SimGANs: Simulator-Based Generative Adversarial Networks for ECG Synthesis to Improve Deep ECG Classification

Tomer Golany [1] **Daniel Freedman** [2] **Kira Radinsky** [1]

Abstract

Generating training examples for supervised tasks is a long sought after goal in AI. We study the problem of heart signal electrocardiogram (ECG) synthesis for improved heartbeat classification. ECG synthesis is challenging: the generation of training examples for such biological-physiological systems is not straightforward, due to their dynamic nature in which the various parts of the system interact in complex ways. However, an understanding of these dynamics has been developed for years in the form of mathematical process simulators. We study how to incorporate this knowledge into the generative process by leveraging a biological simulator for the task of ECG classification. Specifically, we use a system of ordinary differential equations (ODE) representing heart dynamics, and incorporate this ODE system into the optimization process of a generative adversarial network to create biologically plausible ECG training examples. We perform empirical evaluation and show that heart simulation knowledge during the generation process improves ECG classification.

1. Introduction

In the early days of AI research, it was claimed that "the power of an intelligent program to perform its task well depends primarily on the quantity and quality of knowledge it has about that task" (Buchanan & Feigenbaum, 1982). Indeed, in the last two decades it has been shown repeatedly that access to the wealth of knowledge possessed by humans significantly improves machine learning algorithms on various tasks (Gabrilovich & Markovitch, 2009; Zhang et al., 2016). Knowledge has been introduced into machine-learning algorithms in a host of different ways: by enriching the features (Gabrilovich & Markovitch, 2009), by reducing the labeling work (Mintz et al., 2009), and by encoding conditional dependencies (Wainwright & Jordan, 2008). However, it is less straightforward to introduce prior knowledge about biological and physiological systems. This is at least partially due to the fact that the complexity of biological systems stems from their dynamic nature. Each part of the system interacts and behaves collectively to produce distinct physiologic states and functionality. To fully understand the system, one needs to model its dynamics. Consider the problem of detecting cardiac abnormalities from an electrocardiogram (ECG) which measures the electrical activity generated by the heart. In recent years, deep learning algorithms have been applied for the problem, yielding state-of-the-art results (Kachuee et al., 2018; Al Rahhal et al., 2016). While the performance of such algorithms is quite promising, they are purely data-driven; they do not make use of the fact that at a fundamental level the interpretation of an ECG signal involves the understanding of the electrical conduction system of the heart. Intuitively, it seems reasonable that knowledge about the physics of the heart's electrical and mechanical activity might improve the performance of deep learning models. This knowledge has been developed for years in the form of mathematical process simulators. The challenge then becomes how to incorporate this knowledge into machine learning models. In this work we tackle this challenge directly, and propose a technique which allows us to leverage a biological simulator for the task of ECG heartbeat classification.

A simulator is a model of a phenomenon or a process created by a researcher to imitate the operation of a process or system (Banks et al., 2000). Simulation is often used to model natural systems to better understand their behavior. By changing variables in the simulation, one can make predictions about the behavior of the system over time. Simulators have been developed in a variety of fields, including chemistry, biology, and physics. Developing such a model requires deep understanding of how the system is constructed and operates. For example, a heart simulator requires understanding of the heart mechanics and anatomy.

[1]Technion - Israel Institute of Technology, Haifa, Israel [2]Google Research. Correspondence to: Tomer Golany <tomer.golany@cs.technion.ac.il>, Kira Radinsky <kirar@cs.technion.ac.il>, Daniel Freedman <danielfreedman@google.com>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020.

Such a model can be used to mimic the pressure-volume relationship and simulate various conditions and pathologies.

It is common to model systems via a mathematical model, which aims to predict the system behavior given knowledge about its nature. We focus in this work on continuous simulation, where time evolves continuously. In this setting, simulation is usually performed by numerical integration of Ordinary Differential Equations (ODE), that capture the researcher's knowledge regarding the physical or biological nature of the problem. More specifically, since the ODEs cannot generally be solved in an analytic manner, numerical approximation procedures such as Runge-Kutta are used to solve the equations. In this work, we leverage an ECG simulator proposed by (McSharry et al., 2003); the model simulates the electrical and mechanical activity of the heart by a set of ODEs. We explore the direct integration of these ODEs into deep learning models for the purpose of further improving the models, which have already reached state-of-the-art results for the task of ECG classification (Kachuee et al., 2018).

Recently, it has been shown that generating synthetic ECG signals of different arrhythmias to train deep learning models significantly improves the performance of classification models in personalized settings (Golany & Radinsky, 2019). We leverage this architecture to integrate simulator knowledge during the generation process. We hypothesize that this formulation enables generation of ECGs with better morphological characteristic of the different arrhythmias, thus improving classification. We present empirical results on gold-standard ECG datasets and show the superiority of algorithms working with simulator knowledge in the field of ECG heartbeat classification.

Our contribution in this work is threefold: (1) We present a methodology to leverage heart simulator ODEs to improve instance generation by an adversarial network to improve a supervised classification. To the best of our knowledge, this is the first application where a physiological simulator in the form of ODEs was leveraged to improve a supervised classification task. (2) We empirically show that utilizing the synthetically generated ECG instances guided by the heart simulator ODEs significantly improves ECG heartbeat classification using deep learning techniques. (3) We share our code online for further research and experimentation.[1]

The structure of our paper is as follows: Section 3 describes the ECG simulator represented by a system of ordinary differential equations. Section 4 introduces the framework of ECG Simulator GAN (SimGAN), a GAN-based setup which learns to create synthetic data by leveraging knowledge derived from a simulator represented by a system of ODEs (in our specific case, the ECG simulator from Section 3). Section 5 describes how the generated synthetic heartbeats from the SimGAN are used to train a deep network that classifies each heartbeat according to its heartbeat class. We use a ResNet convolutional model which was found to have superior results on the ECG gold-standard dataset (Kachuee et al., 2018)). Finally, in Sections 6-7 we present empirical evaluation comparing to state-of-the-art methods.

