IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received October 7, 2019, accepted October 20, 2019, date of publication October 30, 2019, date of current version November 13, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2950383

ECG Generation With Sequence Generative Adversarial Nets Optimized by Policy Gradient

FEI YE[1,2], FEI ZHU[1,3], YUCHEN FU[2], AND BAIRONG SHEN[4]

[1]School of Computer Science and Technology, Soochow University, Suzhou 215006, China
[2]School of Computer Science and Engineering, Changshu Institute of Technology, Changshu 215500, China
[3]Provincial Key Laboratory for Computer Information Processing Technology, Soochow University, Suzhou 215006, China
[4]Institutes for Systems Genetics, West China Hospital, Sichuan University, Chengdu 610041, China

Corresponding author: Fei Zhu (zhufei@suda.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61303108, in part by the Natural Science Foundation of Jiangsu Higher Education Institutions of China under Grant 17KJA520004, in part by the Suzhou Key Industries Technological Innovation-Prospective Applied Research Project under Grant SYG201804, in part by the Program of the Provincial Key Laboratory for Computer Information Processing Technology, Soochow University, under Grant KJS1524, and in part by the Project Funded by the Priority Academic Program Development of Jiangsu Higher Education Institutions.

ABSTRACT Electrocardiogram (ECG) is a method used by physicians to detect cardiac disease. Requirements for batch processing and accurate recognition of clinical data have led to the applications of deep-learning methods for feature extraction, classification, and denoising of ECGs; however, deep learning requires large amounts of data and multi-feature integration of datasets, with most available methods used for ECGs incapable of extracting global features or resulting in unstable, low quality training. To address these deficiencies, we proposed a novel generative adversarial architecture called RPSeqGAN using a training process reliant upon a sequence generative adversarial network (SeqGAN) algorithm that adopts the policy gradient (PG) in reinforcement learning. Based on clinical records collected from the MIT-BIH arrhythmia database, we compared our proposed model with three deep generative models to evaluate its stability by observing the variance of their loss curves. Additionally, we generated ECGs with five periods and evaluated them according to six metrics suitable for time series. The results indicate that the proposed model showed the highest stability and data quality.

INDEX TERMS Deep learning, generative adversarial networks, policy gradient, electrocardiogram, time series.

I. INTRODUCTION

Increases in the annual incidence of cardiovascular disease have resulted in the use of electrocardiograms (ECGs) as a critical research target in precision medicine, especially for disease classification [1]–[3], morbidity prediction [4], and signal enhancement [5]. However, current analytical methods require an enormous quantity of clinical data with particular labels, there exists the problem that ECG signals are unable to support research requirements. Due to the sensitivity of clinical data represented by ECGs, researchers cannot access relevant data directly without encryption or composition. Therefore, it is imperative to develop a generative method to efficiently and securely synthesize ECG data.

As an initial simulation generation method, McSharry *et al.* [6] presented a dynamic model based on three ordinary differential equations to generate realistic ECG signals. Clifford *et al.* [7], [8] proposed a nonlinear model to generate long-term ECGs combined with realistic linear and nonlinear characteristics to extend the previous model by using a three-dimensional vector-cardiogram formulation. Moreover, Parvaneh and Pashna [9] and Sayadi *et al.* [10] generated ECGs based on a Gaussian combination model. Furthermore, Cao *et al.* [11] designed an ECG system to generate conventional 12-lead ECG signals, and Zhu *et al.* [12] proposed a deep generative model called bidirectional long short-term memory (BiLSTM) convolutional neural network (CNN) generative adversarial network (BCGAN) involving BiLSTMs and several layers of CNNs to generate simulated ECG signals recently.

However, these mathematical methods rely on a few specified formulas or an independent system, which inevitably makes it impossible to learn the variability, diversity, and global characteristics associated with an entire ECG dataset

The associate editor coordinating the review of this manuscript and approving it for publication was Victor S. Sheng.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see http://creativecommons.org/licenses/by/4.0/

over time. Additionally, BCGAN is barely satisfactory, concerning the steadiness of the training process and generation quality, because ECG signals belong to discrete tokens that can result in non-differentiable sub models. Lantao *et al.* [13] proposed the sequence generative adversarial network (SeqGAN) suitable for text in order to address the problem of non-differentiability. It is worth noting that ECG signals belong to time series and display periodicity despite both ECG and text being discrete.

To address these deficiencies, we proposed a new adversarial architecture called RPSeqGAN suitable for ECGs generation. Our main distribution is designing a deep neural network (DNN) called Resnet-Pooling block (RPblock), which adds a max-pooling layer of flexible substructures based on a residual network, and such basic blocks are combined with several nonlinear layers as our discriminator. We addressed the instability during training and low quality of our model by implementing policy gradient (PG) [14] and Monte Carlo (MC) [15] search through the SeqGAN algorithm, followed by generation of different segments of ECGs with high simulation accuracy and different periods.

The paper is organized as follows. In Section II, we reviewed the concept of PG and introduced several generative models. In Section III, we designed the architecture of the discriminator based on a novel structure and described its implementation with a simplified generator mainly comprising bidirectional gated recurrent units (BiGRUs) [16] used as the core network for the SeqGAN algorithm. The dataset analyzed included individual records for patients from MIT-BIH arrhythmia database [17]. In Section IV, we described testing of RPSeqGAN by training with specified steps, followed by observation of their loss curves in order to determine that our model displays the highest degree of stability and the lowest initial value. Models were obtained following training using WGAN-CP [18], WGAN-GP [19], and BCGAN, then we used them to generate ECGs with different lengths. In Section V, we selected three routine metrics for isometric sequences for evaluation and assessed the optimal model separately on each record based on three recursive metrics.

II. RELATED WORK

A. POLICY GRADIENT

In reinforcement learning (RL), policy search [20] is divided into a model-free policy search and a model-based policy search. The model-free policy search is classified as the stochastic policy search and deterministic policy search. Stochastic policy gradient (SPG) belongs to a basic stochastic policy search, which simultaneously integrates the space of states and actions.

