Generative Adversarial Networks in Time Series: A Systematic Literature Review

EOIN BROPHY, Dublin City University
ZHENGWEI WANG, Trinity College Dublin
QI SHE, ByteDance AI Lab
TOMÁS WARD, Dublin City University

Generative adversarial network (GAN) studies have grown exponentially in the past few years. Their impact has been seen mainly in the computer vision field with realistic image and video manipulation, especially generation, making significant advancements. Although these computer vision advances have garnered much attention, GAN applications have diversified across disciplines such as time series and sequence generation. As a relatively new niche for GANs, fieldwork is ongoing to develop high-quality, diverse, and private time series data. In this article, we review GAN variants designed for time series related applications. We propose a classification of discrete-variant GANs and continuous-variant GANs, in which GANs deal with discrete time series and continuous time series data. Here we showcase the latest and most popular literature in this field—their architectures, results, and applications. We also provide a list of the most popular evaluation metrics and their suitability across applications. Also presented is a discussion of privacy measures for these GANs and further protections and directions for dealing with sensitive data. We aim to frame clearly and concisely the latest and state-of-the-art research in this area and their applications to real-world technologies.

CCS Concepts: • **Computing methodologies** → **Machine learning approaches;**

Additional Key Words and Phrases: Generative adversarial networks, time series, discrete-variant GANs, continuous-variant GANs

ACM Reference format:
Eoin Brophy, Zhengwei Wang, Qi She, and Tomás Ward. 2023. Generative Adversarial Networks in Time Series: A Systematic Literature Review. *ACM Comput. Surv.* 55, 10, Article 199 (February 2023), 31 pages.
https://doi.org/10.1145/3559540

1 INTRODUCTION

This review article is designed for those interested in **generative adversarial networks (GANs)** applied to time series data generation. We provide a review of current state-of-the-art and novel time series GANs and their solutions to real-world problems with time series data. GANs have

This work was funded by Science Foundation Ireland under grant numbers 17/RC-PhD/3482 and SFI/12/RC/2289_P2.
Authors' addresses: E. Brophy (corresponding author), Infant Research Centre & School of Computing, Dublin City University, Dublin, Ireland; email: eoin.brophy7@mail.dcu.ie; Z. Wang, V-SENSE, School of Computer Science and Statistics, Trinity College Dublin, Dublin, Ireland; email: villa.wang.zhengwei@gmail.com; Q. She, ByteDance AI Lab, China; email: 1477430657@qq.com; T. Ward, Insight SFI Research Centre for Data Analytics, Dublin City University, Dublin, Ireland; email: tomas.ward@dcu.ie.

0360-0300/2023/02-ART199
https://doi.org/10.1145/3559540

been gaining a lot of traction within the deep learning research community since their inception in 2014 [38]. Their ability to generate and manipulate high-quality data across multiple domains has contributed to their success. The main focus of GANs to date has been in the computer vision domain; however, they have also been successfully applied to others, such as **natural language processing (NLP)** and now time series.

A GAN is a generative model consisting of a generator and discriminator, typically two **neural network (NN)** models. In recent years GANs have demonstrated their ability to produce high-quality image and video generation, style transfer, and image completion. They have also been successfully used for audio generation, sequence forecasting, and imputation, with a movement toward using GANs for time series and sequential data generation and forecasting.

We define a time series as a sequence of vectors dependent on time (t) and can be represented as $xt = x1, \ldots, xn$ for continuous/real time and discrete time. The time series' values can either be defined as continuous or discrete and, depending on the number of values recorded, are univariate or multivariate. In most cases, the time series will take either an integer value or a real value. As Dorffner [25] states, a time series can be viewed, from a practical perspective, as a value sampled at discrete steps in time. This timestep can be as long as years to as short as milliseconds, for example. We define a continuous time series as a signal sampled from a continuous process—that is, the function's domain is from an uncountable set. In contrast, a discrete time series has a countable domain.

The applicability of GANs to time series data can solve many issues that current dataset holders face that cannot or have not been addressed by other machine learning or regressive techniques. Data shortage is often an issue that many practitioners face, and GANs can augment smaller datasets by generating new, previously unseen data. Data can be missing or corrupted in cases; GANs can impute data, such as replace the artifacts with information representative of clean data. GANs are also capable of denoising signals in the case of corrupted data. Data protection, privacy, and sharing have become heavily regulated with the introduction of data protection measures; GANs can ensure an extra layer of data protection by generating differentially private datasets containing no risk of linkage from source to generated datasets.

Time series data generation is not a novel concept in that it has long roots seeded in regression. Furthermore, it initially began as forecasting of timesteps rather than whole sequence generation. One of the most used time series forecasting methods was **autoregressive (AR)** models. Aside from forecasting data points, AR models focus on preserving the temporal dynamics of a sequence. However, they are inherently deterministic in that no randomness is involved in the calculation of future states of the system. This means that AR models are not genuinely generative or probabilistic. For an AR model, the goal is to produce the next timestep (x_{t+1}) in a sequence as a function of the previous n timesteps, where n is the order of the model. The formula for a classic AR model is given in Equation (1).

$$x_{t+1} = c + \theta_1 x_t + \theta_2 x_{t-1} + \epsilon \tag{1}$$

Here, x_t is the value of the sequence at time t, θ is the model parameters, c is a constant, and ϵ is the error term usually chosen as normally distributed noise.

Autoregression was a step shy of time series synthesis. That ultimately came in the form of directed generative networks. When using the term *directed*, we mean a model where the edges are directed and thus indicates which variable's probability distribution is defined in terms of another. In other words, this is a structured probabilistic model with conditional probability distributions. These data-driven generative models offered researchers the option of generating full-length data sequences versus forecasting singular values in the case of the regressive models. It also required little domain knowledge of the time series signal morphology, which was often a necessity for

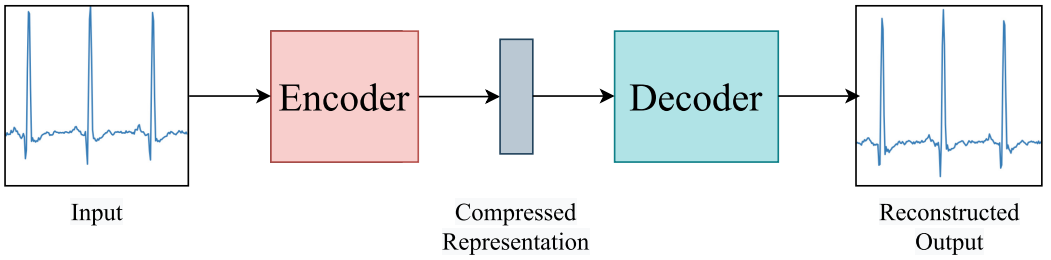

Fig. 1. AE model.

other statistical modeling techniques. This propelled generative modeling forward in the machine learning community for data synthesis techniques.

Several generative methods have been used in the past to generate synthetic data. One such method is the **autoencoder (AE)**, which is designed to efficiently learn an informative representation of an input in a small dimensional space and reconstruct the encoded data back such that the reconstructed input is similar as possible to the original one. The AE model is made of an encoder and decoder NN, as shown in Figure 1. However, other generative models have emerged as front-runners due to the quality of the generated data and inherent privacy protection measures.

Generative models come in many shapes, from **variational autoencoders (VAEs)** and **recurrent neural network (RNN)** variants to GANs, all of which have their pros and cons. For example, VAEs use learned approximate inference to produce synthetic samples efficiently. An inference problem is simply using the value of some variables or probability distributions to predict other values or probability distributions. Approximate inference is when we seek to approximate a true distribution, say $p(y|x)$, by seeking an approximate distribution $q(y|x)$. However, this network approximation conducted by VAEs means that their generated data quality can be degraded compared to samples generated by GANs. However, for all of the benefits that come with GANs, they are not without their own downsides. They are a very useful technology that allows us to reproduce amazingly insightful and powerful datasets, but only if we can address their following challenges.

One of the significant challenges of GANs lies in their inherent instability, which makes them difficult to train. GAN models suffer from issues such as non-convergence, diminishing/vanishing gradients, and mode collapse. A non-converging model does not stabilize and continuously oscillates, causing it to diverge. Diminishing gradients prevent the generator from learning anything, as the discriminator becomes too successful. Mode collapse is when the generator collapses, producing only uniform samples with little to no variety.

The second challenge of GANs lies in its evaluation process. With image-based GANs, researchers have reached a loose consensus [8] surrounding the evaluation of the generated distribution estimated from the training data distribution. Unfortunately for time series GANs, due to the comparatively low numbers of papers published, there has not been an agreement reached on the generated data's evaluation metrics. There have been different approaches put forward, but none established as a front-runner in the metrics space as of yet.

In this review, we present the first complete review and categorization of time series GANs, namely discrete and continuous variants, their applications, architecture, loss functions and how they have improved on their predecessors in terms of variety and quality of their generated data. We also contribute by including experiments for the majority of time series GAN architectures applied to time series synthesis.

2 RELATED WORK

There has been a handful of high-quality GAN review papers published in the past few years. For example, Wang et al. [100] take a taxonomic approach to GANs in computer vision. The authors

split GANs into architecture variants and loss variants. Although they include applications of GANs and mention their applicability to sequential data generation, the work is heavily focused on media manipulation and generation. Gui et al. [40] break down GANs into their constituent parts. They begin by discussing the algorithms and architecture of various GANs and their evaluation metrics, then list their surrounding theory and problems such as mode collapse, among others. Finally, they discuss the applications of GANs and provide a very brief account of GANs used for sequential data. Gonog and Zhou [36] provide a short introduction to GANs, their theory, and explore the variety of plausible models, again listing their applications in image and video manipulation with a mention of sequential data (NLP). In another review, Alqahtani et al. [3] give an overview of GAN fundamentals, variants, and applications. Sequential data applications are mentioned in the form of music and speech synthesis.

As with most review papers, Yinka-Banjo and Ugot [106] give an introduction and overview of GANs. However, they also review GANs as adversarial detectors and discuss their limitations applied to cybersecurity. Yi et al. [105] give a review of GANs and their applications in medical imaging, and explain how they can be used in clinical research and potentially deployed to help practicing clinicians. There is no mention of time series data use cases.

A recurring theme in these works focuses on GAN variants that have mostly been applied to the computer vision domain. To the best of our knowledge, no review paper has been conducted with the main focus on time series GANs. Although these reviews have mentioned the application of these GANs in generating sequential data, they have scratched the surface of what is becoming a growing body of research.

We contribute to lessening this gap by presenting our work, which seeks to provide the latest up-to-date research around time series GANs, their architecture, loss functions, evaluation metrics, trade-offs, and approaches to privacy preservation of their datasets.

3 GENERATIVE ADVERSARIAL NETWORKS

3.1 Background

The introduction of GANs facilitated a significant breakthrough in the generation of synthetic data. These deep learning models typically consist of two NNs: a generator and a discriminator. The generator G takes in random noise $\boldsymbol{z} \in \mathbb{R}^r$ and attempts generates synthetic data that is similar to the training data distribution. The discriminator D attempts to determine if the generated data is real or fake. The generator aims to maximize the failure rate of the discriminator, whereas the discriminator aims to minimize it. Figure 2 shows a simple example of the GAN architecture and the game that the NN models play. The two networks are locked in a two-player minimax game defined by the value function $V(G,D)$ (2), where $D(\boldsymbol{x})$ is the probability that x comes from the real data rather than the generated data [38].

$$\min_G \max_D V(G, D) = \mathbb{E}_{x \sim p_{data}(x)}[logD(\mathbf{x})] + \mathbb{E}_{z \sim p_z(z)}[log(1 - D(G(\mathbf{z})))] \tag{2}$$

GANs belong to the family of generative models and are an alternative method of generating synthetic data that do not require domain expertise. They were conceived in the work by Goodfellow et al. [38] in 2014, where a multi-layer perceptron was used for both the discriminator and the generator. In 2015, Radford et al. [85] subsequently developed the **deep convolutional generative adversarial network (DCGAN)** to generate synthetic images. Since then, researchers have continuously improved on the early GAN architectures, loss functions, and evaluation metrics while innovating on their potential contributions to real-world applications. To appreciate why there has been such concerted activity in the further development of GAN technologies, it is important to understand the limitations of early architectures and the challenges these presented.

