Sparse Autoencoder Based Deep Network in Massive Electrocardiography Signal Classification

Yan Yan, Xingbin Qin, Jianping Fan, and Lei Wang

***Abstract*—The abstract goes here.**

***Index Terms*—big data, electrocardiography classification, sparse autoencoder, deep learning.**

I. Introduction

An era of big data in healthcare is now under way, decades of progress in digitising medical records accumulate vast amounts of medical data, simultaneously mobile healthcare and wearable sensor technologies offer healthcare data from larger population coverage. The noninvasive, inexpensive and well-established technology of electrocardiographic signal in mobile health or personal health has the greatest popularity in heart function analysis. Automated electrocardiography classification provides indispensable assist in long-term clinical monitoring, and a large number of approaches have been proposed for the task, easing the diagnosis of arrhythmic changes as well as further inspection, e.g., heart rate variability or heart turbulence analysis [1].

Lots of algorithms have been proposed for the classification and detection for electrocardiography signals. The electrocardiography classification or detection task had been divided into two parts: the feature extraction process and classifier. Simple classifier such as linear discriminants [2] and kNN [3], more complex classifiers like neural networks [4]–[7], fuzzy inference engines [7], [8], hidden Markov model [9], [10], independent component analysis [11] and support vector machine [3], [12], [13] were also adapted by lots of researchers.

Beyond the classifier, the performance of a recognition system highly depends on the determination of extracted electrocardiography features. Time domain features, frequency domain features, and statistical measures features for six fundamental waves (PQRSTU) had been used in feature extraction process [14]. Time domain features like morphological features include shapes, amplitudes, and durations were adapted primarily in [15]–[17], frequency domain features like wavelet transformation were widely used [18], [19] stationary features like higher order statistics also had been developed. Principal component analysis [20] and Hermite functions [21] have been used in electrocardiography classification and related analysis technologies as well. Almost every single published paper proposes a new set of features to be used, or a new combination of the existing ones [1].

The results from these algorithms or models were not amenable to expert labelling, as well as for the identification of complex relationships between subjects and clinical conditions [22]. But for the ambulatory electrocardiography clinical application, as well as the normal application in daily healthcare monitoring for cardiac function or early warning of heart disease, an automated algorithm or model would have significant meaning. The application of artificial intelligence methods has become an important trend in electrocardiography for the recognition and classification of different arrhythmia types [22]. The data explosion puts forward the new request to the method of data processing and information mining.

Over the past decades computational techniques proliferated in the pattern recognition field, simultaneously the applications in electrocardiography recognition, detection and classification for relevant trends, patterns, and outliers. Most of the literatures in the electrocardiography classification task were focused on the supervised learning methods, as in unsupervised learning methods were infrequently used, which needs a lot of effort in labelling data. The MIT-BIH database [23] was the most widely used data in the classification and detection algorithm developments, while mass unlabelled electrocardiography data had been ignored due to the supervise learning approaches essential. Unsupervised learning methods become crucial in mining or analysing unlabelled data, as the unlabelled electrocardiography data accumulated. Unsupervised learning-based approaches and the application to electrocardiogram classification in literatures mainly include clustering-based techniques [21], [24], [25], self-adaptive neural network-based methods [26], [27] and some hybrid unsupervised learning systems [28].

In this paper, we adopt a big data unsupervised learning approach of sparse autoencoder based deep neural network in large unlabelled ambulatory electrocardiography dataset to learn features automatically, with which the cardiac arrhythmia with electrocardiograms classification task was proposed.

In the following sections, we will first state the experimental setup and methodologies in Section 2. Then the experimental results and the discussion are given in Section 3 and Section 4 respectively.

Y. Yan is with the Shenzhen Key Laboratory for Low-cost Healthcare, Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences. No. 1068, Xueyuan Road, Nanshan District, Shenzhen, Guangdong Province, China-mail: (yan.yan@siat.ac.cn).

X. Qin is with the Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences. No. 1068, Xueyuan Road, Nanshan District, Shenzhen, Guangdong Province, China.

J. Fan is with the Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences. No. 1068, Xueyuan Road, Nanshan District, Shenzhen, Guangdong Province, China.

L. Wang is with the Shenzhen Key Laboratory for Low-cost Healthcare, Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences. No. 1068, Xueyuan Road, Nanshan District, Shenzhen, Guangdong Province, China.

Manuscript received April 19, 2005; revised December 27, 2012.

TABLE I
AAMI CLASSES MAPPED FROM MIT-BIH ARRHYTHMIA & LONG-TERM DATABASE TYPES

AAMI heartbeat classes	N	S	V	F	Q
Description	Any heartbeat not in the S,V,F or Q class	Supraventricular ectopic beat	Ventricular ectopic beat	Fusion beat	Unknown beat
MIT-BIH heartbeat types (codes)	Normal beat (1) Left bundle branch block beat (2) Right bundle branch block beat (3) Nodal escape beat (11) Atrial escape beat (34)	Aberrated atrial premature beat (11) Nodal (junctional) premature beat (7) Atrial premature contraction (8) Premature or ectopic supraventricular beat(9)	Ventricular escape beat(10) Premature ventricular contraction (5) Ventricular flutter wave (31)	Fusion of ventricular & normal beat (6)	Paced beat (12) Unclassifiable beat (13) Fusion of paced and normal (38)
MITBIH-AR(100,687)	89,925	2,774	7,171	802	15
MITBIH-LT(667,347)	600,197	150	64,090	2,906	0

The counts listed in the table may differ from other literatures [2] due to the computation need.