Given a group of state-action trajectories, $\tau : \{s_0, a_0, \ldots s_T, a_T\}$, where T represents the amount of pairs of state-action in a trajectory, we denote

$$R(\tau) = \sum_{t=0}^{T} r(s_t, a_t) \tag{1}$$

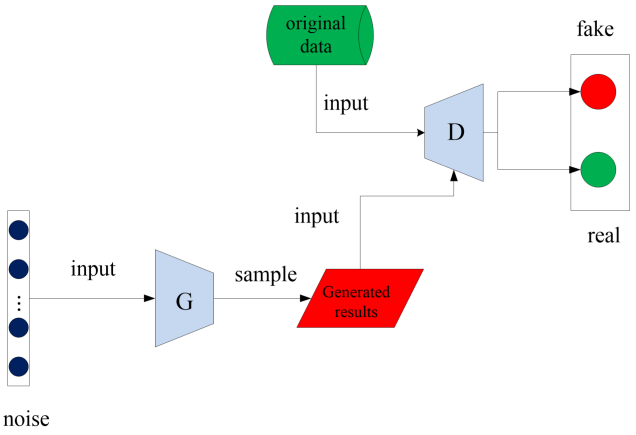

FIGURE 1. GAN architecture.

as the total reward of the trajectory τ, π_θ as the θ-parameterized stochastic policy and

$$P(\tau;\theta) = P(s_0)\prod_{t=0}^{T}\pi_\theta(a_t|s_t)P(s_{t+1}|s_t) \tag{2}$$

as the probability of following τ under policy π_θ, respectively, where $\pi_\theta(a_t|s_t)$ is the probability that the agent selects action a_t at s_t under π_θ, $P(s_{t+1}|s_t)$ is the transition probability from s_t to s_{t+1}. Our goal is to determine the optimum parameter that maximizes the expectation of cumulative rewards:

$$J(\theta) = E_{\pi_\theta}[R(\tau)] = \sum_{\tau} P(\tau;\theta)R(\tau) \tag{3}$$

The gradient of the objective function with respect to θ can be calculated as:

$$\nabla_\theta J(\theta) = \sum_{\tau} P(\tau;\theta)\nabla_\theta \log[P(\tau;\theta)]R(\tau) \tag{4}$$

The steepest-descent method with the learning rate, η, can be used to update θ_{old} *to* θ_{new} as showed in the following expression:

$$\theta_{new} \leftarrow \theta_{old} + \eta\nabla_\theta J(\theta) \tag{5}$$

B. GAN AND SEQGAN

Generative adversarial network (GAN) [21] was proposed as an antagonistic neural network comprising a generator, G, and a discriminator, D, (Fig. 1), both of which obey a zero-sum game during the entire training process, where both G and D achieve Nash equilibrium when they converge.

GAN has been previously applied in multiple fields, including image generation [22], style transfer [23], representation learning [24], [25], and anomaly detection [26]. Furthermore, we find that GAN is useful for denoising [27], [28], synthesis [29], and segmentation [30], [31] of medical images. However, conventional images-support GANs may perform badly during training when faced with sequences consisted of discrete tokens [32]. SeqGAN addresses these problems by regarding sequence generation

as a decision-making process in RL, and it has made advance in text generation and music synthesis [33].

In SeqGAN, the generator, G_θ, is considered as a stochastic θ-parametrized policy; a state $Y_{1:t-1}$ is a set of tokens that have been generated so far; an action is the next token available for selection. The role of the discriminator is to score a completed sequence with length T and output the judgement regarding whether it is true or false. The generator is then updated simultaneously by PG and MC search based on the feedback from the discriminator. The gradient $\nabla_\theta J(\theta)$ of the objective function from SeqGAN is calculated as:

$$\begin{aligned}\nabla_\theta J(\theta) &= E_{Y_{1:t-1}\sim G_\theta}[\sum_{y_t\in A}\nabla_\theta G_\theta(y_t|Y_{1:t-1})\cdot Q^{G_\theta}_{D_\varphi}(Y_{1:t-1},y_t)]\\ &\simeq \frac{1}{T}\sum_{t=1}^{T}\sum_{y_t\in\gamma}\nabla_\theta G_\theta(y_t|Y_{1:t-1})\cdot Q^{G_\theta}_{D_\varphi}(Y_{1:t-1},y_t)\\ &= \frac{1}{T}\sum_{t=1}^{T}E_{y_t\sim G_\theta(y_t|Y_{1:t-1})}[\nabla_\theta \ln^{G_\theta(y_t|Y_{1:t-1})}\cdot Q^{G_\theta}_{D_\varphi}\\ &\quad\times(Y_{1:t-1},y_t)] \end{aligned} \tag{6}$$

At time, t, where A is the set of actions; $G_\theta(y_t|Y_{1:t-1})$ is the probability that G_θ selects the action y_t at current state $Y_{1:t-1}$; $Q^{G_\theta}_{D_\varphi}(Y_{1:t-1},y_t)$ is the action-value function of an intermediate state. The function Q is expressed as:

$$\begin{aligned}&Q^{G_\theta}_{D_\varphi}(s=Y_{1:t-1},a=y_t)\\ &=\begin{cases}\frac{1}{N}\sum_{n=1}^{N}D_\varphi(Y^n_{1:T}),Y^n_{1:T}\in MC^{G_\beta}(Y_{1:t};N) & \text{t<T}\\ \mathrm{D}_\varphi(Y_{1:t}) & \text{t=T}\end{cases}\end{aligned} \tag{7}$$

where $D_\varphi(Y_{1:t})$ denotes the reward provided by the discriminator according to a complete sequence. The N-times MC search is applied with a roll-out policy, G_β, in order to sample the last T-t tokens. Finally, the optimizer (Adam or RMSprop) is selected to execute gradient update according to Eq. (5).

C. WGAN-CP AND WGAN-GP

Different with original GAN theory, WGAN-CP replaces Kullback-Leibler Divergence (KLD) with Earth-Mover (EM) distance for measurement between $\mathrm{P_r}$ (distribution of raw data) and $\mathrm{P_g}$(distribution of reconstructed data), and restricts the parameters updated by the discriminator at each step into a compact space compulsively to satisfy Lipchitz constraints.

However, WGAN-CP might result in weakness of modeling, explosion or vanish of gradient. To address these faultiness, WGAN-GP was proposed by adding a penalty term behind the loss function to stabilize training:

$$\begin{aligned}Loss = &\underset{x'\sim P_g}{E}[D(x')]-\underset{x\sim P_r}{E}[D(x)]\\ &+\lambda\underset{x'\sim P_{x'}}{E}[(||\nabla_{x'}D(x')||_2-1)^2]\end{aligned} \tag{8}$$

where the penalty term is based on the expectation of the $\mathrm{L_2}$ regularization of gradients from D, and λ is the penalty factor.

III. MODEL ARCHITECTURE

A. GENERATOR ARCHITECTURE

The generator G_θ samples N points from noise data, which are subject to a Gaussian normal distribution, with each noise point represented as a d_n-dimension vector $\boldsymbol{v_t}$:

$$\boldsymbol{S_{no}} = \boldsymbol{v_1}\oplus\ldots\oplus\boldsymbol{v_t}\oplus\ldots\oplus\boldsymbol{v_N} \tag{9}$$

where $\oplus$ is the concatenation operator that transforms all noise vectors into the input matrix $\boldsymbol{S_{no}}$. In the generator, BiGRUs are firstly adopted to process the input sequence, $\boldsymbol{S_{no}}$. At time, t, the hidden state, $\boldsymbol{h_t}$, is dependent on the hidden states from a pair of opposite and parallel directions (The symbol '*fw*' and '*bw*' are denoted as the forward and backward directions respectively):

$$\boldsymbol{h_t^{fw}} = BiGRUs_{fw}[\boldsymbol{v_t},\boldsymbol{h_{t-1}^{fw}}] \tag{10}$$

$$\boldsymbol{h_t^{bw}} = BiGRUs_{bw}[\boldsymbol{v_t},\boldsymbol{h_{tC1}^{bw}}] \tag{11}$$