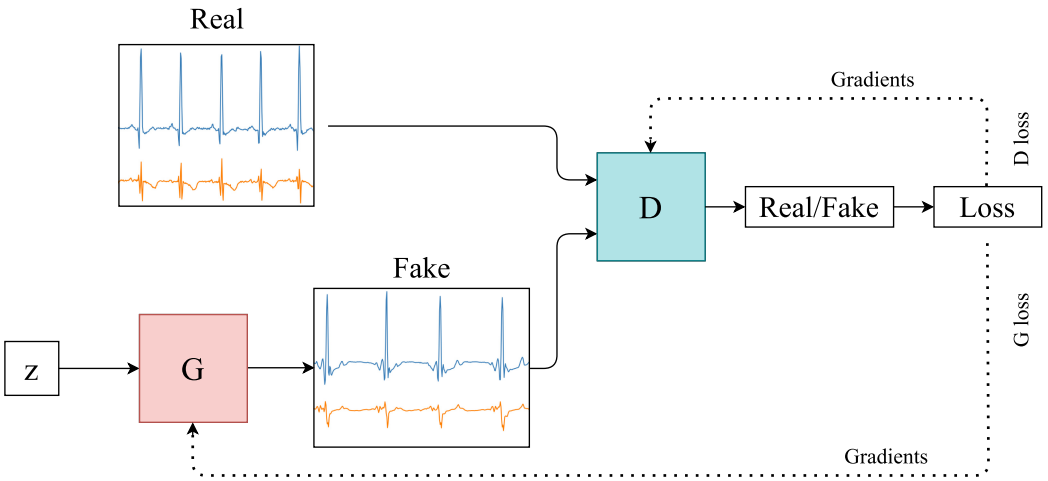

Fig. 2. Generative adversarial network.

We describe these next, and in so doing, we prepare the reader for the particular manifestation of these challenges in the more specific context of time series.

3.2 Challenges

There are three main challenges in the area of time series GANs: training stability, evaluation, and privacy risk associated with synthetic data created by GANs. We explain these three challenges next.

Training stability. The original GAN work has already proved the global optimality and the convergence of GANs during training [38]. However, it still highlights the instability problem that can arise when training a GAN. Two problems are well studied in the literature: vanishing gradients and mode collapse. The vanishing gradient is caused by directly optimizing the loss presented in Equation (2). When D reaches the optimality, optimizing Equation (2) for G can be converted to minimizing the Jensen-Shannon (JS) divergence (details of the derivation can be found in Section 5 of the work of Wang et al. [100]) between the real data distribution (p_{data}) and the generator's distribution (p_g):

$$\mathcal{L}_G = 2 \cdot JS(p_{data} \| p_g) - 2 \cdot \log 2. \tag{3}$$

$\mathcal{L}_G$ stays constant ($log2 = 0.693$) when there is no overlap between p_{data} and p_g, which indicates that the gradient for G using this loss is near 0 in this situation. A non-zero gradient for G only exists when p_{data} and p_g have substantial overlap. In practice, the possibility that p_{data} and p_g are not intersected or have negligible overlap is quite high [4]. To get rid of the vanishing gradient problem for G, the original GAN work [38] highlights the minimization of

$$\mathcal{L}_G = -\mathbb{E}_{\mathbf{x} \sim p_g} \log[D(\mathbf{x})] \tag{4}$$

for updating G. This strategy is able to avoid the vanishing gradient problem but leads to the mode collapse issue. Optimizing Equation (4) can be converted to optimizing the reverse **Kullback-Leibler (KL)** divergence—that is, $KL(p_g || p_{data})$ (details can be found in the work of Wang et al. [100]). When p_{data} contains multiple modes, p_g chooses to recover a single mode and ignores other modes when optimizing the reverse KL divergence. Considering this case, G trained using Equation (4) might be only able to generate few modes from real data. These problems can be

amended by changing the architecture or the loss function, which are reviewed by Wang et al. [100] in detail.

Evaluation. A wide range of evaluation metrics has been proposed to evaluate the performance of GANs [9, 10, 98, 99]. Current evaluations of GANs in computer vision are normally designed to consider two perspectives: quality and quantity of generated data. The most representative qualitative metric is to use human annotation to determine the visual quality of the generated images. Quantitative metrics compare statistical properties between generated and real images: two-sample tests such as maximum mean discrepancy (MMD) [93], inception score (IS) [88], and Fréchet inception distance (FID) [51]. Contrary to evaluating image-based GANs, it is difficult to evaluate time series data from human psycho-perceptual sense qualitatively. In terms of qualitatively evaluating time series based GANs, it normally conducts t-SNE [95] and PCA [13] analyses to visualize how well the generated distributions resemble the original distributions [107]. Quantitative evaluation for time series based GANs can be done by deploying two-sample tests similar to image-based GANs.

Privacy risk. Apart from evaluating the performance of GANs, a wide range of methods have been used to assess the privacy risk associated with synthetic data created by GANs. Choi et al. [17] performed tests for presence disclosure and attribute disclosure. In contrast, others utilized a three-sample test on the training, test, and synthetic data to identify if the synthetic data has overfitted to the training data [17, 31]. It has been shown that common methods of de-identifying data do not prevent attackers from re-identifying individuals using additional data [29, 72]. Sensitive data is usually de-identified by removing personally identifiable information. However, work is ongoing to create frameworks to link different sources of publicly available information together using alternative information to personally identifiable information. Malin and Sweeney [72] developed a software program, REID, to connect individuals contained in publicly available hospital discharge data with their unique DNA records. Culnane et al. [19] re-identified individuals in a de-identified open dataset of Australian medical billing records using unencrypted parts of the records and known information about individuals from other sources. Hejblum et al. [50] developed a probabilistic method to link de-identified **electronic health record (EHR)** data of patients with rheumatoid arthritis. The re-identification of individuals in publicly available datasets can lead to the exposure of their sensitive health information. Health data has been categorized as special personal data by General Data Protection Regulation (GDPR) and is subject to a higher level of protection under the Data Protection Act of 2018 (Section 36(2)) [32]. Consequently, concerned researchers must find alternative methods of protecting sensitive health data to minimize the risk of re-identification. This will be addressed in Section 7.

3.3 Popular Datasets

Unlike image-based datasets (CIFAR, MNIST, ImageNet [22, 61, 64]), there are no standardized or commonly used benchmarking datasets for time series generation. However, we have compiled a list of some of the more popular datasets implemented in the reviewed works, and they are listed in Table 1 along with their year of release/update, data type, and how many instances and attributes they contain. What makes these datasets interesting/applicable to time series GANs is that they are signals made up of highly complex waveforms (physiological and audio) and contain important temporal dynamics crucial to preserve when generating new samples. Furthermore, these signals are the exact data type that are highly regulated and can stand to benefit from being leveraged by GANs to generate further volumes of this kind of data.

There exist two repositories; the UCR Time Series Classification/Clustering database [20] and the UCI Machine Learning Repository [26] that make available several time series datasets. Despite this, there remains no consensus on a standardized dataset used for benchmarking time

Table 1. Popular Datasets Used in the Reviewed Works

Name (Year)	Data Type	Instances	Attributes
Oxford-Man Institute "realized library" (updated daily)	Real multivariate time series	>2,689,487	5
EEG Motor Movement/Imagery Dataset (2004)	Real multivariate time series	1,500	64
ECG 200 (2001)	Real univariate time series	200	1
Epileptic Seizure Recognition Dataset (2001)	Real multivariate time series	11,500	179
TwoLeadECG (2015)	Real multivariate time series	1,162	2
MIMIC-III (2016)	Real, integer, and categorical multivariate time series	–	–
EPILEPSIAE project database (2012)	Real multivariate time series	30	–
PhysioNet/CinC (2015)	Real multivariate time series	750	4
Wrist PPG During Exercise (2017)	Real multivariate time series	19	14
MIT-BIH Arrhythmia Database (2001)	Real multivariate time series	201	2
PhysioNet/CinC (2012)	Real, integer, and categorical multivariate time series	12,000	43
KDD Cup Dataset (2018)	Real, integer, and categorical multivariate time series	282	3
PeMS Database (updated daily)	Integer and categorical multivariate time series	–	8
Nottingham Music Database (2003)	Special text format time series	1,000	–

series GANs, which may be due to the "continuous" nature of the architecture dimensions. GANs designed for continuous time series generation often differ in the length of their input sequence due to either author preference or the constraints placed on their architecture for the generated data's downstream tasks.

4 CLASSIFICATION OF TIME SERIES BASED GANS

We propose a categorization of the following time series based GANs based on two distinct variant types: *discrete variants* (discrete time series) and *continuous variants* (continuous time series). A discrete time series consists of data points separated by time intervals. This type of data might have a data-reporting interval that is infrequent (e.g., 1 point per minute) or irregular (e.g., whenever a user logs in), and gaps where values are missing due to reporting interruptions (e.g., intermittent server or network downtime in a network traffic application). Discrete time series generation involves generating sequences that may have a temporal dependency but contain discrete tokens; these can be commonly found in EHRs (International Classification of Diseases 9 codes) and text

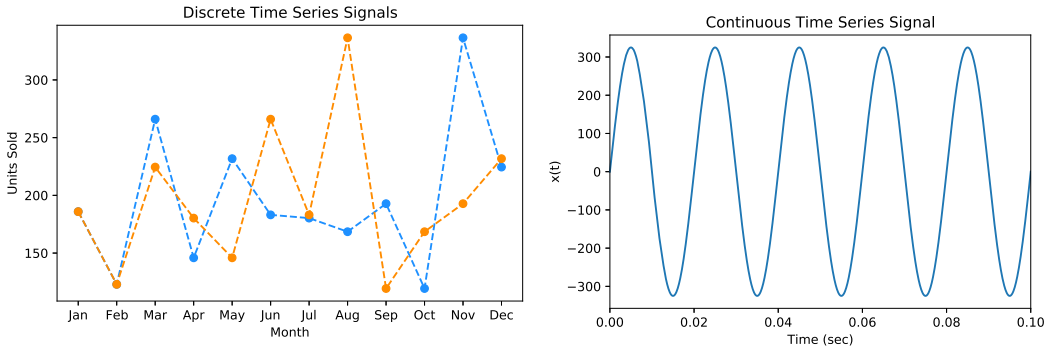

Fig. 3. Example plots of discrete (left) and continuous time series (right).

generation. A continuous time series has a data value corresponding to every moment in time. Continuous data generation is concerned with generating a real-valued signal x with temporal dependencies where $x \in \mathbb{R}$. Figure 3 presents examples of discrete and continuous time series signals.

Challenges with discrete time series generation. GANs struggle with discrete data generation due to the zero gradient nearly everywhere—that is, the distribution on discrete objects are not differentiable with respect to their parameters [52, 108]. This limitation makes the generator untrainable using backpropagation alone. The generator starts with a random sampling and a deterministic transform guided via the gradient of the loss from the discriminator with respect to the output produced by G and the training dataset. This loss leads to a slight change in G's output, pushing it closer to the desired output. Making slight changes to continuous numbers makes sense; adding 0.001 to a value of 10 in financial time series data will bring it to 10.001. However, a discrete token such as the word "penguin"cannot simply undergo the addition of 0.001, as the sum "penguin+0.001" makes no sense. What is important here is the impossibility for the generator to jump from one discrete token to the next because the small change gives the token a new value that does not correspond to any other token over that limited discrete space [37]. This is because there exists zero probability in the space between these tokens, unlike with continuous data.

Challenges with continuous time series generation. Modeling continuous time series data presents a different problem for GANs, which are inherently designed to model continuous data, albeit most commonly in the form of images. The temporal nature of continuous data in time series presents an extra layer of difficulty. Complex correlations exist between the temporal features and their attributes—for example, if using multichannel biometric/physiological data, the electrocardiogram (ECG) characteristics will depend on the individual's age and/or health. In addition, long-term correlations exist in the data, which are not necessarily fixed in dimension compared to image-based data under a fixed dimension. Transforming image dimensions may lead to a degradation in image quality, but it is a recognized practice. This operation becomes more difficult with continuous time series data, as there is no standardized dimension used across time series GAN architectures, which means that benchmarking their performances becomes difficult.

Since their inception in 2014, GANs have shown great success in generating high-quality synthetic images indistinguishable from real images [41, 65, 87]. Although the focus to date has been on developing GANs for improved media generation, there is a growing consensus that GANs can be used for more than image generation and manipulation, which has led to a movement toward generating time series data with GANs.