II. Electrocardiography Datasets and Arrhythmia Classes

In the proposed method for electrocardiography classification, the unlabelled dataset and labelled datasets are both used. In this section, a general description of the unlabelled pretraining dataset is proposed, then the arrhythmia classification classes used in the labelled open datasets were discussed.

A. Arrhythmia Classes

The related experiments in this paper were carried out in two public databased available on Physionet, and a collected unlabelled ambulatory electrocardiography database (unlabelled means there were no electrocardiography experts involved in the interpretation). The ANSI/AAMI EC57: 1998/(R)2008 standard AAMI (2008) recommends to group the heartbeats into five classes: on-ectopic beats (N); supraventricular ectopic beat (S); ventricular ectopic beat (V); fusion of a V and a N (F); unknown beat type (Q). As Table I illustrated the detail of the MIT-BIH Arrhythmia Database and MIT-BIH Long-term Database which mapped to the AAMI heartbeat classes.

B. The Unlabelled Dataset

Data from the ambulatory electrocardiography database were used in this study, which includes recordings of 100 subjects with arrhythmia along with normal sinus rhythm. The database contains 100 recordings, each containing a 3-lead 24-hour long electrocardiography which were bandpass filtered at 0.1-100Hz and sampled at 128Hz. In this study, only the lead I data were adapted after preprocessing in the classification task. The reference average heart beats for each sample has 97,855 beats for the 24-hour long recording, and the reference arrhythmia average is 1,810 beats which were estimated by a commercial software (this statistics aim to indicate the existence for arrhythmia samples, which should not be consider as a experiment preset).

C. The Open Labelled Datasets

The MIT-BIH Arrhythmia Database [23] contains 48 half-hour recordings each containing two 30-min ECG lead signals (lead A and lead B), sampled at 360Hz. As well only the lead I data were used in the proposed method. In agreement with the AAMI recommended practice, the four recordings with paced beats were removed from the analysis. The remaining recordings were divided into two datasets, with small part of which were used as the training set of the fine-tuning process (details would be described in the following part). The MIT-BIT Long-term Database is also used in this study for training and verification, which contains 7 long-term ECG recordings (14 to 22 hours each), with manually reviewed beat annotations and sampled at 128Hz. Similarly, the 7 recordings were divided into two datasets, with part used as the fine-tuning training set. A description of the labelled datasets are illustrated in Table 1.

III. Experimental Methodology

In this section, the proposed approach technic details are described. This section includes descriptions of the deep architectures, the auto-encoder based training method, the data sets used, the classification task settings and other necessary details.

A. Models

Deep learning methods attempt to learn feature hierarchies as higher-level features are formed by the composition of lower-level features. The electrocardiography interpretation has been judged by the medical professionals, which was based on the abstractions of the perceptible features. In this model we consider the higher-level abstractions as the perceptible features, with whose composition the medical professionals can make arrhythmia judgement. The deep architecture automatic learning method is especially important for high-level abstractions, which human often do not know how to specify explicitly in terms of raw sensory input [29]. As [30] discussed, deep learning methods are bused on learning internal representations of data, another important advantage they offer is the ability to naturally leverage: (a) unsupervised data and (b) data from similar tasks to boost performance on large and challenging problems that routinely suffer from a poverty of labelled data. In the electrocardiography classification problem, we got plenty of unsupervised data, and the labelled data was limited as well, so it is a spontaneously idea to adapt deep learning method in this classification problem.

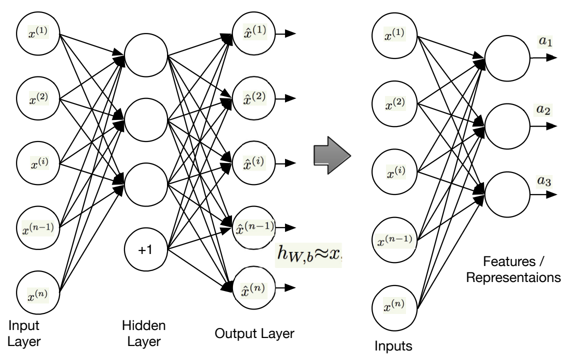

Fig. 1. An one-hidden-layer autoencoder: first preset the output of the autoencoder to be the same as itself, train the neural network, then use the learned weights of the connections to calculate the representations of the raw input, in some literatures the process could be encoder and decoder [31].

1) Deep Neural Networks: The artificial neural network had been widely used in different applications, the basic 3-layer model (with only one hidden layer) is a fairly shallow network which means only shallow features can be learning via the structure. Deep neural networks, meaning ones in which we have multiple hidden layers, with which we can compute much more complex features of the input. Each hidden layer computes a non-linear transformation of the previous layer, a deep network can have significantly greater representational power (i.e., can learn significantly more complex functions) than a shallow one. A typical deep neural network structure makes no different from the normal multi layer neural network.

