Neurocomputing 404 (2020) 122–136

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

# Generating electrocardiogram signals by deep learning

Naren Wulan[a], Wei Wang[a,*], Pengzhong Sun[a], Kuanquan Wang[a,1], Yong Xia[a], Henggui Zhang[b]

[a] School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China
[b] Biological Physics Group, Department of Physics and Astronomy, University of Manchester, Manchester, UK

ARTICLE INFO

*Article history:*
Received 21 June 2019
Revised 31 March 2020
Accepted 11 April 2020
Available online 13 May 2020

Communicated by Dr. Nianyin Zeng

*Keywords:*
Deep learning
Evaluation approach
Generative model
Synthetic ECG
Short-term Fourier transform
Stationary wavelet transform
$\mu$-Law companding transformation

ABSTRACT

Given the importance of a diverse and vast amount of realistic and labeled electrocardiogram (ECG) signals in improving the performance of biomedical signal processing algorithms, and the situation of severe lack of the signals, three generative models based on deep learning are introduced for the generation of ECG signals: The WaveNet-based model, the SpectroGAN model, and the WaveletGAN model. The WaveNet-based model adopts $\mu$-law companding transformation as a preprocessing method and then is followed by a sequence of convolutional layers with dilation; SpectroGAN and WaveletGAN use short-term Fourier transform (STFT) and stationary wavelet transform (SWT) respectively to obtain suitable input form for the generative adversarial networks (GAN). Our proposed models are capable of generating ECG signals containing three different heartbeat types: normal beat, left bundle branch block beat and right bundle branch block beat. The synthetic ECG signals generated by our models are more realistic since deep artificial neural networks can discover intricate structure and characteristics of real ECG signals instead of manually setting specific parameters for synthesis. Besides, ECG signals produced by one of our proposed models could be naturally continuous and be up to more than 20 seconds. Furthermore, we first provide an evaluation approach for quantitatively demonstrating the performance of ECG generative models. The study demonstrates that deep learning is a feasible and effective method for ECG generation. Our proposed ECG generative models can be utilized to assess biomedical signal processing algorithms so as to improve their performance in clinical trials.



## 1. Introduction

ECG has become a promising tool to achieve automatic disease detection by using ECG related signal processing algorithms [1,2]. At present, these algorithms are usually based on those already existed databases such as the Physionet database to evaluate the feasibility and accuracy of the algorithms [3]. However, due to the exponential booming of information, the amount of the existing database is limited and insufficient. In addition, with the fact that data annotation is quite a time, labor and money consuming task, it is impossible to obtain such a large amount of data with labels. Meanwhile, the performance of the algorithms could vary greatly in different clinical conditions, which shows the necessity of large amounts of annotated data to ensure the efficiency of the algorithms. In other words, without the support of plenty of labeled data, even if the algorithms can perform very well on the given database, it may not be wildly used in reality. Therefore, synthesizing realistic artificial ECG signals is of immense significance for helping researchers to improve the performance of their algorithms in the field of ECG signal processing. By generating artificial and natural-looking samples, the researchers can achieve the goal of data augmentation, acquiring abundant samples that are not included in the original database to increase training data, and they can even obtain samples with particular properties.

So far, there are many studies on synthetizing ECG signals by using mathematical modeling. In 2003, McSharry et al. [4] proposed a two-stage dynamic model to generate artificial ECG signals. Firstly, they produced an internal time series by specifying spectral parameters and temporal parameters (mean heart rate and standard deviation) of a real R-R tachogram. Secondly, they constructed equations of motion, a three-dimensional (3-D) trajectory model, and specify locations and heights of the peak of each heartbeat to produce the average morphology of ECGs. P, Q, R, S and T waves are respectively represented by a set of gaussian equations within this trajectory model, and the ECG signal can be regarded as the sum of the gaussian equations of these waves. By adjusting the angular velocity of the trajectory, the waveform of the R-R interval could be changed. In 2005, Li and Ma [5] proposed

* Corresponding author.
*E-mail address:* weiwang_sherry@yeah.net (W. Wang).
[1] Senior Member, IEEE.

https://doi.org/10.1016/j.neucom.2020.04.076
0925-2312/

a data flow graph method based on piecewise curve to model ECG signals. The algorithm needs to model the P wave, QRS wave and T wave respectively and then synthesize a heartbeat, which involves many parameters, thus increasing the complexity of the algorithm. Sameni et al. in 2007 [6] proposed a 3-D dynamic model, which is a single dipole model for the heart and combines a linear model to indicate the temporal motion and rotation of the cardiac dipole. The model can be generalized into a multi-lead model. Compared with the single-lead model proposed by McSharry above, the multi-lead model proposed by Sameni avoids the problem of repeated calculation of parameters during the multi-lead modeling. From the perspective of the dipole model of the heart, the ECG signals of different leads are actually the projection of the dipole vector of the heart onto the axis of the ECG recording electrode. The disadvantage of this algorithm is that the ECG signals reconstructed from this model do not match the real ECG signals very well, especially at low frequencies such as P wave. This shows the limitation of the single dipole model in representing the low-frequency components of ECG. On the basis of Sameni's 3-D dynamic model, Clifford et al. in 2010 [7] proposed a method that used first-order Markov chains to alternately generate normal and abnormal heartbeats and used transition matrix to represent the probability of the transition between normal and abnormal heartbeats. In 2013, Roonizi et al. [8] proposed a signal decomposition-based method to model ECG signals since a single ECG heartbeat can be viewed as a sum of basis functions. They tried many different basis functions in the method such as the gaussian model, polynomial spline models (including Bézier and B-splines) and sinusoidal model. During their experiments, they found that the sinusoidal model and Bézier function didn't meet the requirements on the local control, and B-spline function could provide good local control required for modeling. Those traditional mathematical modeling methods have some deficiencies as follows:

- Although in terms of ECG morphology their model can produce very realistic heartbeats, these synthetic heartbeats are too "standard". In other words, the morphology of each heartbeat in a synthetic ECG signal is too normal and is basically the same.
- Despite that this model is able to be modified to produce abnormal heartbeat, such a modification could be complicated and cumbersome since every type of abnormal heartbeat is corresponding to a certain set of equations.
- Operators using this kind of model need to possess the expertise to make sure the accuracy, rationality, and reliability of the parameter setting.
- Besides, the evaluation of the quality of generated signals mainly relies on visual observation with professional knowledge rather than scientific and objective methods.

On the other hand, the rapid development of deep learning over past years has made it be applied successfully in many domains such as speech recognition [9], image recognition [10], object detection [11], as well as in biomedical field [12–21]. Nowadays, many ECG related detecting algorithms and signal processing algorithms based on deep learning has far exceeded traditional methods in the aspect of accuracy. For example, in 2017, the team of Professor Andrew Y. Ng from Stanford University used a deep convolutional network combined with a residual network to realize single-lead arrhythmia signal detection, which claimed that the detection accuracy of the algorithm was close to or even outstripped professional cardiologists [22]. Besides the above-mentioned discriminative model, the generative model has also progressed dramatically in recent years. In 2014, Ian J. Goodfellow from Université de Montréal proposed Generative Adversarial Networks (GAN) [23], which had been widely used in computer vision for generating human face [24,25], producing a variety of realistic scenes [26–28], painting [29,30], etc. Moreover, GAN is also applied to solve problems of speech and language processing [31,32]. Besides GAN, Google's DeepMind team in 2016 proposed a network WaveNet [33] specifically for generating one-dimensional (1-D) signals. It can produce fairly natural-sounded speech signals with long-range temporal dependencies. This method is also able to be used to generate different voices by conditioning on speaker identity.

In this paper, we presented three methods for synthesizing artificial ECG signals by using deep learning techniques. To our knowledge, this is the first study using deep learning for the purpose of generating ECG signals. Compared with traditional algorithms, the proposed approaches have a number of advantages as follows:

- The synthetic ECG signals are more realistic since they can reproduce the essential characteristics and intricate structure of real ECG signals.
- The synthetic ECG signals have longer dependence and correlation in time, which can last up to 20 seconds.
- The synthetic ECG signals are more abundant in morphology.
- The synthetic ECG signals could be of a certain type of heartbeats.
- The synthetic ECG signals could be of different sampling frequencies.
- The lower complexity of the operation and less man-made control of parameters.

In addition, we first put forward a quantitative approach to evaluate the performance of generative models for ECG generation. This paper is organized as follows: Section II illustrates the database and datasets used in our study and three different methods for generating ECG signals. The results are presented in Section III. Section IV discusses the performance of our proposed three generative models, and we conclude the paper in Section V.

## 2. Methods

We designed three different methods to produce ECG signals depending on the two popular approaches for generating data mentioned above: WaveNet and GAN. Fig. 1 shows the whole architecture of our proposed methods for ECG generation. The first method uses data in the encoded form as the input of WaveNet to produce continuous ECG signals. The second and third approaches are aiming at generating signal heartbeat via data transformation and GAN. The reason for using GAN to merely synthesize a single heartbeat is primarily due to the characteristics of GAN itself and its training method, and we will discuss this in detail later. In the end, an evaluation method based on a support vector machine (SVM) is originally used to demonstrate the performance of our proposed ECG synthetic models by comparing GAN-train and GAN-test scores.

### 2.1. Database and ECG signal preprocessing

In this study, we use the MIT-BIH arrhythmia database [34], which is publicly accessible from PhysioNet, to train our models. The MIT-BIH arrhythmia database contains 48 ECG recordings, each about 30 minutes long with a sampling rate of 360 Hz. Each ECG record contains two leads called MLII and V5. In this study, we only choose the MLII for generation. Our training set includes three types of heartbeats: normal beat (N), left bundle branch block beat (L) and right bundle branch block beat (R). Seven records (#106, #114, #203, #213, #221, #222, #228) of the MIT-BIH arrhythmia database were excluded in this study because they contain too many other types of heartbeat, such as atrial premature beat, which can cause trouble in obtaining continuous data of type N for the purpose of generation. The other four records (#102, #104, #107, #217) were also excluded since they contain paced heartbeats.

