A semi-supervised autoencoder framework for joint generation and classification of breathing

Pastor-Serrano, Oscar; Lathouwers, Danny; Perkó, Zoltán

DOI
10.1016/j.cmpb.2021.106312

Publication date
2021

Document Version
Final published version

Published in
Computer Methods and Programs in Biomedicine

Contents lists available at ScienceDirect

Computer Methods and Programs in Biomedicine

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

A semi-supervised autoencoder framework for joint generation and classification of breathing

Oscar Pastor-Serrano*, Danny Lathouwers[1], Zoltán Perkó[1]

Delft University of Technology, Department of Radiation Science and Technology, Mekelweg 15, Delft 2629JB, Netherlands

ARTICLE INFO

Article history:
Received 25 March 2021
Accepted 21 July 2021

Keywords:
Deep learning
Respiratory motion
Breathing signals
Convolutional neural network
Probabilistic autoencoder
Semi-supervised learning

ABSTRACT

Background and objective: One of the main problems with biomedical signals is the limited amount of patient-specific data and the significant amount of time needed to record the sufficient number of samples needed for diagnostic and treatment purposes. In this study, we present a framework to simultaneously generate and classify biomedical time series based on a modified Adversarial Autoencoder (AAE) algorithm and one-dimensional convolutions. Our work is based on breathing time series, with specific motivation to capture breathing motion during radiotherapy lung cancer treatments.

Methods: First, we explore the potential in using the Variational Autoencoder (VAE) and AAE algorithms to model breathing signals from individual patients. We then extend the AAE algorithm to allow joint semi-supervised classification and generation of different types of signals within a single framework. To simplify the modeling task, we introduce a pre-processing and post-processing compressing algorithm that transforms the multi-dimensional time series into vectors containing time and position values, which are transformed back into time series through an additional neural network.

Results: The resulting models are able to generate realistic and varied samples of breathing. By incorporating 4% and 12% of the labeled samples during training, our model outperforms other purely discriminative networks in classifying breathing baseline shift irregularities from a dataset completely different from the training set, achieving an average macro F1-score of 94.91% and 96.54%, respectively.

Conclusion: To our knowledge, the presented framework is the first approach that unifies generation and classification within a single model for this type of biomedical data, enabling both computer aided diagnosis and augmentation of labeled samples within a single framework.

1. Introduction

Biomedical data is the driving force behind most modern advances in medicine. The use of biomedical records is associated however with a series of problems such as the lack of reliable models capable of simulating data with clinical precision, the absence of personalized models for diagnosis, or the lack of labeled samples since the labels containing personal features that compromise privacy or simply are not recorded [1]. Some of the initial efforts to model biomedical data include analytical approaches: e.g., McSharry et al. [2] developed an electrocardiogram (ECG) model based on three coupled ordinary differential equations, and George et al. [3] introduced a sinusoidal model to represent breathing.

Recent advances in Deep Learning and the introduction of algorithms such as the Variational Autoencoder (VAE) [4,5] and Generative Adversarial Networks (GANs) [6] have resulted in a wide variety of methods capable of generating and classifying biomedical signals, most of them having been applied to ECG data. Regarding classification, Acharya et al. [7,8], Fujita et al. [9], Cimr et al. [10] and Yildirim et al. [11] present classification Convolutional Neural Network (CNN) frameworks for computer aided diagnosis based on biomedical signals. Yildirim et al. [12] propose an efficient algorithm based on autoencoder artificial neural networks (ANNs) that compresses ECG signals but lacks generative capabilities. Recent implementations of CNN architectures [13] and a combination of Long Short Term Memory (LSTM) Networks and convolutional autoencoders [14] result in minimal classification error of arrhythmia in ECG signals. With respect to generation, both Zhu et al. [15] and Delany et al. [16] propose generative models of realistic ECG signals that combine different ANN architectures (recur-

* Corresponding author.
E-mail address: o.pastorserrano@tudelft.nl (O. Pastor-Serrano).
[1] PhD

https://doi.org/10.1016/j.cmpb.2021.106312

rent and convolutional) under a GAN adversarial training objective. Golany and Radinsky [17] present a framework where a GAN generates data for ECG classification, while Wulan et al. [18] introduce an autoregressive model able to produce longer signals with high variability.

Most of the previously proposed methods focus either on generation or classification and result in models that depend on large labeled datasets and supervised training; are resource intensive and require significant amounts of computing power; are inaccurate when the dataset is imbalanced (there are very few labels for some classes of interest), or generate data that lacks variability and has a limited temporal dependence [19,20]. Furthermore, most of the approaches are not capable of capturing the structure of the data in a low-dimensional manifold in which specific regions correspond to similar samples.

In this study, we focus on mechanical breathing signals representing the movement of chest markers during respiration. Among their many applications, these type of biomedical signals are of great importance in radiotherapy cancer treatments, where they are used to quantify the impact of respiration and to design robust lung cancer radiotherapy treatments that withstand the detrimental effect of breathing motion during treatment delivery. Among the most important breathing irregularities are baseline shifts, which are gradual or sudden changes in the exhale position and trend of respiration. Baseline shifts negatively affect the outcome of radiotherapy treatments [21]. To our knowledge, there are no previous studies that develop breathing generative models that result in realistic respiratory traces. Likewise, very few computer-aided diagnostic tools have been presented for physical breathing signals. Abreu et al. [22] present an autoencoder framework that discriminates between apnea and regular breathing, focusing on gating radiotherapy treatments.

We investigate whether it is possible to combine classification and generation of breathing signals within a single model. We propose a semi-supervised framework that simultaneously classifies and generates breathing motion with high accuracy using a small subset of labeled data, and which outperforms purely discriminative models, and could in principle be applied to modeling other biomedical signals. The main contributions of this research are threefold. First, we investigate the suitability of probabilistic generative models based on one-dimensional convolutional filters for the task of modeling breathing signals. Second, building upon these breathing models, we introduce a modified semi-supervised algorithm to train a joint generative-discriminative model using a partially-labeled dataset. The proposed model can be used to simultaneously generate and classify samples of irregular breathing or samples from a population of patients. Third, we develop a pre-processing and post-processing method that transforms back and forth the breathing signals from their original 3-dimensional time series form into a simplified vector form containing pairs of position-time values. This transformation significantly reduces the dimensionality of the inputs and speeds up training.

2. Background

Probabilistic generative models

Consider $\mathbf{x} \in \mathbb{R}^M$ to be a random vector over a vector space $\mathcal{X}$, with unknown underlying probability distribution $p_{data}(\mathbf{x})$. Given a dataset $\mathcal{D} = \{\boldsymbol{x}^{(i)}\}_{i=1}^{N_\mathcal{D}}$ with $N_\mathcal{D}$ independent and identically distributed (i.i.d) data points, the goal is to model a probability distribution $p_{\boldsymbol{\theta}}(\mathbf{x})$ that approximates the unknown true probability distribution generating the data using a probabilistic graphical model with parameters $\boldsymbol{\theta}$. Let this probabilistic model be a latent variable model, which conditions the observed variable $\mathbf{x}$ on the unobserved random variable $\mathbf{z} \in \mathbb{R}^N$ over the latent space $\mathcal{Z}$ containing N latent variables that are assumed to capture the principal factors of variation in the data. The latent variable model represents the joint distribution of observed and unobserved variables and factorizes as $p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z}) = p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})p(\mathbf{z})$. The (target) marginal distribution of the observed variables can be recovered as

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \int_{\mathcal{Z}} p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) d\boldsymbol{z} = \int_{\mathcal{Z}} p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}) p(\boldsymbol{z}) d\boldsymbol{z}, \tag{1}$$

where $p(\boldsymbol{z})$ is the prior probability distribution over $\mathcal{Z}$ and $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$ is a conditional distribution that can be parametrized using neural networks. In principle, the prior could be any function and it is not conditioned on the observations. Point-estimates of the parameters $\boldsymbol{\theta}$ of the latent variable model can be obtained via maximum likelihood estimation, i.e., by maximizing the (log-) marginal distribution of the observed data

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \sum_{\boldsymbol{x} \in \mathcal{D}} \log\left(p_{\boldsymbol{\theta}}(\boldsymbol{x})\right) \simeq \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \ \mathbb{E}_{\mathbf{x} \sim \hat{p}_{data}(\boldsymbol{x})} \log(p_{\boldsymbol{\theta}}(\boldsymbol{x})), \tag{2}$$

where the expected value is computed over the empirical data distribution $\hat{p}_{data}(\boldsymbol{x})$. The empirical data distribution is different from the true underlying data generating distribution $p_{data}(\boldsymbol{x})$ to which we do not have direct access and we want to approximate. $\hat{p}_{data}(\boldsymbol{x})$ is defined as a mixture of Dirac delta distributions $\delta(\boldsymbol{x})$ that assigns probability mass $1/N_\mathcal{D}$ to each data point in $\mathcal{D}$ as

$$\hat{p}_{data}(\boldsymbol{x}) = \frac{1}{N_\mathcal{D}} \sum_{i=1}^{N_\mathcal{D}} \delta(\boldsymbol{x} - \boldsymbol{x}^{(i)}). \tag{3}$$

In practice, computing the integral over the space $\mathcal{Z}$ in Eq. 1 is intractable. Thus, the optimization in Eq. 2 is simplified by maximizing a lower bound on the marginal distribution.

Variational Autoencoder

Kingma and Welling [4], and Rezende et al. [5] present an algorithm that allows to estimate the latent variable model parameters maximizing the Evidence Lower BOund (ELBO). The algorithm, known as Variational Autoencoder (VAE), requires an inference model that approximates the (also) intractable true posterior distribution $p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})$ using a family of probability distributions of the latent variables $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ conditioned on observed data points, with parameters $\boldsymbol{\phi}$ shared across data points $\boldsymbol{x}$. By including the inference model, the ELBO optimization objective is formulated as

$$\begin{aligned} \log\left(p_{\boldsymbol{\theta}}(\boldsymbol{x})\right) \geq\ & \mathbb{E}_{\mathbf{z} \sim q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}[\log\left(p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})\right)] \\ & - D_{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) \| p(\boldsymbol{z})) := \mathrm{ELBO}(\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{x}), \end{aligned} \tag{4}$$

where the second term is the Kullback - Leibler (KL) divergence, denoted $D_{KL}(\cdot||\cdot)$. Essentially, the KL divergence quantifies ”the difference” between distributions. Further details about the ELBO and how to compute the KL-divergence are included in Appendix A.

In the VAE framework, the prior is the multivariate Gaussian $p(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{z}; \mathbf{0}, \boldsymbol{I})$, where $\boldsymbol{I}$ is the identity matrix. The likelihood conditional distribution $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$ is represented as a multivariate Gaussian probability distribution with identity covariance matrix $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}) = \mathcal{N}(\boldsymbol{x}; f_{\boldsymbol{\theta}}(\boldsymbol{z}), \boldsymbol{I})$, where the function $f_{\boldsymbol{\theta}}(\boldsymbol{z}) : \mathcal{Z} \to \mathbb{R}^M$ is parameterized with an ANN referred to as the probabilistic decoder. With this formulation, $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ is an infinite mixture of Gaussian distributions. In the same way as with the probabilistic decoder, it is possible to parameterize the inference model conditional distribution using a neural network that performs a mapping $g_{\boldsymbol{\phi}}(\boldsymbol{x}) : \boldsymbol{x} \in \mathcal{X} \to (\boldsymbol{\mu}(\boldsymbol{x}), \boldsymbol{\sigma}(\boldsymbol{x})) \in \mathbb{R}^{2N}$ and outputs the mean $\boldsymbol{\mu}(\boldsymbol{x})$ and standard deviation $\boldsymbol{\sigma}(\boldsymbol{x})$ of the Gaussian distribution $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{z}; \boldsymbol{\mu}(\boldsymbol{x}), \operatorname{diag} \boldsymbol{\sigma}^2(\boldsymbol{x}))$.

The ELBO balances two terms: the first term encourages the probabilistic decoder to produce samples that resemble the observed data, while the second term forces the approximated posterior distribution obtained from the inference model to be close to the prior distribution. Using the negative ELBO as optimization objective, the minimization problem to solve is:

$$\begin{aligned}\boldsymbol{\theta}^*, \boldsymbol{\phi}^* = \underset{\boldsymbol{\theta},\boldsymbol{\phi}}{\operatorname{argmin}}\, & \mathbb{E}_{\mathbf{x}\sim\hat{p}_{data}(\boldsymbol{x})}[-\mathbb{E}_{\mathbf{z}\sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}[\log(p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}))] \\ & + \beta D_{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z}))],\end{aligned} \tag{5}$$

where β is a hyperparameter that can be used to weigh the reconstruction and regularization terms [23]. The minimization in Eq. 5 can be performed using first order stochastic methods such as Stochastic Gradient Descent (SGD). The reparametrization trick is usually employed to propagate the gradients of the weights through the encoder, as described in [4]. Details on the VAE algorithm and how to estimate its gradients can be found in [4,24].

