Time Series Anomaly Detection with Adversarial Reconstruction Networks

Shenghua Liu*, Bin Zhou *, Quan Ding *, Bryan Hooi†, Xueqi Cheng* and Zhengbo Zhang‡
*Institute of Computing Technology, Chinese Academy of Sciences
†School of Computer Science, National University of Singapore
‡Center for Artificial Intelligence in Medicine, Chinese PLA General Hospital

Abstract—Time series data naturally exist in many domains including medical data analysis, infrastructure sensor monitoring, and motion tracking. However, a very small portion of anomalous time series can be observed, comparing to the whole data. Most existing approaches are based on the supervised classification model requiring representative labels for anomaly class(es), which is challenging in real-world problems. So can we learn how to detect anomalous time ticks in an effective yet efficient way, given mostly normal time series data? Therefore, we propose an unsupervised reconstruction model named BeatGAN which learns to detect anomalies based on normal data, or data which majority of samples are normal. BeatGAN provides a framework to adversarially learn to reconstruct, which can cooperate with both 1-d CNN and RNN. Rarely observed anomalies can result in larger reconstruction errors, which are then detected based on extreme value theory. Moreover, data augmentation with dynamic time warping regularizes reconstruction and provides robustness. In the experiments, effectiveness and sensitivity are studied in both synthetic data and various real-world time series. BeatGAN achieves better accuracy and fast inference.

Index Terms—Time series, Adversarial Reconstruction Networks, Anomaly Detection, Data Augmentation

1 Introduction

Time series is an important type of data object, which has the characteristics of high dimensions, large data volume, and real-time streaming, and it is widely used in various applications. For example, in the medical field, the patient's ECG data and respiratory data are recorded through medical equipment. In the industrial field, sensors of various devices collect state data in real time. In the financial field, exchanges record stock price changes every moment which form time series data. These massive time series data contain various information, and some of them are different from the general mode of the data, and their characteristics are obviously different from most data. We call such data "anomalous data". Although the proportion of anomalous data is very small, it often contains more valuable information and has important research value in various fields. For example, in the medical field, patients have a large amount of monitoring data such as electrocardiogram, blood pressure, respiration rate, etc. Using these data to detect the abnormal state of the patient and early warning is the primary goal of modern medical treatment. So, it is very significant to propose a method detecting anomalous time series segments in large-scale rhythmic time series data.

Another key goal in our research is explainability. In practice, experts in medical and other domains need to know not only whether an anomaly exists but also to understand the mechanism of anomalies. This leads to the following questions:

How can we automatically detect anomalous beats when monitoring multivariate time series? Can we pinpoint the anomalous time ticks that led to our decision?

The traditional anomaly detection algorithm faces the shortcomings of low efficiency and shallow accuracy in today's massive and complex data, and cannot model the characteristics of normal samples precisely. Besides, there are several challenges in time series anomaly detection: 1) massive time series can contain few anomalies, which is insufficient and imbalanced for supervised classification; 2) anomalous segments can be very different from one another, e.g. some anomalous heartbeats are never seen in defined categories; 3) even in healthy patients, the time periods involved in various heartbeat characteristics (e.g. P waves, QRS complex, P-R, and R-R intervals) vary from one beat to another.

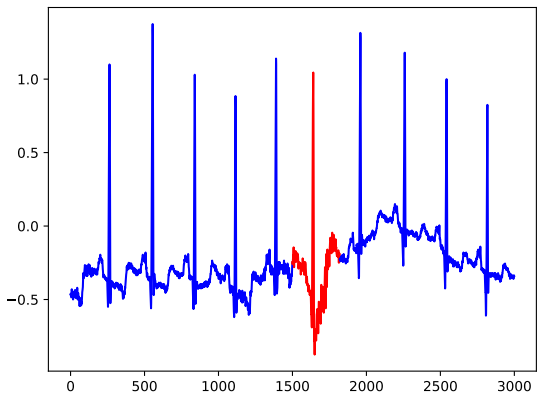


Fig. 1: ECG with anomalies.

Different from supervised methods, unsupervised anomaly detection methods are trained only on *normal* samples, and then used to identify *abnormal* samples. Among many anomaly detection methods, reconstruction-based anomaly detection techniques are widely used. We expect to obtain the true nature of the data by reconstructing them with their lower-dimensional representation, then the reconstruction error can be measured as the anomaly score

Corresponding to Shenghua Liu, liushenghua@ict.ac.cn. Shenghua Liu, Bin Zhou, Quan Ding, and Xueqi Cheng are also with CAS Key Laboratory of Network Data Science & Technology, and University of Chinese Academy of Sciences.

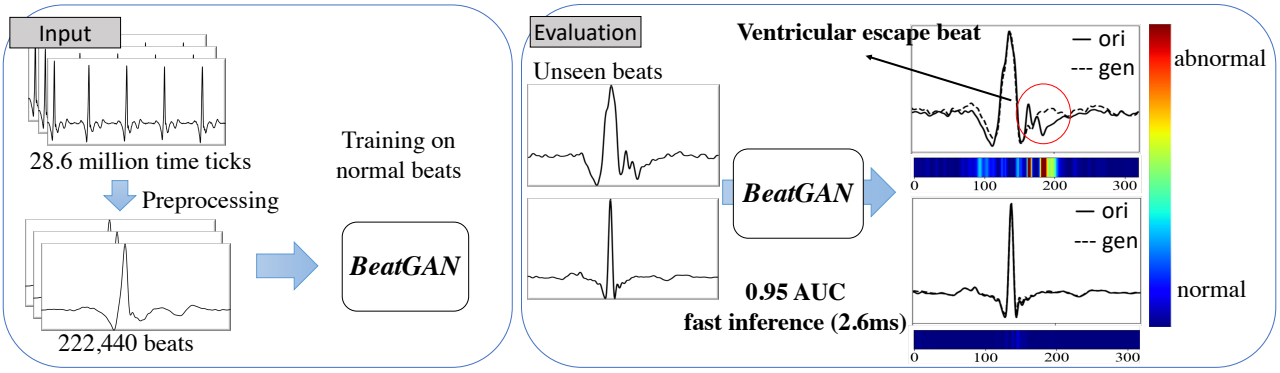


Fig. 2: BeatGAN successfully detects anomalous rhythms, and explains the results. The size of training input is 28.6 million time ticks, and inference can be as fast as 2.6 ms per beat. The original beat is shown by solid lines, and the generated beat is shown by dashed lines.

for given samples. [1] uses reconstruction error between true and latent representation to detect anomalies by the modified SVD decomposition. Autoencoder (AE) [2] allows for more complex patterns by applying nonlinear functions for reconstruction and anomaly detection. However, without proper regularization, such a reconstruction easily leads to overfitting, resulting in low accuracy.

Generative adversarial networks (GANs) is a successful generative model that jointly learns to generate realistic synthetic data while learning a discriminator [3]. Some works utilize both generator and discriminator [4] for anomaly detection and it provides an intuitive way to regularize the reconstruction error.

Therefore, we propose an anomaly detection model, *BeatGAN*, which detects anomalies using adversarially generated time series as shown in Fig 2. The model additionally provides explainable results, pinpointing the time ticks that led to our decision, rather than just an overall anomalousness score.

BeatGAN reconstructs the data robustly and performs regularization using an adversarial generation approach. The generation is fulfilled by reconstruction and regularization of jointly optimizing hidden feature matching and discrimination in a generative adversarial network. Pairwise feature matching loss and the reconstruction loss are used to tune model parameters. To further improve its accuracy, we exploit the characteristics of rhythmic time series by designing a warping method to augment training data in our proposed BeatGAN method. Experiments show that BeatGAN detects anomalies accurately in different types of time series like ECG data from the MIT-BIH arrhythmia database, multivariate time series frim SWaT database, and sensor time series data from the CMU Motion Capture database.

In summary, our main contributions are as follows:

- **Anomaly detection by observing massive normal time series**: We propose to use a reconstruction-error method with the generative adversary network, named BeatGAN to detect anomalous time series, by comparing the difference between generated time series and the original one. Taking advantage of adversarial regularization, BeatGAN is robust. Moreover, time series warping is proposed for data augmentation to improve detection accuracy.
- **Effectiveness**: BeatGAN outperforms existing state-of-the-art methods in identifying anomalies in ECG and SWaT time series, achieving an accuracy of nearly 0.95/0.82 AUC, and very fast inference (2.6 ms per beat).
- **Explainability**: BeatGAN is capable of pinpointing where the anomalous patterns occur, outputting interpretable results by visualization and attention routing (see Fig 2).
- **Generality**: BeatGAN can successfully detect anomalies from the different multivariate sensor time series databases.
- **Reproducibility**: BeatGAN is open-sourced [1].

The rest of the paper is organized as follows. Section 2 summarizes the related works about anomaly detection. In section 3, we describe the key components of our BeatGAN framework, and the corresponding optimization algorithm. In section 4, we design our experiments with several datasets and analysis the results. Section 5 presents the conclusions.

2 Related Work

Time series mining and anomaly detection methods can be categorized into four categories.

2.1 Statistics-based Methods

Statistical-based methods mainly identify abnormal points that deviate from the normal distribution. A classical statistics-based method is 3-sigma criterion [5], which assumes that the data follows a normal distribution, and then the probability that the data is within the 3-sigma range $(\mu - 3\sigma, \mu + 3\sigma)$ is 99.74%. Then the data that falls outside this interval is regarded as anomalous data. Besides, the Grubbs' test [6] is a statistical-based method as well. It also assumes the data follows a normal distribution and can output the confidence of each test sample. These methods require a prior assumption while real world data may not satisfy. So some works use the extreme value theory to detect the anomalies in univariable time series [7], [8]. This method doesn't need to assume the prior distribution of the data, but it still can not handle the multivariable time series well.