$$\boldsymbol{h_t} = F_\sigma(\boldsymbol{h_t^{fw}}\oplus\boldsymbol{h_t^{bw}}) \tag{12}$$

$$p(\boldsymbol{y_t}|\boldsymbol{v_1},\ldots,\boldsymbol{v_t}) = F_o(\boldsymbol{W_o}\cdot\boldsymbol{h_t}+\boldsymbol{b_o}) \tag{13}$$

where an unidirectional hidden unit at time t depends on the current input $\boldsymbol{v_t}$, the forward hidden state flows $\boldsymbol{h_t^{fw}}$, or backward hidden state flows $\boldsymbol{h_t^{bw}}$; F_σ and F_o are the activation functions, where Rectified Linear Unit (ReLU) and Tanh are adopted here respectively. The purpose of using function Tanh at last layer is to normalize the lead of each generated ECG sampling point between -1 and 1. In the output layer, where $\boldsymbol{W_o}$ is the weight matrix and $\boldsymbol{b_o}$ is the basis, $p(\boldsymbol{y_t}|\boldsymbol{v_1},\ldots,\boldsymbol{v_t})$ is the conditional probability of the current output $\boldsymbol{y_t}$:

$$\boldsymbol{y_t} = G_\theta(\boldsymbol{v_1}\oplus\ldots\oplus\boldsymbol{v_t}) \tag{14}$$

when $t=N$, the generator outputs matrix $\boldsymbol{S_g}$ which has equal length to $\boldsymbol{S_{no}}$.

B. DISCRIMINATOR ARCHITECTURE

The generated ECGs might exhibit a normal distribution when the discriminator is based on variations or stacks of CNNs; however, our discriminator is based on RPblocks and not common CNNs combined with linear models, despite the latter performs well in natural-language processing (NLP) [34].

In the discriminator, an RPblock (as Fig. 2 shows), which is the basic substructure of D, consists of a residual network and a pooling layer with a adjustable filter size. The residual network contains a pair of convolution layers with filter size 1×3 and a shortcut connection to transfer the identity $\boldsymbol{x}$. Each convolutional layer has the same stride size and padding size with 1×1, which not only increases the depth of networks but also keeps the output size unchanged. Additionally, we adopted batch normalization (BN), the LeakyRelus function, and the max-pooling operation for the pooling layer, which reduces the complexity of forward computing and captures the relevant features of the input.

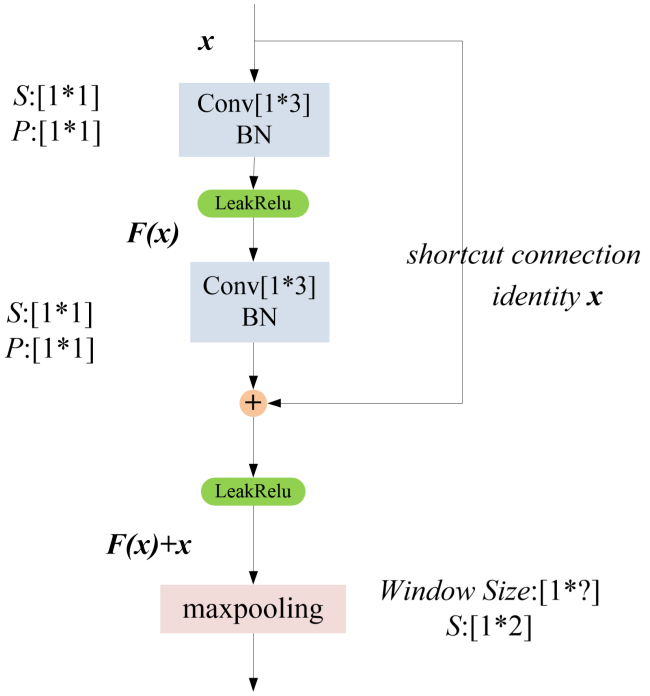

FIGURE 2. Framework of a Resnet-Pooling block. Function *F(x)*is the output of the first convolution layer with linear change and activation of*x*; the second convolution layer undergoes the same linear change and then combines *x*, i.e.*F(x)*+*x*, with activation as the input of the pooling layer.

TABLE 1. Parameter settings for sublayers in D.

Layer	Output	Windows	Stride
Conv1D	1*476	1*50	1*2
Resnet	4*1*476	[1*3, 1*3]	[1*1, 1*1]
Pooling1	4*1*234	1*10	1*2
Resnet	8*1*234	[1*3, 1*3]	[1*1, 1*1]
Pooling2	8*1*112	1*12	1*2
Resnet	16*1*112	[1*3, 1*3]	[1*1, 1*1]
Pooling3	16*1*50	1*14	1*2
Resnet	32*1*50	[1*3, 1*3]	[1*1, 1*1]
Pooling4	32*1*18	1*16	1*2
FC	80	-	-
Softmax	2	-	-
Output	1	-	-

Table 1 shows that the discriminator includes a 1D-convolution layer, several RPblocks, a fully connected (FC) layer, a softmax layer, and the output layer. In the i^{th} block, the number of resnet layers is $2i$+1, and the size of the windows for the pooling layer is 1*($2i$+8). We determined that the number of convolutional layers is $1 + \sum 2i + 2$ $(i = 1, 2, 3, 4)$ in total, and the FC layer contains 80 neurons, resulting in 46,080 neurons mapping between it and the Pooling4 (P4) layer. The first neuron of the softmax layer gives the probability, $\mathrm{P_{real}}$, that the judgement of the input sequence is true, and the second neuron gives the probability, $\mathrm{P_{false}}$, of false. If $\mathrm{P_{real}} \geq \mathrm{P_{false}}$, the discriminator outputs 1; otherwise, it outputs 0.

IV. ALGORITHM DESCRIPTION

The model structures, G_θ and D_φ, are used as the generator and discriminator in Algorithm 1, respectively. Different from the original SeqGAN algorithm, our training data is no longer produced by a target recurrent neural network (RNN), and it is unnecessary to evaluate a network model that produces training data that has nothing to do with ECG signals. In the following sections, we described the SeqGAN algorithm that uses our proposed model (RPSeqGAN).

In Algorithm 1, the roll-out policy, G_β, is set to the same architecture as G_θ initially. It should be noted that we omitted the pre-training of G_θ and D_φ in order to accelerate the training speed of the core architecture. Before training, the corresponding parameters of G_θ and D_φ are initialized with random weights, respectively.

The generator, G_θ, first produces a sequence of signals, $\boldsymbol{S_g}$, stochastically. We computed the action-value function of each intermediate state s_t (i.e., a subsequence consisting of signals that have been generated thus far) in $\boldsymbol{S_g}$ by MC search, where the reward is provided by D_φ. Finally, the generator employs PG to update the self-parameter, θ. The gradient $\nabla_\theta J(\theta)$ depends on action-value functions of all states.