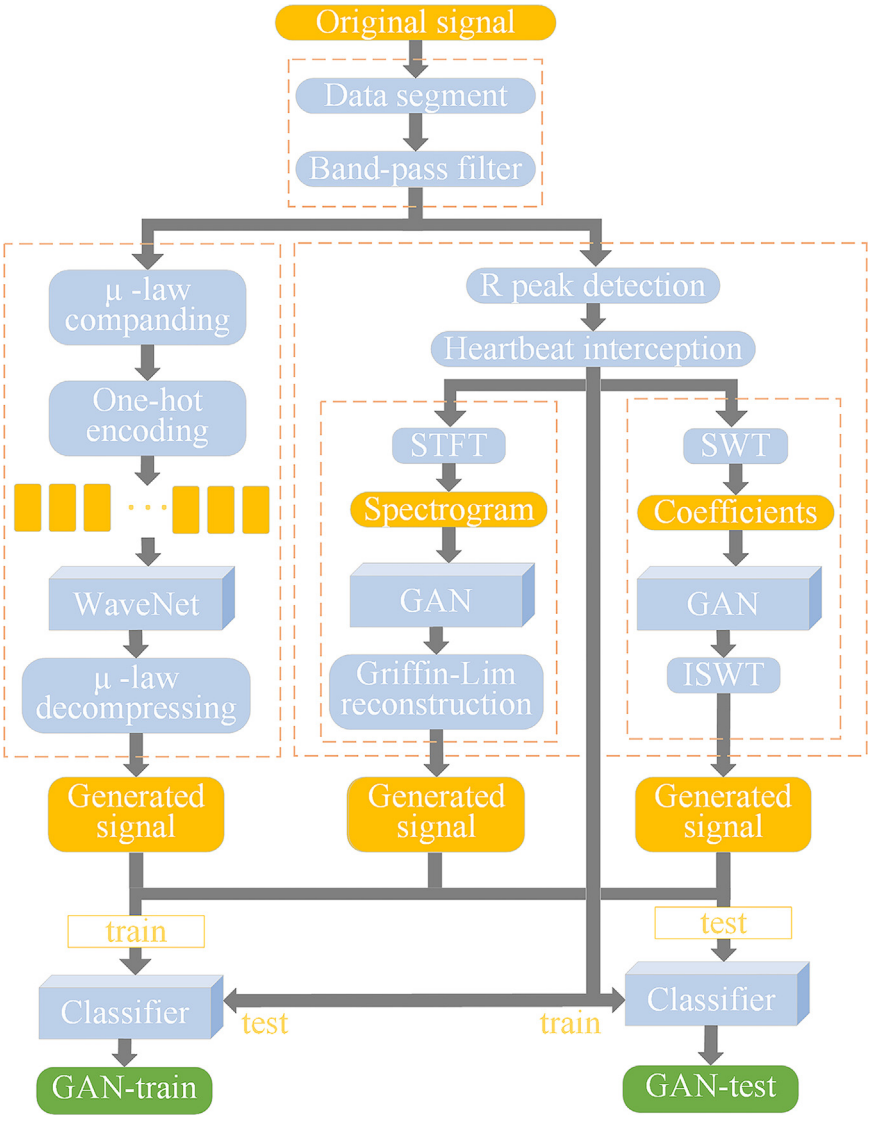


**Fig. 1.** Schematic overview of the proposed approaches.

Every remaining ECG record was segmented by a certain length of time T (4 seconds in our study) to obtain a number of data segments of length T. Each data segment can only be labeled as one type of N, L or R, for example, a data segment of type N means that all heartbeats it contains are N-type. Considering the situation for making the neural networks converge quickly and the training stable, each data segment was filtered using a 0.1–100 Hz band-pass filter to remove noise. Of course, this step could be omitted if one wants to produce ECG signals with noises. In our study, we extracted 10,659 4-s data segments of N-type, 1371 4-s data segments of L-type and 1311 4-s data segments of R-type. We randomly extracted 50 samples from 10,659 data segments of N-type, 50 samples from 1371 data segments of L-type and 50 samples from 1311 data segments of R-type to compose a test set to assess the performance of our ECG signals synthetic methods. It can be obviously observed that the number of samples of the three different types is so unbalanced that we cannot use them to directly build our training set. To address this problem, we expanded the number of data segments of L-type and R-type by four times through replicating those remaining data segments of L-type and R-type, and then randomly extracted 5500 N-type data segments. Ultimately, the training set has 15,828 4-s samples and the test set has 150 4-s samples. This data set is used for the WaveNet method.

Another training set containing complete heartbeat samples was also constructed for our proposed GAN related methods according to their characteristics. We used the Pan-Tompkins algorithm [35] to locate QRS waves on those ECG records we selected. On the basis of the position of QRS wave, we intercepted 100 sampling points forward and 150 sampling points backward (250 points in total) to form a heartbeat. Although this interception method is a rough approach to obtain a heartbeat, it is commonly used by researchers considering the difficulty to detect the starting or ending points of P and T waves. Besides, such an interception method is able to avoid overlap between heartbeats since the original sampling frequency of the MIT-BIH arrhythmia database is 360 Hz. If we use the length 360 points as a unit, we could probably obtain a data segment with incomplete heartbeats, for example including the last half of the former heartbeat and the first half of the latter heartbeat. This will cause a lot of trouble in constructing ECG signals because a complete single heartbeat is essential to build up a very long ECG signal. We marked each heartbeat according to the annotation offered by the MIT-BIH arrhythmia database. Finally, we obtained 42,636 N-type heartbeats,

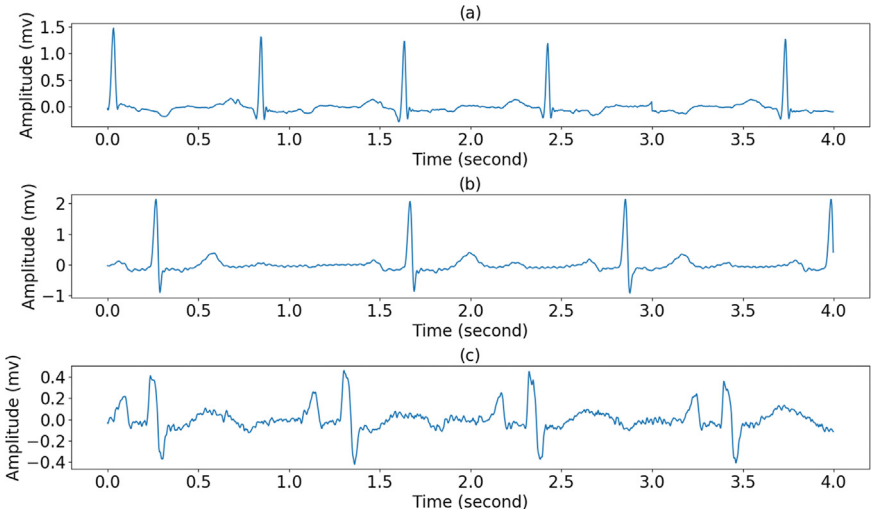


**Fig. 2.** Samples of the training set 1: (a) N-type sample, (b) L-type sample, (c) R-type sample.

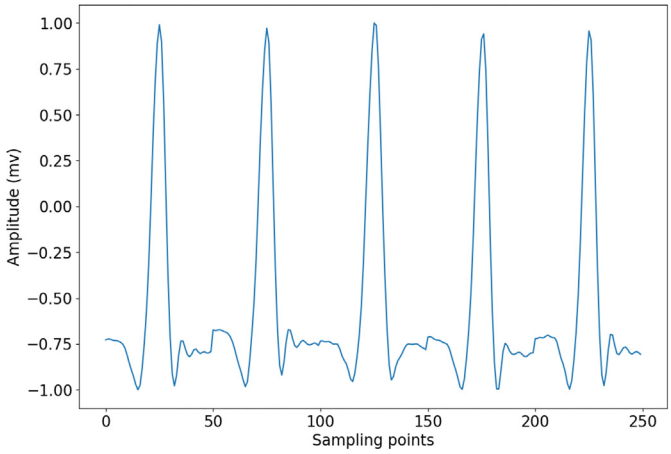


**Fig. 3.** Sample of the training set 3.

7068 L-type heartbeats, and 6254 R-type heartbeats. We randomly selected 5000 heartbeats from each of the three types of data respectively to form a training set and 500 beats from the rest of the data respectively to form a test set.

Besides the two training sets mentioned above, we also built a training set of R-wave peaks of N-type. This R-wave peaks data set represents a simple form of ECG, as a result, we can use this data set to design an initial generating model which is then capable of being modified into an appropriate model for a complete ECG data segment. To build this data set, we first used the Pan-Tompkins algorithm over a data set of N-type to detect QRS waves. Then a window with a width of 50 points is used to intercept each of the 25 sampling points near the left side and right side of each R-peak, finally obtaining an R-peak segment of length 50 points. Each sample in the R-wave peaks data set consists of 5 R segments thus its length is 250 points. Finally, the training set is composed of 10,000 samples, and the corresponding test set includes 500 samples.

For simplicity, we orderly name the above-mentioned data sets as the training set 1, the test set 1, the training set 2, the test set 2, the training set 3 and test set 3. Here we only give images of the training set 1 and the training set 3 as shown in Fig. 2 and Fig. 3, respectively, since the morphology of the sample in the training set 2 is just the same as one single heartbeat in the training set 1.

### 2.2. WaveNet-based method

WaveNet is a likelihood-based autoregressive model, which was initially used for generating audio waveforms like speech or music [33]. The joint probability of an ECG signal $x = \{x_1, x_2, \ldots, x_T\}$ can be expressed in terms of chain of conditional probabilities as follows:

$$p(x) = \prod_{t=1}^{T} p(x_t | x_1, ..., x_{t-1}) \tag{1}$$

Therefore, each sample point $x_i$of an ECG signal is conditioned on sample points of all previous timesteps. WaveNet is built using stacks of convolutional layers with residual, and skip connections in between and is no pooling layers. The last layer of the model is the softmax layer which is used to output a categorical distribution over $x_i$. The most remarkable characteristic of WaveNet is its ability to generate relatively long signals with high quality, and this is the main reason for us to adopt this method for ECG generation. In order to produce long signals, WaveNet uses dilated convolutional layers to exponentially increase the receptive field and to simultaneously make sure that the model gives its prediction in the order that the prediction $p(x_{i+1} | x_1, \ldots, x_i)$ at timestep $i$ can merely depend on the previous timesteps but not on any future timesteps. With dilated convolutional layers, WaveNet is no need superimposing many layers or enlarging filters to expanding the receptive field, which is able to greatly reduce computational cost. Besides, WaveNet uses skip connections [36] to speed up convergence and enable the training of much deeper models [33]. The gradient for the early layers doesn't need to pass through all consecutive layers (that could cause the "vanishing gradient" problem), but can go straight from the output of the neural network through a skip connection. Skip connections allow us to produce long ECG signals by adding more layers without worsening the performance of the network but making it better.

The model takes digitized ECG signal waveform as input, which then flows through these convolutional layers and outputs a waveform sample point. At generating time, this sample point is then used with previous samples to generate the next sample. In order to digitize ECG signal, we first applied a $\mu$-law encoding [37] to every data segment, and then quantized it:

$$f(x_t) = \text{sign}(x_t) \frac{\ln(1 + \mu|x_t|)}{\ln(1 + \mu)} \tag{2}$$

where -1 < $x_t$ < 1 and $\mu$ is an integer which is equivalent to $2^i - 1$ $(i = 1, 2, \ldots)$. After that, we embedded the quantized data segment into a continuous vector by making each sample point of this data segment represented as a one-hot vector. Accordingly, the softmax layer would need to output $\mu$ probabilities per timestep to model all possible values [33].

### 2.3. GAN-based method

WaveNet has two biggest disadvantages. One is the extremely slow speed of training and generation. The other is that the generation of sampling points only depends on the previous information. By comparison, GAN generates time series in a parallel manner. Its generation needs to take context information into consideration and its speed is greatly improved. GAN consists of two models: a generative model G and a discriminative model D. Both G and D can be made up of deep neural networks. D is a two-class classifier that is used to determine whether the input data is real or not. The task of G is to capture the distribution of real data. The training process of this framework is similar to a minimax two-player game, in which the ultimate goal is that the distribution of generated data is as consistent as possible with the distribution of raw data. The loss function of the original GAN is defined as below:

$$\min_G \max_D V(D, G) = \mathrm{E}_{x\sim P_r}[\log(D(x))] + \mathrm{E}_{x\sim P_g}[\log(1 - D(x))] \quad (3)$$

The first $D(x)$ is the discrimination for real data and the second $D(x)$ is the discrimination for generated data. $P_r$ and $P_g$ represent the distribution of real data and generated data, respectively. When optimizing the discriminative model D, we expect the first $D(x)$ to be as close to 1 as possible and the second $D(x)$ to be as close to 0 as possible, that is, maximizing the following equation:

$$\max_D V(D, G) = \mathrm{E}_{x\sim P_r}[\log(D(x))] + \mathrm{E}_{x\sim P_g}[\log(1 - D(x))] \quad (4)$$

After finishing the optimization of the discriminative model D, we begin to optimize the generative model G by getting the second$D(x)$ to be close to 1, which means minimizing the following equation:

$$\min_G V(D, G) = \mathrm{E}_{x\sim P_g}[\log(1 - D(x))] \quad (5)$$

Compared with the original GAN, deep convolutional GAN (DCGAN) [24] almost completely replaces fully connected layers with convolutional layers. D and G are almost symmetrical, and the entire network does not have a pooling layer and an upsampling layer. Instead, it uses convolution kernels with stripes because convolution kernels have a good effect on extracting image features and can also increase the stability of training. Moreover, it is relatively a good and simple GAN at present to generate a relatively high-resolution image, which is exceedingly critical for postprocessing of the GAN-based methods I proposed.