RNNs (Figure 4, left), due to their loop-like structure, are perfect for sequential data applications but by themselves lack the ability to learn long-term dependencies that might be crucial in forecasting future values based on the past. **Long short-term memory (LSTM)** networks (Figure 4,

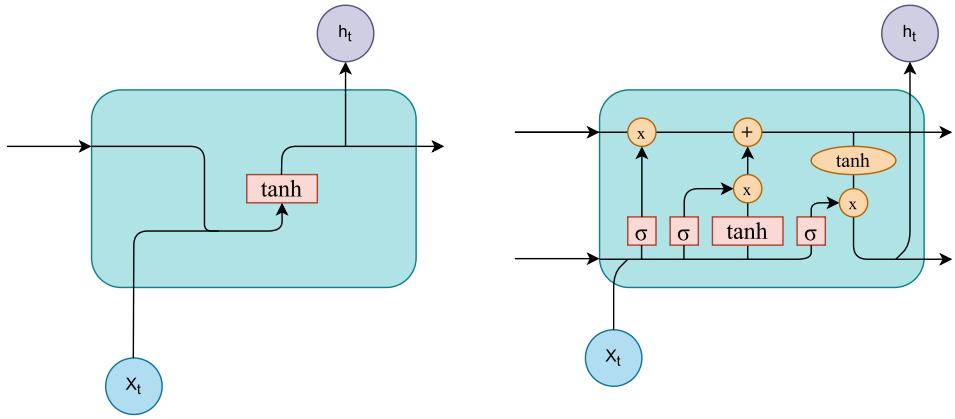

Fig. 4. Block diagram of a standard RNN (left) and an LSTM cell (right).

right) are a specific kind of RNN that have the ability to remember information for long periods of time and, in turn, learn these long-term dependencies that the standard RNN is not capable of doing. In most work reviewed in this article, the majority of the RNN-based architectures are utilizing the LSTM cell.

RNNs can model sequential data such as financial data, medical data, text, and speech, and they have been the foundational architecture for time series GANs. A **recurrent generative adversarial network (RGAN)** was first proposed in 2016. The generator contained a recurrent feedback loop that used both the input and hidden states at each timestep to generate the final output [54]. RGANs often utilize LSTM NNs in their generative models to avoid the vanishing gradient problem associated with more traditional recurrent networks [53]. In the section that follows, we present time series GANs that have either contributed significantly to this space or have made some of the most recent novel advancements in addressing the challenges mentioned previously.

4.1 Discrete-Variant GANs

4.1.1 Sequence GAN (SeqGAN) (Sept. 2016). Yu et al. [108] proposed a sequential data generation framework [108] that could address the issues with generating discrete data as mentioned previously in Section 4. This approach outperformed previous methods for generative modeling on real-world tasks, including a maximum likelihood estimation (MLE)-trained LSTM, scheduled sampling [6], and policy gradient with bilingual evaluation understudy (PG-BLEU) [79]. SeqGAN's generative model comprises RNNs with LSTM cells, and its discriminative model is a **convolutional neural network (CNN)**. Given a dataset of structured sequences, the authors train G to produce a synthetic sequence $Y_{1:T} = (y_1 \ldots, y_t \ldots, y_T), y_t \in \mathcal{Y}$ where $\mathcal{Y}$ is defined as the vocabulary of candidate tokens. G is updated by a policy gradient and **Monte Carlo (MC)** search on the expected reward from D (Figure 5). The authors used two datasets for their experiments. A Chinese poem dataset [62] and a Barack Obama Speech dataset [102] with Adam optimizers and a batch size of 64. Their experiments are available online.[1]

Although the purpose of SeqGAN is to generate discrete sequential data, it opened the door to other GANs in generating continuous sequential and time series data. The authors use a synthetic dataset whose distribution is generated from a randomly initialized LSTM following a normal distribution. They also compare the generated data to real-world examples of poems,

[1]SeqGAN GitHub: https://github.com/LantaoYu/SeqGAN/.

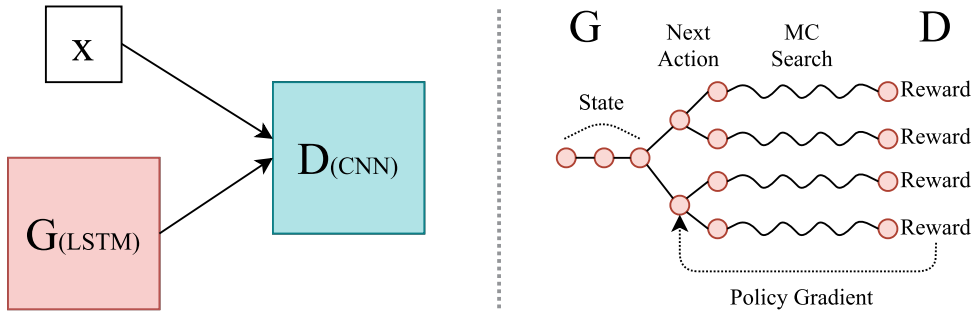

Fig. 5. SeqGAN: D is trained over real and generated data (left), whereas G is trained by policy gradient where the final reward signal is provided by D and is passed back to the intermediate action value via MC search (right).

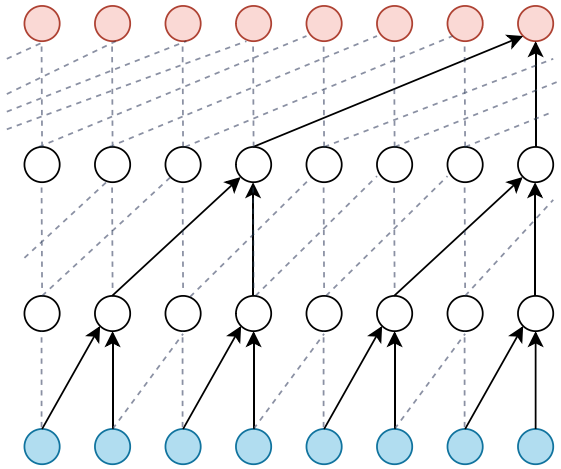

Fig. 6. Dilated causal convolutional layer.

speech-language, and music. SeqGAN showed competitive performance in generating the sequences and contributed heavily toward the further development of the continuous sequential GANs.

4.1.2 Quant GAN (July 2019). Quant GAN is a data-driven model that aims to capture long-range dependencies in financial time series data such as volatility clusters. Both the generator and discriminator use **temporal convolutional networks (TCNs)** with skip connections [101], which are essentially dilated causal convolutional networks. They have the advantage of being suited to model long-range dependencies in continuous sequential data. The generator function is a novel stochastic volatility neural network that consists of a volatility and drift TCN. Temporal blocks are the modules used in the TCN that consist of two dilated causal convolutions layers (Figure 6) and two parametric rectified linear units (PReLU) as activation functions. Data generated by G is passed to D to produce outputs, which can then be averaged to give an MC estimate of D's loss function. The authors used a dataset of daily spot prices of the S&P 500 from May 2009 until December 2018.

The authors aim to capture long-range dependencies in financial time series; however, modeling the series in continuous time over these long time frames would blow up the models'

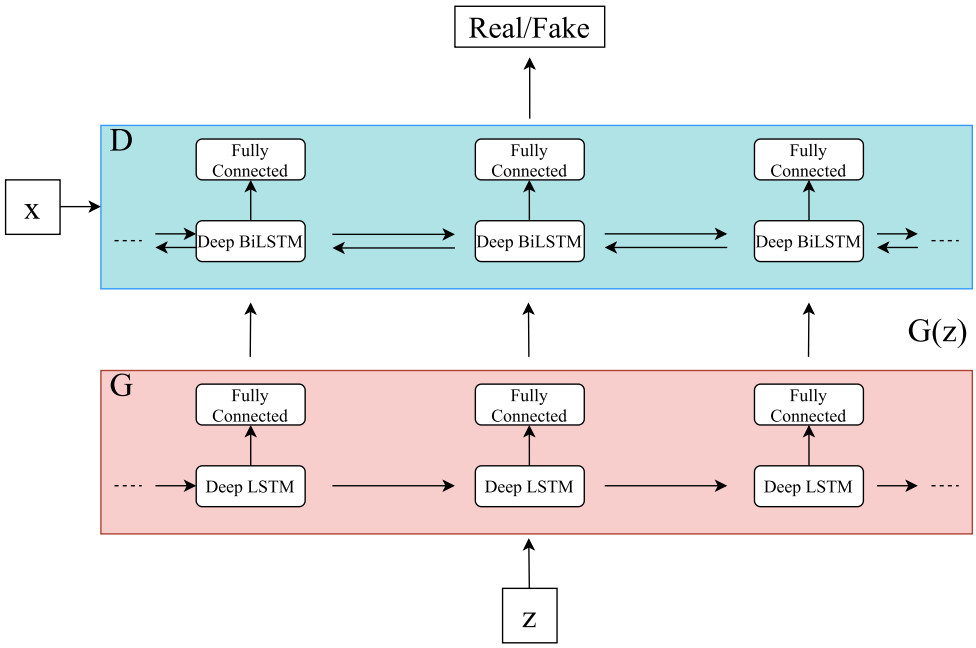

Fig. 7. Structure of C-RNN-GAN's generator and discriminator.

computational complexity. Therefore, this method models the time series in discrete time. The authors report that this approach is capable of outperforming more conventional models from mathematical finance (constrained stochastic volatility NN and generalized AR conditional heteroskedasticity (GARCH) [7]) but state that there remain issues that need to be resolved for this approach to become widely adopted. One such issue concerns the need for a unified metric for quantifying the performance of these GANs, which is a point we discuss further in Section 6.

4.2 Continuous-Variant GANs

Training Stability Developments

4.2.1 Continuous RNN-GAN (C-RNN-GAN) (Nov. 2016). In previous works, RNNs have been applied to modeling music but have generally used a symbolic representation to model this type of sequential data. Mogren [74] proposed the C-RNN-GAN (Figure 7), one of the first examples of using GANs to generate continuous sequential data. The generator is an RNN, and the discriminator a bidirectional RNN, which allows the discriminator to take the sequence context in both directions. The RNNs used in this work were two stacked LSTM layers, with each cell containing 350 hidden units. The loss functions can be seen in Equations (5) and (6), where $z^{(i)}$ is a sequence of uniform random vectors in $[0, 1]^k$, and $x^{(i)}$ is a sequence from the training data. k is the dimensionality of the data in the random sequence.

$$L_G = \frac{1}{m}\sum_{i=1}^{m} log(1 - D(G(z^{(i)}))) \tag{5}$$

$$L_D = \frac{1}{m}\sum_{i=1}^{m}[-logD(x^{(i)}) - log(1 - D(G(z^{(i)})))] \tag{6}$$

The C-RNN-GAN is trained with backpropagation through time (BPTT) and mini-batch stochastic gradient descent with L2 regularization on the weights of both G and D. Freezing was applied to both G and D when one network becomes too strong relative to the other. The dataset used was 3,697 midi files from 160 different composers of classical music with a batch size of 20. Adam and gradient descent optimizers were used during training; full implementation details are available online.[2] Overall, the C-RNN-GAN was capable of learning the characteristics of continuous sequential data and, in turn, generate music. However, the author stated that their approach still needs work, particularly in rigorous evaluation of the generated data quality.

4.2.2 Noise Reduction GAN (NR-GAN) (Oct. 2019). NR-GAN is designed for noise reduction in continuous time series signals but more specifically has been implemented for noise reduction in mice electroencephalogram (EEG) signals [90]. This dataset was provided by the International Institute for Integrative Sleep Medicine (IIIS). EEG is the measure of the brain's electrical activity, and it commonly contains significant noise artifact. NR-GAN's core idea is to reduce or remove the noise present in the frequency domain representation of an EEG signal. The architecture of G is a two-layer 1D CNN with a fully connected layer at the output. D contains almost the same two-layer 1D CNN structure with the fully connected layer replaced by a softmax layer to calculate the probability that the input belongs to the training set. The loss functions are defined in Equations (7) and (8) as

$$G_{loss} = \sum_{x \in S_{ns}} [log(1 - D(G(x))) + \alpha \|x - G(x)\|^2], \tag{7}$$

$$D_{loss} = \sum_{x \in S_{ns}} [log(D(G(x)))] + \sum_{y \in S_{cs}} [log(1 - D(y))], \tag{8}$$

where S_{ns} and S_{cs} are the noisy and clear EEG signals, respectively. α is a hyperparameter that essentially controls the aggressiveness of noise reduction; the authors chose a value of $\alpha = 0.0001$.