When training the discriminator, the current generator, G_θ, is used to generate a complete sequence $\boldsymbol{S_g}$ whose length is equal to $\boldsymbol{S_r}$. We combined $\boldsymbol{S_g}$ and $\boldsymbol{S_r}$ into a tensor with size of $2\times T$, then the tensor is input into D_φ at each step. Notably, the positive sequence, $\boldsymbol{S_r}$, is obtained from the original data by traversing. The cross-entropy between the ground-truth tagging and the predicted probability is calculated as:

$$\min_{\varphi} -E_{\boldsymbol{S}\sim p_{real}}[\ln D_\varphi(\boldsymbol{S})] - E_{\boldsymbol{S}\sim G_\theta}[\ln(1 - D_\varphi(\boldsymbol{S}))] \quad (15)$$

where $\boldsymbol{S}$ is a variable quantity that represents a complete ECG sequence. The self-parameter, φ, which is updated at each step with a designated learning rate, is ultimately replaced by the minimum parameter at the last step.

Algorithm 1 comprises two internal loop branches, where an adversarial and constant process is formed. During this process, G_θ should keep pace with D_φ until both of them arrive at Nash equilibrium. Fig. 3 describes the process of the adversarial training by the proposed model for ECG generation.

V. EXPERIMENTS AND RESULTS

A. DATASET AND PLATFORM

The dataset was downloaded from the official website of the MIT-BIH Arrhythmia Database and it includes 48 30-min excerpts of two-channels ambulatory ECG records sampled at a rate of 360 Hz. Single-channel signals are considered preferential for facilitating this research. Each clinical record combines information concerning physiological characteristics, complicated paroxysms, and diagnostic results. Use of 48 individual records for training will result in ECGs with

Algorithm 1 Training Process of RPSeqGAN

Require: generator policy G_θ; roll-out policy G_β; discriminator D_φ;
an input sequence with T length: $\boldsymbol{S_r}$.

1: Initialize G_θ and D_φ with random weights θ, ϕ respectively.
2. $\beta \leftarrow \theta$
3: **repeat**
4: **for** g-steps **do**
5: Use G_θ to generate a T length sequence $\boldsymbol{S_g}$
6: **for** t in 1:T **do**
7: Compute $Q(\ a_t = y_t, s_t = \boldsymbol{S_g}[1{:}t\text{-}1]\ )$ by Eq. (7)
8: **end for**
9: Update parameters of G_θ via PG by Eq. (5)
10: **end for**
11: **for** d-steps **do**
12: Use current G_θ to generate negative examples and combine with given examples $\boldsymbol{S_r}$
13: Train D_φ for k epochs by Eq. (15)
14: **end for**
15: $\beta \leftarrow \theta$
16: **until** model converges
17: output G_θ

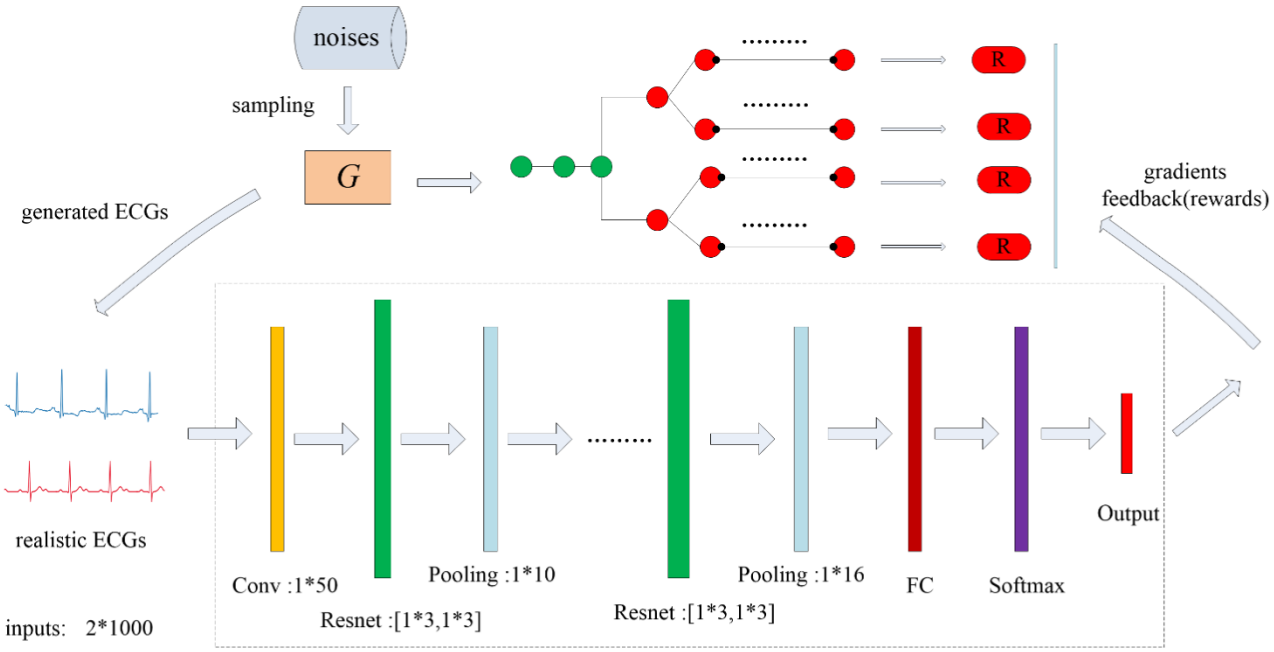

FIGURE 3. Description of the proposed method. The blue curve represents generated ECG signals, the red curve represents the real ECG signals from clinical records. Strips of single colors are used to denote different neural sublayers in *D*. In *G*, the green node is a state, the red node is an action, and the elliptical node is a reward that comes from *D*. *G* updates its parameters automatically using an Adam optimizer according to the feedback from *D*.

multiple features possessing no representative characteristics or practical significance.

To address this, we selected young women (aged 20-40) as our research target. There exists five eligible records from the database: 106, 113, 115, 208, and 212. According to the sampling rate and duration for each record, our dataset contains 3.25 million points and was divided into 3250 subsequences of length 1000.

We trained our dataset using a device with 16 GB of memory and a GeForce GTX 1080 Ti graphics-processing unit (GPU). The operating system is Ubuntu 16.04 LTS, and the development language is Python 2.7. The deep-learning framework is Pytorch, an open-source machine-learning library for Python, which also provides tensor computing with strong GPU acceleration and a tape-based auto grad system for DNNs.