2) Autoencoders and Sparsity: An autoencoder is trained to encode the input x into some representation $c(x)$ so that the input can be reconstructed from that representation [31]. The autoencoder neural network is an unsupervised learning algorithm that applies backpropagation, setting the target values to be equal to the inputs. i.e it uses $y^{(i)} = x^{(i)}$. The autoencoder neural network illustrated in Fig2 tries to learn a function $h_{W,b} \approx x$, which tries to approach a identity function to ensure the output $\hat{x_1}$ close to x. If there is one linear hidden layer and the mean squared error criterion is used to train the network, then the k hidden units learn to project the input in the span of the first k principal components of the data [32]. Since in the neural network the hidden layer is nonlinear, the autoencoder behaves differently from PCA, which has the ability to capture multi-modal aspects of the input distribution (the representation of the input) [33]. The related literature experiments reported in [34] suggest that in practice, when trained with stochastic gradient descent, nonlinear autoencoders with more hidden units than inputs (called overcomplete) yield useful representations (in the sense of classification error measured on a network taking this representation in input). A farther defence of autoencoder can be accessed from [31]. As the theory illustrated, the electrocardiography signal representations can be learned via the autoencoder structures and algorithms.

In the adapted algorithm, we impose a sparsity constraint on the hidden units to guarantee the representations expression ability. The neuron would be considered as "active" or "firing" if its output value by the activation function is close to 1, or as being "inactive" if its output value is close to 0 in which the $sigmoid$ activation function adapted. The correspond numerical value would be 1 and -1 with $tanh$ activation function. $a_j^{(2)}(x)$ denote the activation of hidden unit j in the autoencoder with the given input of x. And farther, let

$$\hat{\rho}_j = \frac{1}{m} \sum_{i=1}^{m} [a_j^{(2)}(x^{(i)})] \tag{1}$$

be the average activation of hidden unit j (averaged over the training set). Approximately enforce the constraint:

$$\hat{\rho}_j = \rho \tag{2}$$

where ρ is a sparsity parameter, typically a small value close to zero (such as $\rho = 0.05$), which means the average activation of each hidden neuron j to be close to zero (0.05 for instance).

The overall cost function of neural network is denoted by $J(W, b)$ which was defined by:

$$\begin{aligned} J(W,b) = [\frac{1}{m} \sum_{i=1}^{m} (\frac{1}{2} \| h_{W,b}(x^{(i)}) - y^{(i)} \|^2)] \\ + \frac{\lambda}{2} \sum_{l=1}^{n_l - 1} \sum_{i=1}^{s_l} \sum_{j=1}^{s_l + 1} {W_{ji}^{(l)}}^2 \end{aligned} \tag{3}$$

as the first term in the definition of $J(W, b)$ is an average sum-of-squares error term. The second term is a regularization term that tends to decrease the magnitude of the weights, and helps prevent overfitting. The definition of λ, s, l etc. would be explained in detail in the appendix part. To satisfy the constraint of sparsity, an extra penalty term to the optimisation objective that penalised $\hat{\rho}_j$ deviating significantly from ρ. The Kullback-Leibler (KL) divergence:

$$\sum_{j=1}^{s_2} KL(\rho || \hat{\rho}) = \sum_{j=1}^{s_2} \rho \log \frac{\rho}{\hat{\rho}_j} + (1 - \rho) \log \frac{1 - \rho}{1 - \hat{\rho}_j} \tag{4}$$

is chosen as the penalty term. KL-divergence is a standard function for measuring how different two different distributions are. So in the autoencoder neural network training, the cost function of $J_{sparse}(W, b)$ was defined as:

$$J_{sparse}(W,b) = J(W,b) + \beta \sum_{j=1}^{s_2} KL(\rho || \hat{\rho_j}) \tag{5}$$

β denotes the weight of the sparsity penalty term.

3) Feature Self-taught Learning: The mechanism of how the autoencoder neural network has been used to learn features from unlabelled data had been illustrated in the above sections. Concretely, the collected unlabelled data training set $\{x_u^{(1)}, x_u^{(2)}, x_u^{(3)}, \ldots, x_u^{(m_u)}\}$ in which u for the unlabelled. The unlabelled data then had been used in the autoencoder training process. Then the activation a would be calculated by the model parameters of $W^{(1)}$, $b^{(1)}$, $W^{(2)}$, $b^{(2)}$ in the autoencoder illustrated in Fig 2, which would be considered as a better representation (or feature) than the raw input x. As well we can adapt the same method in the limited supervised dataset $\{(x_l^{(1)}, y^{(1)}), (x_l^{(2)}, y^{(2)}), \ldots, (x_l^{(m_l)}, y^{(m_l)})\}$, the supervised dataset has been represented in $\{(a_l^{(1)}, y^{(1)}), (a_l^{(2)}, y^{(2)}), \ldots, (a_l^{(m_l)}, y^{(m_l)})\}$. Finally, after the transformation, a supervised learning algorithm (SVM, logistic regression, Softmax, etc.) could be used for classification tasks.