[1] https://github.com/tomerGolany/sim_gan

2. Related Work

In imitation learning, real-world simulation has shown empirical gains. For example, in a racing game, an agent will have access to a simulated environment, which is usually the game engine. (Zadok et al., 2019) showed that autonomous vehicle driving significantly improves after training using imitation learning in a simulated road environment. In reinforcement learning, game simulation has been used to improve learning (Mnih et al., 2013).

Previous work has also exploited generative models for the purposes of creating extra training data. In the computer vision realm, (Shrivastava et al., 2017) presented a GAN model which learns to improve the realism of synthetic labeled images using unlabeled real images. The generator's input is a synthetic image (instead of random noise), and the output is a refined image. The discriminator must learn to classify images as real vs refined. They show that the improved realism enables the training of better models for the estimation of gaze and hand pose by adding the refined labeled images to the training set. In the ECG heartbeat classification task, (Golany et al., 2020) similarly showed that classification performance can be improved by adding synthetic ECG heartbeats generated from a standard GAN to a training set. Our work also generates synthetic ECG signals that can be used in training; however, unlike previous work, we leverage a simulator to improve the deep generative model's ability to generate synthetic labelled ECG heartbeats.

Classic ECG beat-level classification methods focused on extracting interval features and using prior knowledge on the ECG morphology (De Chazal et al., 2004). Each ECG signal is separated into heartbeats using heartbeat detection techniques (Afonso et al., 1999). For each heartbeat, features related to the heartbeat intervals and ECG morphology are calculated. The combined features are then fed into supervised machine-learning models based on linear discriminants (LDs) (Nasrabadi, 2007). In recent years, the application of deep learning models to ECG classification has become popular. It has been applied to numerous tasks, such as cardiologist-level arrhythmia detection (Rajpurkar et al., 2017) and ECG heartbeat classification (Güler & Übeyli, 2005; Prasad & Sahambi, 2003; Kachuee et al., 2018; Al Rahhal et al., 2016). The state-of-the-art method for ECG heartbeat-level classification (Kachuee et al., 2018)

recently showed that superior results are reached by applying a ResNet convolutional model which classifies each heartbeat class separately. In this work, we focus on the training of such deep learning systems trained with additional synthetic ECG heartbeats which are generated from our proposed generative model enhanced with simulator knowledge (Section 5).

3. The ECG Simulator

In a normal resting heart, the physiological rhythm is referred to as the *normal sinus rhythm* (NSR). NSR produces the prototypical pattern of the P wave, followed by the QRS complex, and finally the T wave. To capture this pattern, McSharry et al. represented a model of the heart by a system of three ordinary differential equations (2003). The resulting simulator is able to generate synthetic ECG signals with realistic PQRST morphology, as well as prescribed heart rate dynamics. The simulator is parameterized by specific heart rate statistics, such as the mean and standard deviation of the heart rate, as well as frequency-domain characteristics of the heart rate variability (HRV)(Malik & Camm, 1990).

The simulator is specified by three coupled ODEs, giving rise to a trajectory $(x(t), y(t), z(t))$. The ODEs are as follows, where we have suppressed the time-dependence of x, y, and z for conciseness:

$$\frac{dx}{dt} = \alpha(x,y)x - \omega y \equiv f_x(x,y;\eta) \tag{1}$$

$$\frac{dy}{dt} = \alpha(x,y)y + \omega x \equiv f_y(x,y;\eta) \tag{2}$$

$$\frac{dz}{dt} = -\sum_{\beta\in\mathcal{B}} a_\beta \Delta\theta_\beta(x,y) e^{-\Delta\theta_\beta(x,y)^2/2b_\beta^2} - (z - z_0(t)) \equiv f_z(x,y,z,t;\eta) \tag{3}$$

where:

$$\alpha(x,y) = 1 - \sqrt{x^2+y^2} \tag{4}$$

$$\Delta\theta_\beta(x,y) = (\theta(x,y) - \theta_\beta) \mod 2\pi \tag{5}$$

$$\theta(x,y) = \text{atan2}(y,x)) \in [-\pi,\pi] \tag{6}$$

$$\mathcal{B} = \{P,Q,R,S,T\} \tag{7}$$

The so-called "baseline wander" has been introduced by coupling the baseline value $z_0(t)$ to the respiratory frequency f_2 using:

$$z_0(t) = A\sin(2\pi f_2 t) \tag{8}$$

where $A = 0.15$ mV. The baseline wander is low frequency noise (0.05 - 3 Hz in stress tests)(Sörnmo & Laguna, 2005), which is due to a variety of factors: movement during breathing, patient movement, poor contact between electrode cables and ECG recording equipment, inadequate skin preparation where the electrode has been placed, and dirty electrodes. ω is a scalar representing the angular velocity of the trajectory as it moves around the limit cycle.

Finally, we denote the sum total of all of the parameters of the ODEs as η; they consist of the values θ_β, a_β, and b_β for $\beta \in \mathcal{B} = \{P,Q,R,S,T\}$. Explicitly, we may write

$$\eta = (\theta_P,\theta_Q,\theta_R,\theta_S,\theta_T,a_P,a_Q,a_R,a_S,a_T, b_P,b_Q,b_R,b_S,b_T) \tag{9}$$

The solution of the equations generates a trajectory in a three dimensional state-space with coordinates (x,y,z). The generated ECG signal itself is just the third dimension, i.e. the signal $z(t)$, which is the movement of the trajectory around a limit cycle of unit radius in the (x,y) plane. Each revolution around this circle corresponds to a single heartbeat (or cardiac-cycle).