B. SETTING AND TRAINING

In the pre-processing phase, signals sampled from the database were converted to units of mv prior to max-min normalization as the following expression:

$$normalized[x_{<n>}] = \frac{x_{<n>} - x_{\max}}{x_{\max} - x_{\min}} \tag{16}$$

where $x_{<n>}$, x_{max}, x_{min} denote the value of n^{th} sampling point, the value of the maximum sampling point, the value of the minimum sampling point, respectively. The training data were transformed into a tensor of size 3250 $\times$ 1000 (T = 1000). All correlative hyper parameters in G_θ and D_φ were simultaneously optimized using the Adam optimizer with a learning rate of 1e−6. Both noise and original points were embedded into a 16-dimensional vector and a 32-dimensional vector, respectively, to accelerate the advancement of the generator without pre-training. In the generator, the BiGRUs with hidden units of 16 cells have two layers. We calculated the binary cross-entropy (BCE) loss of D_φ by the following expression for each epoch:

$$\min_{\varphi} \sum_{i=1}^{C} \sum_{j=1}^{B} -E_{\boldsymbol{S_{ij}} \sim p_{real}}[\ln D_\varphi(\boldsymbol{S_{ij}})] - E_{\boldsymbol{S_{ij}} \sim G_\theta}[\ln(1 - D_\varphi(\boldsymbol{S_{ij}}))] \tag{17}$$

where C denotes the steps of the discriminator (simplified expression: d-step), and B denotes the amount of min-batches, which is set to 50 at each step. Therefore, the size of variable tensor $\boldsymbol{S_{ij}}$ was fixed to 50 $\times$ 1000. Moreover, it is necessary to state that the whole dataset was rearranged randomly for each d-step for highly effective training.

We analyzed the performance of RPSeqGAN using self-training steps of D_φ in each epoch and compared our model with other generative models. We first trained RPSeqGAN for 300 epochs with D_φ using 1, 2, 3, and 4 steps per epoch, respectively, in order to find the most stable model. We then compared the performance of the most stable RPSeqGAN result with those from WGAN-CP, WGAN-GP, and BCGAN. We clipped all weights for WGAN-CP to 0.1 and set the lambda term to four. All surplus parameters corresponded to the first part of the experiment.

C. RESULTS

Fig. 4 shows that the initial loss is minimized when D_φ is trained for four steps. The stability of each model was assessed by the slope of the loss curve and the speed of convergence. We found that the curves from the 2-dstep and 4-dstep models are nearly symmetrical at epoch ranks from 25 to 100, indicating similar variations in their slopes. After 100 epochs, the loss in the 4-dstep model is gradually lower than that in the 2-dstep model, and it converges preferentially. In particular, the 2-dstep model initially shows maximal loss, and the convergence speed of the 1-dstep model is clearly lower than that of the others. These results suggest that the RPSeqGAN model is the most stable when its discriminator is trained for 4-dstep.

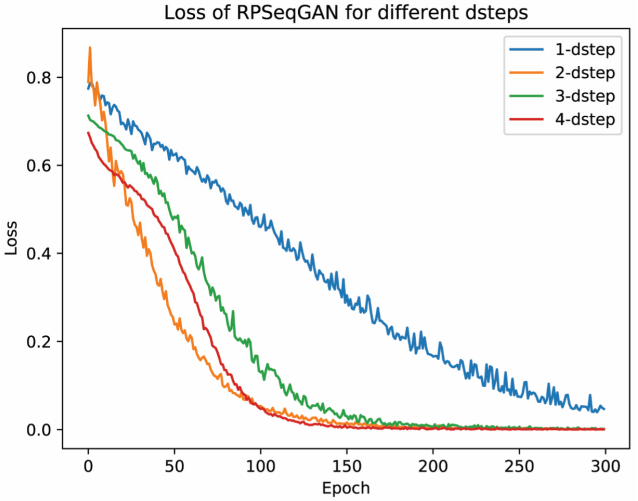

FIGURE 4. RPSeqGAN training according to dsteps. The horizontal axis represents the epoch and the vertical axis represents the loss. Curves of blue, orange, green and red respectively represent loss of 1,2,3, and 4 times of training for D during each epoch. In view of three curves gradually converge after 150th epoch, the horizontal axis shows 300 epochs appropriately.

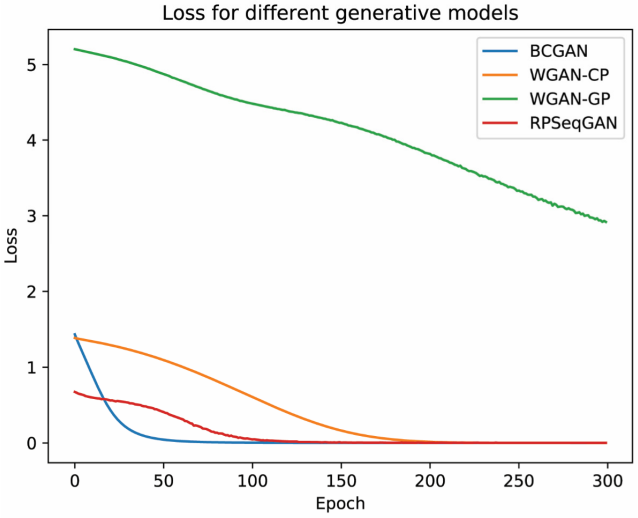

FIGURE 5. Dloss of four generative models. Curves of red, blue, orange, and green represent loss of RPSeqGAN (for 4 dsteps), BCGAN, WGAN-CP, and WGAN-GP, respectively.

Fig. 5 shows that WGAN-GP not only displays the highest loss during 300 epochs but also shows the lowest convergence speed, which is predicted to result in the worst performance. The initial loss demonstrates by the proposed model is less than that of BCGAN and WGAN-CP, and its loss changes smoothly rather than sharply before the 50$^{\text{th}}$ epoch.

In the following section, we described the superiority of our proposed model according to evaluation metrics for the quality of the generated results.

VI. ANALYSIS AND DISCUSSION

A. METRICS

In this part, we denoted $\boldsymbol{S_g}$: $\{x_1, \ldots, x_i, \ldots x_N\}$ as a generated sequence, $\boldsymbol{S_r}$: $\{y_1, \ldots, y_j, \ldots y_M\}$ as a real sequence, $\mathrm{P_r}$

as the probability distribution of generated signals, P_g as the probability distribution of original signals, respectively. We have $N := M$ for metric 1), 2), and 3), and $N \neq M$ for metric 4), 5), and 6).

1) KERNEL MAXIMUM MEAN DIFFERENCE

We took Kernel Maximum Mean Difference (KMMD) [35] to measure the similarity between P_r and P_g via a Gaussian kernel function:

$$\kappa(a, b) = e^{\frac{-|a-b|^2}{2\sigma^2}} \tag{18}$$

with a bandwidth σ which controls the range of the radial direction. We computed KMMD according to:

$$d^2_{KMMD}(P_g, P_r) = \underset{\substack{x_i, x_i' \sim P_g \\ y_j, y_j' \sim P_r}}{E} [\kappa(x_i, x_i') - 2\kappa(x_i, y_j) + \kappa(y_j, y_j')] \tag{19}$$

where lower KMMD means that P_g is closer to P_r.

2) EUCLIDEAN DISTANCE

We used Euclidean Distance (ED) to calculate the absolute distance between P_r and P_g by:

$$d_{ED}(P_g, P_r) = \sqrt{\sum_{k=1}^{N} (x_k - y_k)^2} \tag{20}$$

where a lower value of Euclidean distance indicates a higher generation quality.