Adversarial Autoencoder

Makhzani et al. [25] propose an alternative formulation to the ELBO, where the KL divergence is approximated as the optimal value of an adversarial loss that forces the aggregated posterior distribution $q_{\boldsymbol{\phi}}(\boldsymbol{z})$ to be close to the prior:

$$q_{\boldsymbol{\phi}}(\boldsymbol{z}) = \int_{\mathcal{X}} q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})\hat{p}_{data}(\boldsymbol{x})d\boldsymbol{x} \;\; \simeq p(\boldsymbol{z}). \tag{6}$$

In the original paper, the authors explore the use of both probabilistic encoders and deterministic encoders with $g_{\boldsymbol{\phi}}(\boldsymbol{x})$ as a deterministic mapping. We use a universal approximation probabilistic encoder that in principle is able to learn any arbitrary posterior distribution by employing random noise $\eta \in H \in \mathbb{R}$ with distribution $p(\eta) = \mathcal{N}(\eta; 0, 1)$. Such encoders take additional random noise values to produce samples $\boldsymbol{z} = g_{\boldsymbol{\phi}}(\boldsymbol{x}, \eta)$, and can use different noise values η to map the same input $\boldsymbol{x}$ to a domain in $\mathcal{Z}$. The aggregated posterior can be computed as

$$q_{\boldsymbol{\phi}}(\boldsymbol{z}) = \int_{\mathcal{X}}\int_{H} \delta(\boldsymbol{z} - g_{\boldsymbol{\phi}}(\boldsymbol{x}, \eta))p(\eta)\hat{p}_{data}(\boldsymbol{x})d\eta d\boldsymbol{x}, \tag{7}$$

The adversarial loss is based on GANs. Let the encoder network be $g_{\boldsymbol{\phi}}(\boldsymbol{x}, \eta)$ with parameters $\boldsymbol{\phi}$ that performs a mapping $g_{\boldsymbol{\phi}}(\boldsymbol{x}, \eta) : \mathcal{X} \times H \to \mathcal{Z}$. A discriminator model is introduced, modeled also with an ANN with mapping function $d_{\boldsymbol{\xi}}(\boldsymbol{z}) : \mathcal{Z} \to \mathbb{R}$ that outputs a single scalar logit. The value $S(d_{\boldsymbol{\xi}}(\boldsymbol{z})) \in [0, 1]$ represents the probability that $\boldsymbol{z}$ is a sample from the prior distribution $p(\boldsymbol{z})$ (true samples) rather than being a latent space mapping from the encoder (fake samples), where $S(z) := (1 + e^{-z})^{-1}$ is the logistic sigmoid function. This translates into a min-max optimization problem

$$\begin{aligned}\min_{\boldsymbol{\phi}}\max_{\boldsymbol{\xi}}\, & \mathbb{E}_{\mathbf{z}\sim p(\boldsymbol{z})}[\log(S(d_{\boldsymbol{\xi}}(\boldsymbol{z})))] \\ & + \mathbb{E}_{\mathbf{x}\sim\hat{p}_{data}(\boldsymbol{x})}\mathbb{E}_{\eta\sim p_{(\eta)}}[\log(1 - S(d_{\boldsymbol{\xi}}(g_{\boldsymbol{\phi}}(\boldsymbol{x}, \eta))))],\end{aligned} \tag{8}$$

where first the discriminator is trained to correctly distinguish between real and encoder samples by maximizing the probability of classifying real samples from the prior $\boldsymbol{z}_r$ as real ($S(d_{\boldsymbol{\xi}}(\boldsymbol{z}_r) = 1)$) and fake samples from the encoder $\boldsymbol{z}_f$ as false ($S(d_{\boldsymbol{\xi}}(\boldsymbol{z}_f) = 0)$). Second, the encoder is trained to minimize the probability $1 - S(d_{\boldsymbol{\xi}}(\boldsymbol{z}_f))$ that the discriminator identifies its samples $\boldsymbol{z}_f$ as fake, where $d_{\boldsymbol{\xi}}(\boldsymbol{z}_f) = 1$ means that the discriminator classifies a fake sample as a true sample. Training the probabilistic decoder $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$, the inference model $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ and the discriminator $d_{\boldsymbol{\xi}}(\boldsymbol{z}_f)$ can be done with SGD in two alternating steps: a reconstruction phase forces the decoder to produce realistic samples by using the $\boldsymbol{z}_f$ variables produced by the inference model, and the regularization phase updating the parameters of the encoder and discriminator. As shown in Appendix B, optimizing the adversarial objective results in an approximation to the ELBO, where the optimum discriminator function is $d^*_{\boldsymbol{\xi}} = \log(q(z)/p(z))$ and the regularization term $\mathbb{E}_{\mathbf{x}\sim\hat{p}_{data}(\boldsymbol{x})}[D_{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z}))]$ in Eq. 5 is replaced by $D_{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z})||p(\boldsymbol{z}))$. More details about the adversarial objective can be found in Appendix B.

3. Joint generative-discriminative models

One of the advantages of the AAE algorithm is that the standard architecture can be slightly modified in order to additionally perform semi-supervised classification based on few labeled data points. The most notable difference with respect to the standard AAE architecture in [25] is the introduction of an extra discrete latent variable $\boldsymbol{y} \in \{0, 1\}^C$, which represents the class to which the input belongs over C classes. The class $\boldsymbol{y}$ is practically implemented as a sparse one-hot vector with a 1 entry at the position corresponding to the class. In the case of breathing, the $\boldsymbol{y}$ variable could indicate the presence of irregularities or the patient to which breathing pertains, while for ECG $\boldsymbol{y}$ could represent type of heart arrhythmia. The encoder now outputs the joint distribution $q_{\boldsymbol{\phi}}(\boldsymbol{y}, \boldsymbol{z}|\boldsymbol{x})$ that factorizes as

$$q_{\boldsymbol{\phi}}(\boldsymbol{y}, \boldsymbol{z}|\boldsymbol{x}) = q^c_{\boldsymbol{\phi}}(\boldsymbol{y}|\boldsymbol{x})q^s_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}), \tag{9}$$

where $q^c_{\boldsymbol{\phi}}(\boldsymbol{y}|\boldsymbol{x})$ is a categorical distribution that performs a mapping $softmax(\boldsymbol{\pi}(\boldsymbol{x})) : \mathcal{X} \to [0, 1]^C$ based on the input $\boldsymbol{x}$, and $\boldsymbol{\pi}(\boldsymbol{x})$ is a deterministic function. The use of the softmax non-linearity and one-hot vectors as a target forces sparsity in $q^c_{\boldsymbol{\phi}}(\boldsymbol{y}|\boldsymbol{x})$. We use the Gumbel-softmax reparametrization trick [27,28] to back-propagate the gradients through the categorical distribution. The approximate posterior $q^s_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ is either a distribution or a deterministic mapping, as in the standard AAE. In the original paper [25], the semi-supervised AAE is trained to perform either clustering or generation. Given that our goal is to simultaneously classify and generate new samples given a specific input, we propose a modified AAE architecture that uses a single discriminator for both the classification and style heads. In this way, the aggregated approximated posterior is forced to match the mixture prior distribution

$$q_{\boldsymbol{\phi}}(\boldsymbol{z}, \boldsymbol{y}) = \int_{\mathcal{X}} q^s_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})q^c_{\boldsymbol{\phi}}(\boldsymbol{y}|\boldsymbol{x})\hat{p}_{data}(\boldsymbol{x})d\boldsymbol{x} \;\; \simeq p(\boldsymbol{z}, \boldsymbol{y}). \tag{10}$$

where the prior distribution factorizes as the mixture

$$p(\boldsymbol{z}, \boldsymbol{y}) = p(\boldsymbol{z})p(\boldsymbol{y}) = \mathcal{N}(\boldsymbol{z}; \boldsymbol{0}, \boldsymbol{I})\mathrm{Cat}(\boldsymbol{y}; \boldsymbol{c}),$$

With this setup, each label $\boldsymbol{y}$ is associated with an independent low-dimensional space where $\boldsymbol{z}$ is distributed according to $p(\boldsymbol{z})$. Sampling from each cluster is easy, as opposed to the models presented in [25] that are specifically trained either for clustering or conditional generation of samples, and where $\boldsymbol{z}$ is jointly distributed according to $p(\boldsymbol{z})$ over all $\boldsymbol{y}$ classes.

Semi-supervised models

Let $\hat{p}_{data}(\boldsymbol{x}_l, \boldsymbol{y}_l)$ be the joint empirical distribution of labeled data $\boldsymbol{x}_l$ with labels $\boldsymbol{y}_l$. Our variant of the AAE, named Semi-supervised AAE (SAAE) in the remainder of the paper, is trained in 3 stages: a reconstruction and regularization phase that are identical to the ones in the standard AAE, and a supervised classification phase for the available labels in which the cross-entropy $\alpha \cdot \mathbb{E}_{\mathbf{x}_l, \mathbf{y}_l\sim\hat{p}_{data}(\boldsymbol{x}_l, \boldsymbol{y}_l)}[-\log q^c_{\boldsymbol{\phi}}(\boldsymbol{y_l}|\boldsymbol{x_l})]$ is minimized, where α controls the weight of the classification loss. The optimization problem is defined as

$$\begin{aligned}\text{Regularization: } \max_{\boldsymbol{\xi}}\, & \mathbb{E}_{\mathbf{z},\mathbf{y}\sim p(\boldsymbol{z},\boldsymbol{y})}[\log(S(d_{\boldsymbol{\xi}}(\boldsymbol{z}, \boldsymbol{y})))] \\ & + \mathbb{E}_{\mathbf{x}\sim\hat{p}_{data}(\boldsymbol{x})}\mathbb{E}_{\eta\sim p_{(\eta)}}[\log(1 - S(d_{\boldsymbol{\xi}}(g_{\boldsymbol{\phi}}(\boldsymbol{x}, \eta))))]\end{aligned} \tag{11}$$

$$\text{Classification: } \min_{\boldsymbol{\phi}} \alpha \cdot \mathbb{E}_{\mathbf{x}_l, \mathbf{y}_l\sim\hat{p}_{data}(\boldsymbol{x}_l, \boldsymbol{y}_l)}[-\log q^c_{\boldsymbol{\phi}}(\boldsymbol{y}_l|\boldsymbol{x}_l)] \tag{12}$$

$$\begin{aligned}\text{Reconstruction: } \max_{\boldsymbol{\theta},\boldsymbol{\phi}}\, & \mathbb{E}_{\mathbf{x}\sim\hat{p}_{data}(\boldsymbol{x})}\mathbb{E}_{\mathbf{z},\mathbf{y}\sim q_{\boldsymbol{\phi}}(\boldsymbol{z},\boldsymbol{y}|\boldsymbol{x})}[\log(p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}, \boldsymbol{y}))] \\ & + \mathbb{E}_{\mathbf{x}\sim\hat{p}_{data}(\boldsymbol{x})}\mathbb{E}_{\eta\sim p_{(\eta)}}\left[d_{\boldsymbol{\xi}}\left(g_{\boldsymbol{\phi}}(\boldsymbol{x}, \eta)\right)\right]\end{aligned} \tag{13}$$

4. Methods and materials

First, we investigate the benefits of applying VAE and the AAE to model respiratory motion of individual patients using few latent parameters. Second, using the presented SAAE architecture, we obtain a population breathing model capable of simultaneously classifying and generating specific types of breathing. We base our study on breathing signals, which are time series representing the position of chest markers in lung cancer patients. Fig. 1 shows an overview of the workflow, including the pre-processing, the models for classification and generations, and the final post-processing time series reconstruction step.

Patient and population data

Different breathing signals were obtained with the stereotactic radiosurgery system Cyberknife® (Accuray Inc., Sunnyvale CA, US). Cyberknife® tracks breathing movement using correspondence of markers positioned on the patient's chest [29]. The data used in our study consists of long respiratory traces for 21 different patients. The optical device tracks data with a 26 Hz frequency, for a total duration between ten and thirty minutes. The breathing signals for 15 out of the 21 patients were obtained from the open-access database recorded at Georgetown University Hospital (Washington D.C, United States) [30], with breathing amplitudes in the interval (0.5,10) mm. The 6 remaining respiratory traces were recorded during treatments at Erasmus MC (Rotterdam, Netherlands) and correspond to 6 patients with much smaller amplitudes in the range (0.5,2) mm. The 2 datasets are referred to as the GUH and EMC datasets for the remainder of the paper.