2.2 Classification-based Methods

Supervised classification approaches require a large amount of labeled data, and either manually defined features or hidden variables learned from deep models. If given enough labeled data, these deep learning based approaches can achieve high accuracy [9], [10], [11]. However, the labeled data is usually difficult to obtain in practice [12]. Furthermore, it has difficulty generalizing to anomalies which differ significantly from those it has seen,

1. https://github.com/BGT-M/spartan2-tutorials/blob/master/BeatGAN.ipynb

e.g. when new types of anomalies appear. Based on these defects, Scholkopf proposed One-Class SVM [13], an unsupervised model which learns from normal data to identify anomalies on unseen data. It works by separating the origin from the data instances in a kernel space, corresponding to nonlinear regions describing the normal data in feature space. Each instance's anomalousness score is measured by its normalized distance to the determined decision boundary.

	PCA/SVD	OCSVM	DAGMM	USAD	AnoGAN	Ganomaly	LSTM-AD	MadGAN	MSCRED	BeatGAN
Non-linear		✓	✓	✓	✓	✓	✓	✓	✓	✓
Explains anomalies	✓		✓	✓	✓			✓	✓	✓
Robust	✓			✓	✓	✓	✓	✓	✓	✓
Fast inference	✓	✓	✓	✓		✓	✓		✓	✓
Hyper-parameter free	✓	✓								✓

TABLE 1: Comparison of related approaches .

2.3 Vocabulary-based Methods

Vocabulary-based methods learn a set of vocabularies (patterns) of the given time series. The rare vocabularies are regarded as anomalies. Hidden Markov Models (HMMs) [14] are classic vocabulary-based method. Variants include DynaMMo [15] which uses Dynamic Bayesian Networks and AutoPlait [16] which uses two-level HMMs. Recent work [17] proposed a vocabulary approach named BEATLEX which performs segmentation and forecasting by optimizing minimum description length (MDL). This model automatically segments the given long time series and summarizes the different patterns as vocabularies, these vocabularies are regarded as the basis of the time series and the rare variables are detected as anomalies. And this model can segment long time series more properly and we can get meaningful sub-sequences. In addition, Keogh et al. proposed a series of time series data mining algorithms. Matrix Profile [18], [19] is the most famous algorithm among them and it can effectively cluster the given time series and find out the anomalous patterns. NormA [20] elaborately selects some of sequences and build normal behavior models based on MDL principal.

2.4 Reconstruction-based Methods

Anomalies can be defined as events that deviate significantly from the patterns observed in real data [21]. Thus, many works detect anomalies by computing a synthetic reconstruction of the data and then measuring the deviation between an observed instance and its reconstruction. Principal Component Analysis (PCA) and Singular Value Decomposition (SVD) is the linear reconstruction model. The low-rank decomposition makes the model pays more attention to the "normal" part of the data and the anomalous part is ignored in reconstructed data. Autoencoders can also be used for deep-learning based anomaly detection by inspecting its reconstruction error. And it is more suitable for complex data because of the power of deep learning. Jinwon used autoencoders (AE) and variational autoencoders (VAE) for anomaly detection on several benchmark datasets [2]. Dan also proposed a variational autoencoder based reconstruction model to learn the key features of normal time series, and detect the anomalies using the reconstructed time series [22], [23]. The MSCRED model [24] proposes an autoencoder to reconstruct signature matrix to jointly consider the multi-scale time dependence, robustness and explainability. The USAD [25] utilizes adversely trained autoencoders to learn data features in an unsupervised way while achieving a high performance.

Recently, with the growing interest in generative adversarial networks, researchers have proposed anomaly detection using adversarial training. AnoGAN [4] and Ganomaly [26] are both originally proposed for anomaly detection on visual data, while ours is designed for a series of real numbers that need robustness against speed variations. AnoGAN needs to learn a latent vector for every input for anomaly detection, which is very time consuming and limits its application. Ganomaly uses an encoder-decoder-encoder structure, and identify the anomalies by comparing the latent representations. Zenati uses the BiGAN model which has the ability of both generating and inferencing to detect anomalies and achieves good performance [27]. Inspired by AnoGAN and LSTM-based generative adversarial network [28], Li proposes the MAD-GAN model for industrial time series anomaly detection, the model consists of an LSTM-based generator and LSTM-based discriminator. Our BeatGAN uses data reconstructions, resulting in a more concise model and better performance for time series data. Meanwhile, BeatGAN can give explainable results with such a reconstruction.

Other approaches include [29], [30], [31], which proposed tensor decomposition based methods for time series forecasting and detecting anomalies based on the forecast. The Deep Autoencoder Gaussian Mixture Model(DAGMM) [32] combines a deep autoencoder and a gaussian mixture model to model the distribution of time series data. Series2Graph [33] represents subsequences in low-dimensional embeddings and identifies anomalies with varying lengths.

BeatGAN provides an explainable approach combining autoencoders and generative adversarial networks, incorporating the advantages of both models. Table 1 summarizes existing works related to our problem. Only BeatGAN satisfies all the desired characteristics.

3 PROPOSED MODEL

Let $\mathcal{T} \in \mathbb{R}^{M \times N}$ be a multivariate time series, which is N time ticks in length, and has M dimensions for each time tick, e.g. reading from M sources. A rhythmic time series $\mathcal{T}$ contains subseries, i.e. beats. For example, a beat in the ECG time series consists of a sequence of a P wave followed by a QRS complex, T, and U waves. We fix the window size for a beat in time series, and beats are denoted as $x \in \mathbb{R}^{M \times L}$, where L is large enough for containing a beat. Zero-padding or sampling can be used for fitting irregular beats in such a window if we know the exact length of each beat.

Most beats in time series are normal in practice, and the amount of beats is massive. Our anomaly detection problem can then be described as follows:

Informal Problem 1 (Anomalous beat detection). **Given** a collection of multivariate time series beats $\mathcal{X} = \{x_i, i = 1, 2, \dots, \}$ with most of beats in the normal class,

- **Detect** anomalous beats x in a collection of unseen time series,
- **Such that** they deviate significantly from the reconstructed time series, and can pinpoint anomalous time ticks in x for explanation.

3.1 General Framework

The idea of our model is to detect anomalies by reconstructing their "normal" state. In general, the framework for detecting anomalies based on reconstruction error will have two main components: reconstruction (or generation) model optimization, and anomalousness scoring based on the reconstruction.

The optimization objective for learning the reconstruction model is: for input time series, our model reconstructs a more "normal" time series, the anomaly score can be calculated by comparing the residuals between original time series and reconstructed time series. The general reconstruction-based model's loss function can be formulated as:

$$\begin{aligned} L &= ||X - G(X)||_2 + R(G) \\ &= \sum_x ||x - G(x)||_2 + R(G) \end{aligned} \tag{1}$$

where X is a matrix concatenating each beat matrix $x \in \mathcal{X}$ along its columns. $G(\cdot)$ is the reconstructed model, and $R(G)$ is the regularization term for different models.

And the anomalousness score for a beat x is calculated as:

$$A(x) = ||x - G(x)||_2 \tag{2}$$

which summarizes the distance of each time tick between original beat and reconstructed beat. After calculating anomalousness score for each dimension, we select the top-5 dimension with the maximum score as potential root cause to direct users' attention.

Many reconstruction-based anomaly detection methods can be expressed by this framework, using Eq.(1) as objective and Eq.(2) to calculate the anomalousness score. We list the specific forms of reconstruction-based anomaly detection methods including SVD, AE, VAE, and our BeatGAN in Table 2, The $G(\cdot)$ means the reconstruction function and the $R(G)$ is the regularization loss for reconstruction model optimization. The SVD-based methods reconstruct data using low-rank approximation. u_i and v_i are the i-th columns of U and V respectively obtained by singular value decomposition. Moreover, FBOX uses the same reconstruction model as SVD, but uses a different anomalousness score which applies a threshold based on percentiles of the reconstructed distribution.

The AE-based and VAE-based methods, and BeatGAN are all neural network based model. They reconstruct data using an encoder network $G_E(\cdot)$ and decoder network $G_D(\cdot)$. While AE-based methods do not have any explicit regularization, VAE-based methods use the KL divergence between the approximate posterior distribution $Q(z|x)$ learned by the encoder, and the prior distribution $P(z)$ of the latent variable z often uses Gaussian distribution. While our BeatGAN uses adversarial regularization in its training process.

We will show in the following how BeatGAN regularizes its reconstruction, taking the benefits from generative adversarial networks (GAN).

Fig 2 shows the framework of our proposed method. In this example, we preprocess the ECG data and train the model with normal heartbeats. At test time, for each unseen beat x, we feed it into the trained model and obtain the reconstructed beat x' (showed by dashed lines in Fig. 2). By comparing the residuals between x and x', we capture the anomalies.

Method	$G(\cdot)$	$R(G)$
SVD/ FBOX	$\sum_{k=1}^{p} \sigma_k u_k v_k^T$	$\lambda_1 \left\|\left\|I - U^T U\right\|\right\|_2 + \lambda_2 \left\|\left\|I - V^T V\right\|\right\|_2$
AE	$G_D(G_E(x))$	/
VAE	$G_D(G_E(x))$	$\lambda D_{KL}[Q(z\|x)\|\|P(z)]$
BeatGAN	$G_D(G_E(x))$	adversarial regularization

TABLE 2: Unifying the reconstruction-based methods for anomaly detection. $G(\cdot)$ is the reconstruction function, $R(G)$ is the regularization loss.

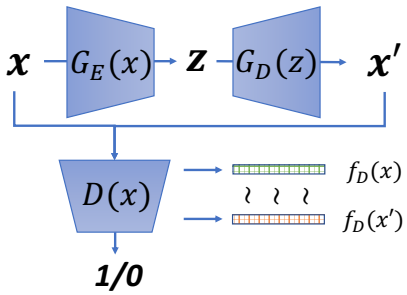


Fig. 3: Illustration of BeatGAN framework.