3) RELATIVE ENTROPY

We applied relative entropy (RE) [36] to measure the difference between two probability distributions P_r and P_g in the same space of the event by:

$$\begin{aligned} d_{RE}(P_g, P_r) &= \sum\nolimits_x \boldsymbol{S_g}(x) \ln[\frac{\boldsymbol{S_g}(x)}{\boldsymbol{S_r}(x)}] \\ &= \sum_{k=1}^{N} x_k (\ln x_k - \ln y_k) \end{aligned} \tag{21}$$

where $P_g \sim \boldsymbol{S_g}(x)$ is the approximation, and $P_r \sim \boldsymbol{S_r}(x)$ is the real distribution that P_g should match. The lower entropy value is, the higher quality generations are.

4) TIME-WARP EDIT DISTANCE

We used Time-warp Edit Distance (TWED) [37] to evaluate tasks of comparing time series segments effectively. TWED is an elastic distance metric that enables warping in the time axis and integrates the edit distance with L_p-norms and denominated by a stiffness parameter, ν, and a mismatch penalty, λ. The process of computing TWED between P_r and P_g is an iteration which based on the following expression:

$$\begin{aligned} g(x_i, y_j) = \min\{&g(x_i, y_j) + \delta(x_i, y_j), g(x_{i-1}, y_j) \\ &+ \delta(x_i), g(x_i, y_{j-1}) + \delta(y_j)\} \end{aligned} \tag{22}$$

For $i = \{1, 2, \ldots, N\}$ and $j = \{1, 2, \ldots, M\}$, with:

$$\begin{aligned} \delta(x_i, y_j) &= \cos t(x_i, y_j) + \cos t(x_{i-1}, y_{j-1}) \\ &\quad + 2\nu \cdot |i - j| \end{aligned} \tag{23}$$

$$\delta(x_i) = \cos t(x_i, x_{i-1}) + \lambda + \nu \tag{24}$$

$$\delta(y_j) = \cos t(y_j, y_{j-1}) + \lambda + \nu \tag{25}$$

To compute TWED, we first initialized $g(x_0, y_0) = 0$, $g(x_i, y_0) = g(x_0, y_j) = \infty$. It should be underlined that we utilized the Mean Squared Error (MSE) function as the cost function in above three formulas, that is:

$$\cos t(x_i, y_j) = |x_i - y_j|^2 \tag{26}$$

When the iteration ends, the dissimilarity value $g(x_N, y_M)$ is regarded as the TWED between P_r and P_g. For better evaluation, we take all 24 possible constant combinations for λ and ν to compute a series of TWED values:

$$\begin{aligned} \boldsymbol{vec}_\lambda &= [0, 0.2, 0.4, 0.6, 0.8, 1] \\ \boldsymbol{vec}_\nu &= [0.001, 0.01, 0.1, 1]. \end{aligned}$$

The minimum value of these TWED values is obtained as the final metric value.

5) DYNAMIC TIME WARPING

We used Dynamic Time Warping (DTW) [38] to measure time series of variable size by dealing with a dynamic program using Bellman's recursion with a quadratic cost. To compute DTW, we first initialized $q(x_0, y_0) = 0$, $q(x_i, y_0) = q(x_0, y_j) = \infty$, where $i = \{1, 2, \ldots, N\}$ and $j = \{1, 2, \ldots, M\}$, then calculated $q(x_N, y_M)$ by the following expression with an iteration:

$$\begin{aligned} q(x_i, y_j) = |x_i - y_j|^2 &+ \min\{q(x_{i-1}, y_j), \\ &\times q(x_i, y_{j-1}), q(x_{i-1}, y_{j-1})\} \end{aligned} \tag{27}$$

When the iteration is over, the final result, $q(x_N, y_M)$ is obtained as the DTW between P_r and P_g.

6) SOFT DYNAMIC TIME WARPING

Soft Dynamic Time Warping (Soft-DTW) [39] is a differentiable loss function evolved from DTW, we used it to obtain the soft-minimum of all alignment costs. In addition to dealing with time series of variable size and the robustness of shifts or inflation across the dimensions associated with DTW, Soft-DTW shows superiority in quadratic time/space complexity.

We calculated Soft-DTW using the same method as DTW and replaced Eq. (27) with Eq. (28):

$$\begin{aligned} q(x_i, y_j) = |x_i - y_j|^2 &+ \overset{\gamma}{\min}\{q(x_{i-1}, y_j), \\ &\times q(x_i, y_{j-1}), q(x_{i-1}, y_{j-1})\} \end{aligned} \tag{28}$$

where exponent γ (set to 0.001) is the argument that controls Soft-DTW, and the operator $\min^\gamma\{\ldots\}$ is calculated by the following expression:

$$\overset{\gamma}{\min}\{a_1, a_2, a_3\} = -\gamma \ln \sum\nolimits_{i=1}^{3} e^{-a_i/\gamma} \tag{29}$$

TABLE 2. Evaluations of different generative models with specified lengths.

Metric	Model	0.5s	1s	1.5s	2s
KMMD	**RPSeqGAN**	**5.536**	**5.262**	**5.323**	**5.342**
	WGAN-GP	6.171	6.304	6.153	6.306
	WGAN-CP	6.521	6.097	6.431	6.444
	BCGAN	6.411	6.278	6.407	6.255
ED	**RPSeqGAN**	**7.385**	10.743	13.166	**14.979**
	WGAN-GP	7.794	11.601	13.235	15.628
	WGAN-CP	7.662	**9.793**	**12.989**	15.040
	BCGAN	7.753	11.678	13.719	15.588
RE	**RPSeqGAN**	**8.270**	**8.597**	**8.585**	**8.461**
	WGAN-GP	9.896	10.507	9.629	9.929
	WGAN-CP	9.826	8.720	9.633	9.600
	BCGAN	9.908	10.563	10.153	9.893

TABLE 3. Evaluations for RPSeqGAN with different clinical records.

Metric	Duration	Rc.106	Rc.113	Rc.115	Rc.208	Rc.212
TWED	0.5s	36.265	32.093	33.318	29.945	**27.467**
	1s	68.188	59.084	59.067	56.415	**52.472**
	1.5s	94.243	87.751	89.152	92.586	**80.742**
	2s	124.687	121.191	121.731	122.231	**112.538**
DTW	0.5s	**81.237**	87.245	89.591	99.237	92.331
	1s	**161.968**	167.166	174.249	184.300	181.509
	1.5s	**241.324**	250.319	270.511	270.621	260.296
	2s	**323.315**	330.770	354.998	362.029	343.227
Soft-DTW	0.5s	**46.250**	**46.053**	59.763	64.452	58.901
	1s	100.978	**98.033**	113.401	120.082	117.786
	1.5s	**150.705**	154.637	179.886	180.908	167.328
	2s	**203.753**	207.551	235.187	242.481	220.370

B. METHOD

We generated ECGs of 0.5s, 1s, 1.5s, and 2s using WGAN-CP, WGAN-GP, BCGAN, and RPSeqGAN, respectively, all of which were trained until convergence. To effectively assess each model, we calculated metrics (KMMD, ED, and RE) between generated ECGs and a segment of normal ECGs. Additionally, to evaluate the quality of samples generated by RPSeqGAN, we analyzed disparities between the generated ECGs and the specified durations of five clinical records and computed these distances by the following expression:

$$value = Median\{item = metric(\boldsymbol{S_g}, \boldsymbol{S_r})\} \tag{30}$$

where $\boldsymbol{S_r}$ is an original segment comprising 1000 real sampling points ($\sim 2.778s$), $\boldsymbol{S_g}$ is a generated segment which has various amounts of reconstructive sampling points. The distance between a generated segment and an entire clinical record is based on the traversal comparison between the generated segment and all of the original segments in a record. Due to that there exists 0.65 million real sampling points in each standard ECG record, an entire ergodic process of a record will last 650 cycles totally. When the ergodic process is finished, the median value is considered as the distance between a generated segment and the corresponding clinical record. Notably, we selected TWED, DTW, and Soft-DTW due to their superiority in dealing with time series of various sizes.