Input data & pre-processing

The first step consists of removing obvious errors in the signal acquisition process that are usually related to machine recalibration during measurement. This results in a 3D time series, where each dimension correspond to a physical dimension in the Cartesian coordinate system. Since the 3D are correlated, the 3D signals are further compressed into a 1D signal by using Principal Component Analysis (PCA) and projecting them onto the main axis of movement, which is the eigenvector with highest eigenvalue. We find that the projection onto the principal axis retains around 95% of the original variance. The resulting trace is divided into different periods $\boldsymbol{\tau}_j$, each of them corresponding to the time between start of different inhales. Each period j is discretized into 4 points with $A_{s,j}$ denoting position and a $\Delta_{s,j}$ representing the difference in time between consecutive points. Thus, a period is parametrized by the vector

$$\boldsymbol{\tau}_j = (A_{\mathrm{EE},j}, \Delta_{\mathrm{EE},j}, A_{\mathrm{MI},j}, A_{\mathrm{EI},j}, \Delta_{\mathrm{EI},j}, A_{\mathrm{ME},j}), \tag{14}$$

where s denotes the stage within each breathing period: EE for the end of exhale (or beginning of inhale), EI for the end of inhale (or beginning of exhale), and ME, MI for the 2 intermediate points between EE and EI. For simplicity, we omit the redundant $\Delta_{\mathrm{ME},j}$ and

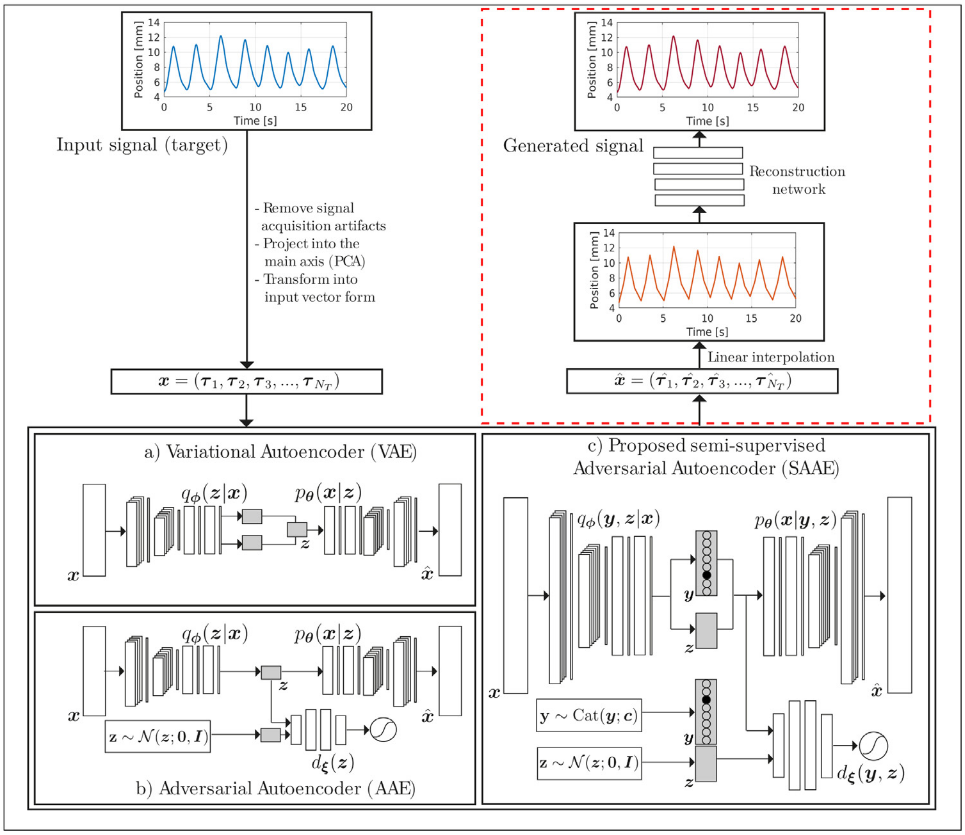
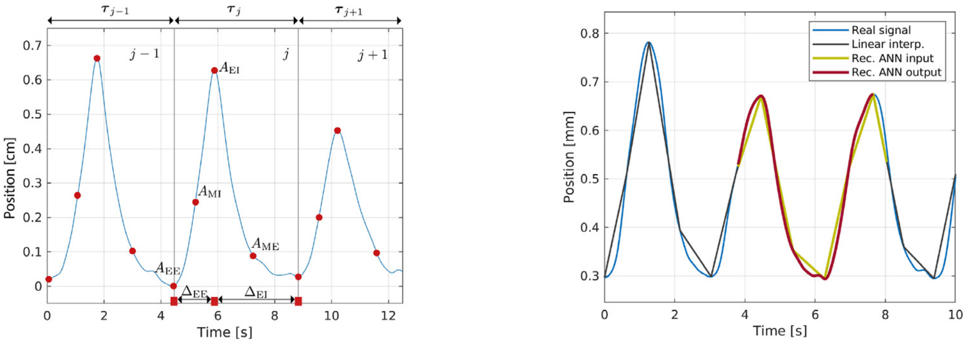

Position [mm]
14
12
10
8
6
4
0
5
10
15
20
Time [s]
Input signal (target)
- Remove signal acquisition artifacts
- Project into the main axis (PCA)
- Transform into input vector form
$\boldsymbol{x} = (\boldsymbol{\tau}_1, \boldsymbol{\tau}_2, \boldsymbol{\tau}_3, ..., \boldsymbol{\tau}_{N_T})$
Generated signal
Reconstruction network
Linear interpolation
$\hat{\boldsymbol{x}} = (\hat{\boldsymbol{\tau}}_1, \hat{\boldsymbol{\tau}}_2, \hat{\boldsymbol{\tau}}_3, ..., \hat{\boldsymbol{\tau}}_{N_T})$
a) Variational Autoencoder (VAE)
$q_\phi(\boldsymbol{z}|\boldsymbol{x})$
$p_\theta(\boldsymbol{x}|\boldsymbol{z})$
$\boldsymbol{x}$
$\boldsymbol{z}$
$\hat{\boldsymbol{x}}$
$z \sim \mathcal{N}(\boldsymbol{z}; 0, \boldsymbol{I})$
$d_\xi(\boldsymbol{z})$
b) Adversarial Autoencoder (AAE)
c) Proposed semi-supervised Adversarial Autoencoder (SAAE)
$q_\phi(\boldsymbol{y}, \boldsymbol{z}|\boldsymbol{x})$
$p_\theta(\boldsymbol{x}|\boldsymbol{y}, \boldsymbol{z})$
$\boldsymbol{y}$
$y \sim \mathrm{Cat}(\boldsymbol{y}; \boldsymbol{c})$
$d_\xi(\boldsymbol{y}, \boldsymbol{z})$

Fig. 1. Summary of the breathing modeling workflow. First, the original time series is pre-processed and projected into the main axis of movement (eigenvector with biggest eigenvalue) using Principal Component Analysis (PCA), from which the input vectors $\boldsymbol{x}$ are obtained. Patient or population models are then obtained through the use of the (a) VAE, (b) AAE and (c) SAAE with one-dimensional convolutional encoder and decoder models. In the VAE and AAE, the encoder (or inference) model produces a low-dimensional latent variable $\boldsymbol{z}$ that ideally captures the factors of variation in the dataset, such as variations in period, amplitude and exhale position. In the SAAE the inference model generates a class label latent variable $\boldsymbol{y}$ besides vector $\boldsymbol{z}$. Labeled data can be leveraged during training in order to learn the classification task in a semi-supervised manner. During generation (red dashed square), the sampled latent variables are transformed into the input vector form. These new vectors $\hat{\boldsymbol{x}}$ are then transformed into a time series with the help of an auxiliary reconstruction neural network. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)



(a) Times series discretization.

(b) Time series reconstruction.

Fig. 2. (a) Discretization of a breathing signal into periods and time-position points. In practice, the time series is discretized into a pair of time-position coordinates that are concatenated for a number of periods covering a certain desired time. (b) Transformation of the vector $\boldsymbol{x}$ into a time series. An additional ANN is trained to generate realistic breathing signals from linearly interpolated time series.

$\Delta_{\mathrm{MI},j}$ time coordinates, since they are equal to $\Delta_{\mathrm{EI},j}/2$ and $\Delta_{\mathrm{EE},j}/2$, respectively.

Fig. 2 a displays a fragment of the time series and its discretization into time-position points. A breathing sample is obtained by concatenating consecutive periods for the desired length of the signal. Each sample is assumed to be i.i.d. and is characterized by a vector $\boldsymbol{x} = (\boldsymbol{\tau}_1, \boldsymbol{\tau}_2, \ldots, \boldsymbol{\tau}_{N_T}) \in \mathbb{R}^{N_T \times 6}$ formed by N_T discretized periods. We use vectors of length $N_T = 25$ to model shorter signals of 1 to 2 minutes, and $N_T = 100$ for longer signals of several minutes corresponding to the typical duration of radiotherapy treatments. This compression step allows reducing the dimensionality of the breathing time series two orders of magnitude.

The pre-processing step results in 36,430 and 4,468 breathing fragments for the GUH and EMC datasets, respectively. Each data sample is assigned a baseline shift label according to the slope of the signal: if the slope of a sample is above a certain threshold value, the breathing sample is labeled as upwards baseline shift. Likewise, if the (negative) slope is below the threshold, the data point is labeled as downwards baseline shift. The threshold values correspond to the 7.5 upper and lower percentile of the distributions of slopes in the GUH dataset.

Convolutional filters

We use one-dimensional convolutional layers for both the encoder and decoder models under the assumption that these provide the encoder and decoder with powerful feature extractors that exploit the order in time and local structure of the periods. A one-dimensional discrete kernel convolution operation (denoted as $\boldsymbol{x} * \mathcal{K}$) over an input $\boldsymbol{x} \in \mathbb{R}^{N_T \times 6}$ with N_T time-steps and 6 channels for the different time and position values, using a kernel $\mathcal{K} \in \mathbb{R}^{K \times 6}$, consists of sliding the kernel matrix through the different j time-steps and computing

$$(\boldsymbol{x} * \mathcal{K})(j) = \sum_{k=1}^{K} \sum_{h=1}^{6} \mathcal{K}_{k,h}\, x_{j-k,h}. \tag{15}$$

Patient-specific models

To investigate the potential and limitations of signal modeling with probabilistic autoencoders, we first apply the standard VAE and AAE algorithms to model breathing signals from individual patients in the dataset separately. We train both the AAE and VAE frameworks using an isotropic Gaussian prior distribution $p(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{z}; \boldsymbol{0}, \boldsymbol{I})$. For the VAE, the parameter β in Eq. 5 is normalized with respect to the input dimension M and latent dimension N (which vary per model) as $\beta_n = (M/N)\beta$. 80% of the patient data is used to train the model, while the remaining 20% is equally split into a validation and a test set. Both the encoder and decoder consist of 4 convolutional layers and 2 fully-connected layers. Details about training and the architecture of the different models in the VAE and AAE are shown in Appendix C. After training the models, the input vector $\boldsymbol{x}$ can be reconstructed by sampling the inference model $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ to obtain $\boldsymbol{z}$, and then sampling the decoder. Artificial breathing signals can be obtained by decoding random samples from the prior $p(\boldsymbol{z})$.

Evaluating patient-specific models

A good model is capable of reconstructing unseen signals and generates artificial signals that distribute according to the training data. We perform several tests to asses the generative performance of the patient-specific model:

- Analyzing reconstruction error. To assess the reconstruction and generalization performance of the patient-specific models, we evaluate the reconstruction error of signals from the test set. For a fixed encoder and decoder architecture, we investigate the effect that varying the dimensionality of the latent space has on the reconstruction error of unseen test data. We verify and quantify the advantages of using convolutional layers by training models purely based on fully-connected layers and compare them to the one-dimensional convolutional models in terms of reconstruction performance.
- Assessing the generative performance. To determine if the model captures the data distribution, we train a classifier to distinguish between reconstructed and artificial samples from the model. Based on the same reasoning as in [34], we use reconstructed data instead of the original input vectors, since the compression through the latent space usually removes high-frequency noise in the original data that can be easily used by the classifier to distinguish samples. The classifier performance is evaluated for different latent space dimensionalities.
- Investigating the structure of the latent space. The presence of ”empty” regions in the latent space where no encodings $\boldsymbol{z}$ data are observed often results in low quality and variability of training samples. To determine the presence of empty regions, we evaluate the distribution of the distance between neighboring $\boldsymbol{z}$ from the dataset. Additionally, we evaluate possible mismatches between the aggregated posterior and prior distribution by comparing the distribution of the L2 norm of the encodings of the training samples and the samples from the prior.