The η parameters: Distinct points on the ECG signal, such as the P, Q, R, S and T waves are described by events corresponding to negative and positive attractors / repellers in the z direction. The wave events locations are determined by the equations parameters $\theta_P,\theta_Q,\theta_R,\theta_S,\theta_T$ which represent fixed angles along the unit circle. When the z trajectory approaches one of these events, it is pushed upwards or downwards away from the limit cycle, and then it moves back toward the limit cycle. The height and length of each of the P, Q, R, S, T wave events is determined by the equations parameters a_P,a_Q,a_R,a_S,a_T and b_P,b_Q,b_R,b_S,b_T respectively. We follow the η parameters suggested by (McSharry et al., 2003) for a normal heartbeat.

Solving the ODE system is done numerically using *Runge-Kutta methods* (Butcher & Butcher, 1987) with a fixed time step $\Delta t = \frac{1}{f_s}$ where $f_s = 360$ is the sampling frequency. Specifically, for the numerical integration of the ordinary differential equations we leverage the simplest form of the Runge Kutta family known as the Euler method (Atkinson, 2008).

The Euler method is based on the finite difference approximation (Milne-Thomson, 2000):

$$\frac{du}{dt}(t) \approx \frac{u(t+\Delta t) - u(t)}{\Delta t} \tag{10}$$

Given an ODE of the form $du/dt = v$, the above may be rearranged to yield the following formula:

$$u(t+\Delta t) = u(t) + v(t)\Delta t \tag{11}$$

We construct the approximate solution as follows. We run Eq. 11 iteratively for L time-steps, where the ℓ^{th} step corresponds to $t_\ell = \ell\Delta t$; L is the number of samples that corresponds to a single signal. This yields

$$u_{\ell+1} = u_\ell + v_\ell \Delta t \tag{12}$$

Notice that approximating the values at time step $\ell + 1$ requires only the values at time step ℓ, which are already known. (Of course we require the initial conditions, u_0, as well.)

Substituting Eqs. 1, 2, 3 into Eq. 12 yields:

$$\begin{aligned} t_{\ell} &= \ell \Delta t \\ x_{\ell+1} &= x_{\ell} + f_x(x_{\ell}, y_{\ell}; \eta)\Delta t \\ y_{\ell+1} &= y_{\ell} + f_y(x_{\ell}, y_{\ell}; \eta)\Delta t \\ z_{\ell+1} &= z_{\ell} + f_z(x_{\ell}, y_{\ell}, z_{\ell}, t_{\ell}; \eta)\Delta t \end{aligned} \tag{13}$$

4. SimGAN Framework: Adding Simulators to GANs

We introduce the framework of ECG Simulator GAN (SimGAN), a GAN-based setup which has been enriched with additional knowledge from the ECG simulator presented in Section 3. Recall that in a regular GAN setting the generator learns to create synthetic data based on input noise. The data is then fed to a discriminator, which attempts to discriminate between synthetic and real data. Although generating ECG signals from noise allows for the creation of samples which do not appear in the training data, it does not yield ECG heartbeats with truly realistic morphology and characteristics. We therefore strive to incorporate the ECG simulator equations directly into the generation process. Following the practice of (Golany & Radinsky, 2019; Golany et al., 2020; Shrivastava et al., 2017), the generated ECG signals are then used to train a deep network (Section 5). We empirically show (Section 7) that the additional generated labeled examples significantly improve the heartbeat classification.

4.1. General GAN Framework

We formulate the generative adversarial networks for ECG heartbeats generation as follows. An ECG signal taken from a patient in one lead is sliced to heartbeats (cardiac cycles), which are labelled h. Each such heartbeat may be written as a fixed length vector $h = (h_1, \ldots, h_L)$, which represents the time evolution of the voltage values of a single heartbeat. The length $L = 216$; these 216 points represent a 600ms range sampled at 360 samples per second, where the range is from 200ms before the R-peak to 400ms after the R-peak. Our ultimate goal is to classify the heartbeats according to their heartbeat type c. We denote the underlying beats distribution of a given class as a conditional probability $p(h|c)$, which reflects the distribution of heartbeat signals given that the heartbeat is taken from class c.

Given a set of heartbeat signals $\{h^{(1)}, \ldots h^{(K)}\}$ drawn from class c, our general GAN framework consists of a discriminator and a generator, where both are class-specific. We denote the discriminator by $D(h; \phi_D^c)$, where the parameters of the discriminator network are given by ϕ_D^c, and depend on the class c. The generator is denoted by $G(\mathbf{m}; \phi_G^c)$, where $\mathbf{m}$ is a random variable with known, fixed distribution (Gaussian), and ϕ_G^c are the class-specific parameters. As is standard, G and D play the following two-player game:

$$\min_{\phi_D} L_D(\phi_D, \phi_G) \tag{14}$$

$$\min_{\phi_G} L_G(\phi_D, \phi_G) \tag{15}$$

However, note that in our case the game is not minimax, as the discriminator and generator will have objective functions which are not equivalent, i.e. $L_D \neq -L_G$. We now elaborate on the specific discriminator and generators we use, and their corresponding loss functions L_D and L_G.

4.2. The ECG Simulator Discriminator

Architecture The discriminator architecture and optimization remains the same as for a conventional GAN. Given positive ECG heartbeat samples from real patients and negative ECG heartbeat samples from the generator, the objective for the discriminator is to maximize the log-probability of assigning the correct labels to both positive and negative samples. The ECG heartbeats are fed through 6 one-dimensional convolutional layers. All weights were initialized from a zero-centered Gaussian distribution with a standard deviation of 0.02. After each layer we perform batch normalization and apply a Leaky ReLU activation function, where the slope of the leak is set 0.2. The final layer is a fully connected layer with sigmoid activation which classifies whether the heartbeat is from the real data or from the generator (fake).