C. DISCUSSION

Table 2 shows that ECGs with different durations generated by RPSeqGAN are significantly better than those generated by WGAN-GP, WGAN-CP, and BCGAN, especially assessed by KMMD and RE. When assessed by KMMD, RPSeqGAN has a metric value between 5.0 and 6.0, whereas those of other models are between 6.0 and 7.0. When assessed by RE, RPSeqGAN has a metric value between 8.0 and 9.0, whereas those of other models are between 9.0 and 11.0. Both of above observations fully demonstrate that under the metric of KMMD and RE, our proposed model is superior to other generative models, whereas other models have smaller differences in metric values relatively. Additionally, WGAN-CP is only more excellent than RPSeqGAN when generating 1s and 1.5s ECGs according to ED assessment. However, WGAN-CP is nowhere near RPSeqGAN in light of other assessments, which has no impact on our model that performs

optimal in ECG generation among all deep generative models generally.

Table 3 shows that all of four generated ECG segments are absolutely as closest to record 212 and record 106 when assessed by TWED and DTW, respectively. For Soft-DTW, two generated ECG segments (0.5s and 1s) are closest to record 113, whereas other segments (1.5s and 2s) are closest to record 106. Additionally, we also observe that Soft-DTW (Duration: 0.5s, Rc.106; 46.250) is nearly equivalent to Soft-DTW (Duration: 0.5s, Rc.113; 46.053). The above analysis suggests that the generated ECGs with different durations are closet to record 106 in distribution and morphology. Furthermore, Table 3 also displays that no matter which metric is selected, the corresponding value increases dramatically with additional 180 sampling points (0.5s) of the generated ECG segments added each time.

VII. CONCLUSION

We designed a novel GAN architecture (RPSeqGAN) using PG and MC search to generate short-term ECGs. The experimental results demonstrate that the proposed method is not only the most stable among popular generative models but also best at generating ECGs regardless of periodicity or morphology. Additionally, we found that the stability of our proposed model hold pace with the increasing training steps of the discriminator. Moreover, although ECG signals are trained by our model with the same sampling rate as the MIT-BIH Arrhythmia Database, we are able to alter the number of points per second and use the trained model to generate ECGs with different sampling rates. Our future research will explore the use of RL methods with generative structures to generate long-term related ECGs.

REFERENCES

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

[2] S. Kiranyaz, T. Ince, and M. Gabbouj, "Real-time patient-specific ECG classification by 1-D convolutional neural networks," *IEEE Trans. Biomed. Eng.*, vol. 63, no. 3, pp. 664–675, Mar. 2016.

[3] S. Shadmand and B. Mashoufi, "A new personalized ECG signal classification algorithm using block-based neural network and particle swarm optimization," *Biomed. Signal Process. Control*, vol. 25, pp. 12–23, Mar. 2016.

[4] E. Volna, M. Kotyrba, and H. Habiballa, "ECG prediction based on classification via neural networks and linguistic fuzzy logic forecaster," *Sci. World J.*, vol. 2015, Nov. 2015, Art. no. 205749.

[5] H. D. Hesar and M. Mohebbi, "ECG denoising using marginalized particle extended Kalman filter with an automatic particle weighting strategy," *IEEE J. Biomed. Health Inform.*, vol. 21, no. 3, pp. 635–644, May 2017.

[6] P. E. McSharry, G. D. Clifford, L. Tarassenko, and L. A. Smith, "A dynamical model for generating synthetic electrocardiogram signals," *IEEE Trans. Biomed. Eng.*, vol. 50, no. 3, pp. 289–294, Mar. 2003.

[7] G. D. Clifford and P. E. McSharry, "Generating 24-hour ECG, BP and respiratory signals with realistic linear and nonlinear clinical characteristics using a nonlinear model," in *Proc. Comput. Cardiol.*, Chicago, IL, USA, Sep. 2004, pp. 709–712.

[8] G. D. Clifford, S. Nemati, and R. Sameni, "An artificial vector model for generating abnormal electrocardiographic rhythms," *Phys. Meas.*, vol. 31, no. 5, pp. 595–609, May 2010.

[9] S. Parvaneh and M. Pashna, "Electrocardiogram synthesis using a Gaussian combination model (GCM)," in *Proc. Comput. Cardiol.*, Durham, NC, USA, Sep./Oct. 2007, pp. 621–624.

[10] O. Sayadi, M. B. Shamsollahi, and G. D. Clifford, "Synthetic ECG generation and Bayesian filtering using a Gaussian wave-based dynamical model," *Phys. Meas.*, vol. 31, no. 10, pp. 1309–1329, Oct. 2010.

[11] H. Cao, H. Li, L. Stocco, and V. C. M. Leung, "Design and evaluation of a novel wireless three-pad ECG system for generating conventional 12-lead signals," in *Proc. BodyNets*, Corfu, Greece, 2010, pp. 84–90.

[12] F. Zhu, F. Ye, Y. Fu, Q. Liu, and B. Shen, "Electrocardiogram generation with a bidirectional LSTM-CNN generative adversarial network," *Sci. Rep.*, vol. 9, May 2019, Art. no. 6734, doi: 10.1038/s41598-019-42516-z.

[13] L. Yu, W. Zhang, J. Wang, and Y. Yu, "SeqGAN: Sequence generative adversarial nets with policy gradient," in *Proc. AAAI Conf. Artif. Intell.*, San Francisco, CA, USA, 2017, pp. 2852–2858.

[14] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Proc. NIPS*, 2000, pp. 1057–1063.

[15] C. B. Browne, E. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton, "A survey of Monte Carlo tree search methods," *IEEE Trans. Comput. Intell. AI in Games*, vol. 4, no. 1, pp. 1–43, Mar. 2012.

[16] X. Luo, W. Zhou, W. Wang, Y. Zhu, and J. Deng, "Attention-based relation extraction with bidirectional gated recurrent unit and highway network in the analysis of geological data," *IEEE Access*, vol. 27, no. 6, pp. 5705–5715, Dec. 2017.

[17] G. B. Moody and R. G. Mark, "The impact of the MIT-BIH arrhythmia database," *IEEE Eng. Med. Biol. Mag.*, vol. 20, no. 3, pp. 45–50, May 2001.