For this work, the generator does not sample from a latent space; rather, it attempts to generate the clear signal from the noisy EEG signal input (Figure 8). The authors found that the NR-GAN is competitive with classical frequency filters in terms of noise reduction. They also state that the experimental conditions may favor the NR-GAN and list some limitations in terms of the amount of noise NR-GAN can handle and the influence of α. However, this is a novel method for noise reduction in continuous sequential data using GANs.

4.2.3 TimeGAN (Dec. 2019). TimeGAN provides a framework that utilizes both the conventional unsupervised GAN training method and the more controllable supervised learning approach [107]. By combining an unsupervised GAN network with a supervised AR model, the network aims to generate time series with preserved temporal dynamics. The architecture of the TimeGAN framework is illustrated in Figure 9. The input to the framework is considered to consist of two elements: a static feature and a temporal feature. $\mathbf{s}$ represents a vector of static features and $\mathbf{x}$ of temporal features at the input to the encoder. The generator takes a tuple of static and temporal random feature vectors drawn from a known distribution. The real and synthetic latent codes $\mathbf{h}$ and $\hat{\mathbf{h}}$ are used to calculate the supervised loss element of this network. The discriminator receives the tuple of real and synthetic latent codes and classifies them as either real (y) or synthetic ($\hat{y}$), and the $\tilde{}$ operator denotes the sample as either real or fake.

The three losses used in TimeGAN are calculated as follows.

$$L_{reconstruction} = \mathbb{E}_{s, x_{1:T} \sim p} \left[\|s - \tilde{s}\|_2 + \sum_t \|x_t - \tilde{x}_t\|_2 \right] \tag{9}$$

[2]C-RNN-GAN GitHub: https://github.com/olofmogren/c-rnn-gan/.

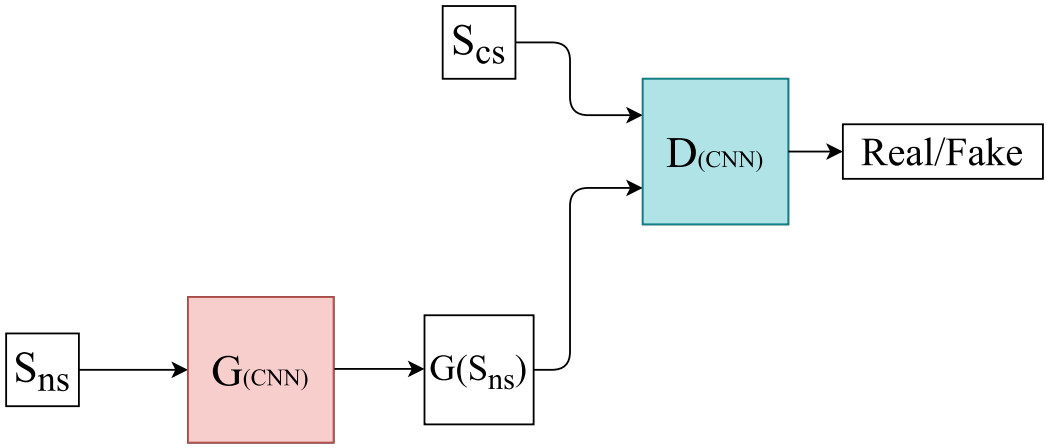

Fig. 8. NR-GAN architecture with noisy EEG input S_{ns} and clean input data S_{cs}.

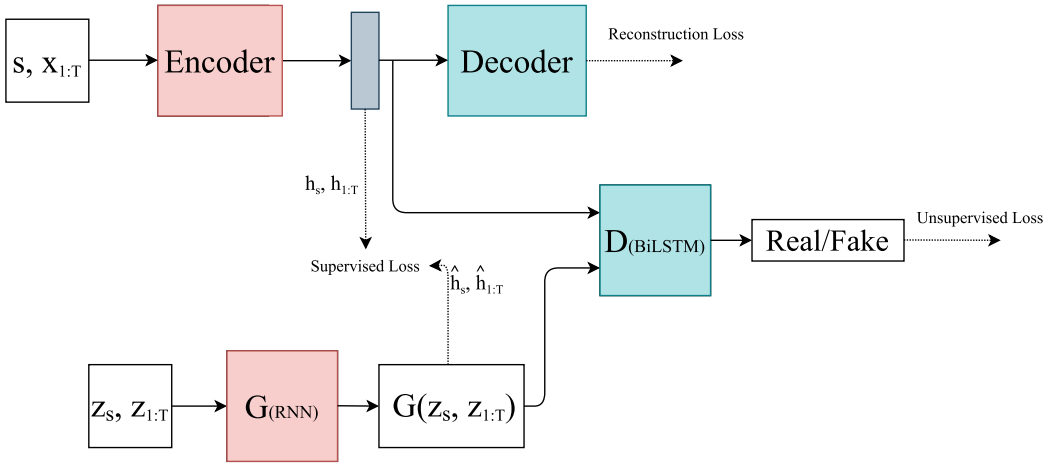

Fig. 9. TimeGAN architecture.

$$L_{unsupervised} = \mathbb{E}_{s,x_{1:T}\sim p}\left[log(y_S) + \sum_t log(y_t)\right] + \mathbb{E}_{s,x_{1:T}\sim \hat{p}}\left[log(1-\hat{y}_S) + \sum_t log(1-\hat{y}_t)\right] \quad (10)$$

$$L_{supervised} = \mathbb{E}_{s,x_{1:T}\sim p}\left[\sum_t \|h_t - g_X(h_S, h_{t-1}, z_t)\|_2\right] \quad (11)$$

The creators of TimeGAN conducted experiments on generating sine waves, stocks (daily historical Google stocks data from 2004 to 2019), energy (UCI Appliances energy prediction dataset) [26], and event (private lung cancer pathways dataset) datasets. A batch size of 128 and Adam optimizer were used for training, and implementation details are available online.[3] The authors demonstrated

[3]TimeGAN GitHub: https://github.com/jsyoon0823/TimeGAN.

improvements over other state-of-the-art time series GANs such as RCGAN, C-RNN-GAN, and WaveGAN.

4.2.4 Conditional Sig-Wasserstein GAN (SigCWGAN) (June 2020) . A problem addressed by Ni et al. [76] is that long time series data streams can greatly increase the dimensionality requirements of generative modeling, which may render such approaches infeasible. To counter this problem, the authors develop a metric named *Signature Wasserstein-1* (Sig-W_1) that captures time series models' temporal dependency and uses it as a discriminator in a time series GAN. It provides an abstract and universal description of complex data streams and does not require costly computation like the Wasserstein metric. A novel generator is also presented that is named *conditional autoregressive feed-forward neural network* (AR-FNN), which captures the AR nature of the time series. The generator is capable of mapping past series and noise into future series. For a rigorous mathematical description of their method, the interested reader should consult the work of Ni et al. [76].

For the AR-FNN generator, the idea is to obtain the step-q estimator $\hat{X}^{(t)}_{t+1:t+q}$. The loss function for D is defined as

$$L(\theta) = \sum_t \left| \mathbb{E}_\mu \left[S_M(X_{t+1:t+q}) | X_{t-p+1:t} \right] - \mathbb{E}_v \left[S_M(\hat{X}^{(t)}_{t+1:t+q}) | X_{t-p+1:t} \right] \right|, \tag{12}$$

where v and μ are the conditional distributions induced by the real data and synthetic generator, respectively. $X_{t-p+1:t}$ is the true past path, $\hat{X}^{(t)}_{t+1:t+q}$ is the forecasted next path, and $X_{t+1:t+q}$ is the true forecast value. S_M is the truncated signature of path X of degree M. Further details of the Ni's algorithm can be found in the appendix of their original paper [76]. SigCWGAN eliminates the problem of approximating a costly D and simplifies training. It is reported to achieve state-of-the-art results on synthetic and empirical datasets compared to TimeGAN, RCGAN, and generative moment matching networks (GMMNs) [68]. The empirical dataset consists of the S&P 500 index (SPX) and Dow Jones index (DJI) and their realized volatility, which is retrieved from the Oxford-Man Institute's "realized library" [55]. A batch size of 200 with the Adam optimizer was used for training.[4]

4.2.5 Decision-Aware Time Series Conditional GAN (DAT-CGAN) (Sept. 2020). This framework is designed to provide support for end users' decision processes, specifically in financial portfolio choices. It uses a multi-Wasserstein loss on structured decision-related quantities [91]. The discriminator loss and generator loss are defined in Equations (13) and (14), respectively. For further details on the loss functions, see Section 3 of the original paper [91] and Equations (15) through (18).

$$\inf_\eta \sup_{\gamma_k, \theta_{j,k}} \sum_{k=1}^{K} \omega_k \left(\mathbb{E}^r_k - \mathbb{E}^{G_\eta}_k \right) + \sum_{k=1}^{K} \sum_{j=1}^{J} \lambda_{j,k} \left(\mathbb{E}^{f,R}_{j,k} - \mathbb{E}^{f,G_\eta}_{j,k} \right) \tag{13}$$

$$\inf_\eta - \sum_k \omega_k \mathbb{E}^{G_\eta}_k - \sum_{k,j} \lambda_{j,k} \mathbb{E}^{f,G_\eta}_{j,k} \tag{14}$$

$$\mathbb{E}^r_k = \mathbb{E}_{r_{t+k} \sim P(r_{t+k}|x_t)}[D_{\gamma k}(r_{t+k}, x_t)] \tag{15}$$

$$\mathbb{E}^{G_\eta}_k = \mathbb{E}_{z_{t,k} \sim P(z_{t,k})}[D_{\gamma k}(r'_{t,k}, x_t)] \tag{16}$$

$$\mathbb{E}^{f,R}_{j,k} = \mathbb{E}_{R_{t,k} \sim P(R_{t,k}|x_t)}[D_{\theta_{j,k}}(f_{j,k}(R_{t,k}, x_t), x_t)] \tag{17}$$

$$\mathbb{E}^{f,G_\eta}_{j,k} = \mathbb{E}_{Z_{t,k} \sim P(Z_{t,k})}[D_{\theta_{j,k}}(f_{j,k}(R'_{t,k}, x_t), x_t)] \tag{18}$$

[4]SigCWGAN GitHub: https://github.com/SigCGANs/Conditional-Sig-Wasserstein-GANs/.

We offer a full description of all terms used in Equations (13) and (14). $D_{\gamma k}$ is the discriminator for the data at look ahead period k with respect to parameters γ. G_η is the generator with parameters η. As this is the conditional case, x_t is the conditioning variable containing relevant information up to time t. $r'_{t,k} = G_\eta(z_{t,k}, x_t)$ is defined as the synthetic data at look ahead point k where the noise is $z_{t,k}$. The discriminator for decision-related quantity j at look ahead period k with respect to parameters $\theta_{j,k}$ is defined as $D_{\theta_{j,k}}$. These decision-related quantities may include mean and covariance, for example. $f_{j,k}(R_{t,k}, x_t)$ represents the decision-related quantity. Finally, ω_k and $\lambda_{j,k}$ are weights and inf and sup are the infimum and supremum or greatest lower bound and least upper bound of a non-empty subset, respectively.

The generator is a two-layer feed-forward NN for each input—assets in this case. G outputs asset returns that are used to compute decision-related quantities. These quantities are fed into D, which is also a two-layer feed-forward NN. Further details about the architecture can be found in the appendix of the work of Sun et al. [91]. The dataset used is daily price data for each of four U.S. Exchange-traded funds (ETFs): Material (XLB), Energy (XLE), Financial (XLF), and Industrial (XLI) ETFs, from 1999 to 2016. The authors found this model capable of high-fidelity time series generation that supports decision processes by end users due to incorporating a decision-aware loss function. However, this approach's limitation is that the computational complexity of this model is vast and requires 1 month of training time for a single generative model.