Loss The discriminator's job is a typical classification problem. Thus, we use a cross-entropy loss:

$$\begin{aligned} L_D(\phi_D, \phi_G) = &- \mathbb{E}_{h \sim p_{data}} \log D(h; \theta_D) \\ &- \mathbb{E}_{m \sim \mathbf{m}} \log(1 - D(G(m; \theta_G); \theta_D)) \end{aligned} \tag{16}$$

which is the standard GAN discriminator loss.

4.3. The ECG Simulator Generator

Architecture The SimGAN Generator architecture differs from that of the classic GAN by leveraging knowledge from the ECG simulator (Section 3). Figure 1 describes the architecture. The generator input layer draws a length 100 vector from a Gaussian distribution with zero mean and identity covariance: $m \sim \mathbf{m} \equiv N(0, I)$. This noise vector m is then fed to 6 one-dimensional deconvolution layers. After each layer batch normalization (Ioffe & Szegedy, 2015) is performed followed by a ReLU activation function. The final one-dimensional deconvolution layer of the generator produces an output vector of size $L = 216$. The output dimensions of the generator therefore corresponds to the same dimension of one heartbeat (cardiac-cycle).

Loss: General Considerations The generator loss function is defined by a combination of the classical cross-entropy loss, which tries to generate cardiac-cycles that will fool the discriminator, and a Mean Square Error (MSE) function which tries to generate fake heartbeats which are morphologically similar to heartbeats from the ECG Simulator (Section 3). We now elaborate on each of these two loss functions.

Cross-Entropy Loss In contrast to the discriminator, the generator aims to minimize the log-probability that the discriminator correctly assigns negative labels to the samples generated by G. That is, the goal is to optimize the generator's weights in such a way that the discriminator network will predict that the generated ECG heartbeat is real. Formally, the cross-entropy loss function of the generator may be written

$$L_G^{CE}(\phi_D, \phi_G) = -\mathbb{E}_{m\sim\mathbf{m}} \log D(G(m;\theta_G);\theta_D) \quad (17)$$

Euler Loss In an ordinary GAN setting the generator learns to create synthetic data based on input noise which is later fed into the discriminator. We wish to leverage our world knowledge coming from the ECG simulator and use it to generalize the heartbeats produced to be more realistic. We therefore introduce the Euler Loss function.

We begin by defining a measure of how well a given heartbeat matches the ECG simulator. If the heartbeat is given by h and the simulator parameters are given by η, then the Simulator Distance is given by

$$\Delta_{sim}(h,\eta) = \sum_{\ell=1}^{L-1} \left(\frac{h_{\ell+1} - h_\ell}{\Delta t} - f_z(x_\ell, y_\ell, h_\ell, t_\ell; \eta) \right)^2 \quad (18)$$

where f_z is defined as in Eq. 3; and the trajectories for x_ℓ and y_ℓ are given by the discrete solution to the coupled ODEs in Eq. 13. In other words, the Simulator Distance attempts to determine how closely the generated heartbeat matches the biophysical model of a heartbeat, as specified by the set of ODEs. In order to do so, the z trajectory – which is the heartbeat itself – is fixed to the generated heartbeat; and the x and y trajectories are fixed to their ODE solutions. The Euler Loss then measures how closely the ODE in z holds. If the generated heartbeat is exactly reproduced by the model, then the ODE in z will hold exactly, leading to a Simulator Distance of 0; the further the deviation between generated heartbeat and the biophysical model's prediction, the larger the Simulator Distance. Note that in order for the Simulator Distance to be properly defined, we need an additional item, the initial conditions for x and y. These are taken to be $x_0 = -0.41, y_0 = -0.91$ as suggested by (McSharry et al., 2003).

Given this definition of the Simulator Distance, we define the Euler Loss as follows:

$$L_G^{EUL}(\phi_G) = \mathbb{E}_{m\sim\mathbf{m},\eta\sim p(\eta|c)}\ \Delta_{sim}(G(m;\theta_G),\eta) \quad (19)$$

That is, the Euler Loss is the expectation of the Simulator Distance over realizations of both the noise vector input m to the generator, as well as the simulator parameters η. The distribution of the simulator parameters is approximated, for a given class c, as a Gaussian, where the mean and covariance matrix are computed from the set of η-vectors corresponding to that class. (For each heartbeat in the class, one may compute the η-vector which generates a simulated heartbeat that is closest to that heartbeat.) The covariance matrix is taken as diagonal for simplicity. It is then straightforward to sample η.

Intuitively, then, the Euler Loss tries to constrain the output from the generator to lie close to the submanifold of signals which can produced by the simulator ODEs. Combining the Euler Loss with the classical cross-entropy loss enables the generator to create synthetic ECG heartbeats with real morphology and characteristics which don't exist in the training set, while preserving the noise which defines the real data.

Optimization The final loss function of the generator is therefore the sum of the two loss functions:

$$L_G(\phi_D, \phi_G) = L_G^{CE}(\phi_D, \phi_G) + L_G^{EUL}(\phi_G) \quad (20)$$

The optimization is performed using Adam optimizer (Kingma & Ba, 2014) and a learning rate of 0.0002 as suggested for training GAN models by (Radford et al., 2015). The generator's parameters are updated twice each iteration while the discriminator's parameters are updated once.

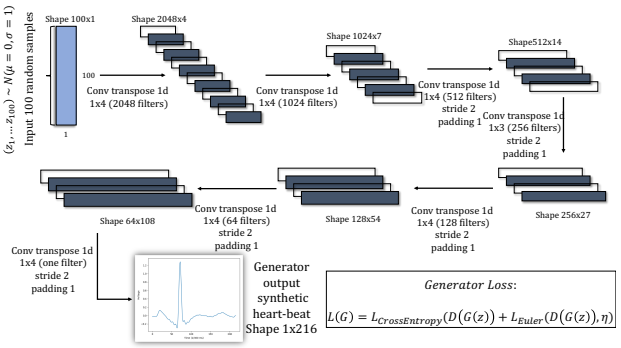


Figure 1. Generator model pipeline (Golany et al., 2020). The generator outputs a single synthetic heartbeat with characteristics similar to both the heartbeats generated from the ECG simulator as well as the real data.