Unlike DCGAN mainly modifying the network structure of the original GAN, WGAN [38] focuses on modifying the loss function of the original GAN in order to address the problem of unstable training theoretically. When the discriminative model D is optimal, the loss function of the generative model G is to add a term independent of G to Eq. (5):

$$\min_G V(D, G) = \mathrm{E}_{x\sim P_r}[\log(D(x))] + \mathrm{E}_{x\sim P_g}[\log(1 - D(x))] \quad (6)$$

Minimizing Eq. (6) is equivalent to minimizing Eq. (5). Since the original GAN uses Jensen-Shannon (JS) divergence to measure the distance between $P_r$ and $P_g$. Therefore, Eq. (6) could be written as the following expression:

$$\min_G V(D, G) = 2JS(P_r||P_g) - 2\log 2 \quad (7)$$

In the paper of WGAN, they demonstrated [38] that as long as $P_r$ and $P_g$ don't overlap at all or their overlap is negligible, their JS divergence is the constant log2, and that means the gradient is zero. At this point, for the optimal discriminator D, the generator G is definitely not going to get any gradient information, that is, the generator G has a great chance of facing the problem of gradient vanishing. Hence, they replaced JS divergence with Wasserstein distance. Compared with JS divergence, Wasserstein distance could still well reflect the distance between $P_r$ and $P_g$ even if there is no overlap between them. The loss function of WGAN is:

$$L = \mathrm{E}_{x\sim P_r}[f_w(x)] - \mathrm{E}_{x\sim P_g}[f_w(x)] \quad (8)$$

$f$ is a continuous function with the Lipschitz constraint. Lipschitz constraint requires that the gradient of the discriminator should not exceed k. This restriction is realized by weight clipping. During the training process, the weight is clipped into a certain range, such as [-0.01, 0.01], to ensure that all parameters of the discriminator are bounded, which ensures that the discriminator cannot give significantly different scores to two slightly different input data, thus indirectly realizing the Lipschitz constraint.

However, weight clipping can cause two huge problems. First, since the discriminator wants to make the difference between scores of true and fake data as large as possible, and weight clipping limits the value range of all parameters, the value of parameters will take extreme values, that is, the maximum value (such as 0.01) or the minimum value (such as -0.01). In this case, the discriminator is very much inclined to learn a simple function like a binary mapping and fails to take full advantage of its strong fitting ability. Then the gradient returned back to the generator also becomes worse. The second problem is that weight clipping can easily lead to gradient vanishing or gradient exploding. If the clipping threshold is set a little bit smaller, the gradient will decrease gradually with each layer of the network and will decline exponentially after multiple layers. Otherwise, there will be a gradient exploding [39].

Gradient penalty [39] is an alternative approach to enforce the Lipschitz constraint in order to resolve the problem of gradient vanishing and gradient exploding when training WGAN. Gradient penalty is to set an additional loss to satisfy the requirement between gradient and $k$, then this additional loss is added with weight to the original loss of WGAN. The objective of WGAN-GP is:

$$L = \mathop{\mathrm{E}}_{\tilde{x}\sim \mathrm{P}_g}[D(\tilde{x})] - \mathop{\mathrm{E}}_{x\sim \mathrm{P}_r}[D(x)] + \lambda \mathop{\mathrm{E}}_{\hat{x}\sim \mathrm{P}_{\hat{x}}}\left[\left(\left\|\nabla_{\hat{x}} D(\hat{x})\right\|_2 - k\right)^2\right] \quad (9)$$

where $k$ is the Lipschitz constraint and $\lambda$ is the penalty factor. $\mathop{\mathrm{E}}_{\tilde{x}\sim \mathrm{P}_g}[D(\tilde{x})] - \mathop{\mathrm{E}}_{x\sim \mathrm{P}_r}[D(x)]$ is the original critic loss and $\lambda \mathop{\mathrm{E}}_{\hat{x}\sim \mathrm{P}_{\hat{x}}}[(\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1)^2]$ is the gradient penalty. WGAN-GP has faster convergence speed than DCGAN and provides a stable GAN training method. In this method, we choose DCGAN for ECG generating. In particular, we merely referred to DCGAN to design our own generator and discriminator rather than directly use the whole structure of DCGAN.

#### 2.3.1. SpectroGAN method

Since DCGAN is initially designed for generating images, we use short-term Fourier transform (STFT) over 1-D ECG signals to obtain a 2-D array as the input of DCGAN. In this way, we are allowed to train the GAN like training an image without modifying the structure of DCGAN into 1-D form. STFT converts a time-domain signal into a 2-D time-frequency representation and shows the frequency domain variation of the signal [17]. Mathematically, the STFT is defined as:

$$STFT\{f(t)\} = \int_R f(t)g(t-u)e^{-j\omega t}dt \quad (10)$$

where $f(t)$ is a long time signal and $g(t)$ is a window function. In order to accelerate the convergence of the network during training, we normalized the matrix before inputting it to the network. We trained our GAN with the gradient penalty method [39] since we found that loss failed to converge if we used the original loss function of DCGAN. After attaining a synthetic matrix generated by our SpectroGAN, we used the Griffin-Lim algorithm [40] to reconstruct the corresponding ECG signal from the generated matrix. The griffin-lim algorithm is able to estimate a signal from its synthetic matrix by minimizing the mean squared error between the matrix of the estimated signal and the synthetic matrix [40]. Simply, we call this method the SpectroGAN.

#### 2.3.2. WaveletGAN method

In the SpectroGAN method, it is vital to choose an appropriate window size during STFT since it is directly connected with obtaining a good resolution. With a narrow window, the frequency resolution is poor but the temporal resolution becomes satisfactory; with a wide window, it is the other way around. Thus, it is less likely to find an accurate trade-off in both time and frequency. Wavelet transform (WT) is another effective approach to analyze ECG signals in the time-frequency domain. A prominent advantage of WT is that it can address the problem of the unchanged window in STFT by automatically changing the window size depending on the frequencies it focuses on: narrowing the window at high frequencies and widening the window at low frequencies. Such a multiresolution analysis allows us to observe the characteristics of the signal concentrated on different scales [41]. The WT of a signal $f(t)$ is defined as:

$$T(a,b) = \frac{1}{a}\int_R f(t)\psi\left(\frac{t-b}{a}\right)dt \tag{11}$$

where $a$ is the scale factor, $b$ is the translation and $\psi$ represents wavelet. With a large $a$, we can have an overall view of the signal because of the expansion of the wavelet, and with a small $a$, we are shown with a localized and detailed view of the signal since the wavelet is shrunk in the time domain.

Among several wavelet transform techniques, we chose the stationary wavelet transform (SWT) [42] because SWT is time-invariant at each decomposition level, which means that the time resolution of each coefficient time series is consistent with that of the original signal segment [43]. In our study, we used SWT with $J$-levels over each data segment to obtain $2J$ ($J$ detail coefficients and $J$ coarse coefficients) time series of each segment. Particularly, when using SWT with $J$-levels, we need to make sure that the signal length is a multiple of $2^J$. Before we input these $2J$ coefficient time series into GAN, we performed normalization over each coefficient time series. Different from conventional operations that the $2J$ coefficient time series are organized into a whole to input into a GAN, we input each coefficient time series into a GAN separately and then train the $2J$ GANs to generate corresponding coefficient time series. This is because we found that training a GAN using all $2J$ coefficient time series as an input was far more difficult than the above-mentioned method due to the more complex the input, which leads to the more complicated the network and then harder training process. Moreover, the GAN with all $2J$ coefficient time series as an input forced us to use 1-D convolution or long short-term memory (LSTM) [44], which can also add difficulty to train the GAN. By contrast, with our proposed method, we can train the GAN of each coefficient time series with only a few layers of a linear transformation. In this way, the training speed has been greatly improved and the difficulty of adjusting parameters has been greatly reduced, especially when the length of the input time series becomes longer.

After finishing training $2J$ GANs, we applied inverse stationary wavelet transform over the $2J$ coefficient time series separately produced by $2J$ GANs to reconstruct the ECG data segment to obtain the synthetic ECG. This ECG-generating approach is called WaveletGAN. Specifically, different from SpectroGAN, WaveletGAN is no need to use the gradient penalty method since the original loss function of DCGAN can work perfectly well.

### 2.4. Evaluation

To evaluating a generative model, we need to test its performance in two aspects: the fidelity of generated data, and the diversity of the data. In other words, if the synthetic data are not genuine enough, the model does not perform well. And even if the data are realistic enough, we still have to confirm whether the data are various. If the model can merely produce limited types of data, it falls into the so-called mode collapse. At present, there are several indicators [45], such as Inception Score (IS) [46], Mode Score [47], Fréchet Inception Distance [48] and so on, for evaluating the performance of a GAN. However, most of them are not suitable to evaluate the generative model for ECG since they are all based on the Inception V3 model. Inception V3 model is trained on a huge ImageNet dataset, therefore, it cannot be used for small ECG datasets.

In our study, we adopted an approach calculating GAN-train and GAN-test scores, which is proposed by Shmelkov [49], to test the performance of our conditional GANs. Firstly, we designed a classification method to classify different types of heartbeats. We perform a multilevel one-dimensional wavelet analysis on each heartbeat with a length of 250 to obtain wavelet decomposition of the heartbeat at level n. Then we extract the coarse-scale approximation coefficients at level n to form a feature vector. Finally, we use SVM to classify different types of heartbeat based on the feature vector. For computing GAN-train score, SVM needs to be trained on fake data produced by our generative models and be tested on real data. GAN-train is capable of measuring the difference between the learned and the target distribution. If SVM can correctly classify real data, it shows that the generated data are similar to the real ones. In other words, GAN-train can be viewed as a recall measure, the higher the GAN-train score, the more diverse the synthetic data. GAN-test is trained on real data and tested on generated data. This measure is akin to precision, a high value suggesting that the distribution of generated data and the unknown distribution of real data is approximate.

## 3. Experiments and results

The Pytorch deep learning framework [50] is used for constructing our synthetic models.

### 3.1. Experiments on WaveNet

The optimizer we used in all the WaveNet related experiments is Adam with an initial learning rate of 0.0001. When loss converges, the learning rate will decrease by a factor of 0.1. In order to determine the best parameters of the WaveNet-based method for synthesizing ECG signals, we conducted more than 30 experiments. Table 1 only shows 15 representative experiments. For estimating the basic parameters to generate complete ECG signals, we first applied the WaveNet to training set 3, each sample of which has a length of 250 and includes 5 R-peak data segments, as shown in Fig. 3. Training set 1 of complete ECG signals containing three heartbeat types N, L and R have higher complexity and variation in the signal waveform than training set 3, therefore, before processing complex data set, relatively simple data set should be processed first, which is helpful for later parameter setting.