Privacy Developments just skip

4.2.6 Recurrent Conditional GAN (RCGAN) (2017). RCGAN for continuous data generation [31] differs architecturally from the C-RNN-GAN. Although the RNN LSTM is used, the discriminator is unidirectional, and the outputs of G are not fed back as inputs at the next timestep. There is also additional information that the model is conditioned on, which makes for a conditional RGAN; see the layout of the model in Figure 10. The purpose of the RCGAN and RGAN in this work is to generate continuous time series with a focus on medical data intended for use in downstream tasks, and this was one of the first works in this area. The loss functions can be seen in Equations (19) and (20), where CE is the average cross-entropy between two sequences. X_n are samples drawn from the training dataset. y_n is the adversarial ground truth; for real sequences, it is a vector of 1s, and conversely, for generated or synthetic sequences, it is a vector of 0s. Z_n is a sequence of points sampled from the latent space, and the valid adversarial ground truth is written here as $\mathbf{1}$.

$$D_{loss}(X_n, y_n) = -CE(D(X_n), y_n) \tag{19}$$

$$G_{loss}(Z_n) = D_{loss}(G(Z_n), \mathbf{1}) = -CE(D(G(Z_n)), \mathbf{1}) \tag{20}$$

In the conditional case, the inputs to D and G are concatenated with some conditional information c_n. This variant of an RNN-GAN facilitates the generation of a synthetic continuous time series dataset with associated labels. Experiments were carried out on generated sine waves, smooth functions sampled from a Gaussian process with a zero-valued mean function, the MNIST dataset as a sequence, and the Philips eICU database [83]. A batch size of 28 with Adam and gradient descent optimizers was used for training. The authors propose a novel method for evaluating their model, which is discussed further in Section 6. Full experimental details can be found online.[5]

4.2.7 Sequentially Coupled GAN (SC-GAN) (April 2019). SC-GAN aims to generate patient-centric medical data to inform of a patient's current state and generate a recommended medication dosage based on the state [97]. It consists of two coupled generators tasked with producing two

[5]RCGAN GitHub: https://github.com/ratschlab/RGAN/.

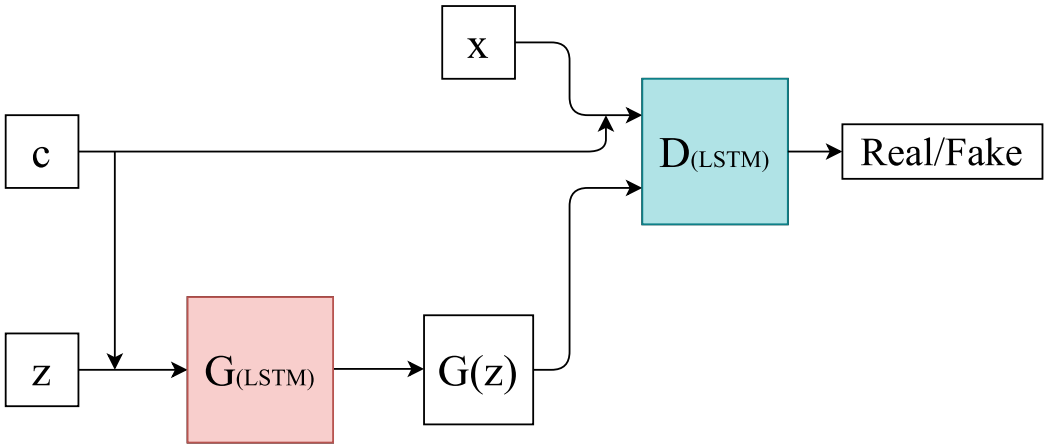

Fig. 10. RCGAN architecture with conditional input **c**, input data **x**, and latent variable **z**.

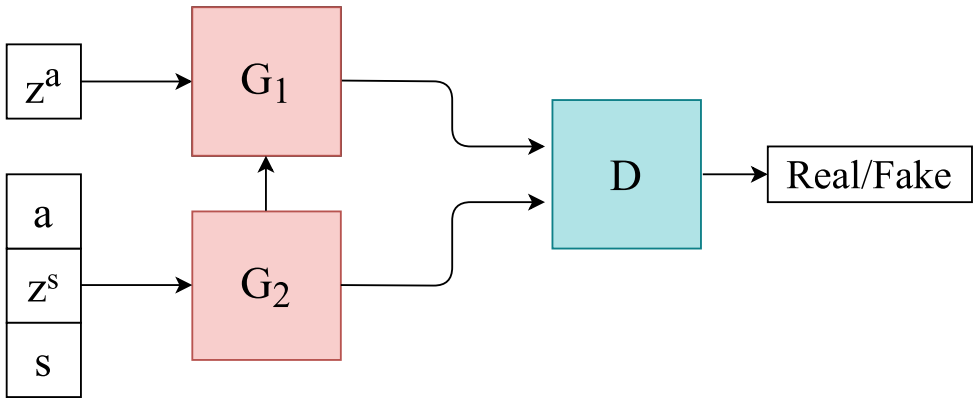

Fig. 11. SC-GAN architecture.

outcomes: one for the current state of an individual and the other for a recommended medication dosage based on the individual's state. The discriminator is a two-layer bidirectional LSTM, and the coupled generators are both two-layer unidirectional LSTMs. Figure 11 presents further details of the architecture.

G_1 generates the recommended medication dosage data $(\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_T)$ with a combined input of the sequential continuous patient state data $(\mathbf{s}_0, \mathbf{s}_1, \ldots, \mathbf{s}_{T-1})$ and a random noise sequence $(\hat{\mathbf{z}}_0^a, \hat{\mathbf{z}}_1^a, \ldots, \hat{\mathbf{z}}_{T-1}^a)$ sampled from a uniform distribution. At each timestep t, the input $\mathbf{z}_t^a$ of G_1 is the concatenation of $\mathbf{s}_t$ and $\hat{\mathbf{z}}_t^a$.

Conversely, G_2 is tasked with generating the patient state data $\mathbf{s}_t$ and at each timestep the input $\mathbf{z}_t^s$ is the concatenation of $\mathbf{s}_{t-1}$, $\mathbf{a}_{t-1}$ and $\hat{\mathbf{z}}_t^s$. This means that the outputs from G_1 and G_2 are the inputs to one another. Combining the generators together leads to the following loss function:

$$L_G = \frac{1}{N}\frac{1}{T}\sum_{i=1}^{N}\sum_{t=1}^{T} log(1 - D(G(\mathbf{z}_{i,t}))), \tag{21}$$

$$G(\mathbf{z}_{i,t}) = \left[G_1(\mathbf{z}_{i,t}^a); G_2(\mathbf{z}_{i,t}^s)\right], \tag{22}$$

where N is the number of patients and T is the time length of the patient record. The SC-GAN has a supervised pre-training step for the generators to avoid an excessively strong D that uses the least-squares loss.

The discriminator is tasked with classifying the sequential patient-centric records as real or synthetic, and the loss function is defined as

$$L_D = -\frac{1}{N}\frac{1}{T}\sum_{i=1}^{N}\sum_{t=1}^{T}(logD(\mathbf{x}_{i,t}) + log(1 - D(G(\mathbf{z}_{i,t})))), \tag{23}$$

where $\mathbf{x}_{i,t} = [\mathbf{s}_t; \mathbf{a}_t]$. This model contains novel coupled generators that coordinate to generate patient state and medication dosage data. It has performance close to real data for the treatment recommendation task. The dataset used in this experiment is MIMIC-III [56]. The authors benchmark their SC-GAN against variants of SeqGAN, C-RNN-GAN, and RCGAN and observe their model to be the best performing for this specific use case.

4.2.8 Synthetic Biomedical Signals GAN (SynSigGAN) (Dec. 2020). SynSigGAN is designed to generate different kinds of continuous physiological/biomedical signal data [49]. It is capable of generating ECG, EEG, electromyography (EMG), and photoplethysmography (PPG) from the MIT-BIH Arrhythmia database [75], Siena Scalp EEG database [23], and BIDMC PPG and Respiration dataset [82]. A novel GAN architecture is proposed here that uses a **bidirectional grid long short-term memory (BiGridLSTM)** for the generator (Figure 12) and a CNN for the discriminator. The BiGridLSTM is a combination of a double GridLSTM (a version of LSTM that can represent the LSTMs in a multidimensional grid) with two directions that can combat the gradient phenomenon from two dimensions and are found to work well in time sequence problems. The authors used the value function defined previously in Equation (2).

SynSigGAN is capable of capturing the different physiological characteristics associated with each of these signal types and has demonstrated an ability to generate biomedical time series data with a max sequence length of 191 data points. The authors also present a preprocessing stage to clean and refine the biomedical signals in this work. They compare their architecture to several variants (BiLSTM-GRU, BiLSTM-CNN GAN, RNN-AE GAN, Bi-RNN, LSTM-AE, BiLSTM-MLP, LSTM-VAE GAN, and RNN-VAE GAN) and found the BiGrid-LSTM as the best-performing model.

Evaluation Developmentsskip to here

As evaluating GANs has been identified as one of their major challenges, we discuss standard evaluation metrics and novel developments formally in Section 6.

5 APPLICATIONS

We have discussed the two classes of time series GANs and their contribution to solving the challenges presented in Section 3.2. Now we will list the wide-ranging applications of time series GANs and the benefits of such to both research and industry.

5.1 Data Augmentation

It is common knowledge in the deep learning community that GANs are among the methods of choice when discussing data augmentation. Reasons for augmenting datasets range from increasing the size/variety of small and imbalanced datasets [2, 44, 59, 77] to reproducing restricted datasets for dissemination.

A well-defined solution to the data shortage problem is transfer learning, and it works well in domain adaptation, which has led to advancements in classification and recognition problems [78]. However, it has been found that augmenting datasets with GANs can lead to further improvements

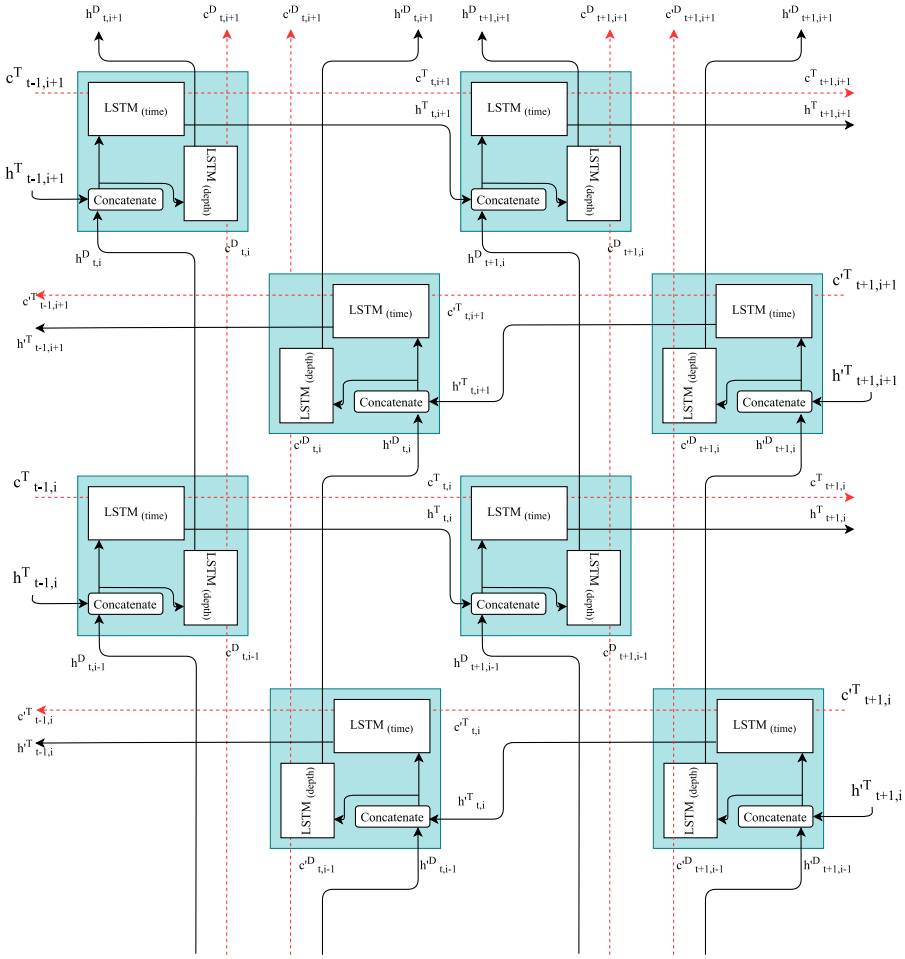

Fig. 12. Architecture of BiGridLSTM with LSTM blocks for the time and depth dimension. The prime (′) symbol indicates reverse in the figure as in the work of Fei and Tan [34].

in certain classification and recognition tasks [110]. Data synthesized by a GAN can adhere to stricter privacy measures discussed in Section 7. This further demonstrates the advantages of augmenting your training dataset with GANs over implementing transfer learning with a pre-trained model from a different domain on a smaller dataset.