5. Deep ECG Classification

In Section 4.1, we described the SimDCGAN – a framework for generating ECG heartbeats that exhibit both a certain arrhythmia and adapt morphological characteristics from

the ECG simulator (Section 3). In this section, we describe how the generated synthetic heartbeats from the generator are used to train a deep network that classifies each heartbeat according to its heartbeat class.

For evaluation we use the ResNet convolutional model which was found to have superior results on the ECG gold-standard dataset (Kachuee et al., 2018). The network is trained over the train portion of the dataset described in Section 6.1. The network architecture consists of a convolutional layer followed by five residual convolutional blocks. Each residual block contains two convolutional layers, two corresponding ReLU activations, a residual skip connection, and a pooling layer. The last residual block is followed by two fully-connected layers with 32 neurons each and a soft-max layer to predict output class probabilities. Each convolutional layer is a one-dimensional convolution through time and has 32 kernels of size 5. Once the model's initial training has converged, we perform further training using the additional generated ECG signals from the trained generator.

6. Experimental Evaluation

6.1. ECG Dataset

Our framework consists of ECG recordings taken from the MIT-BIH arrhythmia database (Moody et al., 2001), which is the most popular public dataset for discovering and clustering arrhythmias, and is considered the gold-standard evaluation data for ECG heartbeat classification tasks. The database contains 48 half-hour ECG records, obtained from patients studied by the BIH Arrhythmia Laboratory between 1975 and 1979. Each record contains two 30-minute ECG lead signals digitized at 360 samples per second. The database contains annotations for both heartbeat class information and timing information verified by independent experts. The database consists of a total of 109,492 heartbeats, each of which has been labelled with one of the following four classes: Ventricular Ectopic Beat, shortened to VEB or V; Supraventricular Ectopic Beat, shortened to SVEB or S; Fusion Beat, shortened to F; Normal beat, shortened to N.

6.2. Experimental Methodology

We follow the dataset partitioning as described by (De Chazal et al., 2004; Al Rahhal et al., 2016; Golany & Radinsky, 2019). All follow the AAMI (Association for the Advancement of Medical Instrumentation) recommendations for the ECG heartbeat classification task. This partition ensures that patient data is not mixed between the train and the test sets. We follow the experimental methodology used in (De Chazal et al., 2004; Al Rahhal et al., 2016; Kutlu & Kuntalp, 2012; Jiang & Kong, 2007), again as recommended by the AAMI; namely, we compute a precision-recall curve separately for each heartbeat class. (Note that in some of the above references, the medical convention is used, namely sensitivity (=recall) vs. positive predictive value (=precision).) For example, for the case of VEB, the precision-recall curve reports the performance in trying to distinguish V from the remaining classes S, F, and normal heartbeat. We report similar precision-recall curves for SVEB and Fusion (no curve is reported for the normal heartbeat class, as it is not an arrhythmia and considered a simple problem with little clinical value). As (Kachuee et al., 2018) follows an idiosyncratic methodology different from the rest of the literature, we have re-implemented the method of (Kachuee et al., 2018) and report results for their technique using the standard settings described above.

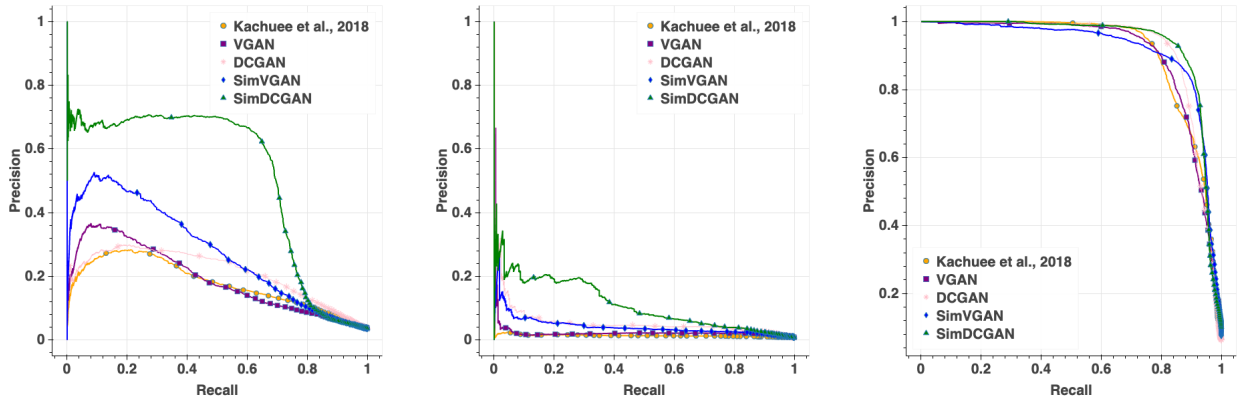
In what follows, we refer to the set of heartbeats from all patients in the train set as the *base set*. We present results for the following different data synthesis regimes:

No ECG Generation The ResNet convolutional model which recently showed state-of-the-art results in the task of ECG heartbeat classification (Kachuee et al., 2018) – see Section 5 – is trained on the base set, with no additional synthesized ECG examples.

ECG Generation using Ordinary GANs We train an ordinary GAN without any special loss terms (e.g. no Euler Loss) to generate synthetic heartbeats. This technique has two flavors:
(1) *Vanilla GAN (VGAN)* – The ResNet convolutional model is trained on the base set with additional synthesized ECG examples from a classical GAN model (Goodfellow et al., 2014).
(2) *DCGAN* – The ResNet convolutional model is trained on the base set with additional synthesized ECG examples from a DCGAN (Radford et al., 2015) model. As DCGANs have shown superior performance on several tasks (Radford et al., 2015; Yeh et al., 2017) we present results on generated examples from this class of generative adversarial networks.