4) Training Stacked Autoencoders: Autoencoders have been used as building blocks to build and initialize a deep multi-layer neural network. The training procedure would be [31]:

1) Train the first layer as an autoencoder to minimise some form for reconstruction error of the raw input. This is unsupervised.
2) The hidden units' outputs of the autoencoder are now used as input for another layer, also trained to be an autoencoder. Here unlabelled representations were used as well.
3) Iterates as in 2) to initialize the desired number of additional layers.
4) Take the last hidden layer output as input to a supervised layer and initialize its parameters (either randomly or by supervised training, keeping the rest of the network fixed).
5) Fine tune all the parameters of this deep architecture with respect to the supervised criterion. Alternately, unfold all the autoencoders into a very deep autoencoder and fine-tune the global reconstruction error.

The greedy layer-wise approach for pre-training a deep network works by training each layer in turn as explained in step 2). Assume $a^{(n)}$ as the deepest activation of the autoencoder network, then $a^{(n)}$ is a higher level representation than any lower layers, which contains what we interested in. Then the higher level representations (the corresponding features in the traditional artificial selected features) can be used as the classifier input.

5) Fine-tuning: For the training method of stacked autoencoders, when the parameters of one layer are being trained, parameters in other layer are kept fixed. In order to achieve better result, fine-tuning using backpropagation can be used to improve the model performance by tuning the parameters of all layers are changed at the same time after the layer-wise train phase.

6) Softmax Classifier: The cardiac arrhythmia classification problem is a kind of classification problems where the class label y may take more than two possible values (the setting of the classes about the classification would be discussed in the following sections). Softmax regression is a supervised learning algorithm which would be adapted as the classifier in conduction with our proposed deep learning (unsupervised) feature (representation) learning methods.

Softmax regression model was generalized from the logistic regression. Similar to the logistic regression, the training set

$$\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})\} \tag{6}$$

of m labelled examples, the input features are $x^{(i)} \in \Re^{(n+1)}$ (with x_0 corresponding to there intercept term). The labels are denoted by

$$y^{(i)} \in \{1, 2, 3, \ldots, k\} \tag{7}$$

which means k classes. Given a test input x, the hypothesis to estimate the probability that $p(y = j|x)$ for each value of $j = 1, \ldots, k$. I.e., the probabilities of the class labels taking on the k different possible values are estimated.

$$h_\theta(x^{(i)}) = \begin{bmatrix} p(y^{(i)} = 1|x^{(i)}; \theta) \\ p(y^{(i)} = 2|x^{(i)}; \theta) \\ \vdots \\ p(y^{(i)} = k|x^{(i)}; \theta) \end{bmatrix} \tag{8}$$

$$= \frac{1}{\sum_{j=1}^{k} e^{\theta_j^T x^{(i)}}} \begin{bmatrix} e^{\theta_1^T x^{(i)}} \\ e^{\theta_2^T x^{(i)}} \\ \vdots \\ e^{\theta_k^T x^{(i)}} \end{bmatrix} \tag{9}$$

in which $\theta_1, \theta_2, \ldots, \theta_k \in \Re^{(n+1)}$ are parameters of the model. The term $\frac{1}{\sum_{j=1}^{k} e^{\theta_j^T x^{(i)}}}$ was normalizes the distribution, so that it sums to one. The cost function adopted for softmax regression is:

$$J(\theta) = -\frac{1}{m}[\sum_{i=1}^{m}\sum_{j=1}^{k} 1\{y^{(i)} = j\}\log\frac{e^{\theta_j^T x^{(i)}}}{\sum_{l=1}^{k} e^{\theta_l^T x^{(i)}}}] \tag{10}$$

where $1\{\cdot\}$ is the indicator function. There is no known closed-form way to solve for the minimum of $J(\theta)$, and an iterative optimisation algorithm synch as gradient descent of L-BFGS could be used for the minimal value. Taking derivatives, the gradient would be:

$$\bigtriangledown_{\theta_j} J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[x^{(i)}(1\{y^{(i)} = j\} - p(y^{(i)} = j|x^{(i)}; \theta))] \tag{11}$$

With this formula for the derivative plugged into an algorithm such as gradient descent, the minimal of $J(\theta)$ could be achieved. The iteration equation of:

$$\theta_j := \theta_j - \alpha \bigtriangledown_{\theta_j} J(\theta)(\text{for each} j = 1, \ldots, k) \tag{12}$$

Generally, the weight decay:

$$\frac{\lambda}{2}\sum_{i=1}^{k}\sum_{j=0}^{n} \theta_{jk}^2 (\lambda > 0) \tag{13}$$

would be incorporated as well, with which the cost function $J(\theta)$ would be strictly convex and a unique solution would be guaranteed. The Hessian is now invertible, and because $J()$ is convex, algorithms such as gradient descent, L-BFGS, etc. are guaranteed to converge to the global minimum. So the cost function and iteration equations would be:

$$J(\theta) = -\frac{1}{m}[\sum_{i=1}^{m}\sum_{j=1}^{k} 1\{y^{(i)} = j\}\log\frac{e^{\theta_j^T x^{(i)}}}{\sum_{l=1}^{k} e^{\theta_l^T x^{(i)}}}] + \frac{\lambda}{2}\sum_{i=1}^{k}\sum_{j=0}^{n} \theta_{jk}^2 (\lambda > 0) \tag{14}$$

and

$$\nabla_{\theta_j} J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[x^{(i)}(1\{y^{(i)}=j\} - p(y^{(i)}=j|x^{(i)};\theta))] + \lambda\theta_j(\lambda > 0) \quad (15)$$

By minimising $J(\theta)$ with respect to θ, the softmax regression classifier would work properly for the classification task.