Initially, we thought that samples of the training set 3 had relatively simple waveform and strong regularity, thus, we set the

**Table 1**
The comparisons of different parameter settings and data sets in WaveNet.

| # | Sr (Hz) | Data set | Length | Heartbeat type | Dilation | $\mu$ | Residue | Skip | Dilation depth | Blocks | Generated |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 360 | Training set 3 | 250 | N | No | 64 | 32 | 256 | 5 | 7 | Yes |
| 2 | 360 | Training set 3 | 300 | N | Yes | 64 | 32 | 256 | 8 | 1 | Yes |
| 3 | 3600 | Training set 3 | 2500 | N | Yes | 32 | 16 | 64 | 10 | 2 | No |
| 4 | 3600 | Training set 3 | 2500 | N | Yes | 64 | 32 | 256 | 10 | 2 | Yes |
| 5 | 360 | Training set 3 | 150 | N | No | 64 | 32 | 256 | 5 | 4 | Yes |
| 6 | 1800 | Training set 3 | 1250 | N | Yes | 64 | 32 | 256 | 10 | 1 | Yes |
| 7 | 360 | Training set 3 | 150 | N | Yes | 64 | 32 | 256 | 7 | 1 | No |
| 8 | 360 | Training set 1 | 1440 | N | Yes | 64 | 32 | 256 | 10 | 1 | No |
| 9 | 360 | Training set 1 | 1440 | N, L, R | Yes | 64 | 32 | 256 | 10 | 1 | No |
| 10 | 360 | Training set 1 | 1440 | N | Yes | 256 | 128 | 512 | 10 | 1 | No |
| 11 | 360 | Training set 1 | 1440 | N | Yes | 64 | 64 | 512 | 10 | 1 | Yes |
| 12 | 360 | Training set 1 | 1440 | N | Yes | 64 | 64 | 1024 | 10 | 1 | Yes |
| 13 | 360 | Training set 1 | 1440 | N | Yes | 32 | 32 | 1024 | 10 | 1 | Yes |
| 14 | 360 | Training set 1 | 1440 | N, L, R | Yes | 64 | 64 | 1024 | 10 | 1 | Yes |
| 15 | 360 | Training set 1 | 1440 | N, L, R | Yes | 32 | 32 | 1024 | 10 | 1 | Yes |

Sr = Sampling rate.
'Length' means the number of sampling points in one sample. '$\mu$'', 'Residue', 'Skip', 'Dilation depth', and 'Blocks' are parameters of WaveNet. 'Generated' stands for whether or not the network can generate ECG signals under its corresponding parameter setting.

parameters $\mu$, *residue* and *skip* very small. Through experiments, however, we found that we can generate nothing in such a parameter setting. Then, we tried something larger, for example, $\mu$=64, *residue*=32, and *skip*=256, as shown in Table 1. Notably, in Table 1, "No" in the column of "Dilation" means the parameter dilation=$2 \times (i+1)$ not dilation=0 in every layer, where $i$ is the number of convolutional layers in WaveNet. The reason why we didn't set dilation=0 was that under the circumstances that dilation was 0 and the length of each sample of the training set 3 was 250, we needed an enormous number of convolutional layers to synthesize data, which would undoubtedly increase the amount of computation and training time. Hence, we use dilation=$2 \times (i+1)$ to reduce the impact of dilation=$2^i$ on the WaveNet generation results to attest the effect of dilation.

(1) In Table 1, according to experiment 1 and 2, we found that for shorter data whose length was less than 500, WaveNet can generate data no matter with or without dilation, but the data generated by experiment 2 was much longer than that of experiment 1. This shows that dilation is indeed beneficial for a signal generation as said in the paper [33] and greatly decrease the number of layers needed by WaveNet, resulting in less training time and computation. In particular, in the case of the length of input data being close to that of the receptive field of WaveNet and being at least one more R-peak segment (whose length is 50) larger than that of the receptive field, we found the generation results of WaveNet would be better. Since the receptive field in experiment 2 was $2^8$=256, if we just use a sample with length 250, we cannot satisfy the above-mentioned finding. Therefore, we used samples with length 300 in our experiment 2 rather than 250.
(2) In order to test generation results of signals with different sampling rates, we up-sampled over training set 3, changing its sampling rate from 360 Hz to 1800 Hz and 3600 Hz, correspondingly, the length of input data changing from 250 to 1250 and 2500. In the light of experiment 4 and 6, WaveNet can also generate signals for upsampling data. Comparing with experiments 1 and 2 which can synthesize very long signals containing more than 20 R-wave peaks, experiments 4 and 6 can merely generate signals with an extremely short length which were composed of 1 to 2 R-wave peaks. We speculated that in terms of high sampling rate data in experiment 4 and 6, if we want to generate longer signals, we might need to input much longer data during training, for instance, data containing more than 10 R-wave peaks instead of the current 5 R-wave peaks, which may contribute to WaveNet better learning the intrinsic characteristics of signal waveforms. Of course, increasing the training time of experiment 4 and 6 might also be of help.
(3) Experiments 3 and 4 show that WaveNet with $\mu$=32, *residue*=16 and *skip*=64 is not suitable for generating R-wave peaks data.
(4) Although data can be synthesized in experiment 5, the R-wave peaks of its generated data were highly dispersed in the comparison of that of experiment 1. We concluded that when input data had fewer R-wave peaks, it is difficult for WaveNet to generate longer realistic signals since such input is harmful to WaveNet to capture the changing pattern of signals in the waveform.
(5) Comparing experiment 7 with experiment 5, we figured that WaveNet with dilation might not be fit for producing data with very short input (less than or equal to 150), which could also prove the finding mentioned above that WaveNet with very short input data has difficulty in producing longer signals.

Fig. 4(a) shows one representative generating result (experiment 1) of the training set 3 by WaveNet, which looks quite good. The seed in Fig. 4, in blue, is the real R-wave peak data segment. Basically, WaveNet can learn the characteristics of each single R-wave and the features of the motion of the data segment as a whole. However, it was noted that the produced R-wave peak data segment is not smooth, but rather a bit rectangular. This is mainly because our quantization parameter $\mu$ is relatively small, which is 64, the original sampling points with similar amplitude are all the same after quantization. If we make $\mu$ bigger the jagged effect will be reduced. However, the network parameters will increase exponentially due to the large size of $\mu$, and our experimental equipment cannot provide such a large computing resource at present. We up-sampled 10 times all the data in training set 3, each data segment containing 2500 sampling points, and then input them into WaveNet for training in experiment 4. The synthetic results are shown in Fig. 4 (b). It can be found that the smoother signal can be generated after up-sampling, but meanwhile, the training time will be greatly prolonged. Moreover, it can only generate relatively short data, including 2 or 3 R-wave peaks, compared with Fig. 4 (a), which contains more than 20 R-wave peaks.

Starting from experiment 8, we applied WaveNet to training set 1 (with all heartbeat types (marked as N, L, R in Table 1) or only normal heartbeat type (marked as N in Table 1)) as shown in Fig. 2. Each data segment in the data set has a length of 4s,

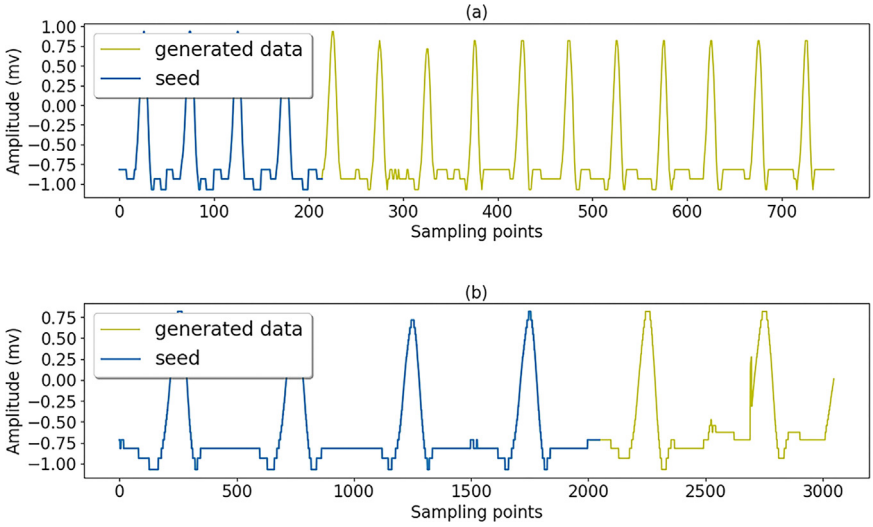


**Fig. 4.** Samples of training set 3 generated by WaveNet:(a) sampling rate=360Hz, the result of experiment 1; (b) sampling rate=3600Hz, the result of experiment 4.

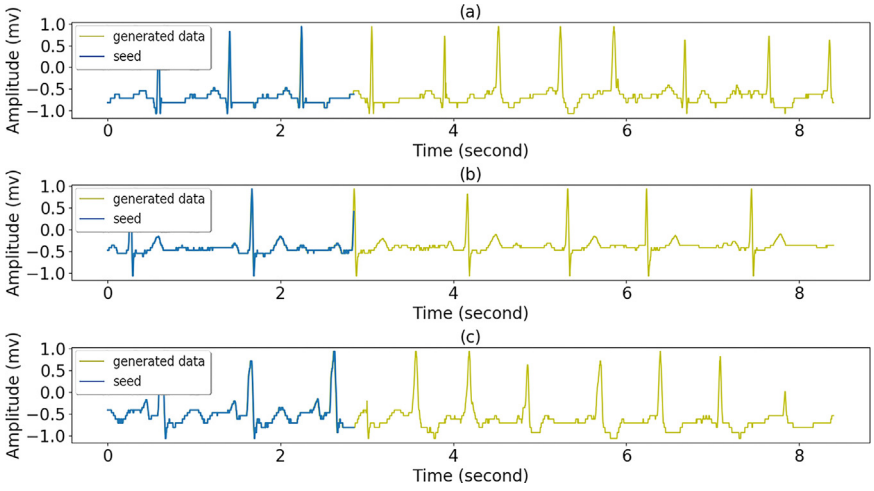


**Fig. 5.** Samples of the training set 1 generated by WaveNet: (a) N-type sample, (b) L-type sample, (c) R-type sample. All of them are the results of experiment 14.

1440 sampling points. The reason why we used 4s data segments not 3s or 5s was that through previous experiments on training set 3, we thought that 3s was so short that it was not conducive to obtaining good generation results, and when parameter dilation depth is 10 and parameter blocks is 1, meaning that the receptive field of WaveNet was 1024, in such a case, input data of 5s (1800 sampling points) was much longer than the receptive field, leading to the poor performance of WaveNet.

(1) Due to the failure of producing ECG signals in experiments 8 and 9, we concluded that $\mu$=64, *residue*=32, and *skip*=256 are not suitable for the generation of complete ECG signals, though they can work well for training set 3.
(2) At first, we thought that larger values for parameters were needed owing to the high complexity of the complete ECG signal in the waveform. However, from the results of experiment 10, we can see that our conjecture was wrong.
(3) Then with experiment 11 and 12, although the synthetic ECG signals were not as perfect as we thought, which might because there were not enough iterations, we could at least make sure that WaveNet with such parameter settings is capable of producing ECG signals and might have the potential to generate a quite good one. On the basis of our observation, the synthetic results of experiment 12 are superior to that of experiment 11, and the results of experiment 13 are better than that of experiment 12. Therefore, we had another finding that large differential between the value of $\mu$ and *skip* was conducive to producing complete ECG signals with relatively good quality.
(4) Finally, due to the generating results of experiment 15 are slightly inferior to that of experiment 14, we decided to use the parameter setting of experiment 14 to generate ECG signals. Specifically, although the parameter settings of experiment 13 and experiment 15 were the same, the effect of 15 on the training set 1 was not as good as that of experiment 14 because we thought that the data in training set 1 were more diverse and complex, and the range of amplitude varies more widely, so it is more advantageous to adopt larger parameter settings to avoid the sampling points that have similar amplitude being quantified into the same value.