Many researchers find that accessing datasets for their deep learning research and models to be time-consuming, laborious work, particularly when the research is concerned with personal sensitive data. Often medical and clinical data are presented as continuous sequential data that can only be accessed by a small contingent of researchers who are not at liberty to disseminate their research openly. This, in turn, may lead to stagnation in the research progress in these domains.

Fortunately, we are beginning to see the uptake of GANs applied to time series with these types of medical and physiological data [12, 21, 31, 49, 111]. Brophy [11] shows that dependent multivariate continuous high-fidelity physiological signal generation is capable via GANs, demonstrating the impressive capability of these networks. Figure 13 presents an example of the real input and synthetic generated data.

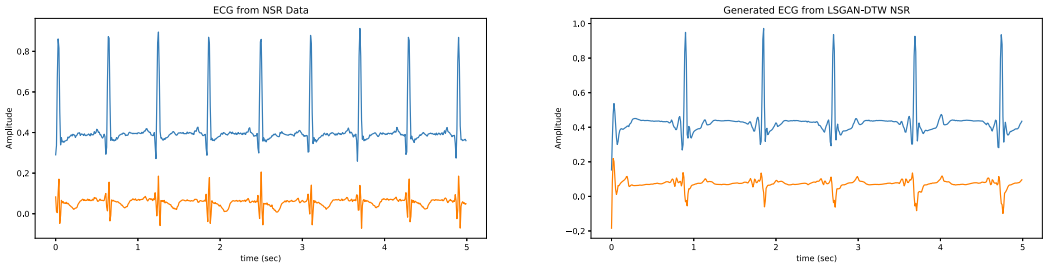

Fig. 13. An example of dependent multichannel ECG data (left) and generated ECG from a multivariate GAN (right) [11]. NSR indicates the training dataset, which is the normal sinus rhythm. The generated data is produced by a GAN named by the authors as *LSGAN-DTW*.

Of course, this is not comprehensive coverage of the research using time series GANs for data synthesis and augmentation. GANs have been applied to time series data for a plethora of use cases.

Audio generation (both music and speech) and text-to-speech (TTS) [57] have been popular areas for researchers to explore with GANs. The C-RNN-GAN described in Section 4.2.1 was one of the seminal works to apply GANs to generating continuous sequential data in the form of music.

In the financial sector, GANs have been implemented to generate data and predict/forecast values. Wiese et al. [101] implemented a GAN to approximate financial time series in discrete time. Ni et al. [76] designed a decision-aware GAN that generates synthetic data and supports decision processes to financial portfolio selection of end users.

Other time series generation/prediction methods range from estimating soil temperature [67] to predicting medicine expenditure based on the current state of patients [58].

5.2 Imputation really what we want!

In real-world datasets, missing or corrupt data is an all too common problem that leads to downstream problems. These issues manifest themselves in further analytics of the dataset and can induce biases in the data. Common methods of dealing with missing or corrupted data in the past have been the deletion of data streams containing the missing segments, statistical modeling of the data, or machine learning imputation approaches. Looking at the latter, we review the work in imputing these data using GANs. Guo et al. [42] designed a GAN-based approach for multivariate time series imputation. Figure 14 presents an example of imputed data from a toy experiment [12].

5.3 Denoising

Artifacts induced in time series data often manifest themselves as noise in the signals. This has become an ever-present challenge in further processing and analytical applications. Corrupted data can cause biases in the datasets or lead to degradation in the performance of critical systems such as those used for health applications. Common methods for dealing with noise include the use of adaptive linear filtering. Another approach recently explored in the work of Sumiya et al. [90] used GANs as a noise-reduction technique in EEG data. Their experiments showed that their proposed NR-GAN (Section 4.2.2) was capable of competitive noise reduction performance compared to more traditional frequency filters.

5.4 Anomaly Detection

Detecting outliers or anomalies in time series data is an important part of many real-world systems and sectors. Whether it is detecting unusual patterns in physiological data that may be a precursor

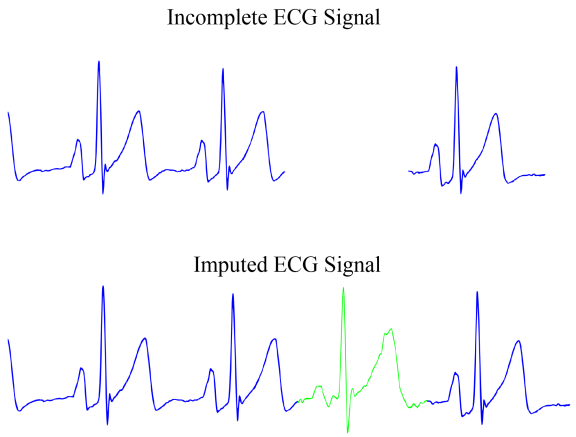

Fig. 14. An example of the incomplete corrupted time series (top) and imputed signal (bottom).

to some more malicious condition or detecting irregular trading patterns on the stock exchange, anomaly detecting can be vital to keeping us informed on important information. Statistical measures of non-stationary time series signals may achieve good performance on the surface, but they might also miss some important outliers present in deeper features. They may also struggle in exploiting large unlabeled datasets; this is where the unsupervised deep learning approaches can outperform the conventional methods. Zhu et al. designed a GAN algorithm for anomaly detection in time series data (ECG and taxi dataset) with LSTMs and GANs, which achieved superior performance compared to conventional, more shallow approaches [112]. Similar approaches have been applied to detect cardiovascular diseases [69], in cyber-physical systems to detect nefarious players [66], and even irregular behaviors such as stock manipulation on the stock markets [63].

5.5 Other Applications

Some works have utilized image-based GANs for time series and sequential data generation by first converting their sequences to images via some transformation function and training the GAN on these images. Once the GAN converges, similar images can be generated; then, a sequence can be retrieved using the inverse of the original transformation function. For example, this approach has been implemented in audio generation with waveforms [16, 24, 60], anomaly detection [18], and physiological time series generation [12].

6 EVALUATION METRICS

As mentioned in Section 3, GANs can be difficult to evaluate, and researchers are yet to agree on what metrics reflect GAN performance best. There have been plenty of metrics proposed in the literature [8], with most of them suited to the computer vision domain. Work is still ongoing to suitably evaluate time series GANs. We can break down evaluation metrics into two categories: qualitative and quantitative. Qualitative evaluation is another term for human visual assessment via the inspection of generated samples from the GAN. However, this cannot be deemed a full evaluation of GAN performance due to the lack of a suitable *objective* evaluation metric. The quantitative evaluation includes the use of metrics associated with statistical measures used for time series analytics and similarity measures such as the Pearson correlation coefficient (PCC), percent root mean square difference (PRD), root mean squared error (RMSE) and mean squared error (MSE), mean relative error (MRE), and mean absolute error (MAE). These metrics are among the

most commonly used for time series evaluation and, as such, are used as a suitable GAN performance metric, as they can reflect the stability between the training data and synthetic generated data, and we show some of these common formulas in Equations (24) through (27).

$$PCC = \frac{\sum_{i=1}^{N}(x_i - \tilde{x})(y_i - \tilde{y})}{\sqrt{\sum_{i=1}^{N}(x_i - \tilde{x})^2 \sum_{i=1}^{N}(y_i - \tilde{y})^2}} \tag{24}$$

$$PRD = \sqrt{\frac{\sum_{i=1}^{N}(x_i - y_i)^2}{\sum_{i=1}^{N}(x_i)^2}} \tag{25}$$

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - y_i)^2} \tag{26}$$

$$MRAE = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{x_i - y_i}{x_i - f_i}\right| \tag{27}$$

Across these formulas, x_i is the actual value of the time series x at time/sample i, and y_i is the generated value of the time series y at time/sample i. $\tilde{x}$ and $\tilde{y}$ represents the mean values of x and y, respectively. f_i is used in the MRAE calculation for the forecast value at time i of a chosen benchmark model. In general, f_i can be chosen to be y_{i-1} for non-seasonal time series and y_{i-M} for seasonal time series, where M is the seasonal period of x.

Several metrics have become well-established choices in evaluating image-based GANs, and some of these have permeated through to the sequential and time series GANs such as IS [88], Fréchet distance, and FID [51]. The structural similarity index (SSIM) is a measure of similarity between two images. However, Parthasarathy et al. [80] use this with time series data, as SSIM does not exclude itself from comparing aligned sequences of fixed length.

Of course, some of these metrics are measures of similarities/dissimilarities between two probability distributions, suitable for many types of data, particularly MMD [39]. In the real world, we do not have access to the underlying distributions of data, and therefore we show an empirical estimate of MMD in Equation (28), which is a quite suitable metric for this task across domains:

$$MMD[\mathcal{F}, X, Y] = \left[\frac{1}{m^2}\sum_{i,j=1}^{m} k(x_i, x_j) - \frac{2}{mn}\sum_{i,j=1}^{m,n} k(x_i, y_i) + \frac{1}{n^2}\sum_{i,j=1}^{n} k(y_i, y_j)\right]^{\frac{1}{2}}, \tag{28}$$

where $\mathcal{F}$ is a class F of smooth functions $f : \mathcal{X} \to \mathbb{R}$. Two observations $X := \{x_1, x_2, \ldots, x_n\}$ and $Y := \{y_1, y_2, \ldots, y_n\}$ are drawn from two distributions p and q with m points sampled from p and n from q. Last, k is the kernel function chosen by the user.

Another metric that generalizes well to the sequential data case is the Wasserstein distance. The Wassterstein-1, or Earth Mover distance, shown in Equation (29), describes the cost it takes to move one cumulative distribution function to another while preserving the shape of the functions, which is done by optimizing the transport plan:

$$W_p(\mu, \nu) = \left(\inf_{\gamma \in \grave{}(\mu,\nu)} \int_{X \times Y} d^p(x, y) d\gamma(x, y)\right)^{\frac{1}{p}}, \tag{29}$$

where $\grave{}(\mu, \nu)$ is the set of all transport plans, $d^p(x, y)$ is the distance function, and $d\gamma(x, y)$ is the amount of "mass" to be moved.

The data generated from GANs have been used in downstream classification tasks. Using the generated data together with the training data has led to the Train on Synthetic, Test on

Real (TSTR) and Train on Real, Test on Synthetic (TRTS) evaluation methods, first proposed by Esteban et al. [31]. In scoring downstream classification applications that use both real and generated data, studies have adopted the precision, recall, and F1 scores to determine the classifier's quality and, in turn, the quality of the generated data. Other accuracy measures of classifier performance include the weighted accuracy (WA) and unweighted average recall (UAR).

Often used distance and similarity measures in time series data are the Euclidean distance (ED) and **dynamic time warping (DTW)** algorithms. Multivariate (in)dependent dynamic time warping (MVDTW), implemented in the work of Brophy [11], can determine similarity measures across both dependent and independent multichannel time series signals. The idea behind DTW is to find the minimum cost, or optimal alignment of the warping path via the cumulative distance function. The MVDTW cumulative distance function is given in Equation (30), which is used to find the path that minimizes the warping cost of multivariate time series signals.

$$D(i,j) = \sum_{m=1}^{M} (q_{i,m} - c_{j,m})^2 + min\{D(i-1,j-1), D(i-1,j), D(i,j-1)\} \tag{30}$$

Other metrics used across different applications include:

- *Financial sector*: Autocorrelation function (ACF) score and DY metric.
- *Temperature estimation*: Nash-Sutcliffe model efficiency coefficient (NS), Willmott index of agreement (WI), and the Legates and McCabe index (LMI).
- *Audio generation*: Normalized source-to-distortion ratio (NSDR), source-to interference ratio (SIR), source-to-artifact ratio (SAR), and t-SNE [95].

For a full list of GAN architectures reviewed in this work, their applications, evaluation metrics, and datasets used in their respective experiments, see Table 2. Results for the sine wave and ECG generation using variants of GAN architectures can be found in Tables 3 and 4, respectively.

7 PRIVACY

As well as evaluating the quality of the data, a wide range of methods have been used to evaluate and mitigate the privacy risk associated with synthetic data created by GANs.

7.1 Differential Privacy

The goal of differential privacy is to preserve the underlying privacy of a database. An algorithm or, more specifically, a GAN achieves differential privacy if, by looking at the generated samples, we cannot identify whether the samples were included in the training set. As GANs attempt to model the training dataset, the problem of privacy lies in capturing and generating useful information about the training set population without the possibility of linkage from generated sample to an individual's data [27].