3.2 Proposed Model with Adversarial Regularization

As Fig 3 shows, our adversarial reconstruction framework has three key components: encoder $G_E(\cdot)$, decoder $G_D(\cdot)$, and discriminator $D(\cdot)$.

To reconstruct time series x, $G_E(x)$ encodes the input x to a hidden vector z that represents its important features. Then $G_D(z)$ generates a time series x' from the hidden vector z. Thus our reconstruction goal is $G(x) = G_D(G_E(x))$. Discriminator $D(\cdot)$ regularizes the reconstruction by minimizing the classification error.

In the framework, we can generally use the same network structure from a broad range of neural networks. We implement two alternative structures: the one-dimensional convolutional neural network (1-D CNN), and recurrent neural networks (RNN).

1-D CNN: The CNN network is effective in learning good local features (as well as the combinations of them) [34]. So it is more suitable for some medical time series since these time series always contain some sub-patterns [9]. In our proposed model BeatGAN, we use filters in CNN to slide in one dimension along the temporal dimension. The structure of 1-D CNN implemented in encoder is illustrated in Fig 4.

RNN networks: Recurrent neural network behaves well on learning the trend of the time series which can be used to handle smoothed time series. We choose the GRU cell as the realization of RNN in our proposed BeatGAN, because GRU cell can avoid the gradient vanishing problem and is more effective than LSTM. GRU cell has two gates: update gate and the reset gate. The cell compute the last hidden state h_t by Eq. (3)

$$\begin{aligned} z_t &= \sigma(W_z \cdot [h_{t-1}; x_t]) \\ r_t &= \sigma(W_r \cdot [h_{t-1}; x_t]) \\ \hat{h_t} &= tanh(W \cdot [r_t * h_{t-1}+; x_t]) \\ h_t &= (1 - z_t) * h_{t-1} + z_t * \hat{h_t} \end{aligned} \tag{3}$$

In terms of regularization, we use the adversarial training framework of GAN in our model. In the GAN framework, the

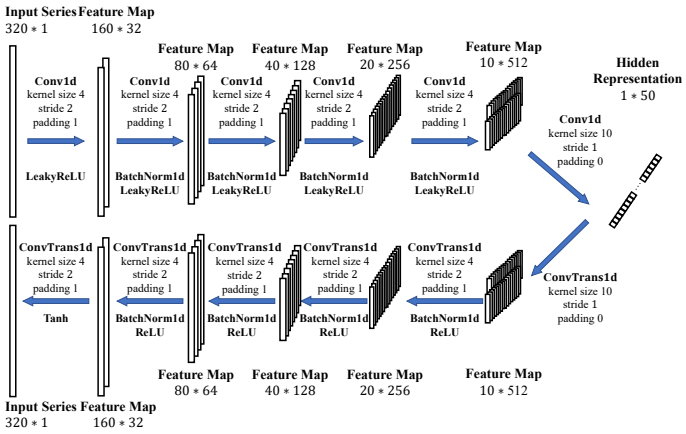


Fig. 4: Illustration of CNN structure in reconstructor (encoder-decoder).

generator and the discriminator compete in a two-player min-max game. The discriminator tries to distinguish real beats from reconstructed beats and the generator tries to generate beats that can fool the discriminator. The generator learns the generating mechanism of data by this adversarial training. In the training process, the discriminator $D(\cdot)$ tries to maximize the loss function:

$$L_D = \mathbb{E}_{x\sim P_r}[\log D(x)] + \mathbb{E}_{z\sim P_z}[\log(1 - D(G(z)))] \quad (4)$$

which discriminates the generated x' from x as different class label 0 and 1. The generator G tries to minimize the following loss function

$$L_G = \mathbb{E}_{z\sim P_z}[\log(1 - D(G(z)))] \quad (5)$$

which makes the generated beats unable to be discriminated by $D(\cdot)$.

Actually, the essence of generating adversarial networks is to optimize the JS divergence of distribution between real data and the generated data. We get the ideal discriminator D^* when the gradient of L_D equal to 0.

$$D^*(x) = \frac{P_r(x)}{P_r(x) + P_g(x)} \quad (6)$$

While $P_g(x)$ is the distribution of generated data. And after we use D^* in L_G, we can get the objective function of the generator

$$\begin{aligned} L_G &= \mathbb{E}_{x\sim P_r}[\log D(x)] + \mathbb{E}_{x\sim P_g}[\log(1 - D(x))] \\ &= \mathbb{E}_{x\sim P_r}[\log \frac{P_r}{P_r(x) + P_g(x)}] + \mathbb{E}_{x\sim P_g}[\log \frac{P_g(x)}{P_r(x) + P_g(x)}] \\ &= JS(P_r||P_g) - 2\log 2 \end{aligned} \quad (7)$$

In practice, directly using Eq. (5) as adversarial regularization does not perform well due to the diminished gradient and mode collapse, which is also the weakness of the JS divergence. Thus, instead of calculating loss by the original x and vector z in hidden space, we set up the relationship between the original x and reconstructed x' via our reconstruction part of the model. Therefore, we consider using pairwise feature matching loss which minimizes differences of the statistics between original and generated time series, learned in hidden layers of the discriminator $D(\cdot)$. Letting $f_D(\cdot)$ be the activation vector on a hidden layer of the discriminator, pairwise feature matching loss between x and x' is:

$$L_{pfm} = ||f_D(x) - f_D(x')||_2 \quad (8)$$

Overall, the objective of reconstruction with adversarial regularization is to minimize the following loss function:

$$L_G = ||x - x'||_2 + \lambda ||f_D(x) - f_D(x')||_2 \quad (9)$$

where $x' = G(x)$, and λ is the weighting parameter adjusting the impact of the adversarial regularization. Meanwhile, the objective of the discriminator is to maximize the following loss function:

$$L_D = \frac{1}{N}\sum_{i}^{N}[\log D(x_i) + \log(1 - D(x_i'))] \quad (10)$$

The discriminator network and the adversarial training process are the biggest differences compared to autoencoders, the adversarial regularization ensures that the reconstruction part will not just compress and decompress the input data.

Finally, we use the Adam optimization algorithm [35] for learning the reconstruction model, as summarized in Alg 1.

Algorithm 1 Training algorithm.

```
θ_G, θ_D ← initialize network parameters
for number of training iterations do
    Sample {x_1, ..., x_m} ~ a batch from the normal data
    Generate {x'_1, ..., x'_m} by G_E and G_D
    Compute L_D by Eq (10)
    θ_D ⟵ θ_D + α∇_{θ_D}(L_D)        // ∇ is the gradient
    Compute L_G by Eq (9)
    θ_G ⟵ θ_G + α∇_{θ_G}(L_G)
end for
```

To perform anomaly detection, we use the reconstruction model to reconstruct a time series x' using our model trained on normal data, for given x. We then evaluate the anomalousness score by comparing the difference between x' and x as in Eq (2). Since anomalies always occur in a portion of time ticks of a beat, the residuals between ticks of x and x' can indicate where the anomaly occurs, routing users' attention to the anomalous portion and providing an explanation.

3.3 Automatic Threshold Selection

During the online anomaly detection, given a multivariate time series with length L, our BeatGAN model computes the anomaly score for each time tick. So the anomaly scores form a univariate time series $\{a_1, a_2, ..., a_L\}$, our goal is to automatically select the anomalous time ticks. Then the problem can be formed as follows:

Informal Problem 2 (Automatic anomaly detection). **Given** a collection of observed normal univariate time series of anomaly score $\{a_1^o, a_2^o, ..., a_n^o\}$ and a collection of test univariate time series of anomaly score $\{a_1^u, a_2^u, ..., a_m^u\}$

- **Determine** whether each time tick in test time series is anomalous (0-normal/1-anomalous)

Here, we use the principle of Extreme Value Theory (EVT) to set a threshold for time tick. Extreme value theory is used to deal with the maximum deviation from the median of probability distribution. So it is often used in the analysis of rare cases (such as once in a century flood and risk analysis in finance). It is more

general than other theory because it doesn't need any assumption for data distribution. In EVT, the most widely used is POT (Peaks Over Threshold) theorem, it uses the GPD (Generalized Pareto Distribution) to estimate the extremum (value greater than a certain threshold) distribution. Like Fig 5 shows, we will use GPD (red dash line) to fit the data larger than threshold.

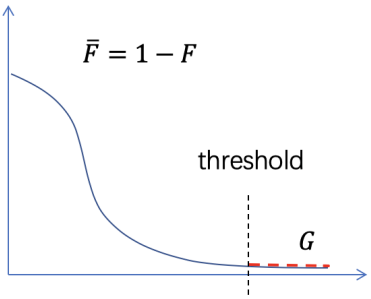


Fig. 5: Extremum distribution fitting.

The idea of POT is to fit the tail portion of the data distribution with GPD with parameter σ, γ [36]

$$f(x) = \frac{1}{\sigma}(1 + \frac{\gamma x}{\sigma})^{-\frac{1}{\gamma}-1}$$

For parameter σ and γ, we can estimate the value by Maximum Likelihood Estimation(MLE), then we can set the threshold with another risk parameter q,

$$z = t + \frac{\sigma}{\gamma}((\frac{qn}{N_t})^{-\gamma} - 1) \tag{11}$$

The automatic threshold selection algorithm is summarized in Alg 2, it is also a streaming algorithm. The algorithm will only store the data that is larger than the given threshold, so it only occupies little memory. For each streaming data, the algorithm will compare it with the current anomaly threshold, if it is larger than the anomaly threshold, this data will be stored in the anomalies set. Otherwise, the algorithm will re-estimate the current Generalized Pareto distribution and compute another anomaly threshold. So in this algorithm, the anomaly thresholds change over time.