B. Setup and Workflow

Similar to the routine of electrocardiography classification task, the workflow consists of three stages: a prepossessing stage, a processing stage, and a classification stage. The prepossessing stage related technologies are not the focus of this study, so the classical methods for prepossessing were adapted and just a brief introduction of the details would be mentioned. The processing stage and classification stage related theories and technologies had been illustrated in last section.

1) ECG Filtering: In the preprocessing stage, filtering algorithms were adapted to remove the artefact signals from the ECG signal. The signals include baseline wander, power line interference, and high-frequency noise. In the traditional ECG classification tasks, the filtering process may affect the results more or less, but in this study, the proposed autoencoder networks structure has the characteristics to predict the missing values from the non-missing values [31], which means even the input signal might be distorted corrupted by the noise interference, the autoencoder has the ability to undo the effect of a corruption process stochastically applied to the input of the autoencoder, a farther description of the essentials is in [35], and in the results section a contrast would be proposed. For the unlabelled database of ambulatory ECG and the MITBIH LT database, the Lead I data were extracted and a resample from 128Hz to 360Hz procedure was adopted for data consistency.

2) Heartbeat Detection: For the heartbeat detection, the MIT-BIH database and unlabelled database, the positions of R waves are determined. The provided fiducial points of R wave had been used as the basis of wave segmentation. The details of the implementation of R wave detection would not be described in this study, and a reference for the R wave detection algorithms had been explored in [36].

3) Heartbeat Segementation: In the heartbeat segmentation process, the segmentation program of Laguna et al.[1] was adapted, which also had been validated by other related work [2]. The segmentation process was focus on the Lead I of the recordings. After the segmentation for the ambulatory ECG database, three batches of heartbeat samples listed in Table II were acquired for the classification task. As for the pre-training, fine-tuning for our proposed task and comparison, we divided all the samples into three groups: the pre-training group as DS1, the fine-tuning group as DS2 and test group as DS3 (illustrated in Table IV). Samples are chosen randomly from the original AR and LT database, the details of the sample class would be described in the experiment result analysis.

[1]"ecgpuwave", check the website of Physionet

TABLE II
SAMPLES AFTER SEGEMENTATION

Ambulatory ECG Database (AECG)	MITBIH-AR	MITBIH-LT
9,785,500	89,925	600,197

TABLE III
SAMPLES DATASET SETTINGS

Dataset	DS1	DS2	DS3
Useage	Pre-training	Fine-tuning	Test
Source (samples)	AECG (9,785,500)		
	AR (50,193)	AR (33,663)	AR (16,831)
	LT (587,347)	LT (50,000)	LT (30,000)
Total	10,423,040	83,633	46,831

4) Pre-training: The stack autoencoder use multilayer "encoder" network to transform high dimensional data into low dimensional code, similarly a "decoder" network can be adopted to recover from the code, which we previously described as a representations or features. For the one-hidden-layer autoencoder input layer and hidden layer as depicted in (a) of Figure 2, as the output was set equal to the input, starting with random weights in the one-hidden layer neural networks, they can be trained together by minimizing the discrepancy between the original input data and its reconstruction. The gradients were obtained by using chain rule of backpropagate error derivatives, the decoder means the raw input can be reconstruct by the learned feature with the trained weight. With large initial weights, autoencoders typically find poor local minima; with small initial weights, the gradients in the early layers are tiny, making it infeasible to train autoencoders with many hidden layers [37].

After learning the feature and network weight in the first layer, we can add hidden layer one by one to get deeper representations, as well the learned weight can be used to reconstruct the input. When training the weight of layer 2, we take the weight in layer 1 fixed replace random initialize because the learned weights are close to a good solution, which means training the parameters of each layer individually while freezing parameters for the remainder of the model. The pre training process is illustrated in Figure 2. As this work focus on the class-action task, so the "decoding" layers were discarded and link the last hidden layer to the softmax classifier.

TABLE IV
SAMPLES DATASET SETTINGS

Number of hidden layer	2	3	4
Nodes	200×100	200×100×100	200×100×100×100

In the experiment, we adopted 2-hidden-layer, 3-hidden-layer, 4-hidden-layer stacked autoencoder for the test and verify.