Joint semi-supervised models of breathing irregularities

We apply the semi-supervised SAAE framework to model and classify baseline shift breathing irregularities, which are gradual downward or upward shifts of the exhale position. First, we perform two simple experiments using an analytical dataset that contains simplified sinusoidal breathing signals. In the first experiment (S1), we vary only the slope of the signals. In the second experiment (S2), we also modify the period and amplitude. The goal of S1 and S2 is to determine whether it is possible to obtain good models that classify and generate signals with upward or downward shift, or no shift at all (regular signals).

In the third experiment, we train the SAAE model using real breathing signals, and investigate the number of labeled samples needed to obtain accurate classification. All models are trained using the GUH dataset as the training set (with 10% as validation data) and tested on the EMC dataset.

Evaluating breathing irregularity models

The evaluation of the joint models is based on the F1-score, which was first introduced by van Rijsbergen [32] and is computed as $\text{F1}(p, r) = 2pr/(p + r)$, where p and r are the per-class precision and recall, respectively. For a given class, the precision is the proportion of correctly predicted samples over the total number of examples labeled as such class, while the recall is the fraction of correctly predicted samples over the total number of true samples for the given class. For multi-label classification, the macro F1-score (mF1) can be used, which is the average of F1-scores for the different classes. The baseline shift breathing irregularity models are tested with regards to both their classification and generative performance.

- Assessing classification performance. The discriminative performance (i.e., the ability to label signals having upward, downward or no baseline shift) is evaluated by comparing the classification accuracy of SAAE models to other neural network models purely optimized for classification. Specifically, convolutional neural network and fully-connected neural network discriminators are trained using a labeled subset of the training data. This additional convolutional classifier is similar to the encoder and inspired by state-of-the-art one-dimensional convolutional ECG models in [7,8,13]. We investigate how the number of labeled examples used during the supervised phase of training affects the classification accuracy of the SAAE by comparing its mF1-score to that of pure classifier networks.
- Evaluating generative performance. Inspired by [33] and [34], we evaluate the generative performance by calculating the Classification Accuracy Score (CAS), which allows to gauge whether the model generates realistic and varied samples. The CAS is obtained by training a discriminative model on data generated by the model, and evaluating the mF1-score on the real data test set.
- Analyzing the reconstruction error. Additionally, we evaluate the reconstruction performance of the model on GUH and EMC test data using 15 and 30 latent variables.

Time series reconstruction

The output vectors $\hat{\boldsymbol{x}}$ from the models have the same structure as the discretized input vector. Therefore, they must be transformed back into a time series by reconstructing the position values between two consecutive points in $\hat{\boldsymbol{x}}$. A first order approximation is a simple linear interpolation between the four position points in each cycle, which requires little time but lacks accuracy.

Alternatively, we reconstruct a realistic breathing time series using an additional feed-forward neural network, which we denote *reconstruction ANN*. The input is the linearly interpolated series, and the ANN learns a general mapping from the linear time series into realistic shapes. The input for the reconstruction ANN is no longer a vector of dimension $M = 6 \times N_T$, but a fragment of 120 position values (see Fig. 2b). The number 120 is a hyperparameter that is selected from a set of different candidate lengths. The output of the ANN is the first 100 transformed values of the input series. By adding 20 extra positions, the network achieves higher accuracy without discontinuities during concatenation of consecutive fragments. Further description of the ANN architecture is included in Appendix C.

The training data for the reconstruction ANN consists of slices with 120 elements of position values from the recorded breathing signals, and the corresponding linear interpolations. During training, the input and output slices are normalized to the interval [0,1]. A single general ANN would allow to reconstruct the time series from any patient in the population and make the process highly scalable. We investigate whether it is possible to train a general reconstruction ANN using only a subset of the data (either data from a single patient or a subset of data from all the patients). For this, we train the reconstruction ANN using (i) data from one patient (referred to as PatBR model from on) and (ii) a subset of data from the GUH data (referred to as PopBR model), while both models are tested using the EMC dataset. The PatBR is trained using a single patient from the GUH dataset, while the PopBR is trained on 10% of the GUH dataset, instead of on all available samples. This is due to the fact that, unlike with the AAE, VAE and SAAE vector inputs, the training dataset for the reconstruction task consists of few million fragments of the breathing time series (vectors with 120 position values) obtained from linear interpolation of the generated vectors.

5. Results

Patient-specific models

The results of the evaluation of the AAE and VAE patient specific models in terms of reconstruction and generative performance are shown in Fig. 3 for 2 randomly selected patients. The models for the first and second random patient were trained using 1890 and 2653 samples, respectively. Fig. 3a displays the reconstruction error on unseen test set data for different latent space dimensionalities. The error values are re-scaled to the interval [0,1] to facilitate comparison, 1 corresponding to the maximum error achieved at weight initialization. We compare the error achieved by models based on one-dimensional convolutional architectures and models purely based on fully connected layers. Although the error always decreases with increasing latent dimension N, the convolutional architectures notably increase the accuracy in the reconstruction. For qualitative evaluation, Fig. 4 shows reconstructions of the original inputs using a convolutional model with a 5-dimensional latent space ($N = 5$).

The generative performance is shown in Fig. 3b, depicting the accuracy of a CNN classifier trained to distinguish reconstructed data points from artificial samples generated by models with varying latent dimensionality. We plot the average and standard deviation of 3 different classifiers trained on distinct artificial data. The data is generated either by sampling the prior $p(\boldsymbol{z})$, or by taking $\boldsymbol{z}$ encodings in the vicinity of $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$, where the latter cover a much smaller domain of the latent space. The auxiliary classifier performs worse when distinguishing real and AAE samples, hinting that these better capture the distribution of the data. Note that the binary cross entropy loss values are almost always above 1 for the $p(\boldsymbol{z})$ classifier, which indicates the presence of uncertainty and significant miss classification errors.

To study the structure of the latent space Fig. 3c shows the distribution of the distance between neighboring encodings. Since the L^n-norm distance metric always increases with the number of latent dimensions N, we divide the L1 norm between nearby $\boldsymbol{z}$ by the latent space dimensionality. The plotted distributions indicate

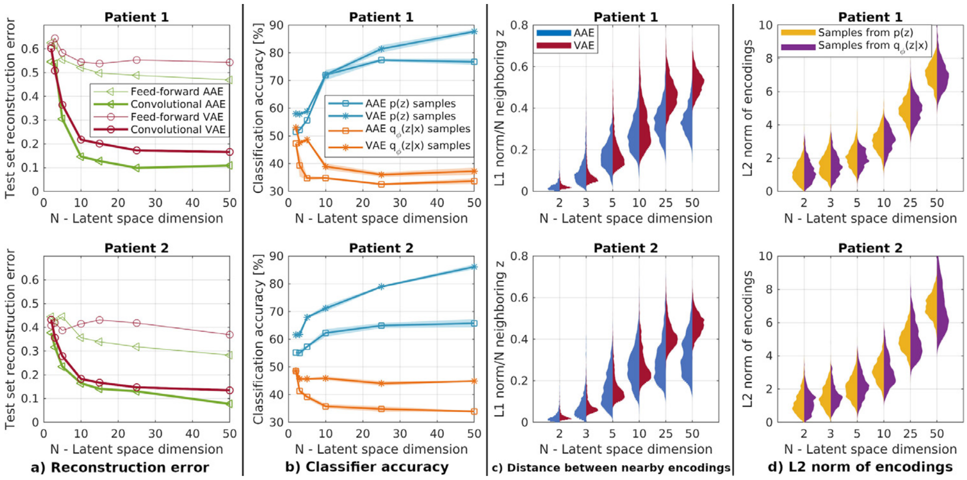


Fig. 3. Summary of the patient-specific model evaluation. (a) Reconstruction error on the test set for different latent space dimensionalities N. (b) Performance of an additional classifier trained to distinguish samples from the dataset from artificial samples from the model. Shaded regions represent the standard deviation around the mean (solid). (c) Distribution of the distance between neighboring encodings, for the AAE (blue) and VAE (red). The L1 norm distance is normalized by dividing by the latent space dimensionality. (d) Distribution of the L2 norm of the real data encodings $\boldsymbol{z}$ (yellow) and sampled encodings from the prior distribution (purple). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

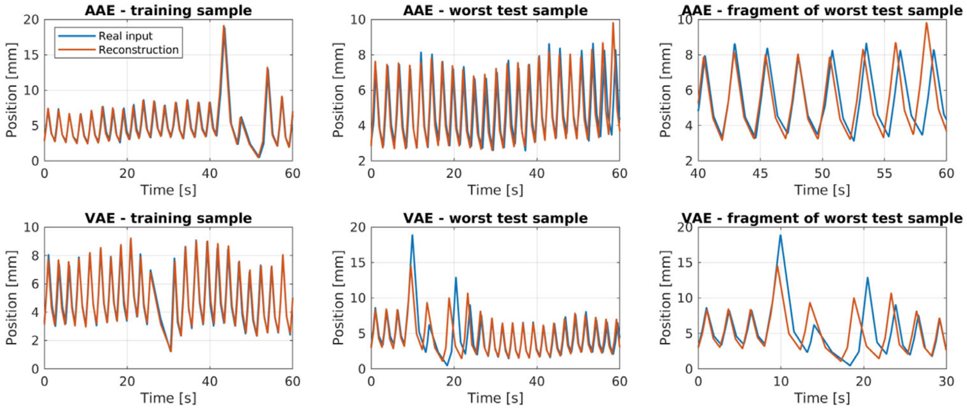


Fig. 4. (Top row) Reconstruction of breathing signals from AAE patient-specific models and (bottom row) VAE-based models for (left) a sample from the training set, (middle) the worst performing sample from the GUH test set, and (right) and a fragment of the worst reconstructed GUH test sample, with the highest reconstruction error. The discretized reconstructed signals are linearly interpolated and transformed back into a time series.

that the AAE encodings are more evenly distributed. This, together with the fact that the classifier in Fig. 3b struggles to distinguish real signals from samples in the vicinity of $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ hints that the latent space is more compact in the AAE-based models. On top of that, the AAE algorithm seems to be a more effective latent space regularizer, whose models have a latent space that closely resembles the prior distribution. This is deduced from Fig. 3d, where the distribution of the L2 norm of the encodings is compared to the distribution of the L2 norm of samples from the prior. The results suggest a possible relationship between more compact and similar to the prior AAE latent space and the lower classifier performance for AAE samples in Fig. 3b. Appendix D directly shows the distribution of the encodings, as well as a visualization of how data is organized in the low-dimensional latent space.

Semi-supervised baseline shift population models

We first evaluate the effect of slope, period and amplitude variations on the classification accuracy by using an artificial dataset based on sinusoidal signals. The SAAE models achieve a mF1-score of 100% in S1 by using as little as 300 labeled examples during the supervised classification phase. Adding period and amplitude variability to the sinusoidal signals in S2 results in additional difficulty, and the models need 1500 labeled examples (around 4% of the training data points) in order to achieve null classification error.

Based on these results, we train a baseline shift model using real data. The performance and added benefits of jointly classifying and modeling breathing signals are evaluated by assessing the classification accuracy, generation variability and the reconstruction error. The classification performance is assessed by comparing the SAAE models to purely discriminative models trained to only classify baseline shifts using a subset of the available labels. Specifically, a feed-forward (MLP) classifier and a convolutional (CNN) classifier were trained using 4% and 12% of the GUH training labeled data. Fig. 5 shows that our SAAE model with 5 to 15 latent variables outperforms both architectures, achieving a mean mF1-score of 94.91 and 96.54 on the unseen test EMC dataset when trained with 4% and 12% of the labels, respectively.