As we have addressed previously, one of the main goals of GANs is to augment existing under-resourced datasets for use in further downstream applications such as upskilling of clinicians where healthcare data is involved. These personal sensitive data must contain privacy guarantees, and the rigorous mathematical definition of differential privacy [28] offers this assurance.

Work is ongoing to develop machine learning methods with privacy-preserving mechanisms such as differential privacy. Abadi et al. [1] demonstrated the ability to train deep NNs with differential privacy and implemented a mechanism for tracking privacy loss. Xie et al. Xie2018 proposed a differentially private GAN (DPGAN) that achieved DP by adding noise gradients to the optimizer during the training phase [103].

Table 2. List of GAN Architectures, Their Applications, and Datasets Used in Their Experiments and Evaluation Metrics Used to Judge the Quality of the Respective GANs

Application	GAN Architecture(s)	Dataset(s)	Evaluation Metrics
Medical/physiological generation	LSTM-LSTM [2, 31, 44, 45, 77, 97] LSTM-CNN [11, 21] BiLSTM-CNN [111] BiGridLSTM-CNN [49] CNN-CNN [33, 47] AE-CNN [81] FCNN [104]	EEG, ECG, EHRs, PPG, EMG, speech, NAF, MNIST, synthetic sets	TSTR, MMD, reconstruction error, DTW, PCC, IS, FID, ED, S-WD, RMSE, MAE, FD, PRD, averaging samples, WA, UAR, MV-DTW
Financial time series generation/prediction	TimeGAN [107] SigCWGAN [76] DAT-GAN [91] QuantGAN [101]	S&P 500 index (SPX), Dow Jones index (DJI), ETFs	Marginal distributions, dependencies, TSTR, Wasserstein distance, EM distance, DY metric, ACF score, leverage effect score, discriminative score, predictive score
Time series estimation/prediction	LSTM-NN [67] LSTM-CNN [58] LSTM-MLP [58]	Meteorological data, Truven MarketScan dataset	RMSE, MAE, NS, WI, LMI
Audio generation	C-RNN-GAN [74] TGAN (variant) [16] RNN-FCN [109] DCGAN (variant) [60] CNN-CNN [57]	Nottingham dataset, midi music files, MIR-1K, TheSession, speech	Human perception, polyphony, scale consistency, tone span, repetitions, NSDR, SIR, SAR, FD, t-SNE, distribution of notes
Time series imputation/repairing	MTS-GAN [42] CNN-CNN [84] DCGAN (variant) [43] AE-GRUI [71] RGAN [92] FCN-FCN [15] GRUI-GRUI [70]	TEP, point machine, wind turbine data, PeMS, PhysioNet Challenge 2012, KDD CUP 2018, parking lot data,	Visually, MMD, MAE, MSE, RMSE, MRE, spatial similarity, AUC score
Anomaly detection	LSTM-LSTM [63] LSTM-(LSTM&CNN) [112] LSTM-LSTM (MAD-GAN) [66]	SET50, NYC taxi data, ECG, SWaT, WADI	Manipulated data used as a test set, ROC curve, precision, recall, F1, accuracy
Other time series generation	VAE-CNN [80]	Fixed length time series “vehicle and engine speed”	DTW, SSIM

For novel approaches, the GAN name is given as they have been covered already in Section 4.

7.2 Decentralized/Federated Learning

Distributed or decentralized learning is another method for limiting the privacy risk associated with personal and personal sensitive data in machine learning. Standard approaches to machine learning require that all training data be kept on one server. Decentralized/distributed approaches to GAN algorithms require large communication bandwidth to ensure convergence [5, 46] and

Table 3. Experimental Results Comparing the Performance of Time Series GANs for Sinewave Generation

Architecture	Loss Function	Toy Sine Dataset		
		MMD	DTW	MSE
LSTM-LSTM	BCE	0.9527	91.1071	0.2308
	MSE	0.0078	54.1644	**0.1480**
BiLSTM-LSTM	BCE	0.1215	428.4310	3.0700
	MSE	0.9515	79.5607	0.2362
LSTM-CNN	BCE	0.006	55.3620	0.3154
	MSE	0.5757	86.7357	0.5643
BiLSTM-CNN	BCE	**1.129E-05**	129.9257	0.9193
	MSE	0.4891	43.2694	0.1869
GRU-CNN	BCE	0.0244	**37.1630**	0.2303
	MSE	0.3727	42.7348	0.22823
FC-CNN	BCE	0.0039	58.3565	0.3048
	MSE	0.0117	43.3611	0.2972

Table 4. Experimental Results Comparing the Performance of Time Series GANs for ECG Generation on the MIT-BIH Dataset

Architecture	Loss Function	MIT-BIH Arrhythmia Dataset		
		MMD	DTW	MSE
LSTM-LSTM	BCE	0.9931	30.1816	0.0867
	MSE	0.8842	44.4553	0.1389
BiLSTM-LSTM	BCE	0.9916	22.8634	0.0699
	MSE	0.9737	23.5533	0.0806
LSTM-CNN	BCE	0.5519	**13.0158**	**0.0151**
	MSE	**0.0005**	24.7306	0.0457
BiLSTM-CNN	BCE	0.9246	117.3994	0.2272
	MSE	0.0687	22.6740	0.0586
GRU-CNN	BCE	0.0055	20.4845	0.0335
	MSE	0.7704	108.4124	0.1948
FC-CNN	BCE	0.2068	23.9910	0.0309
	MSE	0.3082	18.2340	0.0212

are also subject to strict privacy constraints. A new method that enables communication efficient collaborative learning on a shared model while keeping all of the training data decentralized is known as *federated learning* [73]. Rasouli et al. [86] applied a federated learning algorithm to a GAN for communication-efficient distributed learning and proved the convergence of their federated learning GAN (FedGAN) [86]. However, it should be noted that they did not experiment with differential privacy in this study but note that it as an avenue of future work.

Combining the preceding techniques of federated learning and differential privacy in developing new GAN algorithms would lead to a fully decentralized private GAN capable of generating data without leakage of private information to the source data. This is clearly an open research avenue for the community.

7.3 Assessment of Privacy Preservation

We can also assess how well the generative model was able to protect our privacy through tests known as attribute and presence disclosure [17]. The latter test is more commonly known in the machine learning space as a membership inference attack. This has become a quantitative assessment of how machine learning models leak information about the individual data records on which they were trained [89]. Membership inference attacks attempt to detect the data that was used to train a target model without the attacker having access to the model's parameters. A nefarious actor creates random records for a target machine learning model. The attacker then feeds each record into the model. The model will return a confidence score, and based on this score, the records will be fine tuned until a higher confidence score is returned. This process will continue until the model returns a very high score, and at this stage the record will be nearly identical to one of the examples used in the training dataset. These steps will be repeated until enough dataset examples are generated. The fake records will then be used to train an ensemble of models to predict whether a data record was used in the training set of the target model.

Hayes et al. [48] carried out membership inference attacks on synthetic images and concluded that for acceptable levels of privacy in the GAN, the quality of the data generated is sacrificed. Conversely, others have followed this approach and found that differential privacy networks can successfully generate data that adheres to differential privacy and resists membership inference attacks without too much degradation in the quality of the generated data [11, 21, 31].

8 DISCUSSION

We have presented a survey of time series GAN variants that have made significant progress in addressing the primary challenges identified in Section 3.2. These GANs introduced the idea of both discrete and continuous sequential data generation and have made incremental improvements over one another via an architecture variant or a modified objective function capable of capturing the spatio-temporal dependencies present in these data types. The loss functions implemented in these works for some architectures will not necessarily generalize to others; hence, they become architecture specific. The architecture choices of the time series GANs affect both the quality and diversity of the data. However, there remain open problems in terms of the practical implementation of the generated data and GANs in real-world applications, particularly in health applications where the performance of these models can directly affect patients' quality of care/treatment.

The "best" GAN architecture and objective function is yet to be determined. This is because humans have manually designed most architectures. As a result, there is growing interest in automated neural architecture search (NAS) methods [30], whereby automating the architecture engineering aspect of machine learning. It is a growing branch of automatic machine learning (AutoML) and automatic deep learning (AutoDL) that seeks to optimize the processes around machine learning. Work has been done in the image domain space with neural architecture search and GANs [35]. This method, named *AutoGAN*, achieved highly competitive performance compared to state-of-the-art human-engineered GANs. This is a promising area for time series GANs; to the authors' knowledge, it is yet unexplored.

As it stands, GANs tend to be application specific; they perform well for their intended purpose but do not generalize well beyond their original domain. Furthermore, a major limitation of time series GANs is the restrictions placed on the length of the sequence specified that the architecture can manage; documented experiments validating how well a time series GAN can adapt to varying data lengths are notably absent at the time of writing. However, glimpses of work in the NLP literature in the form of Transformers [96] have demonstrated some applicability to dealing with varying sequence lengths that may prove beneficial in addressing this issue and might emerge in time series generation given time.

Other aspects not in the scope of this survey article but important to note is how GANs can deal with issues such as scalability and real-time data. Given its importance, we present some draft ideas and direct the interested reader to further resources for full-stack machine learning in general. Thankfully, the emerging practice of machine learning operations (MLOps) addresses most concerns surrounding retraining models once real-time data begins to diverge from the original dataset it was trained on [14, 94]. This can be applied to GANs, whereby the datasets encountered in production can be driven through a metric process to assess divergence from the original data and subsequent data for retraining, allowing for reliable machine learning solutions that scale. For a parallel computing approach, we would consider federated learning, as referenced previously, where you can train the GAN on subsets of the data and can combine the models following training.

9 CONCLUSION

This article reviews a niche but growing use of GANs for time series data based mainly around architectural evolution and loss function variants. We see that each GAN provides application-specific performance and does not necessarily generalise well to other applications—for example, a GAN for generating high-quality physiological time series may not produce high-fidelity audio due to some limitation imposed by the architecture or loss function. A detailed review of the applications of time series GANs to real-world problems has been provided, along with their datasets and the evaluation metrics used for each domain. As stated in the work of Wang et al. [100], GAN-related research for time series lags that of computer vision both in terms of performance and defined rules for generalization of models. This review has highlighted the open challenges in this area and offers directions for future work and technological innovation, particularly for those GAN aspects related to evaluation, privacy, and decentralized learning.

REFERENCES

[1] Martín Abadi, H. Brendan McMahan, Andy Chu, Ilya Mironov, Li Zhang, Ian Goodfellow, and Kunal Talwar. 2016. Deep learning with differential privacy. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS'16)*. 308–318. https://doi.org/10.1145/2976749.2978318

[2] Sherif M. Abdelfattah, Ghodai M. Abdelrahman, and Min Wang. 2018. Augmenting the size of EEG datasets using generative adversarial networks. In *Proceedings of the 2018 International Joint Conference on Neural Networks (IJCNN'18)*. IEEE, Los Alamitos, CA, 1–6. https://doi.org/10.1109/IJCNN.2018.8489727

[3] Hamed Alqahtani, Manolya Kavakli-Thorne, and Gulshan Kumar. 2019. Applications of generative adversarial networks (GANs): An updated review. *Archives of Computational Methods in Engineering* 28 (Dec. 2019), 525–552. https://doi.org/10.1007/s11831-019-09388-y

[4] Martin Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein GAN. *arXiv preprint arXiv:1701.07875*.

[5] Sean Augenstein, H. Brendan McMahan, Daniel Ramage, Swaroop Ramaswamy, Peter Kairouz, Mingqing Chen, Rajiv Mathews, and Blaise Aguera y Arcas. 2020. Generative models for effective ML on private, decentralized datasets. *arXiv:1911.06679 [cs.LG]*.

[6] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. *arXiv:1506.03099* [cs.LG].

[7] Tim Bollerslev. 1986. Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics* 31, 3 (1986), 307–327. https://EconPapers.repec.org/RePEc:eee:econom:v:31:y:1986:i:3:p:307-327.

[8] Ali Borji. 2018. Pros and cons of GAN evaluation measures. *arXiv:1802.03446* [cs.CV].

[9] Ali Borji. 2019. Pros and cons of GAN evaluation measures. *Computer Vision and Image Understanding* 179 (2019), 41–65. https://doi.org/10.1016/j.cviu.2018.10.009

[10] Ali Borji. 2021. Pros and cons of GAN evaluation measures: New developments. *arXiv:2103.09396* [cs.LG].