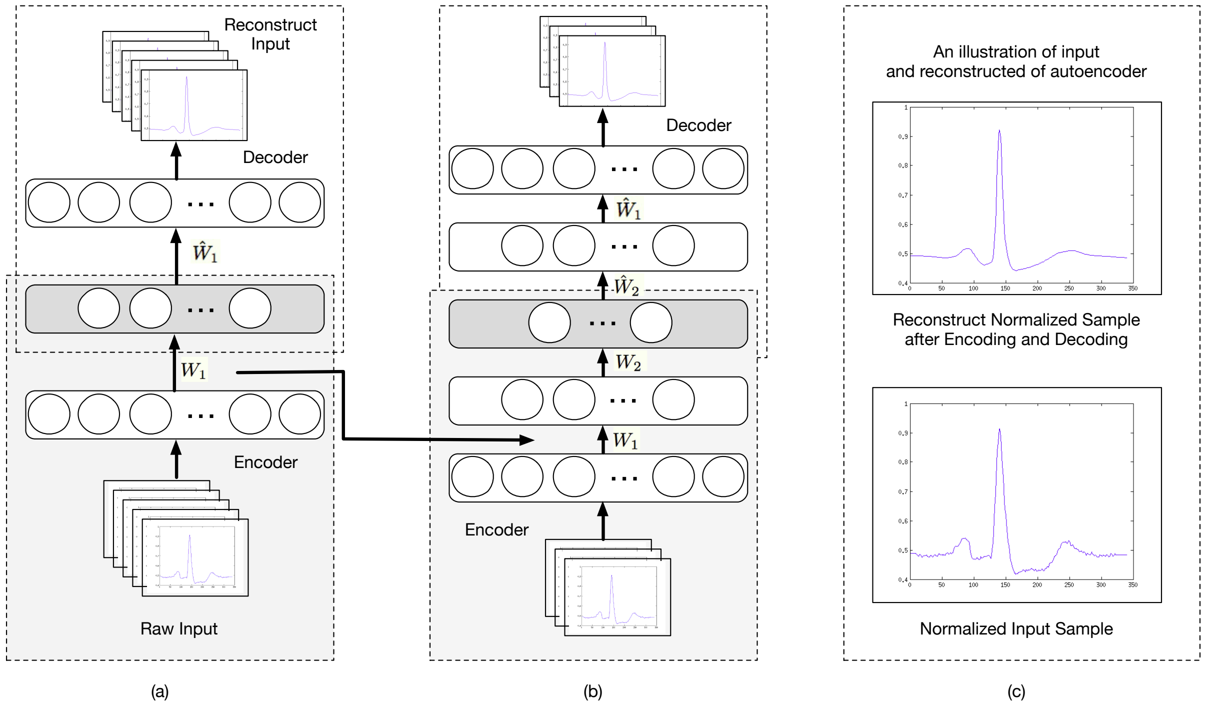

Fig. 2. The autoencoder pre-training process. (a) of figure 2 illustrated the one-hidden-layer autoencoder encoding and decoding processes; (b) of figure 2 illustrated a deeper network of 2-hidden-layer autoencoder which used the weight W_1 and outputs of feature layer (the gray layer) learned in (a); (c) of figure 2 is a comparison of normorlized data of raw input sample and reconstructed sample.

5) Fine-tuning: Fine tuning is a strategy that widely used in deep learning, which can be used to greatly improve the performance of a stacked autoencoder. After pre-training multiple layers of feature detectors, the model is "unfold" to produce encoder and decoder networks that initially use the same weights [37]. The weights learned can be used for classification implementation after adding one classifier after the feature layer. In this study, a softmax classifier was added (Figure 3). In the fine-tuning initialization, the parameters learned in the autoencoder pre-training were used, and the weights W and biases b of softmax classifier (the last layer of the network) were initialized randomly. The training set of DS2 were used in the supervised learning pre-training while the backpropagation algorithm as usual of multi-layer perceptrons to minimise the output prediction error has been adopted.

6) Test and Classifier Performance Assessment: After the pre-training and fine-tuning process, the deep network parameters were acquired. Then we use the parameters and the test data set DS3 to predict the class of samples. It is necessary to mention that in DS2 and DS3, the labelled data used in pre-training and fine-tuning were divided randomly, which satisfy the requirement of Holdout cross-validation scheme so that the test results were meaningful for the classification task performance improvement.

The following statistical parameters of test performance were used in the study:

1) Specificity: number of correctly classified normal beats over total number of normal beats.

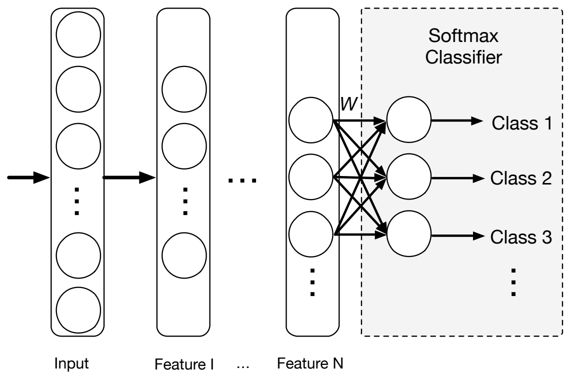

Fig. 3. The softmax classifier. In each layer a bias node was included which had not been illustrated in the figure.

2) Sensitivity: number of correctly classified abnormal beats over total number of the given abnormal beats.
3) Overall classification accuracy: number of correctly classified beats over number of total beat.

IV. Experimental Results

A. The Test Results

As previously mentioned, we adopted three different layer strategies for the classification task. In the 2-hidden-layer autoencoder network, we got a accuracy of 99.33%. For the N class the specificity is 99.76%, the sensitivity of S class is

80.08%, the sensitivity of V class is 98.13%, the sensitivity of F class is 85.48% as illustrated in Table V.

TABLE V
TEST RESULT FOR 2-HIDDEN-LAYER AUTOENCODER NETWORK

		Algorithm label					
		N	S	V	F	Q	T
Reference label	N	41,965	39	45	13	6	42,068
	S	91	398	6	2	0	497
	V	63	3	3,940	5	4	4,015
	F	23	0	13	212	0	248
	Q	2	1	0	1	0	3

The test accuracy is about 99.33%.