The generative performance and sample variability are evaluated with the CAS mF1-score. A CNN classifier is trained using 36,430 randomly generated samples from the SAAE model, which allows a fair comparison with the model trained using the real GUH data. The classifier is then evaluated on EMC data, achieving a remarkable 93.90 mF1-score for the model with 10 latent variables trained with 12% of the labels, which is on par with the performance of the feed-forward and CNN classifiers trained with real data observed in Fig. 5 (MLP and CNN in the two left plots). As

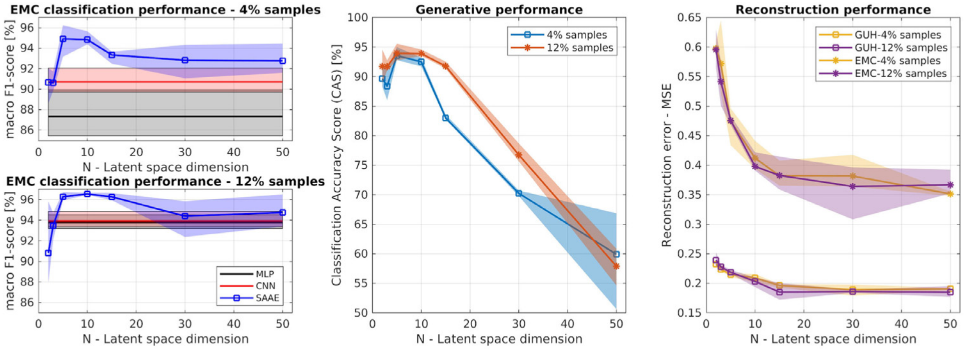


Fig. 5. Classification, generation and reconstruction performance of the SAAE semi-supervised models, for varying latent space dimension. The models use 4% or 12% of the training data during the supervised classification phase, which corresponds to 1500 and 6000 data points, respectively. We show the mean, maximum and minimum values observed from training 3 independent models with different training-test dataset splits and weight initialization. The relative reconstruction error is expressed as a percentage, where 100% corresponds to the maximum error corresponding to a model with randomly initialized weights.

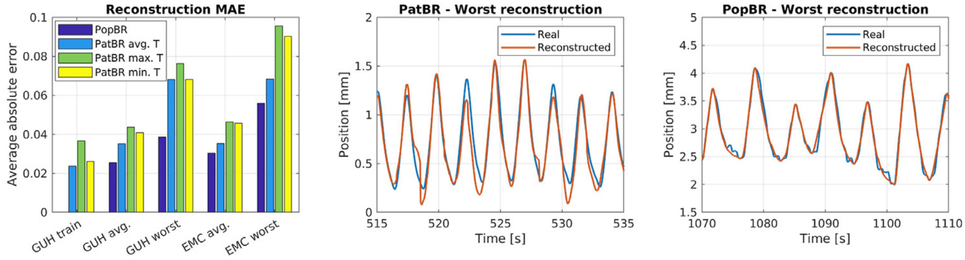


Fig. 6. (Left) Average absolute error achieved by the PatBR and PopBR models in the reconstruction of breathing time series. The error is shown for the training patient(s), the worst-performing patient and the entire set of patients present in each of the GUH and EMC datasets. (Middle) Reconstruction of the EMC signal fragment with highest error, using the PatBR model trained with data from the patient with maximum amplitude. (Right) Worst-performing reconstruction over all the EMC dataset using the PopBR model.

with the patient-specific models, the generative performance significantly degrades for higher latent space dimensionality.

Finally, the reconstruction error on test set data is shown in Fig. 5. As with the patient-specific models, the error is expressed relative to the maximum error corresponding to predictions from a randomly initialized model. The models perform similarly when using more than 10 latent variables. Higher latent space dimensionality seems to beneficial in the complicated task of reconstructing EMC samples that follow a different distribution, where the models achieve similar reconstruction performance to the feed-forward patient-specific models in Fig. 3a.

Time series reconstruction

Three different PatBR models are trained using the data from three patients: the patients with the largest and lowest breathing period in the dataset, and one of the patients with an average period. From these PatBR models, the former (largest period) achieves the largest error, precisely on signals of the patient with the lowest period. For each of the PatBR models and the PopBR model, a comparison of the average absolute error (average L1-norm) on the training set, the test set and the worst performing patient from the test set is shown in the left plot of Fig. 6. The average absolute error is calculated as the average L1-norm $|\boldsymbol{w}_{\text{real}} - \boldsymbol{w}_{\text{rec}}|$ between all position points in the recorded and reconstructed time series vectors $\boldsymbol{w}$. The middle and right plots in Fig. 6 show the worst EMC test sample reconstruction from the PatBR and PopBR models, respectively.

6. Discussion

Reconstruction accuracy and effect of convolutions

The standard VAE, standard AAE and SAAE architectures result in breathing models that capture the variability of respiration through few latent variables, as opposed to approaches that use implicit adversarial models [17,18]. The models are easy to sample and the decoders generate realistic breathing samples. The convolutional layers result in 25% reduction of the reconstruction error on test data. AAEs outperform standard VAE models in reconstruction, generalization and generative performance. Much of the AAE success seems to be related to their more compact latent space: their aggregated posterior distributions are closer to the prior, and their encodings are more evenly spaced, as seen in Fig. 3c and Fig. 3d. The problem of aggregated posterior-prior mismatch in VAEs is not new, and our findings support previous studies [35–37].

Effect of latent space dimensionality

For the set of all possible models, the reconstruction performance is in theory independent of the latent dimension. Very powerful autoencoders with deep encoders and decoders could perfectly reconstruct the input using as few as one latent dimension, but this is not observed in practice. In general, the performance can be practically improved by adding more latent variables or increasing the capacity of the model. However, it has been observed that very powerful decoder architectures tend to ignore the information encoded in $\boldsymbol{z}$ [38–40]. In concordance with Fig. 3, adding dimensions helps, especially in low-dimensional latent spaces. Nevertheless, there is a certain latent space dimensionality beyond which adding more latent units seems to add little information. For the VAE, this may manifest as ”inactive latent variables”, where some latent units remain equal to the prior distribution during the whole training process [41,42]. For the specific case of breathing and given the presented encoder and decoder convolutional architectures, the limit seems to be around 10 latent variables. This is supported by the fact that the test reconstruction error and classifier performance plateau around $N = 10$ in Fig. 3.

Semi-supervised models

Even though SAAE models are mainly trained to reconstruct breathing signals, they outperform pure discriminative architectures based on state-of-the-art one-dimensional convolutional models [7,8]. The fact that a single model can (better) classify and selectively sample types of signals is a novelty with respect to previous architectures that specialize in only one of such tasks [15,18]. One interesting remark is the fact that there seems to be a latent dimension range between 5 and 15 where SAAE models are superior in the classification task. In general, increasing the number of latent variables means that less information about the input is encoded per latent variable. We hypothesize that some of the information encoded in $\boldsymbol{y}$ may leak into the style variables $\boldsymbol{z}$ and cause loss of accuracy for increasing latent space dimensions. Models with a large enough number of latent variables would not benefit from the joint discriminative-generative modeling task, since they could completely encode the input using $\boldsymbol{z}$ and simply learn the label $\boldsymbol{y}$ separately. However, this should be confirmed in future research.

The generative performance of the SAAE models degrades with increasing latent dimensionality. As in the patient-specific models, we hypothesize that this is the result of an ”emptier” latent space with larger distance between encodings. This directly follows from the increasing volume of the multi-variate Gaussian latent space and the fixed number of samples used to cover such volume during training. Additionally, SAAE models perform similarly to the patient-specific models in terms of reconstruction and generalization on test samples from the same distribution, as indicated by the reconstruction error on GUH test samples. Although the reconstruction accuracy significantly decreases, the SAAE models also perform reasonably well in the much more complicated task of generalizing to test samples from the EMC dataset with different distribution, and their reconstruction error is on par with feed-forward patient-specific models (Fig. 3a). As in the patient-specific models, the SAAE models seem to benefit little from adding extra latent variables for latent space dimensionalities above 10. Since the classification and generative performance attain their maximum between 5 and 10 latent variables, we conclude that the optimum latent space dimension lays around 10.

Time series reconstruction accuracy

The PopBR reconstruction ANN consistently outperforms the single patient PatBR networks and opens the door to using a single model to reconstruct breathing signals for any patient. PatBR models fail to reconstruct time series from other patients, especially when they are evaluated on patients whose period significantly differs from that of the samples used for training, as seen in the left plot of Fig. 6. The generalization error of the PopBR model is very low and it provides accurate reconstructions for patients whose breathing signal was recorded in a different location and machine. The error could in principle be further decreased by training a specific PatBr for each specific patient, at the expense of slightly longer computation time.

Usefulness of breathing models

The models presented in this paper can be applied to a wide range of tasks involving signal generation and classification. Regarding generation, the models can be used to capture the variability in breathing of a patient and generate artificial patient samples. Our specific application is proton therapy, where a very narrow (1–3 mm) proton beam is used to actively scan the tumor. The movement of the beam and the breathing motion are on comparable time scales, leading to the so-called “interplay effect”, which can degrade therapeutic effectiveness. The presented generative framework presents significant advantages in addressing this problem compared to the commonly used simple sinusoidal artificial signals that fail to capture irregular motion and the true variability of the breathing. The realistic generated samples can be incorporated into treatment design in order to make treatments less sensitive to breathing motion during dose delivery. Since each generated breathing sample results in a different virtual delivered dose, repeated sampling allows deriving the distribution of plausible treatment outcomes, which can subsequently be used to assess treatment plan robustness before actual delivery or to directly optimize treatment plans to be robust against breathing movements. As a result, the desired clinical outcomes can be better ensured or the likelihood that a patient will present a certain type of breathing can be estimated - tasks that are infeasible with currently available methods.

The SAAE framework can in principle be applied for computer aided diagnosis of breathing abnormalities, as well as for dataset augmentation when the available data for a patient is scarce. An example is classifying breathing irregularities and generating additional samples that present the identified irregularity. One of the advantages of training the proposed framework in a semi-supervised way is the possibility to build such models requiring only a small subset of labeled data.

Our models can in principle be applied to any other kind of biomedical data that shows a repetitive or periodic structure, much like a breathing signal is composed of well-defined randomly varying periods with changing amplitude. To our knowledge, some of these signals could be ECG, electroglottograph (EGG), magnetoencephalography (MEG) or magnetocardiography (MCG). The added advantage of our generative approach with respect to other models in the literature that do not explicitly model the data distribution such as [15] or [16] is the possibility to map the data samples to specific regions or classes in latent space enabling classification and generation of data by sampling $\boldsymbol{z}$ from the desired regions.

Limitations

A notable drawback is the uninformative prior $p(\boldsymbol{y})$ in the semi-supervised model, which assumes no previous knowledge about the proportion between different classes. For cases when there is class imbalance, i.e., many more samples of regular breathing compared to irregular breathing, using such uninformative prior may result in the model miss-classifying some samples in order to match the uniform prior. The solution to this problem is dataset-dependent approach and involves determining the naturally occurring proportion of classes.

Computational cost

An important advantage of the presented methodology is the fact that it achieves feasible compute times. We reduce training times by using Graphics Processing Units (GPUs), which are needed to train the presented convolutional architectures due to the requirements of the latest version of the Tensorflow package [43]. We perform most of the training using an NVIDIA® Tesla® K80, and the training times vary around 10 minutes for the VAE and AAE patient-specfic models, 30 minutes for the reconstruction PopBR and PatBR models, and 20 minutes for the SAAE models. Generating and classifying breathing samples is almost instantaneous.

7. Conclusion

We present a semi-supervised algorithm based on the AAE that allows simultaneous classification and generation of biomedical signals within a single framework, using few labeled data points. The resulting models classify signals with greater accuracy than discriminative models specifically trained for classification; are easy to sample, and compress the data into a reduced latent space with few independent parameters with known probability distributions. We show that 10 of such latent variables are able to capture most of the variation in the data and achieve excellent reconstruction and generation of samples. For the particular case of breathing, we demonstrate that the adversarial objective used in AAEs is a better regularizer of the latent space and overcomes some of the previously studied problems of the VAE framework.

Given the length of the input time series, we train the models on compressed input vectors containing information about the period and amplitude of the biomedical signal. The compressed output vectors of the generative models can be transformed back into a time series with the help of an additional reconstruction network. We demonstrate that a reconstruction model trained with the data of a single patient (PatBR) does not achieve good generalization when evaluated on other patients, and it is outperformed by a population model (PopBR) trained with a subset of the data of a population of patients. The population model is trained only once and achieves great accuracy when applied to new unseen data. Even though we base our study on mechanical breathing signals, the framework shows potential applicability to simulation and diagnostic purposes using any other biomedical signal with a quasi-periodic structure.

8. Code availability

The code implementing training and evaluation of the AAE, VAE and SAAE, as well as the PopBR and PatBR reconstruction networks, is available at: https://bitbucket.org/zperko_TU/autoencodersforbiomedicalsignals.

Declaration of Competing Interest

The authors whose names are listed immediately below certify that they have NO affiliations with or involvement in any organization or entity with any financial interest (such as honoraria; educational grants; participation in speakers' bureaus; membership, employment, consultancies, stock ownership, or other equity interest; and expert testimony or patent-licensing arrangements), or non-financial interest (such as personal or professional relationships, affiliations, knowledge or beliefs) in the subject matter or materials discussed in this manuscript.