3.4 Data Augmentation Using Time Warping

For the time series similarity metric, Euclidean Distance is mainly used in many applications since it is simple and effective. While there is another special similarity metric, named dynamic time warping (DTW) [37]. DTW can compare the "shape" of two time series more accurately and it measures similarity in a way that is more robust against variations in speed (i.e. 'time warping'). For example, heartbeats naturally and slightly speed up or slow down. However, since DTW is not differentiable, we cannot directly use it for reconstruction error in objective Eq (9). [38] propsed a variation of DTW called SoftDTW which can be used to calculate the similarity of time series. We use such method to replace the Euclidean Distance to calculate the loss of reconstruction as:

$$L_G = SoftDTW(x, x') + \lambda \cdot SoftDTW(f_D(x), f_D(x')) \tag{12}$$

Furthermore, we propose a modified time warping for data augmentation to make our model robust against natural variability involving time warping in real time series. We augment our training data as follows. For each training beat x, we sample uniformly at random a small number k of time ticks to "slow down" and a different k time ticks to "speed up" to keep the fixed length of time series. For each time tick to "speed up", we will directly delete the data value at that time tick. While for each time tick to "slow down", we will insert a new data value just before that time tick, whose value is set to the average of the data values at the 2 adjacent time ticks. This results in a modified version of x, which we use as additional training data for our model.

Algorithm 2 Anomaly Detection.

Input: parameter q
$A \leftarrow \emptyset$
$th \leftarrow$ initial threshold
$Y = \{a_i^o - th | a_i^o > th\}$
$\gamma, \sigma \leftarrow MLE(Y)$
$z_q = CalThreshold(q, \gamma, \sigma, k, |Y|, th)$ // by Eq.(11)
$k \leftarrow n$
for $i > 0$ **do**
  if $a_i^u > z_q$ **then**
    Add(a_i^u) in A
  else if $a_i^u > th$ **then**
    Add i in Y
    $\gamma, \sigma \leftarrow MLE(Y)$
    $z_q = CalThreshold(q, \gamma, \sigma, k, |Y|, th)$ // by Eq.(11)
  else
    $k \leftarrow k + 1$
  end if
end for

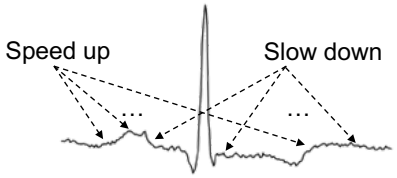


Fig. 6: Data Augmentation.

4 Experiment

In this section, we will design different experiments to answer the following questions:

Q1. Accuracy: How accurately does our method find out anomalies compared with other baselines?

Q2. Robustness: How robust can our model achieve with data augmentation?

Q3. Explainability: How well does BeatGAN pinpoint anomalous portions of input time series, and route people's attention?

Q4. Efficiency: How fast does BeatGAN test on samples compare with other GAN-based baselines?

Q5. Sensitivity: How sensitive does BeatGAN for amplitude and frequency changes? What is the performance of BeatGAN on controlled synthetic time series?

Method	MIT-BIH ECG		SWaT	
	Best-F1 score	AUC score	Best-F1 score	AUC score
PCA	0.7087 ± 0.0047	0.8164 ± 0.0037	0.7731 ± 0.0011	$\mathbf{0.8222 \pm 0.0023}$
OCSVM	0.6759 ± 0.0019	0.7917 ± 0.0018	0.2761 ± 0.0042	0.6336 ± 0.0031
DAGMM	0.6769 ± 0.0005	0.6999 ± 0.0002	$\mathbf{0.7848 \pm 0.0012}$	0.7324 ± 0.0001
USAD	0.3837 ± 0.0001	0.3521 ± 0.0001	0.6365 ± 0.0002	0.7901 ± 0.0002
AnoGAN	0.7091 ± 0.0074	0.8642 ± 0.0100	0.7556 ± 0.0023	0.7792 ± 0.0021
Ganomaly	0.7708 ± 0.0187	0.9083 ± 0.0122	0.7557 ± 0.0025	0.8115 ± 0.0090
LSTMAD	0.7105 ± 0.0013	0.5960 ± 0.0008	0.7632 ± 0.0039	0.8148 ± 0.0044
MadGAN	$--$	$--$	0.77	$--$
MSCRED	0.7783 ± 0.0000	0.6273 ± 0.0002	0.5960 ± 0.0029	0.7466 ± 0.0003
BeatGAN(CNN)	$\mathbf{0.8159 \pm 0.0102}$	$\mathbf{0.9447 \pm 0.0053}$	0.7814 ± 0.0020	$\mathbf{0.8192 \pm 0.0031}$
BeatGAN(RNN)	0.6910 ± 0.0233	0.8361 ± 0.0307	$\mathbf{0.7838 \pm 0.0023}$	0.8185 ± 0.0033

TABLE 3: Results of MIT-BIH ECG dataset and SWaT dataset.

Dataset	dimension	size (time tick)	anomalous ratio
MIT-BIH	1	28.6 M	11.1
SWaT	25	946 K	5.8
CMU Motion	4	10.3 K	- -

anomalous ratio is the percentage of anomalous data in total data

TABLE 4: Description of the datasets.

4.1 Data

We evaluate our proposed model on ECG time series from MIT-BIH Arrhythmia Database[2] [39], sensor time series from SWaT database[3]. And we use motion capture data from the CMU motion capture database[4] to do case study to show the explainability of our method.

MIT-BIH ECG dataset. The MIT-BIH arrhythmia dataset contains 48 ECG records from test subjects from Beth Israel Hospital. The ground-truth labels are annotated on the R-peak of each beat by two or more independent cardiologists indicating positions and types of each heartbeat. As recommended by the AAMI [40], the normal beats include the beats annotated with labels N, L, and R[5], and the records named 102, 104, 107, and 218 are removed due to insufficient signal quality. In total, the dataset contains 97,568 beats and 28.6 million time ticks. In experiments, we augment the data and normalize them between -1 and 1 by min-max scaling.

SWaT dataset The secure water treatment system(SWaT) dataset [41] is collected from a six-stage Secure Water Treatment system. The testbed is designed for representing a large water refreshing plant which can refresh 5 gallons water per minute in modern cities. Based on this testbed, we can simulate two state modes: normal and abnormal. A total of 25 dimensions which record readings of sensors are regarded as input dimensions. Generally, this dataset is divided into train and test sections. The former section contains 496,800 samples collected in the first several days with no anomalies at all, while 449,919 samples are recorded in the latter section to simulate anomalies. In experiments, we normalize each time series x between -1 and 1 by min-max scaling.

CMU Motion Capture dataset. The dataset contains motion-captured subjects performing different motions (walking, jogging, running, etc.). We choose 4 dimensions from different sensors on the subject's body, i.e. left-right arms and legs. We select 16 walking records of 6,616 ticks in total, 10 jogging records of 1,608 ticks in total, and 1 jumping record of 2,085 ticks for training and testing separately. Thus we obtain 10,309 time ticks in total. In experiments, we normalize each time series x between -1 and 1 by min-max scaling.

2. https://physionet.org/content/mitdb/1.0.0/
3. https://itrust.sutd.edu.sg/itrust-labs_datasets/dataset_info/
4. http://mocap.cs.cmu.edu/
5. N is Normal beat, L is Left bundle branch block beat and R is Right bundle branch block beat

4.2 Accuracy

BeatGAN calculates the anomalousness score for each time tick of a beat. Given an evaluation set $\mathcal{Z}$, BeatGAN calculates the score of beat s_i in different ways for different models and forms a score set S, i.e. $\mathcal{S} = \{s_i : A(x_i), x_i \in \mathcal{Z}\}$. For the ECG set, we average the scores of each time tick as score of the beat. For SWaT dataset, we treat the last time tick as the anomalousness score of the whole "beat". To calculate metrics, we first standardize the scores between 0 and 1 by min-max scaling.

$$s_i' = \frac{s_i - min(\mathcal{S})}{max(\mathcal{S}) - min(\mathcal{S})}$$

Then we calculate the two metrics, AUC (Area Under ROC Curve) and Best F1 score. For AUC metric, we first compute the ROC curve for the model and AUC is calculated as the area under the curve. For Best F1 score, we will enumerate each threshold of scores in test dataset and compute the corresponding F1 score. Then we select the highest F1 score among them as our Best F1 score metric.

We compare the performance with PCA-based method, one-class SVM (OCSVM) [13] for anomalous class (using the top 50 features selected by PCA method). DAGMM [32] is a density based model which combine the autoencoder and gaussain mixture model. USAD [25] uses autoencoder structure with adversarial learning to isolate anomalies. AnoGAN [4] is a deep convolutional generative adversarial network and Ganomaly [26] utilizes a conditional generative adversarial network to reconstruct. Besides, we choose LSTMAD and MAD-GAN as baselines of BeatGAN(RNN). LSTMAD is an LSTM-based AutoEncoder [42]. The model MAD-GAN uses the framework of AnoGAN, and replaces the underlying convolutional network with LSTM network. In table 3, we directly use the result of SWaT reported in this paper [43]. Finally, we also compare with MSCRED [24], which employs a convolutional encoder-decoder and attention-based ConvLSTM network to perform anomaly detection task.