In the 3-hidden-layer autoencoder network, we got a accuracy of 99.07%. For the N class the specificity is 99.64%, the sensitivity of S class is 75.14%, the sensitivity of V class is 97.58%, the sensitivity of F class is 80.33% as illustrated in Table VI.

TABLE VI
TEST RESULT FOR 3-HIDDEN-LAYER AUTOENCODER NETWORK

		Algorithm label					
		N	S	V	F	Q	T
Reference label	N	41,721	66	66	19	0	41,872
	S	120	405	13	1	0	539
	V	74	10	4,073	17	0	4,174
	F	27	2	19	196	0	244
	Q	2	0	0	1	0	2

The test accuracy is about 99.07%.

In the 4-hidden-layer autoencoder network, we got a accuracy of 99.34%. For the N class the specificity is 99.74%, the sensitivity of S class is 82.29%, the sensitivity of V class is 98.31%, the sensitivity of F class is 87.71% as illustrated in Table VII.

TABLE VII
TEST RESULT FOR 4-HIDDEN-LAYER AUTOENCODER NETWORK

		Algorithm label					
		N	S	V	F	Q	T
Reference label	N	41,778	38	48	17	5	41,886
	S	93	460	3	1	2	559
	V	52	1	4,067	11	6	4,137
	F	15	0	13	214	2	244
	Q	1	0	1	1	1	5

The test accuracy is about 99.34%.

B.

C.

D.

E.

F.

V. DISCUSSION AND CONCLUSIONS

A. Subsection Heading Here

Subsection text here.

1) Subsubsection Heading Here: Subsubsection text here.

APPENDIX A
PROOF OF THE FIRST ZONKLAR EQUATION

Appendix one text goes here.

APPENDIX B

Appendix two text goes here.

ACKNOWLEDGMENT

The authors would like to thank...

REFERENCES

[1] T. Mar, S. Zaunseder, J. Martinez, M. Llamedo, and R. Poll, "Optimization of ecg classification by means of feature selection," *Biomedical Engineering, IEEE Transactions on*, vol. 58, no. 8, pp. 2168–2177, Aug 2011.

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

[3] F. Melgani and Y. Bazi, "Classification of electrocardiogram signals with support vector machines and particle swarm optimization," *Information Technology in Biomedicine, IEEE Transactions on*, vol. 12, no. 5, pp. 667–677, Sept 2008.

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

[5] T. Olmez, "Classification of ecg waveforms by using rce neural network and genetic algorithms," *Electronics Letters*, vol. 33, no. 18, pp. 1561–1562, Aug 1997.

[6] C.-W. Lin, Y.-T. Yang, J.-S. Wang, and Y.-C. Yang, "A wearable sensor module with a neural-network-based activity classification algorithm for daily energy expenditure estimation," *Information Technology in Biomedicine, IEEE Transactions on*, vol. 16, no. 5, pp. 991–998, Sept 2012.

[7] S. Osowski and T. H. Linh, "Ecg beat recognition using fuzzy hybrid neural network," *Biomedical Engineering, IEEE Transactions on*, vol. 48, no. 11, pp. 1265–1271, Nov 2001.

[8] M. Kundu, M. Nasipuri, and D. Basu, "A knowledge-based approach to ecg interpretation using fuzzy logic," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 28, no. 2, pp. 237–243, Apr 1998.

[9] R. Andreao, B. Dorizzi, and J. Boudy, "Ecg signal analysis through hidden markov models," *Biomedical Engineering, IEEE Transactions on*, vol. 53, no. 8, pp. 1541–1549, 2006.

[10] D. Coast, R. Stern, G. Cano, and S. Briller, "An approach to cardiac arrhythmia analysis using hidden markov models," *Biomedical Engineering, IEEE Transactions on*, vol. 37, no. 9, pp. 826–836, 1990.

[11] Y. Zhu, A. Shayan, W. Zhang, T. L. Chen, T.-P. Jung, J.-R. Duann, S. Makeig, and C.-K. Cheng, "Analyzing high-density ecg signals using ica," *Biomedical Engineering, IEEE Transactions on*, vol. 55, no. 11, pp. 2528–2537, Nov 2008.

[12] A. Kampouraki, G. Manis, and C. Nikou, "Heartbeat time series classification with support vector machines," *Information Technology in Biomedicine, IEEE Transactions on*, vol. 13, no. 4, pp. 512–518, 2009.

[13] A. Khandoker, M. Palaniswami, and C. Karmakar, "Support vector machines for automated recognition of obstructive sleep apnea syndrome from ecg recordings," *Information Technology in Biomedicine, IEEE Transactions on*, vol. 13, no. 1, pp. 37–48, Jan 2009.

[14] C.-P. Shen, W.-C. Kao, Y.-Y. Yang, M.-C. Hsu, Y.-T. Wu, and F. Lai, "Detection of cardiac arrhythmia in electrocardiograms using adaptive feature extraction and modified support vector machines," *Expert Systems with Applications*, vol. 39, no. 9, pp. 7845 – 7852, 2012.