CRediT authorship contribution statement

Oscar Pastor-Serrano: Conceptualization, Data curation, Methodology, Investigation, Formal analysis, Visualization, Validation, Software, Writing – original draft. **Danny Lathouwers:** Supervision, Writing – review & editing. **Zoltán Perkó:** Conceptualization, Supervision, Methodology, Formal analysis, Funding acquisition, Project administration, Writing – review & editing.

Acknowledgements

This work is supported by KWF Kanker Bestrijding [grant number 11711], and part of the KWF research project PAREL. The authors are also thankful to Dr. Dennis Schaart for the interesting discussions regarding the manuscript, as well as Dr. Mischa Hoogeman and Dr. Steven Habraken for the insightful discussions and providing one of the datasets.

Appendix A. Evidence Lower Bound

Deriving the ELBO

Even though there are different ways to obtain the ELBO, the most common derivation is based on Jensen's inequality. For a concave function such as the natural logarithm the Jensen inequality states that

$$\log\left(\mathbb{E}[\boldsymbol{x}]\right) \geq \mathbb{E}\left[\log(\boldsymbol{x})\right].$$

Starting from the marginal likelihood of the probabilistic model, the expression of the ELBO can be obtained as

$$\begin{aligned}
\log\left(p_{\boldsymbol{\theta}}(\boldsymbol{x})\right) &= \log \int_{\mathcal{Z}} p_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{z}) d\boldsymbol{z} && \text{(A.1)}\\
&= \log \int_{\mathcal{Z}} p_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{z}) \frac{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} d\boldsymbol{z} && \text{(A.2)}\\
&= \log \mathbb{E}_{\mathbf{z}\sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}\left[\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{z})}{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}\right] && \text{(A.3)}\\
&\geq \mathbb{E}_{\mathbf{z}\sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}\left[\log\left(\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{z})}{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}\right)\right] && \text{(A.4)}\\
&= \mathbb{E}_{\mathbf{z}\sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}\left[\log\left(\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})\; p(\boldsymbol{z}))}{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}\right)\right] && \text{(A.5)}\\
&= \mathbb{E}_{\mathbf{z}\sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}[\log\; p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})] - D_{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z})), && \text{(A.6)}
\end{aligned}$$

where the KL-divergence D_{KL} is defined as

$$D_{KL}(p(x)||q(x)) = \int \log\left(\frac{p(x)}{q(x)}\right) p(x)\, dx = \mathbb{E}_{\mathrm{x}\sim p(x)} \log\left(\frac{p(x)}{q(x)}\right). \quad \text{(A.7)}$$

Dissecting the ELBO

The output of the probabilistic decoder is the likelihood conditional distribution $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$. This distribution is represented as a multivariate Gaussian probability distribution with identity covariance matrix $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}) = \mathcal{N}(\boldsymbol{x}; f_{\boldsymbol{\theta}}(\boldsymbol{z}), \boldsymbol{I})$, where the function $f_{\boldsymbol{\theta}}(z) : \mathcal{Z} \to \mathbb{R}^M$ is parametrized with an ANN and represents the mean. The log-likelihood is formulated as

$$\begin{aligned}
\log(p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})) &= \log\left(\frac{1}{\sqrt{(2\pi)^M|\boldsymbol{I}|}} \exp\left(-\frac{1}{2}(\boldsymbol{x} - f_{\boldsymbol{\theta}}(\boldsymbol{z}))^T \boldsymbol{I}^{-1}(\boldsymbol{x} - f_{\boldsymbol{\theta}}(\boldsymbol{z}))\right)\right)\\
&= C - \frac{1}{2}\|\boldsymbol{x} - f_{\boldsymbol{\theta}}(\boldsymbol{z})\|_2^2, && \text{(A.8)}
\end{aligned}$$

where C is a constant. The result has the same form as the squared error (SE), which is computed for the model output $\hat{\boldsymbol{x}}$ approximating the true output $\boldsymbol{x}$ as

$$\mathrm{SE} = \|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2^2. \quad \text{(A.9)}$$

Thus, minimizing the log-likelihood with respect to the parameters $\boldsymbol{\theta}$ (which is done by approximating the expectation $\mathbb{E}_{\mathbf{z}\sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} \log(p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}))$ by taking Monte Carlo samples for $\mathbf{z} \sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$) yields the same result as minimizing the SE. On the other hand, when p and q are both Gaussian distributions, the KL-divergence can be computed in closed form. In our case the prior is $p(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{z}; \boldsymbol{0}, \boldsymbol{I})$ and the encoder distribution is $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{z}; \boldsymbol{\mu}(\boldsymbol{x}), \operatorname{diag} \boldsymbol{\sigma}(\boldsymbol{x})^2)$. For an N-dimensional latent space, the KL-divergence can be analytically computed as:

$$D_{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z})) = \frac{1}{2}\left(-\sum_i^N(\log \sigma(\boldsymbol{x})_i^2 + 1) + \sum_i^N \sigma(\boldsymbol{x})_i^2 + \sum_i^N \mu(\boldsymbol{x})_i^2\right).$$

(A.10)

Note that the contribution of the KL-divergence to the ELBO scales linearly with the latent dimensionality, so an increase in the ELBO caused by an increase of the latent space dimensionality could in theory be compensated by increasing the variance of the approximated posterior $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ (lower KL-divergence per latent dimension).

Appendix B. Adversarial variational objective

AAEs do not exactly optimize the ELBO. This section describes the approximated variational objective in AAEs. In [25], the authors propose to regularize the latent space by introducing a discriminator model, modeled also with an ANN with mapping function $d_{\boldsymbol{\xi}}(\boldsymbol{z}) : \mathcal{Z} \rightarrow \mathbb{R}$ that outputs a single scalar logit. The discriminator is assumed to be capable of approximating any function. Given the encoder mapping $g_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}, \eta) : \mathcal{X} \times H \rightarrow \mathcal{Z}$, and the approximated posterior distribution $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) = \int_H \delta(\boldsymbol{z} - g_{\boldsymbol{\phi}}(\boldsymbol{x}, \eta))p(\eta)d\eta$, the adversarial regularization objective maximization can be formulated as

$$\max_{\boldsymbol{\xi}} \mathbb{E}_{\mathbf{z}\sim p(\boldsymbol{z})}[\log(S(d_{\boldsymbol{\xi}}(\boldsymbol{z})))] + \mathbb{E}_{\mathbf{x}\sim \hat{p}_{data}(\boldsymbol{x})}\mathbb{E}_{\mathbf{z}\sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}[\log(1 - S(d_{\boldsymbol{\xi}}(\boldsymbol{z})))] \tag{B.1}$$

$$= \max_{\boldsymbol{\xi}} \int p(\boldsymbol{z}) \log(S(d_{\boldsymbol{\xi}}(\boldsymbol{z})))d\boldsymbol{z} + \int\int \hat{p}_{data}(\boldsymbol{x})q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) \log(1 - S(d_{\boldsymbol{\xi}}(\boldsymbol{z})))d\boldsymbol{z}d\boldsymbol{x} \tag{B.2}$$

$$= \max_{\boldsymbol{\xi}} \int \left[p(\boldsymbol{z}) \log(S(d_{\boldsymbol{\xi}}(\boldsymbol{z}))) + \int \hat{p}_{data}(\boldsymbol{x})q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) \log(1 - S(d_{\boldsymbol{\xi}}(\boldsymbol{z})))d\boldsymbol{x}\right]d\boldsymbol{z}. \tag{B.3}$$

In the last step, we applied Fubini's theorem to change the order in the integration. As in [6] and [26], it can be shown that the discriminator achieves its optimum value at

$$d^*_{\boldsymbol{\xi}}(\boldsymbol{z}) = \log(p(\boldsymbol{z})) - \log\left(\int_{\mathcal{X}} q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})\hat{p}_{data}(\boldsymbol{x})d\boldsymbol{x}\right) = \log(p(\boldsymbol{z})) - \log(q_{\boldsymbol{\phi}}(\boldsymbol{z})). \tag{B.4}$$

This follows from the fact that for any $(a, b) \in \mathbb{R}^2 \setminus [0, 0]$, a function that has the form $f(h) = a\log h + b\log(1 - h)$ attains it maximum in [0,1] at $h = a/(a + b)$. Thus,

$$S(d^*_{\boldsymbol{\xi}}(\boldsymbol{z})) = \frac{p(\boldsymbol{z})}{p(\boldsymbol{z}) + \int_{\mathcal{X}} q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})\hat{p}_{data}(\boldsymbol{x})d\boldsymbol{x}}, \tag{B.5}$$

which is equivalent to Eq. B.4. The ELBO in Eq. 5 can be reformulated based on the definition of the KL divergence in Eq. A.7 as

$$\mathbb{E}_{\mathbf{x}\sim \hat{p}_{data}(\boldsymbol{x})}[\log(p_{\boldsymbol{\theta}}(\boldsymbol{x}))]$$
$$\geq \mathbb{E}_{\mathbf{x}\sim \hat{p}_{data}(\boldsymbol{x})}\mathbb{E}_{\mathbf{z}\sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}[\log(p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}))] - \mathbb{E}_{\mathbf{x}\sim \hat{p}_{data}(\boldsymbol{x})}[D_{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z}))] \tag{B.6}$$

$$= \mathbb{E}_{\mathbf{x}\sim \hat{p}_{data}(\boldsymbol{x})}\mathbb{E}_{\mathbf{z}\sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}[\log(p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}))]$$
$$+ \mathbb{E}_{\mathbf{x}\sim \hat{p}_{data}(\boldsymbol{x})}\mathbb{E}_{\mathbf{z}\sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}[\log(p(\boldsymbol{z})) - \log(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}))]. \tag{B.7}$$

As described in [25], the AAE algorithm replaces the last term in Eq. B.7 (regularization term, equivalent to the KL term) with ”an adversarial procedure that encourages $q_{\boldsymbol{\phi}}(\boldsymbol{z}))$ to match to the whole distribution of $p(\boldsymbol{z})$”. Mathematically, this translates into replacing the KL term with $\mathbb{E}_{\mathbf{x}\sim \hat{p}_{data}(\boldsymbol{x})}\mathbb{E}_{\mathbf{z}\sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}[d^*_{\boldsymbol{\xi}}(\boldsymbol{z})]$, effectively approximating the variational bound as

$$\mathbb{E}_{\mathbf{x}\sim \hat{p}_{data}(\boldsymbol{x})} \log\left(p_{\boldsymbol{\theta}}(\boldsymbol{x})\right) \geq \mathbb{E}_{\mathbf{x}\sim \hat{p}_{data}(\boldsymbol{x})}\mathbb{E}_{\mathbf{z}\sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}[\log(p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}))]$$
$$+ \mathbb{E}_{\mathbf{x}\sim \hat{p}_{data}(\boldsymbol{x})}\mathbb{E}_{\mathbf{z}\sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}[d^*_{\boldsymbol{\xi}}(\boldsymbol{z})] \tag{B.8}$$

$$= \mathbb{E}_{\mathbf{x}\sim \hat{p}_{data}(\boldsymbol{x})}\mathbb{E}_{\mathbf{z}\sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}[\log(p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}))]$$
$$- D_{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z})||p(\boldsymbol{z})), \tag{B.9}$$

where, compared to the ELBO in Eq. B.6, the term $\mathbb{E}_{\mathbf{x}\sim \hat{p}_{data}(\boldsymbol{x})}$ $[D_{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z}))]$ is approximated with $D_{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z})||p(\boldsymbol{z}))$. As a result, the AAE translates into a modified variational objective that does not preserve the original formulation.

Appendix C. Implementation details

VAE architecture

The architecture of the VAE models is shown in Fig. C.7. We find that using BatchNormalization [31] and Dropout [44] between layers significantly improves convergence and results in significantly better generalization. The encoder contains one-dimensional max. pooling layers and the decoder uses dilation rates bigger than 1, which seem to positively affect reconstruction performance. A β_n of 0.02 yields optimum balance between a Gaussian latent space that is closer to the prior and good reconstruction performance, with lower values slightly favoring more accurate reconstructions but aggregated posterior distributions with larger standard deviations that do not match the prior. We use a batch size of 256 samples and the Adam optimizer for training [45], with learning rate 10^{-4}.