ECG Generation using RefineGAN – Generating ECG heartbeats with a DCGAN framework where the inputs to the generator are synthetic heartbeats that comes out from the ECG Simulator. We would like to compare the contribution of integrating the ECG formulation as a loss term (The Euler loss) to a GAN, against a GAN setting where the input to the Generator are synthetic ECG heartbeats which are produced from the ECG Simulator instead of random noise. (Same as (Shrivastava et al., 2017))

ECG Generation using ECG Simulator We would like to test the contribution of integrating the ECG formulation to GAN as compared to directly using generated samples from an ECG simulator. We therefore propose a baseline model which trains the ResNet convolutional model on the base set with additional synthesized ECG examples



(a) SVEB heartbeat class (b) Fusion heartbeat class (c) VEB heartbeat class

Figure 2. Precision - Recall curves of the 3 heartbeat classes evaluated on the test-set. Each subfigure shows curves for each of the four data synthesis regimes described in Section 6, as well as the state of the art method of (Kachuee et al., 2018).

from the ECG Simulator (Section 3). For each of the three type of heartbeats to be classified, we calculate the values of their P, Q, R, S and T waves (Figure 4c). The average values for each wave is taken and is inserted into the ECG simulator as the wave parameters $\eta = (\theta_P, \theta_Q, \theta_R, \theta_S, \theta_T, a_P, a_Q, a_R, a_S, a_T, b_P, b_Q, b_R, b_S, b_T)$ (Section 3). For each heartbeat generated from the ECG simulator, we add additional Gaussian noise with zero mean and identity covariance to the the wave parameters η in order to generate multiple heartbeats from the same class.

ECG Generation using SimGANs The main approach proposed in this paper. The ResNet convolutional model is trained on heartbeats from all patients from the training set with additional synthesized heartbeats from a SimGAN model. As with ordinary GANs, two GAN architectures with the Euler additional optimization loss are tested:
(1) *SimVGAN*, using the VGAN architecture;
(2) *SimDCGAN*, using the DCGAN architecture.
The number of synthetically generated beats which are to be added to the training set is a parameter of the model, and depends on the number of samples from each class. For a heartbeat class with n samples in the base set, we experimented with the following values: $0.1n, 0.3n, 0.5n, 0.8n, n, 1.5n, 2n$.

In Appendix A we provide additional experiments which include training a Wasserstein-GAN (Arjovsky et al., 2017) with and without the additional Euler loss, and classification results using a not deep learning based classifier.

7. Results

Comparison with State-Of-The-Art Making a fair comparison in the task of ECG heartbeat classification is somewhat challenging due to the fact that different papers use different settings. For example, previous papers presented results where heartbeats from the same subjects were shared between train and test sets (Jiang et al., 2006); classifiers which used more than one lead during training (Llamedo & Martínez, 2012); classifiers which used RR intervals (Lin & Yang, 2014); and semi-supervised heartbeat classifiers which had access to unlabeled data from the test (Golany & Radinsky, 2019). In our setting, by contrast, we only make use of the raw ECG signals. We therefore compare our results to techniques which are state of the art for this setting: (Kachuee et al., 2018) and (Al Rahhal et al., 2016).

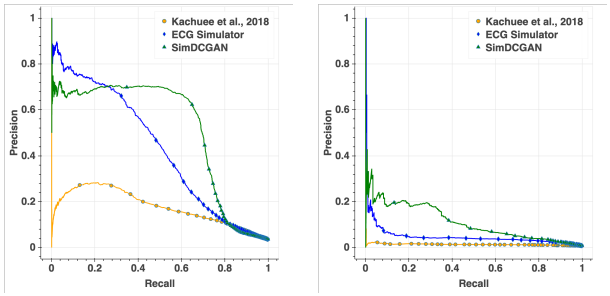
Figures 2(a) - 2(c) present precision recall curves for the task of classifying the MIT-BIH ECG test-set over the heartbeat classes SVEB, Fusion and VEB respectively. We show curves for each of the four data synthesis regimes described in Section 6, as well as the state of the art method of (Kachuee et al., 2018). The SVEB and Fusion heartbeats are quite rare and only comprise 3% and 2% of the dataset, respectively. We see that the performance of (Kachuee et al., 2018) on those heartbeat classes is quite low (Figures 2(a) and 2(b), yellow curves). Adding synthetic heartbeats from any of the generative models improves the classification performance. However, the results of our proposed SimDCGAN model are demonstrably superior to those of any other generative model: the green curves in Figures 2(a) and 2(b) are considerably higher than any of the other curves. This phenomenon is still present in the case of the much more common VEB heartbeats, though it is less pronounced, see Figure 2(c). In particular, the performance of (Kachuee et al., 2018) (yellow curve) is already quite high. Nevertheless, the SimDCGAN model, again shown as the green curve, is clearly better; for example, taking two arbitrary points on the two curves, (Kachuee et al., 2018) achieves (Recall, Precision) = (0.85, 0.75), vs. (0.88, 0.89) for SimDCGAN.

Table 1. Comparison between our method and (Al Rahhal et al., 2016). The methodology of (Al Rahhal et al., 2016) is to test against the entire MIT-BIH database. Re and Pr denote Recall and Precision. Best results are shown in bold and *.