[15] I. Jekova, G. Bortolan, and I. Christov, "Assessment and comparison of different methods for heartbeat classification," *Medical Engineering & Physics*, vol. 30, no. 2, pp. 248–257, 2008.

[16] I. Christov, G. Gomez-Herrero, I. Krasteva, I. Jekova, A. Gotchev, and K. Egiazarian, "Comparative study of morphological and time frequency ecg descriptors for heartbeat classification," *Medical Engineering & Physics*, vol. 28, no. 9, pp. 876–887, 2006.

[17] C. Ye, B. Kumar, and M. Coimbra, "Heartbeat classification using morphological and dynamic features of ecg signals," *Biomedical Engineering, IEEE Transactions on*, vol. 59, no. 10, pp. 2930–2941, Oct 2012.
[18] O. T. Inan, L. Giovangrandi, and G. T. A. Kovacs, "Robust neural-network-based classification of premature ventricular contractions using wavelet transform and timing interval features," *IEEE Transactions on Biomedical Engineering*, vol. 53, no. 12, pp. 2507–2515, 2006.
[19] S. Banerjee and M. Mitra, "Application of cross wavelet transform for ecg pattern analysis and classification," *IEEE Transactions on Instrumentation and Measurement*, vol. 63, no. 2, pp. 326–333, 2014.
[20] T. Stamkopoulos, K. Diamantaras, N. Maglaveras, and M. Strintzis, "Ecg analysis using nonlinear pca neural networks for ischemia detection," *Signal Processing, IEEE Transactions on*, vol. 46, no. 11, pp. 3058–3067, Nov 1998.
[21] M. Lagerholm, C. Peterson, G. Braccini, L. Edenbrandt, and L. Sornmo, "Clustering ecg complexes using hermite functions and self-organizing maps," *Biomedical Engineering, IEEE Transactions on*, vol. 47, no. 7, pp. 838–848, Jul 2000.
[22] G. D. Clifford, F. Azuaje, and P. McSharry, *Advanced Methods And Tools for ECG Data Analysis*. Norwood, MA, USA: Artech House, Inc., 2006.
[23] A. L. Goldberger, L. A. N. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley, "PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals," *Circulation*, vol. 101, no. 23, pp. e215–e220, 2000 (June 13).
[24] H. Nishizawa, T. Obi, M. Yamaguchi, and N. Ohyama, "Hierarchical clustering method for extraction of knowledge from a large amount of data," *Optical Review*, vol. 6, no. 4, pp. 302–307, 1999.
[25] C. Maier, H. Dickhaus, and J. Gittinger, "Unsupervised morphological classification of qrs complexes," in *Computers in Cardiology, 1999*, 1999, pp. 683–686.
[26] S. Palreddy, W. J. Tompkins, and Y. H. Hu, "Customization of ecg beat classifiers developed using som and lvq," in *Engineering in Medicine and Biology Society, 1995., IEEE 17th Annual Conference*, vol. 1, Sep 1995, pp. 813–814 vol.1.
[27] M. Risk, J. Sobh, and J. Saul, "Beat detection and classification of ecg using self organizing maps," in *Engineering in Medicine and Biology Society, 1997. Proceedings of the 19th Annual International Conference of the IEEE*, vol. 1, Oct 1997, pp. 89–91 vol.1.
[28] P. Tadejko and W. Rakowski, "Hybrid wavelet-mathematical morphology feature extraction for heartbeat classification," in *EUROCON, 2007. The International Conference on Computer as a Tool*, Sept 2007, pp. 127–132.
[29] D. Erhan, P.-A. Manzagol, Y. Bengio, S. Bengio, and P. Vincent, "The difficulty of training deep architectures and the effect of unsupervised pre-training," in *International Conference on Artificial Intelligence and Statistics*, 2009, pp. 153–160.
[30] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proceedings of the 25th international conference on Machine learning*. ACM, 2008, pp. 160–167.
[31] Y. Bengio, "Learning deep architectures for ai," *Found. Trends Mach. Learn.*, vol. 2, no. 1, pp. 1–127, Jan. 2009.
[32] H. Bourlard and Y. Kamp, "Auto-association by multilayer perceptrons and singular value decomposition," *Biological Cybernetics*, vol. 59, no. 4-5, pp. 291–294, 1988.
[33] N. Japkowicz, S. Jose Hanson, and M. A. Gluck, "Nonlinear autoassociation is not equivalent to pca," *Neural Comput.*, vol. 12, no. 3, pp. 531–545, Mar. 2000.
[34] Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle, U. D. Montral, and M. Qubec, "Greedy layer-wise training of deep networks," in *In NIPS*. MIT Press, 2007.
[35] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proceedings of the 25th International Conference on Machine Learning*, ser. ICML '08. New York, NY, USA: ACM, 2008, pp. 1096–1103.
[36] V. Afonso, W. J. Tompkins, T. Nguyen, and S. Luo, "Ecg beat detection using filter banks," *Biomedical Engineering, IEEE Transactions on*, vol. 46, no. 2, pp. 192–202, Feb 1999.
[37] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, Jul. 2006.

PLACE
PHOTO
HERE

Jan Doe Biography text here.

John Doe Biography text here.

Jane Doe Biography text here.