As can be seen from Fig. 5, WaveNet can perfectly generate ECG data with three different types of heartbeats. The generated single heartbeat accords with the characteristics of the corresponding types of real heartbeats, and the overall waveform fluctuation also has a natural transition.

However, one of the biggest problems of WaveNet is that synthesis is seriously time-consuming. If we want to produce an

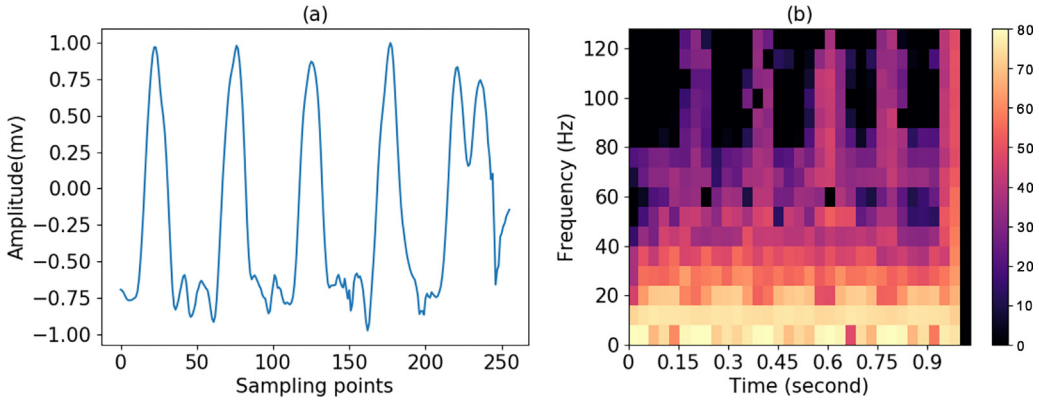


**Fig. 6.** One sample of training set 3 generated by SpectroGAN: (a) signal, (b) spectrogram.

**Table 2**
The detailed architecture of SpectroGAN.

| Generator | Kernel | Stride | Pad | Output shape |
|---|---|---|---|---|
| Latent vector | - | - | - | 100 × 1 × 1 |
| ConvTrans2d IN and Relu | 2 × 4 | 1 | 0 | 256 × 2 × 4 |
| ConvTransd IN and Relu | 2 × 4 | 2 | 0 × 1 | 256 × 4 × 8 |
| ConvTrans2d IN and Relu | 2 × 4 | 2 | 0 × 1 | 128 × 8 × 16 |
| ConvTrans2d Sigmoid | 3 × 5 | 2 | 0 × 1 | 1x 17 × 33 |
| **Discriminator** | **Kernel** | **Stride** | **Pad** | **Output shape** |
| Input data | - | - | - | 1 × 17 × 33 |
| Conv2d LRelu | 3 × 5 | 2 | 0 × 1 | 32 × 8 × 16 |
| Conv2d IN and LRelu | 2 × 4 | 2 | 0 × 1 | 128 × 4 × 8 |
| Conv2d IN and LRelu | 2 × 4 | 2 | 0 × 1 | 256 × 2 × 4 |
| Conv2d Sigmoid | 2 × 4 | 2 | 0 | 1 × 1 × 1 |

IN: Instance normalization, LRelu: LeakyRelu.

ECG signal in high quality, we need to iterate a set of parameters tens of thousands or even hundreds of thousands of times, costing several months with the help of GPU cluster. Because of the limited computing power of our computer (2 CPUs at 2.1 GHz, 1 NVIDIA Tesla K40c GPU, and 32-Gb memory), it is difficult for us to iterate thousands of times during training to verify whether a set of parameters is able to generate the ECG signals, and to simultaneously conduct a great deal of control experiments to determine the optimal set of parameters. Moreover, it is difficult for WaveNet to parallel owing to the structure of its networks, which is another reason for long training time. Therefore, in all of our experiments related to the WaveNet-based method, each experiment, that is, a set of parameters, has approximately 1,000 iterations, taking roughly half a month, and ultimate loss is about 0.1–0.3. Even though the synthetic results of some experiments may not as good as expected, at least we can see from the existing experiments that WaveNet-based method can certainly generate ECG signals with relatively good quality.

### 3.2. Experiments on SpectroGAN

For the same purpose of the WaveNet related experiments, we also used GAN on training set 3. At first, we applied STFT over the data set to obtain a matrix of each data segment and normalized the absolute value of each matrix to [0, 1]. Then, GAN accepted the normalized matrix whose size is 16 × 32 as input. Actually, since *nfft* is set to 32 and hop length is set to 8, the size of the matrix after STFT is 17 × 33. During training, we cropped the size of the matrix to 16 × 32, and when using the Griffin-Lim algorithm, we expanded the matrix by padding zeros to the size of 17 × 33. In the end, the signal was reconstructed in a 100 iterations of Griffin-Lim phase reconstruction algorithm. The details of our network settings are shown in Table 2. We used RMSprop with initial learning rate 0.0001.

In general, the last layer of the generator in DCGAN is tanh function, which means that the output matrix needs to be normalized to [−1, 1]. In our experiments, however, the network can hardly converge when the last layer is tanh function. Hence, we take the place of tanh function with a sigmoid function. Besides, according to WGAN-GP, the sigmoid function of the last layer of the discriminator is removed, whereas in our study, the loss of the network decreases only when we keep sigmoid function. In addition, we chose instance normalization rather than batch normalization in our study. Batch normalization is to normalize all the data in the whole batch, whereas instance normalization is to normalize a single data in a batch. For data generation, the overall information obtained by batch normalization will not bring any benefits but will bring about the noise that will weaken the independence of instances. Thus, it is better to use instance normalization.

Fig. 6 shows that although some local regions are not well generated, the R-wave peak data segment restored by the Griffin-Lim phase reconstruction algorithm is good. After this, we conducted our experiments on the training set 2. The network structure of SpectroGAN and the values of parameters are identical to Table 2, except that we added a condition to SpectroGAN and modified learning rate to 0.00015 and iterations of Griffin-Lim algorithm to 500. The condition was the one-hot encoding of the type of heartbeat: N, L, and R. Fig. 7 shows the generated heartbeats of three types and their corresponding spectrograms. As can be seen from spectrogram in Figs. 6 and 7, energy is concentrated in the low-frequency part of 0–40 Hz, while the high-frequency component greater than 40 Hz is lower in energy. This is also consistent with the energy distribution of the frequency of the general ECG signal. And since the spectrogram is obtained by STFT, we can see how the peak changes over time, similar to how it changes in the time domain.

To obtain a long time ECG signal, here we first generate a single heartbeat, and then we construct an ECG signal by replicating the single heartbeat, as presented in Fig. 8. This is because it is very difficult for SpectroGAN to directly generate an STFT 2-D matrix of a long signal with high quality. The longer the signal, the larger the size of STFT of the signal, and the higher quality of STFT the method needs to reconstruct the signal by the Griffin-Lim algorithm. At present, GAN's generating ability is not very strong. As a result, it is less likely to generate large-high-quality pictures,

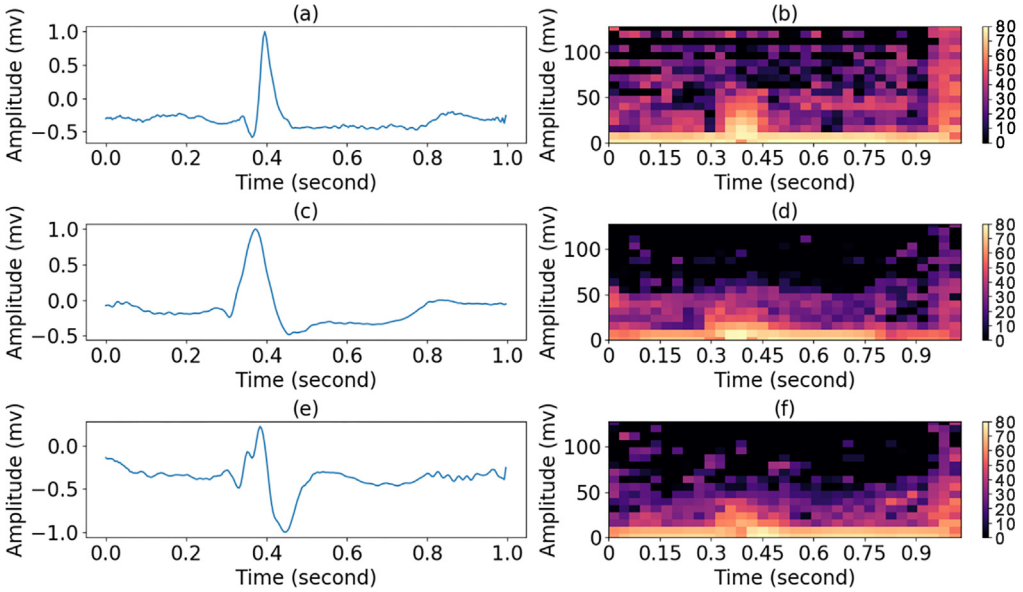


**Fig. 7.** Samples of the training set 2 generated by SpectroGAN: (a) heartbeat of N-type, (b) spectrogram of the N-type heartbeat, (c) heartbeat of L-type, (d) spectrogram of the L-type heartbeat, (e) heartbeat of R-type, (f) spectrogram of the R-type heartbeat.

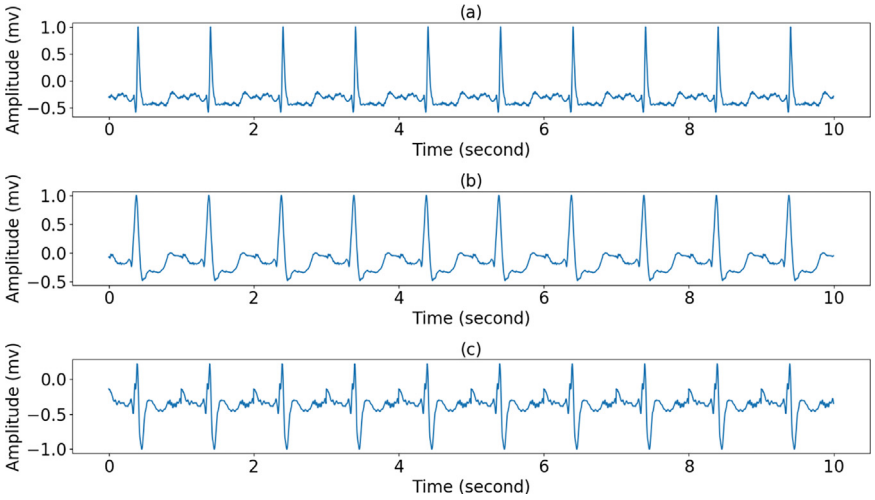


**Fig. 8.** Signals generated by SpectroGAN: (a) signal of N-type heartbeat, (b) signal of L-type heartbeat, (c) signal of R-type heartbeat.

that is, the STFT 2-D matrix in our study. Thus, we have difficulty in directly reconstructing long ECG signals. Moreover, the network structure has to be modified a lot and the parameters have to be readjustment if we want to generate signals of different lengths, which is a time-consuming process. Therefore, we replicated the generated single heartbeat to build a long ECG signal, which not only avoids modifying the network structure and adjusting parameters for generating signals of different lengths but also ensures the quality of the generated single heartbeat and the whole signal. Of course, one can also construct a long ECG signal by combining multiple different heartbeats of one kind or mixed kinds according to their research purposes. Here for simplicity, we only give the results of a single heartbeat construction. It is worth noting that although many sampling points have been removed from the previous interceptive operation in Section II and the ECG signal obtained here by repeating a single heartbeat is not really a complete signal, the omitted data are actually a smooth transition part, which does not contain any important information, so this will not affect the use of the ECG in signal processing algorithms and diagnostic algorithms.