Standard AAE architecture

The architecture of the different models composing the AAE is shown in Fig. C.7. For this framework, the order of the Batch Normalization and activation layers greatly affects convergence and stability during training, with Batch Normalization placed in between the activation and Dropout yielding the best results. Using Leaky ReLU activation functions with slope 0.1 in the discriminator also seems to help to stabilize training, in concordance with [46]. The models are trained using a batch size of 256 samples and the Adam optimizer with unequal learning rates: $2 \cdot 10^{-4}$ in the reconstruction phase and 10^{-4} for the discriminator. The squared error reconstruction loss is approximately 4 times lower than the cross-entropy loss used for the discriminator, and therefore multiplied by 4 during training.

Semi-supervised AAE architecture

Fig. C.7 shows the architecture of the encoder, decoder and discriminator models for the semi-supervised modified AAE architecture. We find that Batch Normalization between layers in the encoder and decoder significantly boosts performance and helps stabilize training, as well as using unequal learning rates for the Adam optimizer: 10^{-4} in the reconstruction and supervised classification phase and $2 \cdot 10^{-4}$ for the discriminator. The models are trained using a batch size of 256 samples. As with the standard AAE architecture, the cross-entropy loss is approximately 4 times higher than the reconstruction error, and so the latter is equalized during training. We find that α values of around 5–10 significantly enhance classification when the number of labels is limited, while higher values do not improve and even hinder performance.

Reconstruction network

The architectures for the PatBr and PopBR models are identical and are shown in Fig. C.7. The learning rate is set to 10^{-4} with a decay rate of 10^{-6} per epoch, and the batch size is 256 samples per batch.

Appendix D. Additional results

Latent space structure

To visualize how the latent space is structured, we train the AAE and VAE frameworks using a two-dimensional latent space. Fig. D.8a and Fig. D.8b show samples from a grid of equally spaced $\boldsymbol{z}$ in such latent space. The square grid is defined as 25 equally

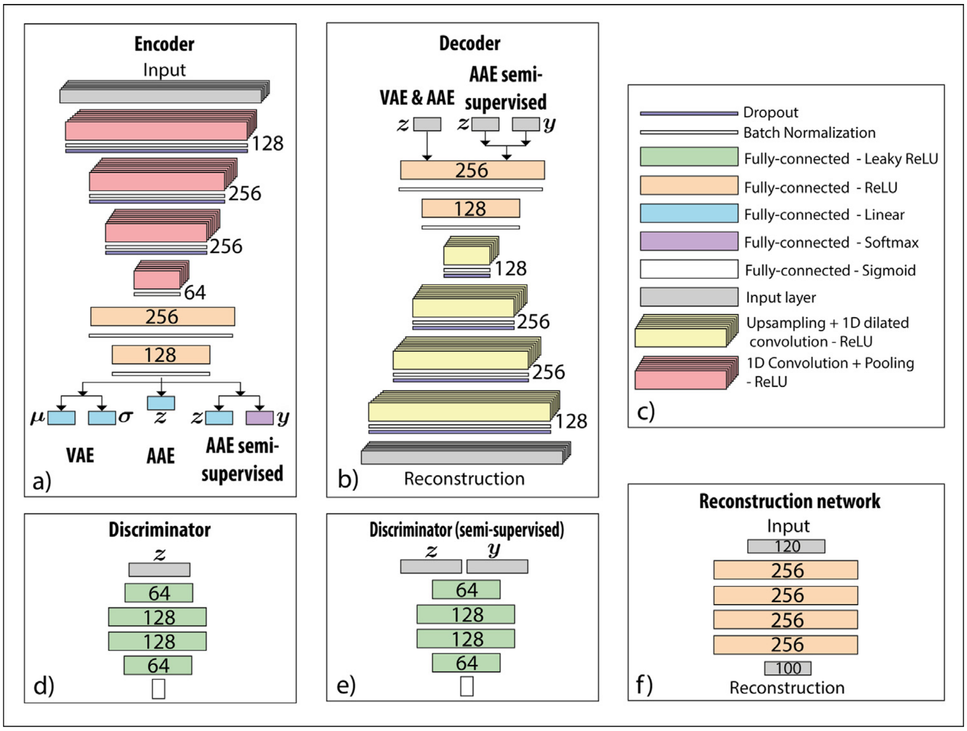


Fig. C1. Architecture of the different networks used in the AAE and VAE algorithms. (a) Convolutional encoder architecture with 4 one-dimensional convolutional layers and 2 fully-connected layers. A 1-D max-pooling layer follows each convolution, and Batch Normalization and Dropout with probability 0.1 are applied after each pooling layer. (b) Convolutional decoder architecture, with 2 fully-connected layers followed by 4 up-sampling dilated one-dimensional convolutional layers. Batch Normalization and Dropout with probability 0.3 follow each of the convolutions. (c) Color code for the layers used in the different models. (d) Discriminator architecture for the AAE, containing 4 fully-connected hidden layers followed by a sigmoid unit. (e) Discriminator for the SAAE. (f) Reconstruction network transforming the interpolated time series into realistic shapes.

spaced points between [-1.5, 1.5] in each of the two axis. According to the prior Gaussian distribution on the latent space, the samples in the center are more likely to be observed than the ones at the corners. Signals from nearby regions in the latent space show similar traits, such as the same type of irregularities or similar amplitudes and exhale positions.

To visualize any possible mismatch between the Gaussian prior and the aggregated posterior in the latent space, we plot the distribution of the encodings of all points in the dataset (i.e. the approximated aggregated posterior distribution). Fig. D.8c and Fig. D.8d show the distribution over a five-dimensional latent space for the AAE and VAE, respectively. The encodings of the AAE encoder are closely distributed to the prior Gaussian distribution.

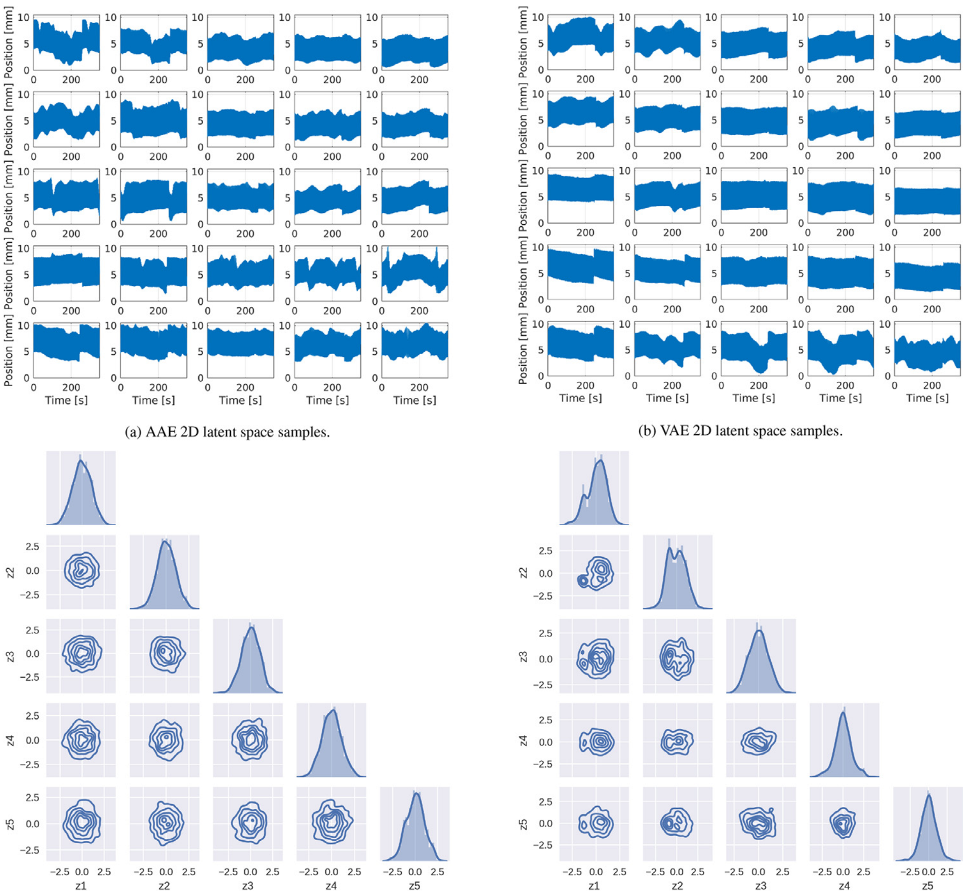
Semi-supervised population model

Using the SAAE framework, it is possible to train a population model that classifies and generates data from all the patients in the GUH dataset. The encoder classifies each signal into 15 classes corresponding to each of the patients in the dataset. The models are trained using a 80%-10%-10% train, validation and test set split. The population model can be used to classify and assign a new breathing sample to the most similar patient, and subsequently generate breathing samples from such patient. Fig. D.9 shows the classification performance using 300 and 600 labeled data points per class during the supervised classification training phase, which corresponds to approximately 12.5% and 25% of the labels in the dataset. For comparison, we plot the performance when the labels of all the data points are used during training. The dimensionality of the latent space and the classification head is set to 15 ($C = 15$, $N = 15$).

One of the main limitiations when training patient-specific models is the size of the dataset. Deep learning methods are data-driven and require a significant amount of different examples to achieve good generalization. The GUH dataset is formed by long breathing signals (in some cases multiple signals per patient) from which between 1200 and 5000 samples can be obtained for each patient. This is not generally the case for the data recorded in clinics on a regular basis, usually consisting of short breathing signals of few minutes, as is the case for the majority of the EMC dataset. This highlights the need for population models in the specific case of breathing.

Sampling the semi-supervised models

The SAAE models can generate breathing signals that present a certain type of irregularity or resemble breathing from a certain patient. First, a class $\boldsymbol{y}$ is obtained from the encoder or sampled from the categorical prior, and then the Gaussian sub-manifold representing breathing of that particular class is sampled according to the prior distribution $p(\boldsymbol{z})$. Fig. D.10a displays samples for each of the three classes in the baseline shift model trained using 12% of labels in the dataset, while Fig. D.10b shows samples from each patient in the population based model that is trained using 600 labeled examples per class.



(a) AAE 2D latent space samples.

(b) VAE 2D latent space samples.

(c) AAE encodings distribution.

(d) VAE encodings distribution.

Fig. D1. (a, b) Sampled signals corresponding to a grid of evenly spaced encodings in a two-dimensional latent space. The grid consists of 25 equally spaced points covering the squared region with corner coordinates (-1.5,-1.5), (-1.5,1.5), (1.5,-1.5), (1.5,1.5). (c,d) Distribution of the dataset encodings in a five-dimensional latent space.

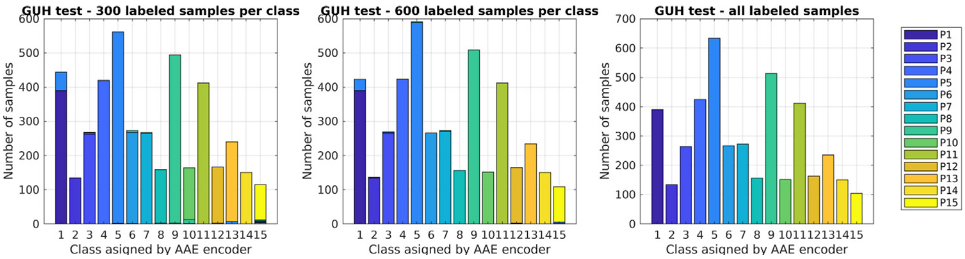
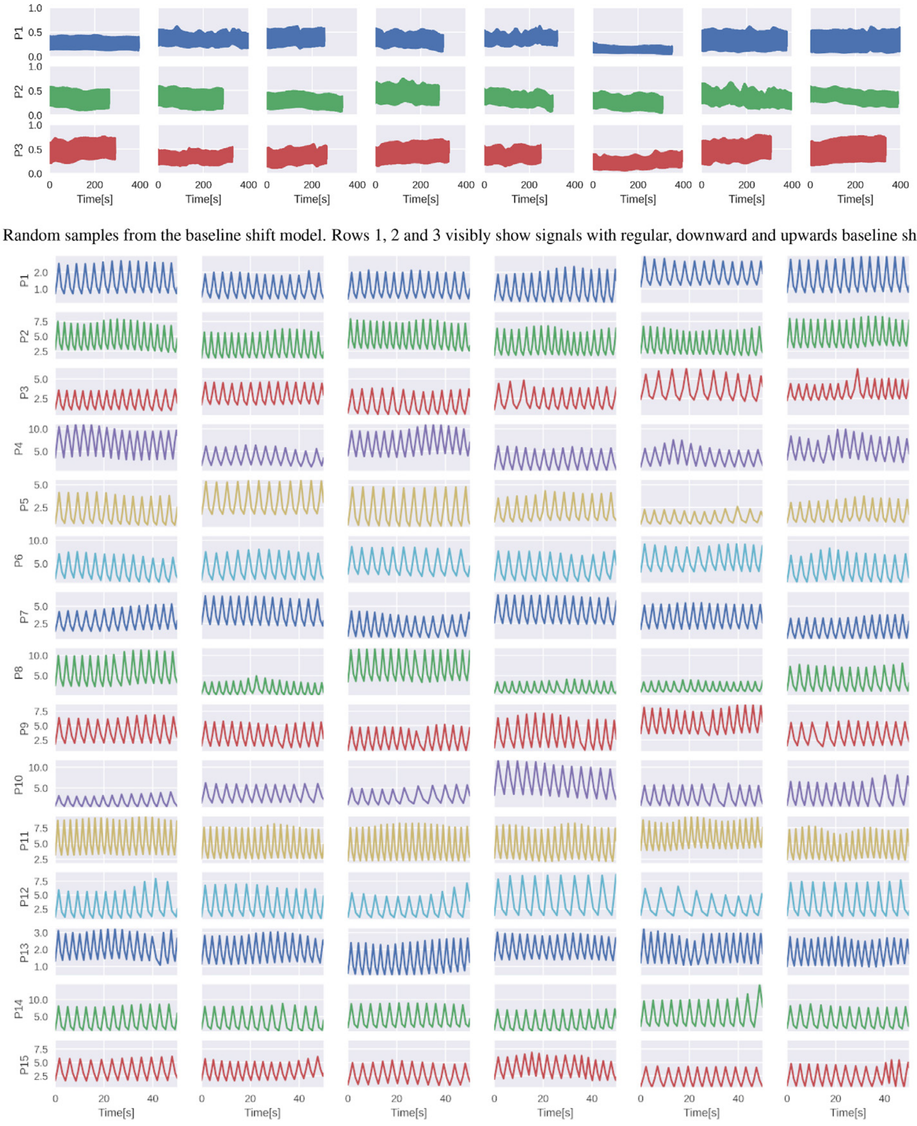