4.2.1 Evaluation on ECG Dataset

Experimental setup. We first use a filter [44] to remove the noise in ECG sequences. We choose 320 time ticks as the window size for a beat: 140 time ticks before the given R-peak and 180 ticks after it. We set the dimension size of latent space as 50, $\lambda = 1.0$ for objective (9). The structure of G_D learns

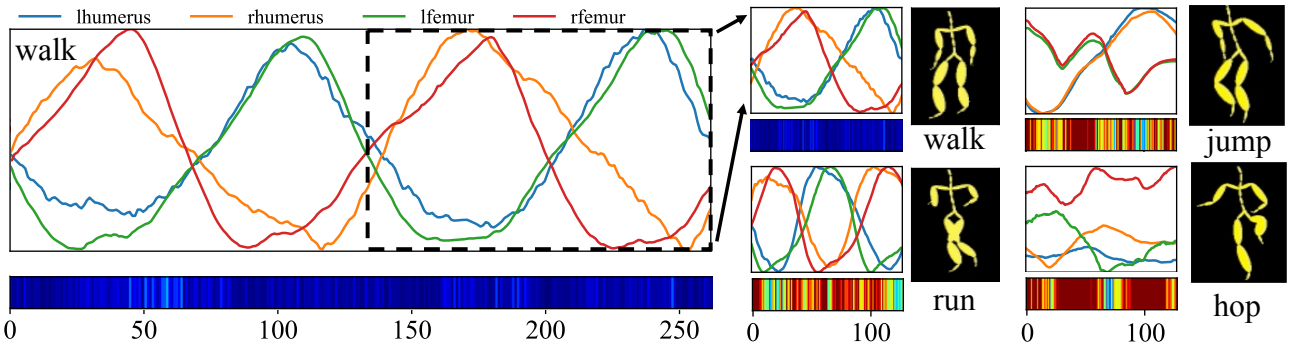


Fig. 7: Example of anomaly detection on motion capture time series(4-dimensions). The right side shows the original time series and heatmaps to pinpoint the anomalies of jumping/running/hopping from walking motions.

the architecture of the generator from DCGAN [45]. We use 5 1D transposed convolutional layers followed by batch-norm and leaky ReLU activation, with slope of the leak set to 0.2. The transposed convolutional kernel's size and number of each layer are 512(10/1)-256(4/2)-128(4/2)-64(4/2)-32(4/2): e.g. 512(10/1) means that the number of filters is 512, the size of filter is 10 and the stride is 1. G_E's structure is a mirrored version of G_D and D has the same architectural details as G_E, which is shown in Fig. 4. We also design the RNN-based BeatGAN model, we choose GRU as the underlying cell, the hidden units are set as 50 and the number of layer is 1. We use Adam optimizer with an initial learning rate $lr = 0.0001$, and momentums $\beta_1 = 0.5$, $\beta_2 = 0.999$. Moreover, we use 5-fold cross-validation for each method, and report the averaged metrics and standard deviations (std).

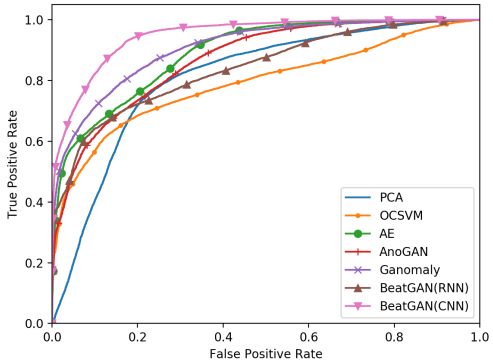


Fig. 8: Roc curve of different model for MIT-BIH dataset.

Result. As shown in Table 3, the averaged results and std of the non-linear methods generally have better performance in both AUC and Best F1 as compared to PCA and OCSVM, providing evidence that non-linear models have advantages on the complex ECG time series. Besides, The results show that BeatGAN(CNN) perform the best among the baselines (p-value<0.01). From the Table 3 and Fig 8, we can also notice that the BeatGAN(RNN) perform much worse than BeatGAN(CNN), the reason maybe that the convolutional neural network is more effective in learning good local features (as well as the combinations of them) [34] since the ECG time series contains many patterns, like P waves, QRS complex and T waves. And so the 1-D CNN is widely used in many ECG time series [9], [46]

We also design the experiment to verify the performance of the automatic threshold selection algorithm. Here we select the $3-\sigma$ rule as the baseline, the $3-\sigma$ rule detects the anomalies as following:

- assume the distribution of the anomaly score is Gaussian.
- compute the parameters the the Gaussian distribution μ and σ
- for each test anomaly score, if it is higher than $\mu + 3\sigma$, then regards it as anomaly

For MIT-BIH dataset, we set the parameter $q = 0.04$ and the initial threshold th as 98% percentile. The result is shown in Fig 9, we can find out that our algorithm can achieve the better performance than baseline, and the final F1 score is only little lower than the best F1 score, which means it is suitable for the real-world task.

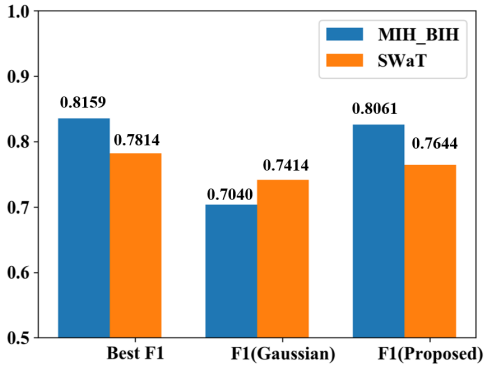


Fig. 9: Result of automatic threshold selection algorithm.

4.2.2 Evaluation on SWaT Dataset

Experimental setup. For SWaT dataset, since the anomalies only occur at the end of the time series, we naturally split the training part and the testing part (the data collected during the first seven days is training data and the data collected during the last four days is testing data). We first down-sample to original time series to one measurement every 5 seconds, then normalize the multivariate time series to $[-1, 1]$ by min-max scale. We slide a window of 60 time ticks along the original multivariate time series to generate the segments, the stride size is 1 tick. Since the annotated label is located at each time tick, we use the reconstruction error of last time tick as the anomaly score for this "beat". The dimension size of latent space is 20 and $\lambda = 0.1$. We use both the 1-D

CNN based model and RNN-based model to do the experiments. For decoder G_D of 1-D CNN based BeatGAN model, we use 4 1D transposed convolutional layers followed by batch-norm and leaky ReLU activation, with a slope of the leak set to 0.2. The transposed convolutional kernel's size and thr number of each layer are 256(4/1)-128(4/2)-64(4/2)-32(4/2): e.g. 256(4/1) means that the number of filters is 256, the size of the filter is 4 and the stride is 1. G_E's structure is a mirrored version of G_D and D has the same architectural details as G_E. As for RNN-based BeatGAN model, we also choose the GRU cell since it is more concise and effective, the hidden units are set as 20 and the number of layer is 1. The learning rate of thr Adam optimizer is 0.001, we test each model with 5 runs and give the mean and std. The result is listed in Table 3

Result. In the experiment of SWaT dataset, we also compare different baselines. From Table 3, we can find out that BeatGAN model achieves a high best F1 score, outperforming other baselines except the DAGMM. Because of the more simple pattern in SWaT dataset, the PCA model can also achieve a high AUC and Best F1, and both CNN-based BeatGAN model and RNN-based BeatGAN model have good performance. The performance of One Class SVM is worst and the reason may be that the dimension of the time series is too high. While other models can have good performance in this anomaly detection task.

Then we also test the performance of the automatic threshold selection algorithm compared to the baseline: $3-\sigma$ rule. For SWaT dataset, we set the parameter $q = 0.1$ and the initial threshold th as 98% percentile. The result is shown in Fig 9, we can also find that our algorithm performs better than the baseline, and it proves the algorithm is suitable for general tasks.

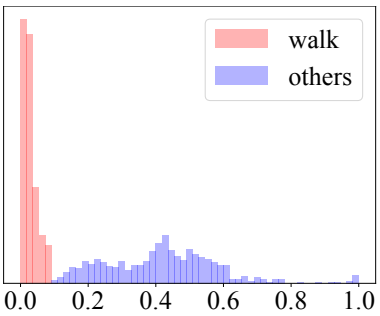


Fig. 10: Normalized anomalousness score distributions.

4.2.3 *Case Study on Motion Capture Dataset*

Experimental setup. In this experiment, walking is considered as our normal class. We evaluate BeatGAN on detecting unusual motions of jogging and jumping time series. We slide a window of 64 time ticks along the original multivariate time series to generate beats, and the stride size for the sliding window is 5 ticks. Thus we obtain 1,729 beats, with 10,309 time ticks in total, some of which overlap. Since the data is sparse and small, we test the performance of the RNN-based BeatGAN model for the study case. We use the GRU cell with 10 neural units and 1 layer. The dimension size of latent space is 10 and $\lambda = 0.01$.

Result. Fig 10 shows the histogram of normalized anomalousness scores on evaluation data. The results show that the score distributions of walking and others are clearly separated (red represents the anomalousness scores of walk and blue represents others), which means our BeatGAN can perfectly discriminate between unusual motions (jogging/jumping) and usual motions (walking), by only using time series of walking for training.

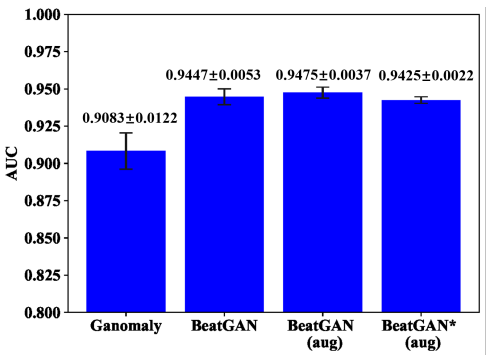


Fig. 11: Performance with data augmentation.