**Table 3**
The detailed architecture of WaveletGAN.

| Generator | Output shape | Discriminator | Output shape |
|---|---|---|---|
| Latent vector | 100 × 1 | Input data | 256 × 1 |
| Linear | 512 × 1 | Linear | 350 × 1 |
| Relu | | LeakyRelu | |
| Linear | 350 × 1 | Linear | 512 × 1 |
| Relu | | LeakyRelu | |
| Linear | 256 × 1 | Linear | 1 × 1 |
| Tanh | | Sigmoid | |

### 3.3. Experiments on WaveletGAN

In WaveletGAN, SWT with 4 levels was first used on the training set 3 to obtain 8 coefficient time series of each sample of the data set. Each coefficient time series was normalized to [−1, 1] and then was input into the same network. Daubechies 2 wavelet is chosen as the mother wavelet to implement the wavelet transform. Our network settings are shown in Table 3.

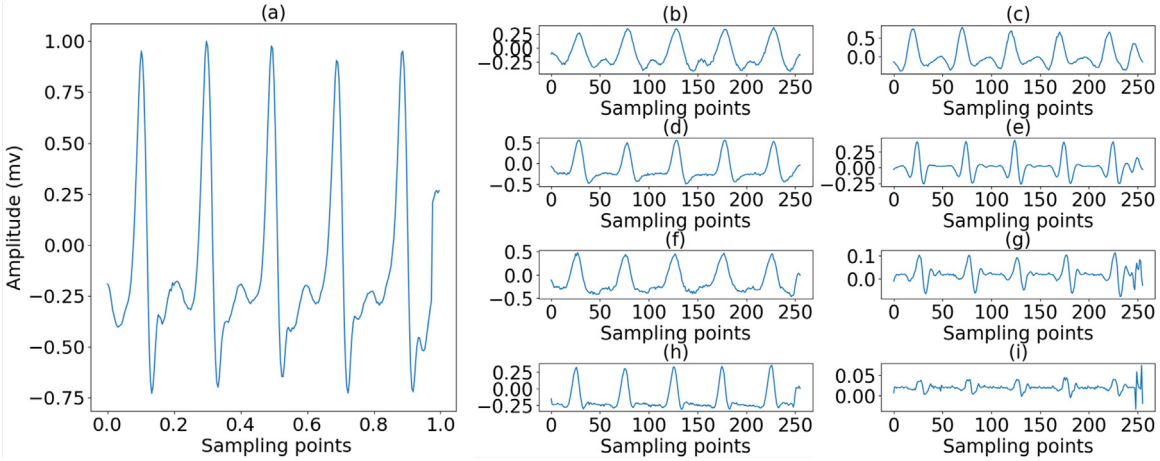


**Fig. 9.** Sample of the training set 3 generated by WaveletGAN: (a) R-peak segment, (b) C1, (c) D1, (d) C2, (e) D2, (f) C3, (g) D3, (h) C4, (i) D4, C=coarse coefficients, D= detailed coefficients.

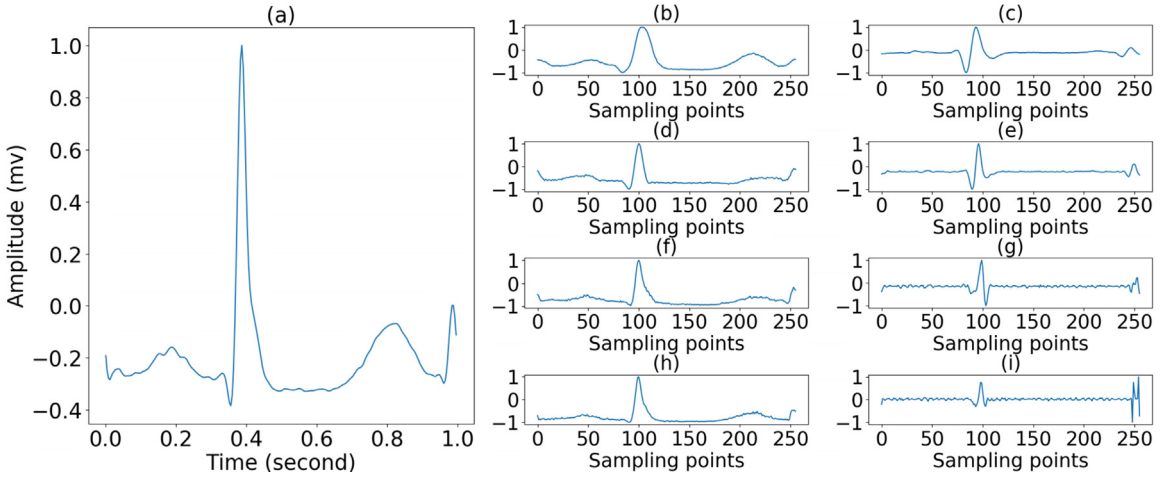


**Fig. 10.** N-type sample of training set 2 generated by WaveletGAN: (a) heartbeat, (b) C1, (c) D1, (d) C2, (e) D2, (f) C3, (g) D3, (h) C4, (i) D4 C=coarse coefficients, D= detailed coefficients.

We used RMSprop with an initial learning rate of 0.00015. Fig. 9 presents the synthetic coefficient time series and the signal reconstructed by these time series. We can see that the 8 coefficient time series generated by WaveletGAN are very realistic, and the reconstructed R-wave peak data segment is also very good.

Then, experiments are conducted on the training set 2, and the results are shown in Figs. 10–12. All parameter settings are the same as Table 3, except that we added a condition to WaveletGAN and modified the learning rate to 0.00006. Although there is some tiny noise in the generated heartbeats, the overall effect is good, and the characteristics of each type of heartbeat are similar to the real ones.

Fig. 13 presents the synthetic signals constructed by replicating the single heartbeat of Figs. 10–12. The duration time the ECG signals is about 10 s. The overall signals shown in Fig. 13 seem to be very similar to real ECG signals except for lacking some transition part.

### 3.4. Evaluation

In the evaluation section, we used a db6 wavelet to decompose each heartbeat at 4 levels. Since the length of each heartbeat was 250, we obtained approximation coefficients at level 4 with a length of 25 and used this coefficient as a feature vector. Before the feature vector is input into SVM, it needs to be normalized into [0,1]. We used the feature vector extracted from data in training set 2 to train SVM, and used the feature vector extracted from data in the test set 2 to test in order to obtain the benchmark of classification accuracy.

For computing the GAN-test score of SpectroGAN, we used this trained SVM to test 500 heartbeats of each type produced by SpectroGAN. The calculation of GAN-test score for WaveletGAN and WaveNet follows the same operation. As for GAN-train score for SpectroGAN, we extracted feature vector from 15,000 heartbeats with each type of 5000 generated by SpectroGAN to train SVM and then used the test set 2 to test the performance of the SVM. We did the same operation for heartbeats produced by WaveletGAN to calculate GAN-train score for WaveletGAN. Nevertheless, here it is difficult for us to give the exact GAN-train score for WaveNet because WaveNet requires to input real data as a seed to generate fake data, which means that if we want to produce 5,000 heartbeats of L-type, we at least need 500 real data to generate fake data since we can obtain 10 heartbeats depending on one seed. However, under the circumstances that the severe lack of data of L-type and R type that we mentioned in Section II and we have to make sure the variety of the training set 1 in order to train a good WaveNet, it is hard for us to extract enough real data from training set 1 to serve as seeds to generate enough fake data for the purpose of training SVM. Thus, the GAN-train

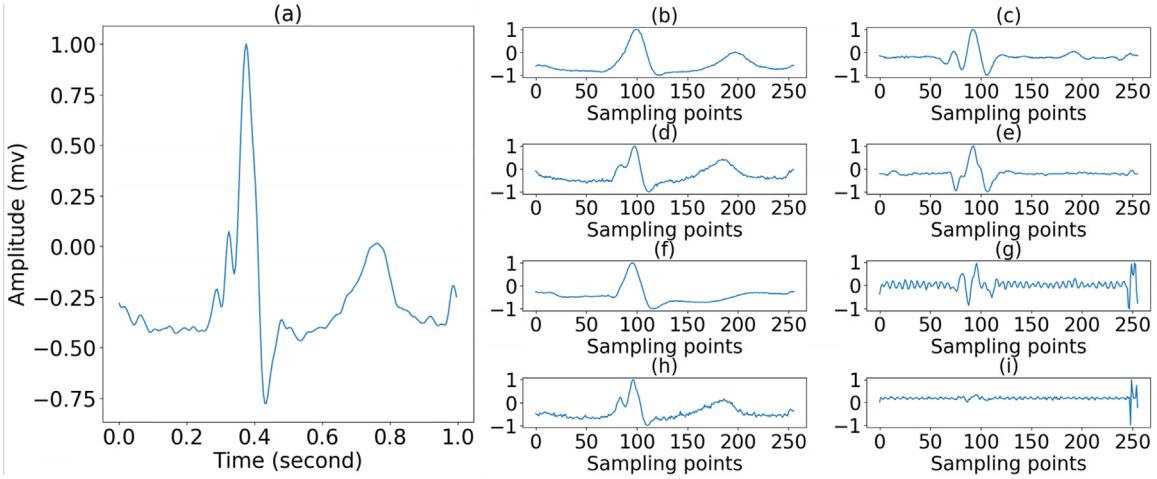


**Fig. 11.** L-type sample of training set 2 generated by WaveletGAN: (a) heartbeat, (b) C1, (c) D1, (d) C2, (e) D2, (f) C3, (g) D3, (h) C4, (i) D4.