	(Al Rahhal et al., 2016)		VGAN		DCGAN		SimVGAN		SimDCGAN	
Heartbeat class	Re	Pr	Re	Pr	Re	Pr	Re	Pr	Re	Pr
SVEB (S)	0.41	0.43	0.41	0.63	0.41	0.58	0.41	0.64	***0.41**	***0.80**
VEB (V)	0.91	0.79	0.91	0.70	0.91	0.72	0.91	0.82	***0.91**	***0.84**
Fusion (F)	0.60	0.04	0.60	0.10	0.60	0.20	0.60	0.25	***0.60**	***0.40**

In Table 1 we compare our method to another state-of-the-art deep learning technique, that of (Al Rahhal et al., 2016). We perform this comparison separately, as (Al Rahhal et al., 2016) report their results over the entire MIT-BIH dataset, that is, the combination of both train and test sets. As can be seen from Table 1, SimDCGAN far outperforms (Al Rahhal et al., 2016) on SVEB and Fusion heartbeats, nearly doubling the precision for the same level of recall for SVEB, and far more than doubling the precision for the same level of recall for Fusion. On the more common VEB heartbeats, the improvement is smaller but still present: a 6% relative increase in precision for the same level of recall.

Different Generative Architectures In Figures 2(a) - 2(c) and Table 1, we demonstrate that learning to adapt to the ECG simulator morphology helps for different types of GAN architectures – both VGAN and DCGAN. Beginning with the vanilla GANs, we observe that for all types of heartbeat, results for the model trained with additional data from SimVGAN outperforms the standard VGAN: in Figures 2(a) - 2(c), the blue curve is always significantly above the purple curve. A similar phenomena occurs also for the DCGAN model, in which SimDCGAN outperforms ordinary DCGAN for all classes. Indeed, as we have already noted, SimDCGAN outperforms all generative models across all classes, reaching state-of-the-art performance. (Table 1 supports the same conclusions numerically.) We therefore conclude that while adding synthetic ECG heartbeats always improves ECG classification, the SimDCGAN generation method yields the highest performance gains.



(a) SVEB heartbeat class (b) Fusion heartbeat class

Figure 3. Comparison of our method against generation of synthetic heartbeats directly from the ECG Simulator. Precision-Recall curves of the two more challenging heartbeat classes.

The Importance of the Generative Component By way of an ablation study, we test whether the SimDCGAN model, which relies on the Euler Loss, brings more value as compared to simply adding synthetic heartbeats from the ECG simulator to the training set directly. Figures 3(a) and 3(b) show the precision-recall curves for the two more challenging classes, SVEB and Fusion. We observe that adding plain data from the ECG simulator outperforms the ResNet convolutional model trained only with the original training data for both heartbeat classes. It is therefore clear that the addition of the simulator alone is of high value. However, SimDCGAN outperforms this model across both classes (with the exception of a small regime of very low recall in the case of the SVEB class). We speculate this is due to the fact that the data generated by the simulator does not contain the noise that exists in the real signals. It therefore does not model the real ECG data quite as well as SimDCGAN.

Table 2. Comparison between our method to a RefineGAN. Best results are shown in bold and *.

	SimDCGAN		RefineGAN	
Heartbeat class	Re	Pr	Re	Pr
SVEB (S)	***0.45**	***0.7**	0.4	0.47
VEB (V)	***0.88**	***0.89**	0.85	0.73

Refining ECG Simulator output We study a different approach to integrate the knowledge from the ECG Simulator into a GAN framework. We adapt (Shrivastava et al., 2017) method, by inserting to the generator synthetic ECG heartbeats which comes directly from the ECG Simulator, instead of the regular setting, in which the generator receives as input random noise. In this setting the loss terms for the generator and discriminator are the conventional cross-entropy losses, without the additional Euler loss term. Table 2 shows significant points on the precision-recall curves for the SVEB and VEB classes. SimDCGAN outperforms this model across both classes.

Qualitative Examples Figure 4 presents a qualitative example of a real patient ECG wave and synthetic waves produced by DCGAN and SimDCGAN. Figure 4(c) is a typical real heartbeat taken from the dataset. The real heartbeat is slightly noisy but the PQRST waves can be observed. Figures 4(a) and 4(b) both present synthetic waves. Although both look realistic, the wave 4(a), which shows a heartbeat produced by a DCGAN model, lacks a T wave (as shown by

the red circle) and is much smoother than a real heartbeat. Figure 4b shows a heartbeat generated from the SimDCGAN. The heartbeat contains real ECG morphology with PQRST waves and measurement noise as well.

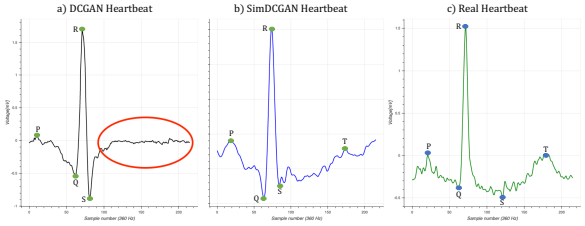


Figure 4. **a**: Heartbeat produced by DCGAN with no simulator knowledge. The red circle shows an incorrect T wave. **b**: Heartbeat produced by SimDCGAN. **c**: Real patient heartbeat.

8. Conclusions

We have presented a new technique for integrating prior knowledge in the form of a mathematical process simulator into a deep learning pipeline. We have shown how an ODE model describing cardiac cycles can be incorporated into a GAN-based generative model of these cycles using a novel Euler Loss. Empirically, we have demonstrated that by using the synthetic cardiac cycles generated from such a model as training data, we can improve the classification accuracy of a standard ResNet convolutional ECG heartbeat classifier, attaining state-of-the-art results. In particular, the results attained are better than those when examples are added either (a) directly from the simulator, or (b) from ordinary GANs which do not take the simulator into account.

The approach presented here is not specific to cardiac cycles; it applies equally well to any system described mathematically by differential equations. In the future, we will investigate the application of the approach to other systems described by ODEs, such as calcium dynamics in neuronal models (De Schutter & Smolen, 1998) and chemical reactions; as well as systems described by PDEs.

References

Afonso, V. X., Tompkins, W. J., Nguyen, T. Q., and Luo, S. Ecg beat detection using filter banks. *IEEE transactions on biomedical engineering*, 46(2):192–202, 1999.