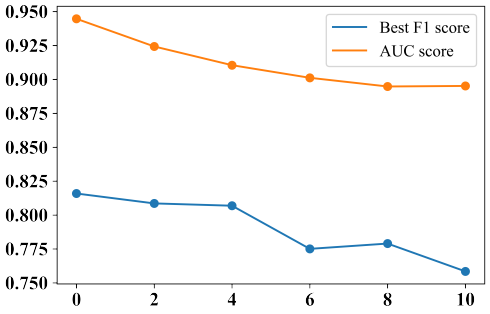
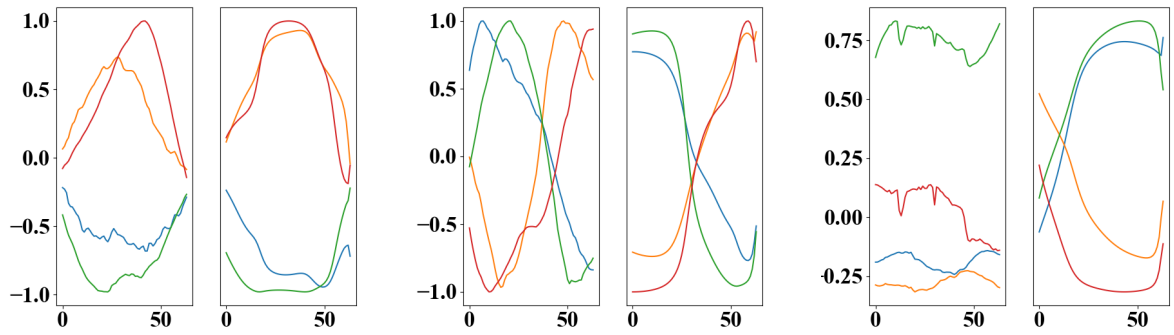


Fig. 12: Performance degradation with noised data.

4.3 Data Augmentation and Robustness

In this part, we will test our model to investigate the effectiveness and robustness of the data augmentation strategy. We test the data augmentation strategy on the ECG dataset, and we set $k = 16$ for data augmentation. The result of data augmentation is shown in Fig 11. In this experiment, all BeatGAN models are based on 1D-CNN. And the BeatGAN(aug) means we augment the training data size to $3\times$ with time warping strategy. For BeatGAN*(aug), we add the 0.1% anomalous time series to the training data for evaluating robustness. The result shows our data augmentation strategy can slightly improve the AUC score of BeatGAN model. Besides, as shown in Fig 12, with little anomalies(2% - 10%) in training data, the performance of our BeatGAN model will drop slightly which also shows the robustness of the model.

[38] modifies DTW and makes it a differentiable algorithm that can replace the MSE loss in Eq (9). We use BeatGAN CNN model to verify its effectiveness. BeatGAN(softdtw-softdtw) stands for replacing MSE loss with SoftDTW loss for both reconstruction error and adversarial regularization. And BeatGAN(softdtw-mse) stands for only replacing the error of reconstruction. BeatGAN(mse-mse) is the same as BeatGAN(CNN) in Table 3. The results shown in Table 5 demonstrate that the SoftDTW-based model performs slightly worse than the MSE-based model while both of them outperform other baselines. Furthermore, the lower standard error shows greater stability of SoftDTW-based loss which can be utilized more in the future.



(a) Reconstruction of walk case.(left is the input time series, right is the reconstructed time series)

(b) Reconstruction of run case.(left is the input time series, right is the reconstructed time series)

(c) Reconstruction of hop case.(left is the input time series, right is the reconstructed time series)

Fig. 13: Reconstruction samples on motion capture time series(4-dimensions).

Method	Best F1 Score	AUC Score
BeatGAN(softdtw-softdtw)	0.7841 ± 0.0009	0.9215 ± 0.0003
BeatGAN(softdtw-mse)	0.7844 ± 0.0009	0.9223 ± 0.0002
BeatGAN(mse-mse)	$\mathbf{0.8159 \pm 0.0102}$	$\mathbf{0.9447 \pm 0.0053}$

TABLE 5: Results of Soft-DTW and MSE loss.

4.4 Explainability

Next, we show that BeatGAN can pinpoint the time ticks when the anomalous pattern occurs. In the right part of Fig 2, we compute the residual for each time tick between input beat and generated beat: $res(t) = (x(t) - x'(t))^2$ at time t , and show the heatmap of residual values. As we observe, our model gives high anomalousness scores for the abnormal beat (top right) and low anomalousness scores for the normal beat (bottom right). This abnormal beat is a 'ventricular escape beat' and our model correctly identifies that the abnormal time ticks occur in its QRS complex (circled region). Besides, our model generates the "normal" generated beat (dashed lines), which provides additional explainability by allowing users to compare the generated beat to the observed beat, to understand how the observed beat differs.

In Fig 7, the left time series is the record of walking. On the right, we illustrate the results of a jogging, jumping or hopping time series, using a heatmap whose color indicates the size of the residual at each time tick. We compute the residual for each time tick by $res(t) = \max(x(t) - x'(t))^2$, where $x(t)$ is a 4-dimensional vector at time t, and $\max$ takes the max value over the 4 dimensions, which we use as the anomalousness score of time tick t. The heatmap shows that BeatGAN cannot well reconstruct the time series of jogging/jumping/hopping, thus correctly assigning them high anomalousness, since we only use walking time series for training.

Besides, we also visualize the reconstruction results to see what the model learns. In Fig 13, we show the reconstruction samples of walking, running and hopping time series. The Fig 13(a) shows the reconstruction example of the walking time series where left is the original input time series. We can see that the model can reconstruct this walking time series well, resulting in a low anomalousness score. And for Fig 13(b) and Fig 13(c), the reconstruction examples of running and hopping time series are shown. For running time series (left picture in Fig 13(b)), it has the high frequency compare to walking time series(left picture in Fig 13(a)), while the reconstructed time series is more like "walking" state, leading to a high anomalousness score and so for the hopping time series. From these, we may find out that our model can reconstruct the input time serise as more "normal" time series and this is in line with our expectations.

4.5 Efficiency

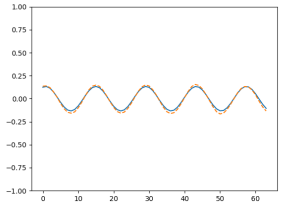
To verify the efficiency of the BeatGAN, We ran the inferences of BeatGAN, AnoGAN and Ganomaly, which are all based on generative adversarial network. We run the experiment on a server with a Tesla K80 GPU using ECG datase. Three models are all implemented in PyTorch. Besides, we set the iteration number of AnoGAN as 500 as in [4].

Fig 16 (the y-axis is a logarithmic scale) shows that BeatGAN only takes 2.6 ms per beat, which is $1.5\times$ faster than Ganomaly and $1415\times$ faster than AnoGAN. Actually, BeatGAN is fast for inference at test time since the adversarial generation of BeatGAN is one-pass through the feed-forward neural network. In contrast, the baseline AnoGAN is a two-step anomaly detection model. AnoGAN needs an extra step to find the corresponding latent vector for the given time series by iterative computation, so it is slowest in the test phase. While the model Ganomaly is also end-to-end but it has another encoder network which is more complex compared with BeatGAN.

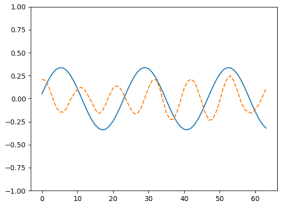
4.6 Sensitivity experiments on synthetic dataset

To further understand the ability of our BeatGAN model, we test the model on the synthetic time series. We generate the sine waves as our toy data. There are two main parameters for sine waves: amplitude and frequency. We create our training dataset with various amplitudes and frequencies, i.e. low amplitudes randomly in $(0.1, 0.5)$, high amplitudes randomly in $(0.5, 0.9)$; low frequencies randomly in $(1.0, 5.0)$ and high frequencies randomly in $(5.0, 10)$, where (a, b) means any real number larger than a and less then b.

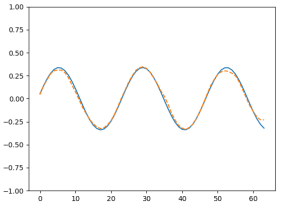
So we have 4 combinations for our training dataset to train our model. For testing, we want to know how well BeatGAN can reconstruct the sine time series with various parameters, i.e. amplitudes, and frequencies. Fig.14(a) -14(p) shows the different



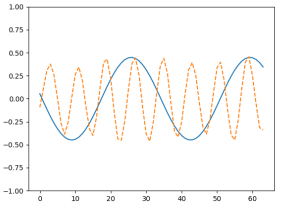
(a) **Training with low Amp./low Freq. Testing with low Amp./low Freq.**



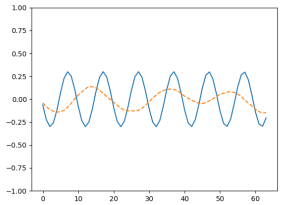
(b) Training with low Amp./high Freq. Testing with low Amp./low Freq.



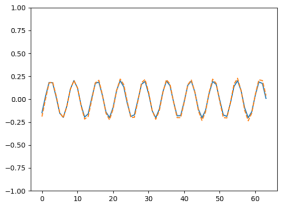
(c) Training with high Amp./low Freq. Testing with low Amp./low Freq.



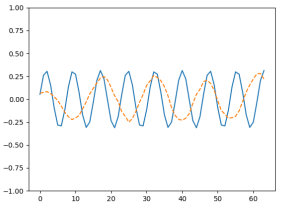
(d) Training with high Amp./high Freq. Testing with low Amp./low Freq.



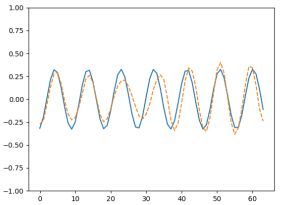
(e) Training with low Amp./low Freq. Testing with low Amp./high Freq.



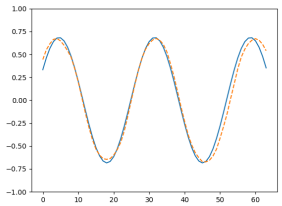
(f) **Training with low Amp./high Freq. Testing with low Amp./high Freq.**



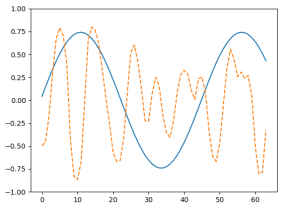
(g) Training with high Amp./low Freq. Testing with low Amp./high Freq.