Fig. D2. Performance of the population SAAE classification head on the test GUH data, when using 300 and 600 labels per class, and all the available labeled samples during the supervised training step. The abscissa displays the label assigned by the encoder, while the legend shows the color code for the true labels. The color of each of the bars shows the true label of the samples assigned to a certain class by the encoder.



(a) Random samples from the baseline shift model. Rows 1, 2 and 3 visibly show signals with regular, downward and upwards baseline shifts.

(b) Random samples from the population model. Each of the rows represents a patient from the dataset.

Fig. D3. Randomly generated signals for each class $\boldsymbol{y}$ in the baseline shifts and the population model. Each row represents (a) a type of baseline shift — regular (C1), downwards baseline shifts (C2) and upwards baseline shifts (C3) — or (b) the patient in a cohort. For each class, different $\boldsymbol{z}$ values are independently sampled from the isotropic Gaussian distribution $\mathcal{N}(\boldsymbol{z}; \mathbf{0}, \boldsymbol{I})$. Note that amplitudes can sometimes notably differ (P8) and periods are usually similar within each patient (P11).

References

[1] M.L. Neal, R. Kerckhoffs, Current progress in patient-specific modeling, Brief. Bioinformatics 11 (1) (2010) 111–126.

[2] P.E. McSharry, G.D. Clifford, L. Tarassenko, L.A. Smith, A dynamical model for generating synthetic electrocardiogram signals, IEEE Trans. Biomed. Eng. 50 (3) (2003) 289–294, doi:10.1109/TBME.2003.808805.

[3] R. George, S.S. Vedam, T.D. Chung, V. Ramakrishnan, P.J. Keall, The application of the sinusoidal model to lung cancer patient respiratory motion, Med Phys 32 (9) (2005) 2850–2861, doi:10.1118/1.2001220.

[4] D.P. Kingma, M. Welling, Auto-encoding variational bayes, arXiv preprint (2013) ArXiv:1312.6114.

[5] D.J. Rezende, S. Mohamed, D. Wierstra, Stochastic Backpropagation and Approximate Inference in Deep Generative Models, In International conference on machine learning (pp. 1278–1286), PMLR, 2014.

[6] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative Adversarial Nets, In Advances in neural information processing systems (pp. 2672–2680), 2014.

[7] U.R. Acharya, H. Fujita, S.L. Oh, Y. Hagiwara, J.H. Tan, M. Adam, Application of deep convolutional neural network for automated detection of myocardial infarction using ECG signals, Inf Sci (Ny) 415 (2017) 190–198, doi:10.1016/j.ins.2017.06.027.

[8] U.R. Acharya, H. Fujita, O.S. Lih, Y. Hagiwara, J.H. Tan, M. Adam, Automated detection of arrhythmias using different intervals of tachycardia ECG segments with convolutional neural network, Inf Sci (Ny) 405 (2017) 81–90, doi:10.1016/j.ins.2017.04.012.

[9] H. Fujita, D. Cimr, Computer aided detection for fibrillations and flutters using deep convolutional neural network, Inf Sci (Ny) 486 (2019) 231–239, doi:10.1016/j.ins.2019.02.065.

[10] D. Cimr, F. Studnicka, H. Fujita, H. Tomaskova, R. Cimler, J. Kuhnova, J. Slegr, Computer aided detection of breathing disorder from ballistocardiography signal using convolutional neural network, Inf Sci (Ny) (2020), doi:10.1016/j.ins.2019.02.065.

[11] O. Yildirim, P. Plawiak, R.S. Tan, U.R. Acharya, Arrhythmia detection using deep convolutional neural network with long duration ECG signals, Comput. Biol. Med. 102 (2018) 411–420, doi:10.1016/j.compbiomed.2018.09.009.

[12] O. Yildirim, R. San Tan, U.R. Acharya, An efficient compression of ECG signals using deep convolutional autoencoders, Cogn Syst Res 52 (2018) 198–211, doi:10.1016/j.cogsys.2018.07.004.

[13] X. Chen, Z. Cheng, S. Wang, G. Lu, G. Xv, Q. Liu, X. Zhu, Atrial fibrillation detection based on multi-feature extraction and convolutional neural network for processing ECG signals, Comput Methods Programs Biomed (2021) 106009, doi:10.1016/j.cmpb.2021.106009.

[14] O. Yildirim, U.B. Baloglu, R.S. Tan, E.J. Ciaccio, U.R. Acharya, A new approach for arrhythmia classification using deep coded features and LSTM networks, Comput Methods Programs Biomed 176 (2019) 121–133, doi:10.1016/j.cmpb.2019.05.004.

[15] F. Zhu, F. Ye, Y. Fu, Q. Liu, B. Shen, Electrocardiogram generation with a bidirectional LSTM-CNN generative adversarial network, Sci Rep 9 (1) (2019) 1–11.

[16] A.M. Delaney, E. Brophy, T.E. Ward, Synthesis of realistic ECG using generative adversarial networks, arXiv preprint (2019). ArXiv: 1909.09150

[17] T. Golany, K. Radinsky, PGANS: personalized generative adversarial networks for ECG synthesis to improve patient-specific deep ECG classification, In Proceedings of the AAAI Conference on Artificial Intelligence 33 (2019) 557–564, doi:10.1609/aaai.v33i01.3301557.

[18] N. Wulan, W. Wang, P. Sun, K. Wang, Y. Xia, H. Zhang, Generating electrocardiogram signals by deep learning, Neurocomputing (2020), doi:10.1016/j.neucom.2020.04.076.

[19] C. Xiao, E. Choi, J. Sun, Opportunities and challenges in developing deep learning models using electronic health records data: a systematic review, Journal of the American Medical Informatics Association 25 (10) (2018) 1419–1428, doi:10.1093/jamia/ocy068.

[20] S. Hong, Y. Zhou, J. Shang, C. Xiao, J. Sun, Opportunities and challenges of deep learning methods for electrocardiogram data: a systematic review, Comput. Biol. Med. (2020) 103801.

[21] S. Takao, N. Miyamoto, T. Matsuura, R. Onimaru, N. Katoh, T. Inoue, S. Shimizu, Intrafractional baseline shift or drift of lung tumor motion during gated radiation therapy with a real-time tumor-tracking system, International Journal of Radiation Oncology* Biology* Physics 94 (1) (2016) 172–180, doi:10.1016/j.ijrobp.2015.09.024.

[22] M. Abreu, A. Fred, J. Valente, C. Wang, H.P. da Silva, Morphological autoencoders for apnea detection in respiratory gating radiotherapy, Comput Methods Programs Biomed 195 (2020) 105675, doi:10.1016/j.cmpb.2020.105675.

[23] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, A. Lerchner, beta-vae: Learning basic visual concepts with a constrained variational framework, 2016.

[24] D.P. Kingma, M. Welling, An introduction to variational autoencoders, arXiv preprint (2019). ArXiv: 1906.02691

[25] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, B. Frey, Adversarial autoencoders, arXiv preprint (2015). ArXiv: 1511.05644

[26] L. Mescheder, S. Nowozin, A. Geiger, Adversarial variational bayes: unifying variational autoencoders and generative adversarial networks, arXiv preprint (2017). ArXiv: 1701.04722

[27] E. Jang, S. Gu, B. Poole, Categorical reparameterization with gumbel-softmax, arXiv preprint (2016). ArXiv: 1611.01144

[28] C.J. Maddison, A. Mnih, Y.W. Teh, The concrete distribution: a continuous relaxation of discrete random variables, arXiv preprint (2016). ArXiv: 1611.00712

[29] E. Coste-Manière, D. Olender, W. Kilby, R.A. Schulz, Robotic whole body stereotactic radiosurgery: clinical advantages of the cyberknife integrated system, Int J Med Robot 1 (2) (2005) 28–39, doi:10.1002/rcs.39.

[30] F. Ernst, Compensating for quasi-periodic motion in robotic radiosurgery, Springer Science & Business Media (2011), doi:10.1007/978-1-4614-1912-9.

[31] S. Ioffe, C. Szegedy, Batch normalization: accelerating deep network training by reducing internal covariate shift, arXiv preprint (2015). ArXiv: 1502.03167

[32] C.J. van Rijsbergen, Information retrieval, Butterworths, London, 1979.

[33] S. Ravuri, O. Vinyals, Classification accuracy score for conditional generative models, arXiv preprint arXiv:1905.10887 (2019). ArXiv: 1905.10887

[34] A. Razavi, A.V.D. Oord, O. Vinyals, Generating diverse high-fidelity images with vq-vae-2, arXiv preprint arXiv:1906.00446 (2019). ArXiv: 1906.00446

[35] D.J. Rezende, F. Viola, Taming vaes, arXiv preprint arXiv:1810.00597 (2018). ArXiv: 1810.00597

[36] M. Rosca, B. Lakshminarayanan, S. Mohamed, Distribution matching in variational inference, arXiv preprint (2018). ArXiv: 1802.06847

[37] B. Dai, D. Wipf, Diagnosing and enhancing VAE models, arXiv preprint (2019). ArXiv: 1903.05789

[38] S.R. Bowman, L. Vilnis, O. Vinyals, A.M. Dai, R. Jozefowicz, S. Bengio, Generating sentences from a continuous space, arXiv preprint arXiv:1511.06349 (2015). ArXiv: 1511.06349

[39] S. Zhao, J. Song, S. Ermon, Infovae: information maximizing variational autoencoders, arXiv preprint (2017). ArXiv: 1706.02262

[40] X. Chen, D.P. Kingma, T. Salimans, Y. Duan, P. Dhariwal, J. Schulman, I. Sutskever, P. Abbeel, Variational lossy autoencoder, arXiv preprint (2016). ArXiv: 1611.02731

[41] Y. Burda, R. Grosse, R. Salakhutdinov, Importance weighted autoencoders, arXiv preprint arXiv:1509.00519 (2015). ArXiv: 1509.00519

[42] C.K. Sønderby, T. Raiko, L. Maaløe, S.K. Sønderby, O. Winther, Ladder variational autoencoders, In Advances in neural information processing systems (2016) 3738–3746.

[43] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, Tensorflow: A system for large-scale machine learning, 2016. In 12th USENIX symposium on operating systems design and implementation (OSDI 16), 265–283

[44] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, The journal of machine learning research 15 (1) (2014) 1929–1958. http://jmlr.org/papers/v15/srivastava14a.html

[45] D.P. Kingma, J. Ba, Adam: a method for stochastic optimization, arXiv preprint (2014). ArXiv: 1412.6980

[46] A. Radford, L. Metz, S. Chintala, Unsupervised representation learning with deep convolutional generative adversarial networks, arXiv preprint (2015). ArXiv: 1511.06434