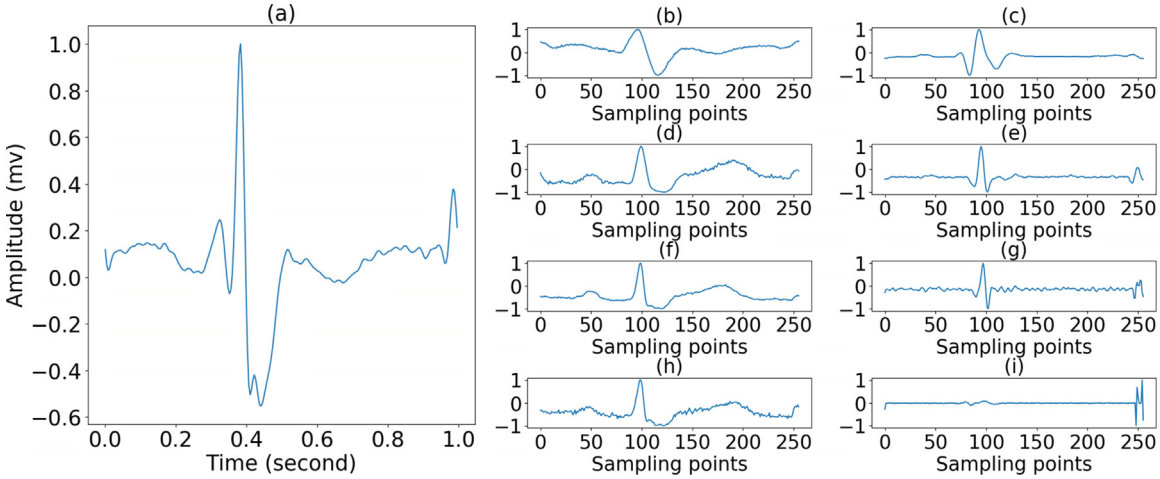


**Fig. 12.** R-type sample of training set 2 generated by WaveletGAN: (a) heartbeat, (b) C1, (c) D1, (d) C2, (e) D2, (f) C3, (g) D3, (h) C4, (i) D4.

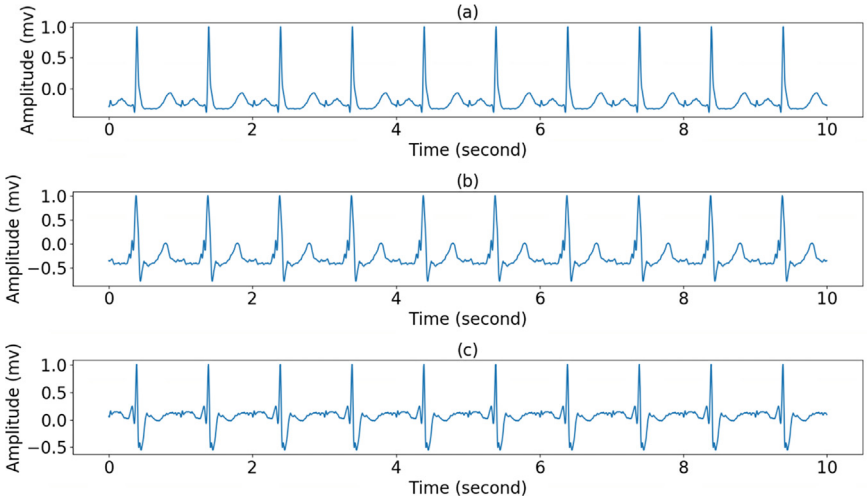


**Fig. 13.** Signals generated by WaveletGAN: (a) signal of N-type heartbeat, (b) signal of the L-type heartbeat, (c) signal of R-type heartbeat.

score for WaveNet in Table 4 was calculated by using only 500 heartbeats as a training set for SVM.

The results are shown in Table 4, we can see that the GAN-train score of WaveNet and SpectroGAN are nearly 35% and 30% respectively lower the benchmark, which manifests that the generated data have less variety and reality. By contrast, the GAN-train score of WaveletGAN is almost 20% higher than that of SpectroGAN and only 10% lower than the benchmark, from which we can speculate that WaveletGAN is superior to SpectroGAN in the diversity of synthetic data. In the comparison of the GAN-test score of WaveNet,

**Table 4**
The comparison of results between different generative models proposed by us.

| Model | GAN-train (%) | GAN-test (%) |
|---|---|---|
| Real data | 99.93 | 99.93 |
| WaveNet | 54.67 | 73.40 |
| SpectroGAN | 68.80 | 99.96 |
| WaveletGAN | 89.07 | 92.33 |

GAN-train score of real data means using real data to train and test.

SpectroGAN, and WaveletGAN, it is obvious that the score of the latter two is much higher than the first one, indicating that the distribution of the generated data of SpectroGAN and WaveletGAN is approximate to that of real data.

## 4. Discussion

This is the first study successfully generate ECG signals using deep learning. We proposed three generative models to produce realistic ECG signals, each of which is able to generate three different types of heartbeats. As we can see in Table 5, compared with traditional algorithms that need the expertise to prescribe specific parameters to control characteristics of ECG signals, which may introduce uncertainty and inaccuracy, our models are capable of discovering some authentic characteristics of real ECG signals by deep learning, contributing to the generated data getting closer to real data in waveform and other features. These methods can also provide much convenience to operators since they don't have to spend a great amount of time in seeking suitable parameters for generation. Besides, most traditional algorithms are dependent on 3-D trajectory model, which requires the design of very complex mathematical formulas to model a single heartbeat of a single lead ECG recording. Such a complex modeling process not only brings a great deal of work to the researchers but also greatly limits the performance of model extendibility in generating multi-lead ECG signals. Because they may need to design models higher than 3-D and introduce more parameters to better serve the purpose of multi-lead signal generation [6]. However, our proposed methods do not need to design complicated formulas for the generative model but only need to design the neural network architecture. The proposed methods can also easily achieve multi-lead signal generation via feeding data from other different leads into the currently used single-lead signal generation model nearly without modifying any parameter setting.

One advantage of traditional algorithms is that they can provide local control. For example, they could adjust the position of P-wave by modifying some parameters in their models. Our proposed methods now do not have this characteristic. The main reason is that we don't think this kind of local control really makes much sense, especially since we've been able to generate many types of heartbeats. In traditional algorithms, local control can be used to enrich the morphological diversity of a certain type of heartbeat or to generate an abnormal heartbeat. However, in our proposed methods, the morphological diversity of a certain type of heartbeat can be realized by feeding random seeds to the network. The generation of different specific types of heartbeats is exactly one of the major contributions of our proposed methods. That is, the deep neural network designed by us can directly generate a specific abnormal heartbeat such as left bundle branch block beat (L) or right bundle branch block beat (R). However, the traditional methods can only generate heartbeats that are different from normal heartbeats by adjusting some parameters of the 3-D trajectory model and call them abnormal heartbeats. In other words, traditional methods can't easily generate a particular type of abnormal heartbeat. The specific type of heartbeat produced by our methods actually eliminates the heavy process of heartbeat labeling and provides great convenience for heartbeat classification algorithms.

In addition, traditional algorithms use a model of the heart rate dynamics to produce a long time ECG signal, in the meantime, they also need a first-order Markov model and a state transition matrix (STM) to switch between normal and abnormal beat dynamics. The dynamic heart rate model makes all R-R intervals of a generated ECG signal the same, while in fact the R-R intervals are different in a certain ECG signal. Moreover, the first-order Markov model means that the state at time T is only related to the state at time T-1 and has nothing to do with the state before or after. However, cognitively, for the time series, the state at time T is related to the state at all previous moments. Therefore, the first-order Markov model is actually not suitable. In traditional algorithms, they have to use the higher-order Markov model to realize the long-term dependency, which undoubtedly increases the complexity of calculation. Nevertheless, our WaveNet model can produce long time ECG signal with natural transition and different R-R intervals within a generated ECG signal easily and directly, as shown in Fig. 5.

Specifically, the WaveNet model can produce continuous ECG signals rather than obtain long ECG signals by replicating one heartbeat, which means it can generate ECG signals with verisimilar transition information since every R-R interval of one ECG signal may be different. However, data generated by WaveNet are a bit rectangular. In contrast, the overall data synthesized by SpectroGAN and WaveletGAN are relatively smoother. Besides, compared with WaveNet, SpectroGAN and WaveletGAN can synthesize data by using noise coming from normal distribution as seed, which is able to significantly increase the diversity and abundance of synthetic data and is more advantageous to generate large-scale data. Nevertheless, a minor drawback of SpectroGAN and WaveletGAN is that they cannot generate data of arbitrary length like WaveNet. They can only generate new data of the same length as those in their training set. Besides, our proposed evaluation method is able to help researchers quantitatively analyze and judge the advantages and disadvantages of the generation algorithms. However, traditional algorithms do not provide a general and quantitative evaluation indicator to explain the quality of the generated data, that is, the degree to which the distribution of generated data conforms to that of real data.

Finally, there are still some problems with our method worth discussing. By examining the data synthesized by SpectroGAN, we found a fact that the diversity within each category of the gen-

**Table 5**
The comparison of performance of ECG generating algorithms.

| Method | Expertise in parameter setting | Modeling complexity | Local control | Specific abnormal beats | Temporal continuity | Multi-leads extendibility | Evaluation indicator |
|---|---|---|---|---|---|---|---|
| McSharry et al. [4] | Yes | High | Yes | No | No | Low | No |
| Li and Ma [5] | Yes | High | Yes | No | No | Low | No |
| Sameni et al. [6] | Yes | High | Yes | No | No | High | No |
| Clifford et al. [7] | Yes | High | Yes | No | No | High | No |
| Roonizi and Sameni [8] | Yes | High | Yes | No | No | Low | No |
| Our proposed methods | No | Low | No | Yes | Yes | High | Yes |

erated data is indeed insufficient. For example, in our training set 2, there are two different patterns of L-type heartbeat, whereas in data set generated by SpectroGAN there is only one form. We have tried to train generator once every five times of training discriminator in order to improve the ability of discriminator to avoid the generated data within one class converging into one similar form. Nonetheless, this strategy is basically useless for our experiments.

Moreover, from the results of evaluation in Table 4, it indicates that in our experiments and settings, WaveNet is less capable of producing data in high quality and capturing the target distribution. Considering the excellent performance of WaveNet in generating speech and music, we think that the following three reasons may account for its unsatisfactory results:

- Too few samples of the training set. In the task of speech generation, the number of training data is usually tens of thousands or even hundreds of thousands. In contrast, we only have 5,000 data of each category and a part of them are obtained by data augmentation.
- Too short of the length of training samples. It is not conducive to generate data with long-term dependence if the duration time of training data is short since this may result in that the data generated later is of poor quality. In our experiments, after weighing the sample length and the number of samples, the maximum length we can give is 4 seconds, which is far less than dozens of seconds of data used in speech generation.
- Too small of the value of $\mu$-law companding. From Fig 4, we can see that the generated data is distinctly rectangular instead of smooth. Due to the limitation of the experimental environment and resources, we can only set 64 as the maximum value of $\mu$-law companding. However, if using 256, the synthetic effect could be better because sampling points with similar amplitudes can be encoded into different values without being grouped into the same value.

Admitted to these limitations described above, we have demonstrated the feasibility and simplicity in ECG generation by deep learning and achieved relatively good synthetic effect.

## 5. Conclusion

Our proposed models greatly improve the generating results and performance compared with traditional methods. It is believed that our generative models can contribute to the development of ECG signal processing techniques or ECG classification algorithms, then finally help to achieve better results for clinical applications.

## Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## CRediT authorship contribution statement

**Naren Wulan:** Conceptualization, Investigation, Methodology, Software, Data curation, Visualization, Writing - original draft, Writing - review & editing. **Wei Wang:** Visualization, Writing - original draft, Writing - review & editing, Supervision. **Pengzhong Sun:** Software, Data curation, Visualization. **Kuanquan Wang:** Supervision, Writing - review & editing. **Yong Xia:** Supervision. **Henggui Zhang:** Supervision.

## Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grant 61571165.