[18] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Proc. ICML*, Sydney, NSW, Australia, 2017, pp. 214–223.

[19] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of Wasserstein GANS," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, Long Beach, CA, USA, 2017, pp. 5767–5777.

[20] A. Y. Ng and M. I. Jordan, "PEGASUS: A policy search method for large MDPs and POMDPs," in *Proc. UAI*, San Francisco, CA, USA, 2000, pp. 406–415.

[21] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, Montpellier, QC, Canada, 2014, pp. 2672–2680.

[22] E. L. Denton, S. Chintala, and R. Fergus, "Deep generative image models using a Laplacian pyramid of adversarial networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, Montreal, QC, Canada, 2015, pp. 1486–1494.

[23] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proc. IEEE ICCV*, Venice, Italy, Oct. 2017, pp. 2223–2232.

[24] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," in *Proc. ICLR*, San Juan, PR, USA, 2016, pp. 1–16.

[25] M. F. Mathieu, J. J. Zhao, J. Zhao, A. Ramesh, P. Sprechmann, and Y. LeCun, "Disentangling factors of variation in deep representation using adversarial training," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, Barcelona, Spain, 2016, pp. 5040–5048.

[26] T. Schlegl, P. Seeböck, S. M. Waldstein, U. Schmidt-Erfurth, and G. Langs, "Unsupervised anomaly detection with generative adversarial networks to guide marker discovery," in *Proc. IPMI*, Boone, NC, USA, 2017, pp. 146–157.

[27] Q. Yang, P. Yan, Y. Zhang, H. Yu, Y. Shi, X. Mou, M. K. Kalra, Y. Zhang, L. Sun, and G. Wang, "Low-dose CT image denoising using a generative adversarial network with Wasserstein distance and perceptual loss," *IEEE Trans. Med. Imag.*, vol. 37, no. 6, pp. 1348–1357, Jun. 2018.

[28] M. Frid-Adar, I. Diamant, E. Klang, M. Amitai, J. Goldberger, and H. Greenspan, "GAN-based synthetic medical image augmentation for increased CNN performance in liver lesion classification," *Neurocomputing*, vol. 321, pp. 321–331, Dec. 2018.

[29] D. Nie, R. Trullo, J. Lian, C. Petitjean, S. Ruan, Q. Wang, and D. Shen, "Medical image synthesis with context—Aware generative adversarial networks," in *Proc. MICCAI*, Quebec City, QC, Canada, 2017, pp. 417–425.

[30] A. Lahiri, K. Ayush, P. K. Biswas, and P. Mitra, "Generative adversarial learning for reducing manual annotation in semantic segmentation on large scale miscroscopy images: Automated vessel segmentation in retinal fundus image as test case," in *Proc. IEEE CVPRW*, Honolulu, HI, USA, Jul. 2017, pp. 42–48.

[31] Y. Xue, T. Xu, H. Zhang, L. R. Long, and X. Huang, "SegAN: Adversarial network with multi-scale L_1 loss for medical image segmentation," *Neuroinformatics*, vol. 16, pp. 383–392, Oct. 2018.
[32] R. F. Huszár, "How (not) to train your generative model: Scheduled sampling, likelihood, adversary," Nov. 2015, *arXiv:1511.05101*. [Online]. Available: https://arxiv.org/abs/1511.05101
[33] S.-G. Lee, U. Hwang, S. Min, and S. Yoon, "Polyphonic music generation with sequence generative adversarial networks," Oct. 2017, *arXiv:1710.11418*. [Online]. Available: https://arxiv.org/abs/1710.11418.
[34] Y. Kim, "Convolutional neural networks for sentence classification," in *Proc. EMNLP*, Doha, Qatar, 2014, pp. 1746–1751.
[35] K. M. Borgwardt, A. Gretton, M. J. Rasch, H.-P. Kriegel, B. Schölkopf, and A. J. Smola, "Integrating structured biological data by kernel maximum mean discrepancy," *Bioinformatics*, vol. 22, no. 14, pp. 49–57, Jul. 2006.
[36] F. Pérez-Cruz, "Kullback–Leibler divergence estimation of continuous distributions," in *Proc. IEEE ISIT*, Toronto, ON, Canada, Jul. 2008, pp. 1666–1670.
[37] P.-F. Marteau, "Time warp edit distance with stiffness adjustment for time series matching," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 2, pp. 306–318, Feb. 2009.
[38] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-26, no. 1, pp. 43–49, Feb. 1978.
[39] M. Cuturi and M. Blondel, "Soft-DTW: A differentiable loss function for time-series," in *Proc. ICML*, Sydney, NSW, Australia, 2017, pp. 894–903.

FEI YE is currently pursuing the master's degree with the School of Computer Science and Technology, Soochow University. His main research interests include deep learning and biomedical informatics.

FEI ZHU received the Ph.D. degree. He is currently an Associate Professor with the School of Computer Science and Technology, Soochow University. He studies to design and applies machine learning algorithms to solve health data science, health informatics, predictive analytics, and personalized data-driven decision support problems. His main research interests include deep learning, reinforcement learning, text mining, and pattern recognition. He is a member of the China Computer Federation.

YUCHEN FU received the Ph.D. degree. He is currently a Professor with the School of Computer Science and Engineering, Changshu Institute of Technology. His main research interests include reinforcement learning, intelligence information processing, and machine learning. He is the Middle-Young Age Science and Technology Leader of the 333 Project in Jiangsu Province, the Training Target of Six Talent Peaks in Jiangsu Province, an Advanced Member of the China Computer Confederation, the Deputy Secretary General of the China Association of Artificial Intelligence, the Secretary of Artificial Intelligence Education Committee, Suzhou Artificial Intelligence Association, and a member of the Big Data Committee, Jiangsu Computer Federation, the Data Mining Committee, Jiangsu Computer Federation, and the Intelligent Big Data Committee of Suchow Artificial Intelligent Association. He was Chairman of the Suzhou Sub-Forum Academic Committee of CCF YOCSEF and an Executive Member of the Suzhou Division, China Computer Federation.

BAIRONG SHEN started his bioinformatics research, in June 1999, and got a Postdoctoral training at the University of Tampere, Finland, after that he was recruited as an Assistant Professor in the beginning of 2014. From 2004 to 2008, he taught eight courses for graduated students and these covered most subfields in bioinformatics. He returned to China, in June 2008, and was appointed as a Professor of systems biology and started to establish the Center for Systems Biology with Soochow University, where he acted as the Director. He is also the Founding Chair of the International Conference on Translational Biomedical Informatics (ICTBI) and the Worldwide Chinese Translational Biomedical Informatics Workshop (WC-TBIW). He acted as a Peer-Reviewer for more than 20 journals such as *Bioinformatics*, *Briefs in Bioinformatics*, *Nucleic Acids Research*, the *Journal of Molecular Cell Biology*, *RNA Biology*, *Clinical Chemistry*, *PLoS ONE*, *BMC Systems Biology*, *BMC Bioinformatics*, and *BMC Genomics*.