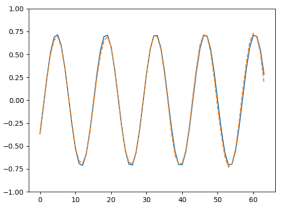
(h) Training with high Amp./high Freq. Testing with low Amp./high Freq.



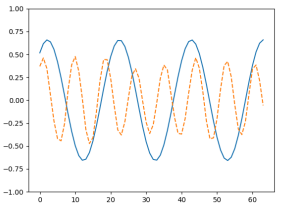
(i) Training with low Amp./low Freq. Testing with high Amp./low Freq.



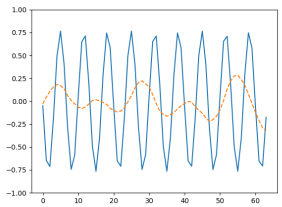
(j) Training with low Amp./high Freq. Testing with high Amp./low Freq.



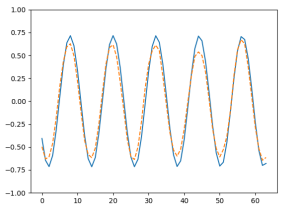
(k) **Training with high Amp./low Freq. Testing with high Amp./low Freq.**



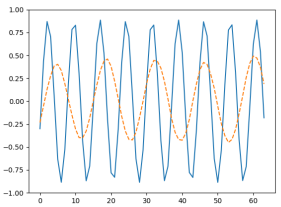
(l) Training with high Amp./high Freq. Testing with high Amp./low Freq.



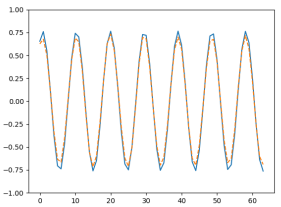
(m) Training with low Amp./low Freq. Testing with high Amp./high Freq.



(n) Training with low Amp./high Freq. Testing with high Amp./high Freq.

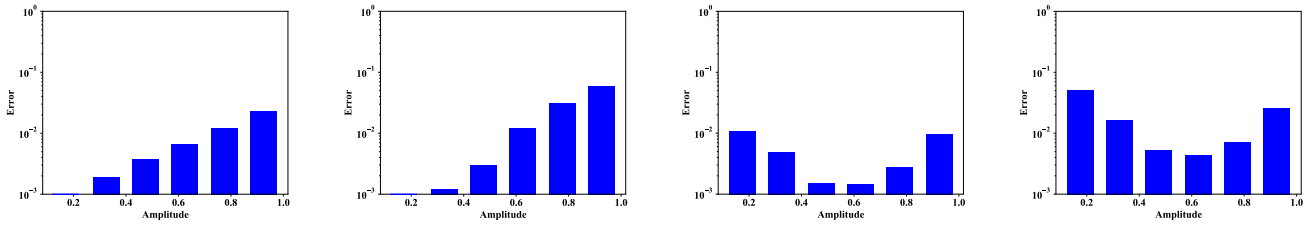


(o) Training with high Amp./low Freq. Testing with high Amp./high Freq.

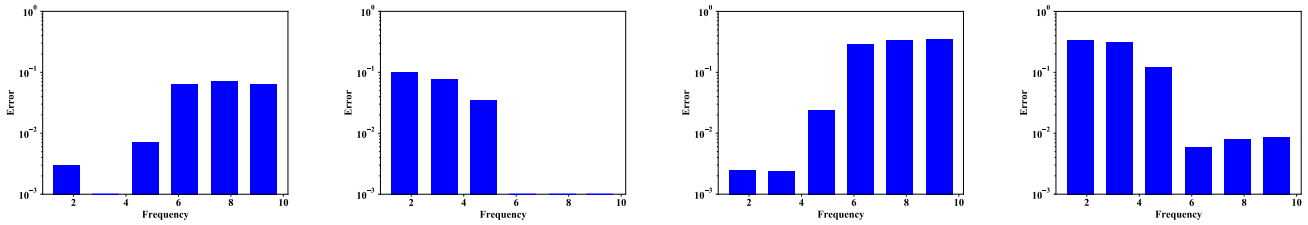


(p) **Training with high Amp./high Freq. Testing with high Amp./high Freq.**

Fig. 14: Reconstruction samples on synthetic sine waves. The original time series is plotted by solid lines, and the reconstruction is plotted by dashed lines. Amp. and Freq. are amplitudes and frequencies respectively. The cases with bold captions are trained and tested by the time series of similar amplitudes and frequencies.



(a) training with low Amp./low Freq. testing on different Amp. (b) training with low Amp./high Freq. testing on different Amp. (c) training with high Amp./low Freq. testing on different Amp. (d) training with high Amp./high Freq. testing on different Amp.



(e) training with low Amp./low Freq. testing on different Freq. (f) training with low Amp./high Freq. testing on different Freq. (g) training with high Amp./low Freq. testing on different Freq. (h) training with high Amp./high Freq. testing on different Freq.

Fig. 15: Mean reconstruction errors of training and testing on various amplitudes and frequencies.

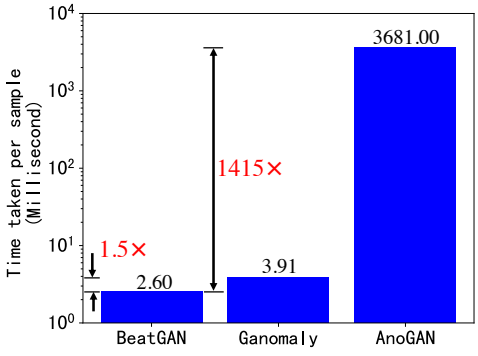


Fig. 16: BeatGAN has fast inference (2.6ms).

reconstruction examples with different models. In the first column of figures, the model is trained by time series of low amplitude and low frequency and tested separately on time series of low amplitude and low frequency, low amplitude, and high frequency, high amplitude and low frequency, and high amplitude, and high frequency. The second column of figures shows the results of the model which is trained by time series of low amplitude and high frequency. And the 3rd and 4th columns are illustrated as such. From these figures, we can see that BeatGAN reconstructs well with various amplitudes and similar frequencies. But BeatGAN is sensitive to frequency, which reconstructions deviate further away from the originals if the training and testing time series have quite different frequencies. In another word, the model can reconstruct well even the amplitude of test sample is not observed (higher or lower) in training time series, while the model reconstruct poorly if the frequency of the test sample is not seen in training.

We also design the quantitative experiments to support this idea. Fig.15 shows the quantitative result. The first row shows the mean reconstruction error with amplitude changes of different models and the second row shows the mean reconstruction error with frequency changes of different models. The first column shows the result of model trained by low amplitude and low frequency, the second column shows the model trained by low amplitude and high frequency, the third column shows the model trained by high amplitude and low frequency, and the fourth shows the model trained by high amplitude and high frequency. Note that the y-axis is a logarithmic scale. We can find out that the mean reconstruction error is more steady while the mean reconstruction error changes much more when frequency changes. The low reconstruction error may mean the model regards the test samples are generated from same mechanism as its training data while high reconstruction error means the high probability of different generation mechanism. This phenomenon reveals that the model may not actually learn the sine function and the complex pattern like frequency is hard to learn.

5 Conclusion

We propose a reconstruction based model with automatic threshold selection algorithm for time series anomaly detection. The model detects anomalies using the adversarially generated time series. For underlying structure, we design both the 1D CNN based and the RNN based neural network, while the CNN based model is more suitable for medical time series and the RNN based model is more suitable for smoothed time series. In conclusion, our model has the following advantages:

- Unsupervised: BeatGAN is the reconstruction based anomaly detection model. It detects anomalies by observing massive normal time series.
- Effectiveness: BeatGAN outperforms baselines in both accuracy and inference speed, achieving high accuracy in three datasets and very fast inference.
- Explainability: BeatGAN can pinpoint the anomalous ticks to route people's attention as shown in Fig 2;
- Generality: BeatGAN also can successfully detects anomalies in many time series of different domains.

ACKNOWLEDGMENTS

This material is based upon work supported by the Strategic Priority Research Program of CAS (XDA19020400), NSF of China (61425016, 61772498, 91746301), and the Beijing NSF (4172059).

REFERENCES

[1] N. Shah, A. Beutel, B. Gallagher, and C. Faloutsos, "Spotting suspicious link behavior with fbox: An adversarial perspective," in *Data Mining (ICDM), 2014 IEEE International Conference on*. IEEE, 2014, pp. 959–964.

[2] J. An and S. Cho, "Variational autoencoder based anomaly detection using reconstruction probability," *Special Lecture on IE*, vol. 2, pp. 1–18, 2015.

[3] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[4] T. Schlegl, P. Seeböck, S. M. Waldstein, U. Schmidt-Erfurth, and G. Langs, "Unsupervised anomaly detection with generative adversarial networks to guide marker discovery," in *International Conference on Information Processing in Medical Imaging*. Springer, 2017, pp. 146–157.

[5] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM computing surveys (CSUR)*, vol. 41, no. 3, pp. 1–58, 2009.

[6] F. E. Grubbs *et al.*, "Sample criteria for testing outlying observations," *The Annals of Mathematical Statistics*, vol. 21, no. 1, pp. 27–58, 1950.

[7] A. Siffer, P.-A. Fouque, A. Termier, and C. Largouet, "Anomaly detection in streams with extreme value theory," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017, pp. 1067–1075.

[8] B. Hooi, K. Shin, H. Lamba, and C. Faloutsos, "Telltail: Fast scoring and detection of dense subgraphs," in *The Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI)*. Association for the Advancement of Artificial Intelligence, 2020.