## References

[1] S. Ladavich, B. Ghoraani, Rate-independent detection of atrial fibrillation by statistical modeling of atrial activity, Biomed. Signal Process. Control 18 (2015) 274–281.
[2] S. Asgari, A. Mehrnia, M. Moussavi, Automatic detection of atrial fibrillation using stationary wavelet transform and support vector machine, Comput. Biol. Med. 60 (2015) 132–142.
[3] A.L. Goldberger, L.A. Amaral, L. Glass, J.M. Hausdorff, P.C. Ivanov, R.G. Mark, J.E. Mietus, G.B. Moody, C.-K. Peng, H.E. Stanley, PhysioBank, PhysioToolkit, and PhysioNet: components of a new research resource for complex physiologic signals, Circulation 101 (23) (2000) e215–e220.
[4] P.E. McSharry, G.D. Clifford, L. Tarassenko, L.A. Smith, A dynamical model for generating synthetic electrocardiogram signals, IEEE Trans. Biomed. Eng. 50 (3) (2003) 289–294.
[5] Z. Li, M. Ma, ECG modeling with DFG, in: Proceedings of the 27th IEEE Annual Conference Engineering in Medicine and Biology, 2005, pp. 2691–2694.
[6] R. Sameni, G.D. Clifford, C. Jutten, M.B. Shamsollahi, Multichannel ECG and noise modeling: Application to maternal and fetal ECG signals, EURASIP J. Adv. Signal Process. 1 (2007) 043407.
[7] G.D. Clifford, S. Nemati, R. Sameni, An artificial vector model for generating abnormal electrocardiographic rhythms, Physiol. Meas. 31 (5) (2010) 595.
[8] E.K. Roonizi, R. Sameni, Morphological modeling of cardiac signals based on signal decomposition, Comput. Biol. Med. 43 (10) (2013) 1453–1461.
[9] A. Graves, A.-r. Mohamed, and G. Hinton, Speech Recognition with Deep Recurrent Neural Networks. pp. 6645-6649.
[10] O. M. Parkhi, A. Vedaldi, and A. Zisserman, Deep Face Recognition. p. 6.
[11] W. Ouyang, X. Wang, X. Zeng, S. Qiu, P. Luo, Y. Tian, H. Li, S. Yang, Z. Wang, and C.-C. Loy, Deepid-net: Deformable Deep Convolutional Neural Networks for Object Detection. pp. 2403-2412.
[12] Y. Xia, N. Wulan, K. Wang, H. Zhang, Detecting atrial fibrillation by deep convolutional neural networks, Comput. Biol. Med. 93 (2018) 84–92.
[13] S. Min, B. Lee, S. Yoon, Deep learning in bioinformatics, Brief. Bioinform. 18 (5) (2017) 851–869.
[14] M. Drozdzal, E. Vorontsov, G. Chartrand, S. Kadoury, C. Pal, The importance of skip connections in biomedical image segmentation, Deep Learning in Medical Image Analysis (DLMIA), Athens, Greece, 2016, pp. 179–187.
[15] L. Yang, Y. Zhang, J. Chen, S. Zhang, and D. Z. Chen, Suggestive Annotation: A Deep Active Learning Framework for Biomedical Image Segmentation. pp. 399-407.
[16] D. Shen, G. Wu, H.-I. Suk, Deep learning in medical image analysis, Annu. Rev. Biomed. Eng. 19 (2017) 221–248.
[17] N. Zeng, Z. Wang, H. Zhang, K.-E. Kim, Y. Li, X. Liu, An improved particle filter with a novel hybrid proposal distribution for quantitative analysis of gold immunochromatographic strips, IEEE Trans. Nanotechnol. 18 (2019) 819–829.
[18] N. Zeng, Z. Wang, H. Zhang, W. Liu, F.E. Alsaadi, Deep belief networks for quantitative analysis of a gold immunochromatographic strip, Cogn. Comput. 8 (4) (2016) 684–692.
[19] N. Zeng, Z. Wang, B. Zineddin, Y. Li, M. Du, L. Xiao, X. Liu, T. Young, Image-based quantitative analysis of gold immunochromatographic strip via cellular neural network approach, IEEE Trans. Med. Imaging 33 (5) (2014) 1129–1136.
[20] S. Panwar, P. Rad, T.-P. Jung, Y. Huang, Modeling EEG Data Distribution with a Wasserstein Generative Adversarial Network to Predict RSVP Events, 2019, arXiv preprint arXiv:1911.04379.
[21] Y. Xia, N. Wulan, K. Wang, H. Zhang, Atrial fibrillation detection using stationary wavelet transform and deep learning, Comput. Cardiol. 44 (2017) 1–4.
[22] P. Rajpurkar, A. Y. Hannun, M. Haghpanahi, C. Bourn, and A. Y. Ng, Cardiologist-level Arrhythmia Detection with Convolutional Neural Networks, 2017, arXiv preprint arXiv:1707.01836.
[23] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, Generative Adversarial Nets. pp. 2672-2680.
[24] A. Radford, L. Metz, and S. Chintala, Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks, 2015, arXiv preprint arXiv:1511.06434.
[25] D. Berthelot, T. Schumm, and L. Metz, Began: Boundary Equilibrium Generative Adversarial Networks, 2017, arXiv preprint arXiv:1703.10717.
[26] Z. Yi, H. Zhang, P. Tan, and M. Gong, Dualgan: Unsupervised Dual Learning for Image-To-Image Translation. pp. 2849-2857.
[27] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, Self-Attention Generative Adversarial Networks, 2018, arXiv preprint arXiv:1805.08318.
[28] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, and D. N. Metaxas, Stackgan: Text to Photo-Realistic Image Synthesis with Stacked Generative Adversarial Networks. pp. 5907-5915.
[29] W. R. Tan, C. S. Chan, H. E. Aguirre, and K. Tanaka, ArtGAN: Artwork synthesis with Conditional Categorical GANs. pp. 3760-3764.
[30] A. Elgammal, B. Liu, M. Elhoseiny, and M. Mazzone, Can: Creative Adversarial Networks, Generating Art by Learning About Styles and Deviating from Style Norms, 2017, arXiv preprint arXiv:1706.07068.
[31] L. Yu, W. Zhang, J. Wang, and Y. Yu, Seqgan: Sequence Generative Adversarial Nets With Policy Gradient."
[32] J. Li, W. Monroe, T. Shi, S. Jean, A. Ritter, and D. Jurafsky, Adversarial Learning for Neural Dialogue Generation, 2017, arXiv preprint arXiv:1701.06547.
[33] A. Van Den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A.W. Senior, K. Kavukcuoglu, WaveNet: A Generative Model for Raw Audio, 125, SSW, 2016.

[34] R. Mark, G. Moody, MIT-BIH Arrhythmia Database Directory, Massachusetts Institute of Technology, Cambridge, 1988.
[35] J. Pan, W.J. Tompkins, A real-time QRS detection algorithm, IEEE Trans. Biomed. Eng. 32 (3) (1985) 230–236.
[36] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, Aggregated Residual Transformations for Deep Neural Networks. pp. 1492-1500.
[37] C. Recommendation, Pulse Code Modulation (PCM) of Voice Frequencies, ITU, 1988.
[38] M. Arjovsky, S. Chintala, L. Bottou, Wasserstein gan, 2017, arXiv preprint arXiv:1701.07875.
[39] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, Improved Training of Wasserstein gans. pp. 5767-5777.
[40] D. Griffin, J. Lim, Signal estimation from modified short-time Fourier transform, IEEE Trans. Acoust. Speech Signal Process. 32 (2) (1984) 236–243.
[41] P. R. Gomes, F. O. Soares, J. Correia, and C. Lima, ECG Data-Acquisition and Classification System by Using Wavelet-Domain Hidden Markov Models. pp. 4670-4673.
[42] G.P. Nason, B.W. Silverman, The stationary wavelet transform and some statistical applications, Wavelets Stat. 103 (1995) 281–299.
[43] A. Salmanpour, L. J. Brown, and J. K. Shoemaker, Performance Analysis of Stationary and Discrete Wavelet Transform for Action Potential Detection from Sympathetic Nerve Recordings in Humans. pp. 2932-2935.
[44] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.
[45] Q. Xu, G. Huang, Y. Yuan, C. Guo, Y. Sun, F. Wu, and K. Weinberger, An Empirical Study on Evaluation Metrics of Generative Adversarial Networks, 2018, arXiv preprint arXiv:1806.07755.
[46] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, X. Chen, Improved techniques for training gans, Neural Information Processing Systems (NIPS), Barcelona, Spain, 2016, pp. 2234–2242.
[47] T. Che, Y. Li, A.P. Jacob, Y. Bengio, W. Li, Mode Regularized Generative Adversarial Networks, 2016, arXiv preprint arXiv:1612.02136.
[48] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, S. Hochreiter, Gans trained by a two time-scale update rule converge to a local Nash equilibrium, Neural Information Processing Systems (NIPS), Long Beach, CA, USA, 2017, pp. 6626–6637.
[49] K. Shmelkov, C. Schmid, and K. Alahari, How Good is my GAN?. pp. 213-229.
[50] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, PyTorch: an imperative style, high-performance deep learning library, Neural Information Processing Systems (NIPS), Vancouver, Canada, 2019, pp. 8024–8035.

**Naren Wulan** received her B.S and M.S degrees in computer science from Harbin Institute of Technology, China. Her research interests include digital signal processing, medical image processing and pattern recognition.

**Wei Wang** is a postdoc in the School of Computer Science and Technology, Harbin Institute of Technology, China. She received her Ph.D. degree at the University of Manchester, UK, and had her B.S. and M.S. degrees in computer science from Harbin Institute of Technology, China. Her research interests include computational modeling of cardiac electrophysiology, medical image processing and simulation studies of cardiac arrhythmia.

**Pengzhong Sun** is currently pursuing a Ph.D. degree in School of Computer Science and Technology, Harbin Institute of Technology, China. He received a B.S. degree in computer science and technology from Harbin Institute of Technology, China. His research interests include pattern recognition, medical image analysis, and computational pathology.

**Kuanquan Wang** (M'01–SM'07) is a full professor and Ph.D. supervisor with School of Computer Science and Technology, and the director of Research Center of Perception and Computing at Harbin Institute of Technology. Also, he was an associate dean of School of Computer Science and Technology, HIT at Harbin, and the dean of School of Computer Science and Technology, HIT at Weihai from 2011 to 2014. He is a senior member of IEEE, a senior member of China Computer Federation (CCF) and ACM, and a senior member of Chinese Society of Biomedical Engineering. His main research areas include Image Processing and Pattern Recognition, Biometrics, Biocomputing, Modeling and Simulation, Virtual Reality and Visualization. He has published over 300 papers and 6 books, got more than 10 patents, and won the second prize of National Teaching Achievement.

**Yong Xia** is an associate professor with the School of Computer Science and Technology, Harbin Institute of Technology at Weihai. He received his Ph.D. degree in Institute of Automation, Chinese Academy of Sciences. His research interest is pattern recognition.

**Henggui Zhang** received the Ph.D. degree in mathematical cardiology from the University of Leeds, in 1994. Then, he worked as a Postdoctoral Research Fellow with The Johns Hopkins University School of Medicine (1994–1995) and the University of Leeds (1996–2000), and then as a Senior Research Fellow with the University of Leeds (2000–2001). In October 2001, he moved to UMIST to take up a lectureship. Since then, he worked as a Lecturer with UMIST (2001–2004), and as a Senior Lecturer (2004–2006) and Reader (2006–2009) with The University of Manchester.

He currently holds the Chair of the Biological Physics Group, School of Physics and Astronomy, The University of Manchester. He is also a Professor of biological physics. He has published more than 400 scientific papers, among them over 200 papers were published in prestigious peer-reviewed journals in his field. Related works have attracted wide public interests, and have been covered by many prestigious media such as BBC. He has been elected as a Fellow of the world-renowned societies as the recognition of distinctions.