Al Rahhal, M. M., Bazi, Y., AlHichri, H., Alajlan, N., Melgani, F., and Yager, R. R. Deep learning approach for active classification of electrocardiogram signals. *Information Sciences*, 345:340–354, 2016.

Arjovsky, M., Chintala, S., and Bottou, L. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

Atkinson, K. E. *An introduction to numerical analysis*. John Wiley & Sons, 2008.

Banks, J., Carson, J. S., Nelson, B. L., and Nicol, D. M. *Discrete-Event System Simulation (3rd Edition)*. Prentice Hall, 2000.

Buchanan, B. G. and Feigenbaum, E. A. *Knowledge-Based Systems in Artificial Intelligence*. McGrawHill, 1982.

Butcher, J. C. and Butcher, J. *The numerical analysis of ordinary differential equations: Runge-Kutta and general linear methods*, volume 512. Wiley New York, 1987.

De Chazal, P., O'Dwyer, M., and Reilly, R. B. Automatic classification of heartbeats using ecg morphology and heartbeat interval features. *IEEE transactions on biomedical engineering*, 51(7):1196–1206, 2004.

De Schutter, E. and Smolen, P. Calcium dynamics in large neuronal models. *Methods in neuronal modeling: From ions to networks*, 2, 1998.

Gabrilovich, E. and Markovitch, S. Wikipedia-based semantic interpretation for natural language processing. *J. Artif. Int. Res.*, 34(1):443–498, March 2009.

Golany, T. and Radinsky, K. Pgans: Personalized generative adversarial networks for ecg synthesis to improve patient-specific deep ecg classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 557–564, 2019.

Golany, T., Lavee, G., Yarden, S. T., and Radinsky, K. Improving ecg classification using generative adversarial networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *NIPS*. 2014.

Güler, I. and Übeyli, E. D. Ecg beat classifier designed by combined neural network model. *Pattern recognition*, 38 (2):199–208, 2005.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

Jiang, W. and Kong, S. G. Block-based neural networks for personalized ecg signal classification. *IEEE Transactions on Neural Networks*, 18(6):1750–1761, 2007.

Jiang, X., Zhang, L., Zhao, Q., and Albayrak, S. Ecg arrhythmias recognition system based on independent component analysis feature extraction. In *TENCON 2006-2006 IEEE Region 10 Conference*, pp. 1–4. IEEE, 2006.

Kachuee, M., Fazeli, S., and Sarrafzadeh, M. Ecg heartbeat classification: A deep transferable representation. In *2018 IEEE International Conference on Healthcare Informatics (ICHI)*, pp. 443–444. IEEE, 2018.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kutlu, Y. and Kuntalp, D. Feature extraction for ecg heartbeats using higher order statistics of wpd coefficients. *Computer methods and programs in biomedicine*, 105(3): 257–267, 2012.

Lin, C.-C. and Yang, C.-M. Heartbeat classification using normalized rr intervals and morphological features. *Mathematical Problems in Engineering*, 2014, 2014.

Llamedo, M. and Martínez, J. P. An automatic patient-adapted ecg heartbeat classifier allowing expert assistance. *IEEE Transactions on Biomedical Engineering*, 59(8): 2312–2320, 2012.

Malik, M. and Camm, A. J. Heart rate variability. *Clinical cardiology*, 13(8):570–576, 1990.

McSharry, P. E., Clifford, G. D., Tarassenko, L., and Smith, L. A. A dynamical model for generating synthetic electrocardiogram signals. *IEEE transactions on biomedical engineering*, 50(3):289–294, 2003.

Milne-Thomson, L. M. *The calculus of finite differences*. American Mathematical Soc., 2000.

Mintz, M., Bills, S., Snow, R., and Jurafsky, D. Distant supervision for relation extraction without labeled data. In *ACL '09*, 2009.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Moody, G. B., Mark, R. G., and Goldberger, A. L. Physionet: a web-based resource for the study of physiologic signals. *IEEE Engineering in Medicine and Biology Magazine*, 20(3):70–75, 2001.

Nasrabadi, N. M. Pattern recognition and machine learning. *Journal of electronic imaging*, 16(4):049901, 2007.

Prasad, G. K. and Sahambi, J. Classification of ecg arrhythmias using multi-resolution analysis and neural networks. In *TENCON 2003. Conference on Convergent Technologies for the Asia-Pacific Region*, volume 1, pp. 227–231. IEEE, 2003.

Radford, A., Metz, L., and Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

Rajpurkar, P., Hannun, A. Y., Haghpanahi, M., Bourn, C., and Ng, A. Y. Cardiologist-level arrhythmia detection with convolutional neural networks. *CoRR*, 2017.

Shrivastava, A., Pfister, T., Tuzel, O., Susskind, J., Wang, W., and Webb, R. Learning from simulated and unsupervised images through adversarial training. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2107–2116, 2017.

Sörnmo, L. and Laguna, P. *Bioelectrical signal processing in cardiac and neurological applications*, volume 8. Academic Press, 2005.

Wainwright, M. J. and Jordan, M. I. Graphical models, exponential families, and variational inference. *Found. Trends Mach. Learn.*, 1(1-2):1–305, January 2008.

Yeh, R. A., Chen, C., Lim, T.-Y., Schwing, A. G., Hasegawa-Johnson, M., and Do, M. N. Semantic image inpainting with deep generative models. In *CVPR*, 2017.

Zadok, D., Hirshberg, T., Biran, A., Radinsky, K., and Kapoor, A. Explorations and lessons learned in building an autonomous formula sae car from simulations. *SIMULTECH*, 2019.

Zhang, F., Yuan, N. J., Lian, D., Xie, X., and Ma, W.-Y. Collaborative knowledge base embedding for recommender systems. KDD'16, 2016.