[9] P. Rajpurkar, A. Y. Hannun, M. Haghpanahi, C. Bourn, and A. Y. Ng, "Cardiologist-level arrhythmia detection with convolutional neural networks," *arXiv preprint arXiv:1707.01836*, 2017.

[10] S. Kiranyaz, T. Ince, and M. Gabbouj, "Real-time patient-specific ecg classification by 1-d convolutional neural networks," *IEEE Transactions on Biomedical Engineering*, vol. 63, no. 3, pp. 664–675, 2015.

[11] Y. Shen, M. Voisin, A. Aliamiri, A. Avati, A. Hannun, and A. Ng, "Ambulatory atrial fibrillation monitoring using wearable photoplethysmography with deep learning," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 1909–1916.

[12] Y. Roh, G. Heo, and S. E. Whang, "A survey on data collection for machine learning: a big data-ai integration perspective," *IEEE Transactions on Knowledge and Data Engineering*, 2019.

[13] B. Schölkopf, R. C. Williamson, A. J. Smola, J. Shawe-Taylor, and J. C. Platt, "Support vector method for novelty detection," in *Advances in neural information processing systems*, 2000, pp. 582–588.

[14] L. E. Baum and T. Petrie, "Statistical inference for probabilistic functions of finite state markov chains," *The annals of mathematical statistics*, vol. 37, no. 6, pp. 1554–1563, 1966.

[15] L. Li, J. McCann, N. S. Pollard, and C. Faloutsos, "Dynammo: Mining and summarization of coevolving sequences with missing values," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2009, pp. 507–516.

[16] Y. Matsubara, Y. Sakurai, and C. Faloutsos, "Autoplait: Automatic mining of co-evolving time sequences," in *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*. ACM, 2014, pp. 193–204.

[17] B. Hooi, S. Liu, A. Smailagic, and C. Faloutsos, "BEATLEX: Summarizing and forecasting time series with patterns," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2017, pp. 3–19.

[18] C.-C. M. Yeh, Y. Zhu, L. Ulanova, N. Begum, Y. Ding, H. A. Dau, D. F. Silva, A. Mueen, and E. Keogh, "Matrix profile i: all pairs similarity joins for time series: a unifying view that includes motifs, discords and shapelets," in *2016 IEEE 16th international conference on data mining (ICDM)*. Ieee, 2016, pp. 1317–1322.

[19] F. Madrid, S. Imani, R. Mercer, Z. Zimmerman, N. Shakibay, and E. Keogh, "Matrix profile xx: Finding and visualizing time series motifs of all lengths using the matrix profile," in *2019 IEEE International Conference on Big Knowledge (ICBK)*. IEEE, 2019, pp. 175–182.

[20] P. Boniol, M. Linardi, F. Roncallo, T. Palpanas, M. Meftah, and E. Remy, "Unsupervised and scalable subsequence anomaly detection in large data series," *The VLDB Journal*, pp. 1–23, 2021.

[21] D. M. Hawkins, *Identification of outliers*. Springer, 1980, vol. 11.

[22] H. Xu, W. Chen, N. Zhao, Z. Li, J. Bu, Z. Li, Y. Liu, Y. Zhao, D. Pei, Y. Feng *et al.*, "Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications," in *Proceedings of the 2018 World Wide Web Conference*, 2018, pp. 187–196.

[23] Y. Su, Y. Zhao, C. Niu, R. Liu, W. Sun, and D. Pei, "Robust anomaly detection for multivariate time series through stochastic recurrent neural network," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 2828–2837.

[24] C. Zhang, D. Song, Y. Chen, X. Feng, C. Lumezanu, W. Cheng, J. Ni, B. Zong, H. Chen, and N. V. Chawla, "A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 1409–1416.

[25] J. Audibert, P. Michiardi, F. Guyard, S. Marti, and M. A. Zuluaga, "Usad: Unsupervised anomaly detection on multivariate time series," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 3395–3404.

[26] S. Akcay, A. Atapour-Abarghouei, and T. P. Breckon, "Ganomaly: Semi-supervised anomaly detection via adversarial training," *arXiv preprint arXiv:1805.06725*, 2018.

[27] H. Zenati, C. S. Foo, B. Lecouat, G. Manek, and V. R. Chandrasekhar, "Efficient gan-based anomaly detection," *arXiv preprint arXiv:1802.06222*, 2018.

[28] H. Arnelid, E. L. Zec, and N. Mohammadiha, "Recurrent conditional generative adversarial networks for autonomous driving sensor modelling," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2019, pp. 1613–1618.

[29] H. A. Song, B. Hooi, M. Jereminov, A. Pandey, L. Pileggi, and C. Faloutsos, "Powercast: Mining and forecasting power grid sequences," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2017, pp. 606–621.

[30] B. Hooi, H. A. Song, A. Pandey, M. Jereminov, L. Pileggi, and C. Faloutsos, "Streamcast: Fast and online mining of power grid time sequences," in *Proceedings of the 2018 SIAM International Conference on Data Mining*. SIAM, 2018, pp. 531–539.

[31] P. Chen, S. Liu, C. Shi, B. Hooi, B. Wang, and X. Cheng, "Neucast: Seasonal neural forecast of power grid time series." in *IJCAI*, 2018, pp. 3315–3321.

[32] B. Zong, Q. Song, M. R. Min, W. Cheng, C. Lumezanu, D. Cho, and H. Chen, "Deep autoencoding gaussian mixture model for unsupervised anomaly detection," in *International conference on learning representations*, 2018.

[33] P. Boniol and T. Palpanas, "Series2graph: Graph-based subsequence anomaly detection for time series," *Proceedings of the VLDB Endowment*, vol. 13, no. 12, pp. 1821–1834, 2020.

[34] L. Guo, G. Sim, and B. Matuszewski, "Inter-patient ecg classification with convolutional and recurrent neural networks," *Biocybernetics and Biomedical Engineering*, vol. 39, no. 3, pp. 868–879, 2019.

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

[36] J. Jocković, "Quantile estimation for the generalized pareto distribution with application to finance," *Yugoslav Journal of Operations Research*, vol. 22, no. 2, 2016.

[37] T. K. Vintsyuk, "Speech discrimination by dynamic programming," *Cybernetics and Systems Analysis*, vol. 4, no. 1, pp. 52–57, 1968.

[38] M. Cuturi and M. Blondel, "Soft-dtw: a differentiable loss function for time-series," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 894–903.

[39] G. B. Moody and R. G. Mark, "The impact of the mit-bih arrhythmia database," *IEEE Engineering in Medicine and Biology Magazine*, vol. 20, no. 3, pp. 45–50, 2001.

[40] AAMI, "Testing and reporting performance results of cardiac rhythm and st segment measurement algorithms," *ANSI/AAMI EC38*, 1998.

[41] A. P. Mathur and N. O. Tippenhauer, "Swat: A water treatment testbed for research and training on ics security," in *2016 International Workshop on Cyber-physical Systems for Smart Water Networks (CySWater)*. IEEE, 2016, pp. 31–36.

[42] P. Malhotra, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, and G. Shroff, "Lstm-based encoder-decoder for multi-sensor anomaly detection," *arXiv preprint arXiv:1607.00148*, 2016.
[43] D. Li, D. Chen, L. Shi, B. Jin, J. Goh, and S.-K. N. Mad-gan, "Multivariate anomaly detection for time series data with generative adversarial networks," *arXiv preprint arXiv:1901.04997*, 2019.
[44] C. Carreiras, A. P. Alves, A. Lourenço, F. Canento, H. Silva, A. Fred *et al.*, "BioSPPy: Biosignal processing in Python," 2015–, [Online; accessed 2019-1-12]. [Online]. Available: https://github.com/PIA-Group/BioSPPy/
[45] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.
[46] S. Hong, Y. Zhou, J. Shang, C. Xiao, and J. Sun, "Opportunities and challenges in deep learning methods on electrocardiogram data: A systematic review," *arXiv preprint arXiv:2001.01550*, 2019.

Shenghua Liu is an Associate Professor at the Institute of Computing Technology, Chinese Academy of Sciences. He received his Ph.D. degree from the Computer Science and Technology Department, Tsinghua University. He was a visiting scholar at the University of California, Los Angeles and Carnegie Mellon University respectively. His current research interests are designing intelligent and automated algorithms for big data mining problems, related to big graphs and series.

Bin Zhou received the B.S. degree from University of Science and Technology Beijing in 2017. He is currently a master student in University of Chinese Academy of Sciences. His research interests are in time series mining, machine learning.

Quan Ding received his B.S. degree from NanKai University in 2020. He is currently a master student in University of Chinese Academy of Sciences. His main research insterests are time series mining and machine learning.

Bryan Hooi is a Ph.D. student jointly enrolled in the Machine Learning Department and the Department of Statistics at Carnegie Mellon University. He received his BS and MS degrees from Stanford University. His main research interests include graph mining, anomaly detection, and time series data analysis.

Xueqi Cheng is a Professor at the Institute of Computing Technology, Chinese Academy of Sciences (ICT-CAS), and the director of the Research Center of Web Data Science & Engineering (WDSE) in ICT-CAS. His main research interests include Network Science etc. He has published over 100 publications in prestigious journals and conferences, including TKDE, Physical Review E., SIGIR, WWW etc.

Zhengbo Zhang is an Associate Professor at the Center for Artificial Intelligence in Medicine, Chinese PLA General Hospital. He received his Ph. D. degree from the Institute of Medical Equipment, Academy of Military Medical Sciences. He was a visiting scientist in the Laboratory for Computational Physiology, Massachusetts Institute of Technology, and Center for Dynamical Biomarkers at Beth Israel Deaconess Medical Center / Harvard Medical School respectively. His current research interests are secondary analysis of electronic health records, predictive analysis of physiological time series, and application of wearable physiological monitoring systems, Internet of Things and Artificial Intelligence in clinical scenario.