1, pp. 1–11, Jan. 2007.

[41] W. L. Xing and X. W. He, "Applications of artificial neural networks on signal processing of piezoelectric crystal sensors," *Sensors and Actuators: B. Chemical*, vol. 66, no. 1, pp. 272–276, July 2000.

[42] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, *Learning internal representations by error propagation*, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[43] P. Vincent, H. Larochelle, I. Lajoie, Y. H. Bengio, and P. A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *The Journal of Machine Learning Research*, vol. 11, pp. 3371–3408, Dec. 2010.

[44] G. E. Hinton and R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[45] Y. Bengio, "Learning deep architectures for AI," *Foundations and Trends in Machine Learning*, vol. 2, no. 1, pp. 1–127, Nov. 2009.

[46] I. Arel, D. C. Rose, and T. P. Karnowski, "Deep machine learning-a new frontier in artificial intelligence research," *IEEE Computational Intelligence Magazine*, vol. 5, no. 4, pp. 13–18, Nov. 2010.

[47] G. E. Hinton, "A practical guide to training restricted boltzmann machines," 2010, UTML Tech Report, Univ. Toronto.

[48] Y. W. Teh G. E. Hinton, S. Osindero, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, July 2006.

[49] G. E. Hinton, "Learning multiple layers of representation," *Trends in Cognitive Sciences*, vol. 11, no. 10, pp. 428–434, July 2007.

[50] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Computation*, vol. 14, no. 8, pp. 1711–1800, Aug. 2002.

[51] P. Pavlidis, J. Weston, J. Cai, and W.N. Grundy, "Gene functional classification from heterogeneous data," in *International Conference on Computational Biology*, 2001, pp. 62–66.

[52] V. N. Vapnik, *The Nature of Statistical Learning Theory*, Springer-Verlag, New York, 1995.

[53] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," *Advances in Neural Information Processing Systems*, vol. 19, pp. 153–160, 2007.

[54] G. E. Hinton, "Training a deep autoencoder or a classifier on MNIST digits," `http://www.cs.toronto.edu/~hinton/MatlabForSciencePaper.html`.

[55] M. Zijlmans, D. Flanagan, and J. Gotman, "Heart rate changes and ECG abnormalities during epileptic seizures: prevalence and definition of an objective clinical sign," *Epilepsia*, vol. 43, no. 8, pp. 847–854, Aug. 2002.

[56] D. Erhan, Y. Bengio, A. Courville, P. A. Manzagol, P. Vincent, and S. Bengio, "Why does unsupervised pre-training help deep learning?," *The Journal of Machine Learning Research*, vol. 11, pp. 625–660, Feb. 2010.

[57] P. M. Rautaharju, B. Surawicz, and L. S. Gettes, "AHA/ACCF/HRS recommendations for the standardization and interpretation of the electrocardiogram," *Journal of the American College of Cardiology*, vol. 53, no. 11, pp. 982–991, 2009.

[58] J. S. Steinberg *et al.*, "Value of the P-wave signal-averaged ECG for predicting atrial fibrillation after cardiac surgery," *Circulation*, vol. 88, no. 6, pp. 2618–2622, 1993.

[59] D. M. Bloomfield *et al.*, "Microvolt T-wave alternans and the risk of death or sustained ventricular arrhythmias in patients with left ventricular dysfunction," *Journal of the American College of Cardiology*, vol. 47, no. 2, pp. 456–463, 2006.

[60] J. Pan and W. J. Tompkins, "A real-time QRS detection algorithm," *IEEE Transactions on Biomedical Engineering*, vol. 32, no. 3, pp. 230–236, Mar. 1985.

[61] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *13th Int. Conf. on Artificial Intelligence and Statistics*, May 2010, pp. 249–256.

[62] L. V. D. Maaten and G. E. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.

[63] N. Japkowicz and S. Stephen, "The class imbalance problem: A systematic study," *Intelligent Data Analysis*, vol. 6, no. 5, pp. 429–449, 2002.

[64] S. S. Xu, M. W. Mak, and C. C. Cheung, "Patient-specific heartbeat classification based on i-vector adapted deep neural networks," in *IEEE Int. Conf. on Bioinformatics and Biomedicine (BIBM)*, Dec. 2018, pp. 784–787.

[65] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE/ACM Transactions on Audio Speech and Language Processing*, vol. 19, no. 4, pp. 788–798, Aug. 2011.

[66] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust dnn embeddings for speaker recognition," in *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 5329–5333.

[67] E. Variani, X. Lei, E. McDermott, I. Moreno, and J. Gonzalez-Dominguez, "Deep neural networks for small footprint text-dependent speaker verification," in *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 4052–4056.

[68] W. M. Campbell, D. E. Sturim, and D. A. Reynolds, "Support vector machines using GMM supervectors for speaker verification," *IEEE Signal Processing Letters*, vol. 13, no. 5, pp. 308–311, May 2006.

[69] S. S. Xu, M. W. Mak, and C. C. Cheung, "Towards end-to-end ECG classification with raw signal extraction and deep neural networks," *IEEE Journal of Biomedical and Health Informatics*, vol. 23, no. 4, pp. 1574–1584, Jul. 2019.

[70] M. W. Mak, "Lecture notes on factor analysis and i-vectors," Tech. Rep., Dept. of Electronic and Information Engineering, The Hong Kong Polytechnic University, 2016.

[71] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer-Verlag New York Inc., 2006.

[72] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013.

[73] A. Mohamed, G.Hinton, and G. Penn, "Understanding how deep belief networks perform acoustic modelling," in *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, Mar. 2012, pp. 4273–4276.

[74] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv*:1412.6980, Dec. 2014.

[75] T. N. Sainath, R. J. Weiss, A. Senior, K. W. Wilson, and O. Vinyals, "Learning the speech front-end with raw waveform CLDNNs," in *Proc. INTERSPEECH*, Sep. 2015, pp. 1–5.

[76] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, pp. 2672–2680, 2014.

[77] A. Taddei, G. Distante, M. Emdin, P. Pisani, G. B. Moody, C. Zeelenberg, and C. Marchesi, "The European ST-T database: standard for evaluating systems for the analysis of ST-T changes in ambulatory electrocardiography," *European Heart Journal*, vol. 13, pp. 1164–1172, Apr. 1992.