[11] Eoin Brophy. 2020. Synthesis of dependent multichannel ECG using generative adversarial networks. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM'20)*. ACM, New York, NY, 3229–3232. https://doi.org/10.1145/3340531.3418509

[12] Eoin Brophy, Zhengwei Wang, and Tomas E. Ward. 2019. Quick and easy time series generation with established image-based GANs. *arXiv:1902.05624* [cs.LG].

[13] Fred B. Bryant and Paul R. Yarnold. 1995. Principal-components analysis and exploratory and confirmatory factor analysis. In *Reading and Understanding Multivariate Statistics*. American Psychological Association, Washington, DC, 99–136.

[14] Andriy Burkov. 2020. *Machine Learning Engineering*. True Positive Inc.

[15] Yuanyuan Chen, Yisheng Lv, and Fei-Yue Wang. 2020. Traffic flow imputation using parallel data and generative adversarial networks. *IEEE Transactions on Intelligent Transportation Systems* 21, 4 (April 2020), 1624–1630. https://doi.org/10.1109/TITS.2019.2910295

[16] Ping-Sung Cheng, Chieh-Ying Lai, Chun-Chieh Chang, Shu-Fen Chiou, and Yu-Chieh Yang. 2020. A variant model of TGAN for music generation. In *Proceedings of the 2020 Asia Service Sciences and Software Engineering Conference*. ACM, New York, NY, 40–45. https://doi.org/10.1145/3399871.3399888

[17] Edward Choi, Siddharth Biswal, Bradley Malin, Jon Duke, Walter F. Stewart, and Jimeng Sun. 2017. Generating multi-label discrete patient records using generative adversarial networks. *arXiv:1703.06490*.

[18] Yeji Choi, Hyunki Lim, Heeseung Choi, and Ig-Jae Kim. 2020. GAN-based anomaly detection and localization of multivariate time series data for power plant. In *Proceedings of the 2020 IEEE International Conference on Big Data and Smart Computing (BigComp'20)*. IEEE, Los Alamitos, CA, 71–74. https://doi.org/10.1109/BigComp48618.2020.00-97

[19] Chris Culnane, Benjamin I. P. Rubinstein, and Vanessa Teague. 2017. Health data in an open world. *arXiv:1712.05627*.

[20] Hoang Anh Dau, Eamonn Keogh, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Ann Ratanamahatana, et al. 2018. UCR Time Series Classification Archive. Retrieved September 7, 2022 from https://www.cs.ucr.edu/~eamonn/time_series_data_2018/.

[21] Anne Marie Delaney, Eoin Brophy, and Tomás E. Ward. 2019. Synthesis of realistic ECG using generative adversarial networks. *arXiv:1909.09150*.

[22] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. ImageNet: A large-scale hierarchical image database. In *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, Los Alamitos, CA, 248–255. https://doi.org/10.1109/CVPR.2009.5206848

[23] Paolo Detti, Giampaolo Vatti, and Garazi Zabalo Manrique de Lara. 2020. EEG synchronization analysis for seizure prediction: A study on data of noninvasive recordings. *Processes* 8, 7 (2020), 846. https://doi.org/10.3390/pr8070846

[24] Chris Donahue, Julian McAuley, and Miller Puckette. 2019. Adversarial audio synthesis. *arXiv:1802.04208* [cs.SD].

[25] Georg Dorffner. 1996. Neural networks for time series processing. *Neural Network World* 6 (1996), 447–468.

[26] Dheeru Dua and Casey Graff. 2017. UCI Machine Learning Repository. Retreived September 7, 2022 from http://archive.ics.uci.edu/ml.

[27] Cynthia Dwork. 2006. Differential privacy. In *Proceedings of the 33rd International Conference on Automata, Languages, and Programming—Volume Part II (ICALP'06)*. 1–12. https://doi.org/10.1007/11787006_1

[28] Cynthia Dwork and Aaron Roth. 2014. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science* 9, 3–4 (Aug. 2014), 211–407. https://doi.org/10.1561/0400000042

[29] Khaled El Emam, Elizabeth Jonker, Luk Arbuckle, and Bradley Malin. 2011. A systematic review of re-identification attacks on health data. *PLoS One* 6 (2011), e28071. https://doi.org/10.1371/journal.pone.0028071

[30] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. 2019. Neural architecture search: A survey. *Journal of Machine Learning Research* 20, 1 (2019), 1997–2017.

[31] Cristóbal Esteban, Stephanie L. Hyland, and Gunnar Rätsch. 2017. Real-valued (medical) time series generation with recurrent conditional GANs. *arXiv:1706.02633*.

[32] European Union. 2018. Data Protection Act 2018 (Section36(2)). Retrieved September 7, 2022 from http://www.irishstatutebook.ie/eli/2018/si/314/made/en/pdf.

[33] Fatemeh Fahimi, Zhuo Zhang, Wooi Boon Goh, Kai Keng Ang, and Cuntai Guan. 2019. Towards EEG generation using GANs for BCI applications. In *Proceedings of the 2019 IEEE EMBS International Conference on Biomedical and Health Informatics (BHI'19)*. IEEE, Los Alamitos, CA, 1–4. https://doi.org/10.1109/BHI.2019.8834503

[34] Hongxiao Fei and Fengyun Tan. 2018. Bidirectional grid long short-term memory (BiGridLSTM): A method to address context-sensitivity and vanishing gradient. *Algorithms* 11, 11 (2018), 172. https://doi.org/10.3390/a11110172

[35] Xinyu Gong, Shiyu Chang, Yifan Jiang, and Zhangyang Wang. 2019. AutoGAN: Neural architecture search for generative adversarial networks. *arXiv:1908.03835*.

[36] Liang Gonog and Yimin Zhou. 2019. A review: Generative adversarial networks. In *Proceedings of the 2019 14th IEEE Conference on Industrial Electronics and Applications (ICIEA'19)*. IEEE, Los Alamitos, CA, 505–510. https://doi.org/10.1109/ICIEA.2019.8833686

[37] Ian Goodfellow. 2016. Generative Adversarial Networks for Text. Retrieved September 7, 2022 from https://www.reddit.com/r/MachineLearning/comments/40ldq6/generative_adversarial_networks_for_text/.

[38] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger (Eds.), Vol. 27. Curran Associates, Montréal, Canada. https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf.

[39] Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander Smola. 2012. A kernel two-sample test. *Journal of Machine Learning Research* 13 (March 2012), 723–773.
[40] Jie Gui, Zhenan Sun, Yonggang Wen, Dacheng Tao, and Jieping Ye. 2020. A review on generative adversarial networks: Algorithms, theory, and applications. *arXiv:2001.06937* [cs.LG].
[41] John T. Guibas, Tejpal S. Virdi, and Peter S. Li. 2017. Synthetic medical images from dual generative adversarial networks. *arXiv:1709.01872*.
[42] Zijian Guo, Yiming Wan, and Hao Ye. 2019. A data imputation method for multivariate time series based on generative adversarial network. *Neurocomputing* 360 (Sept. 2019), 185–197. https://doi.org/10.1016/j.neucom.2019.06.007
[43] Lingyi Han, Kan Zheng, Long Zhao, Xianbin Wang, and Huimin Wen. 2020. Content-aware traffic data completion in ITS based on generative adversarial nets. *IEEE Transactions on Vehicular Technology* 69, 10 (Oct. 2020), 11950–11962. https://doi.org/10.1109/TVT.2020.3007025
[44] Shota Harada, Hideaki Hayashi, and Seiichi Uchida. 2018. Biosignal data augmentation based on generative adversarial networks. In *Proceedings of the 2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC'18)*. IEEE, Los Alamitos, CA, 368–371. https://doi.org/10.1109/EMBC.2018.8512396
[45] Shota Harada, Hideaki Hayashi, and Seiichi Uchida. 2019. Biosignal generation and latent variable analysis with recurrent generative adversarial networks. *IEEE Access* 7 (2019), 144292–144302. https://doi.org/10.1109/ACCESS.2019.2934928
[46] Corentin Hardy, Erwan Le Merrer, and Bruno Sericola. 2019. MD-GAN: Multi-discriminator generative adversarial networks for distributed datasets. *arXiv:1811.03850* [cs.LG].
[47] Kay Gregor Hartmann, Robin Tibor Schirrmeister, and Tonio Ball. 2018. EEG-GAN: Generative adversarial networks for electroencephalograhic (EEG) brain signals. *arXiv:1806.01875*.
[48] Jamie Hayes, Luca Melis, George Danezis, and Emiliano De Cristofaro. 2019. LOGAN: Membership inference attacks against generative models. *Proceedings on Privacy Enhancing Technologies* 2019, 1 (2019), 133–152. https://doi.org/10.2478/popets-2019-0008
[49] Debapriya Hazra and Yung-Cheol Byun. 2020. SynSigGAN: Generative adversarial networks for synthetic biomedical signal generation. *Biology* 9, 12 (Dec. 2020), 441. https://doi.org/10.3390/biology9120441
[50] Boris P. Hejblum, Griffin M. Weber, Katherine P. Liao, Nathan P. Palmer, Susanne Churchill, Nancy A. Shadick, Peter Szolovits, Shawn N. Murphy, Isaac S. Kohane, and Tianxi Cai. 2019. Probabilistic record linkage of de-identified research datasets with discrepancies using diagnosis codes. *Scientific Data* 6 (2019), Article 180298, 11 pages. https://doi.org/10.1038/sdata.2018.298
[51] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. 2018. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. *arXiv:1706.08500* [cs.LG].
[52] R. Devon Hjelm, Athul Paul Jacob, Tong Che, Adam Trischler, Kyunghyun Cho, and Yoshua Bengio. 2018. Boundary-seeking generative adversarial networks. *arXiv:1702.08431* [stat.ML].
[53] Sepp Hochreiter and Jurgen Schmidhuber. 1997. Long short-term memory. *Neural Computation* 9, 8 (1997), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735
[54] Daniel Jiwoong Im, Chris Dongjoo Kim, Hui Jiang, and Roland Memisevic. 2016. Generating images with recurrent adversarial networks. *arXiv:1602.05110*.
[55] Oxford-Man Institute. 2021. Oxford-Man Institute of Quantitative Finance: Realized Library. Retrieved April 30, 2021 from https://realized.oxford-man.ox.ac.uk.
[56] Alistair E. W. Johnson, Tom J. Pollard, Lu Shen, LiWei H. Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G. Mark. 2016. MIMIC-III, a freely accessible critical care database. *Scientific Data* 3 (2016), 160035.
[57] Lauri Juvela, Bajibabu Bollepalli, Junichi Yamagishi, and Paavo Alku. 2019. Waveform generation for text-to-speech synthesis using pitch-synchronous multi-scale generative adversarial networks. In *Proceedings of the 2019 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'19)*. IEEE, Los Alamitos, CA, 6915–6919. https://doi.org/10.1109/ICASSP.2019.8683271
[58] Shruti Kaushik, Abhinav Choudhury, Sayee Natarajan, Larry A. Pickett, and Varun Dutt. 2020. Medicine expenditure prediction via a variance-based generative adversarial network. *IEEE Access* 8 (2020), 110947–110958. https://doi.org/10.1109/ACCESS.2020.3002346
[59] Dani Kiyasseh, Girmaw Abebe Tadesse, Le Nguyen Thanh Nhan, Le Van Tan, Louise Thwaites, Tingting Zhu, and David Clifton. 2020. PlethAugment: GAN-based PPG augmentation for medical diagnosis in low-resource settings. *IEEE Journal of Biomedical and Health Informatics* 24, 11 (Nov. 2020), 3226–3235. https://doi.org/10.1109/JBHI.2020.2979608
[60] Antonina Kolokolova, Mitchell Billard, Robert Bishop, Moustafa Elsisy, Zachary Northcott, Laura Graves, Vineel Nagisetty, and Heather Patey. 2020. GANs & reels: Creating Irish music using a generative adversarial network. *arXiv:2010.15772* [cs.SD].

[61] Alex Krizhevsky and Geoffrey Hinton. 2009. *Learning Multiple Layers of Features from Tiny Images*. Technical Report. University of Toronto, Toronto, Ontario.
[62] Mirella Lapata. 2015. EMNLP14. Retrieved April 30, 2021 from http://homepages.inf.ed.ac.uk/mlap/Data/EMNLP14/.
[63] Teema Leangarun, Poj Tangamchit, and Suttipong Thajchayapong. 2018. Stock price manipulation detection using generative adversarial networks. In *Proceedings of the 2018 IEEE Symposium Series on Computational Intelligence (SSCI'18)*. IEEE, Los Alamitos, CA, 2104–2111. https://doi.org/10.1109/SSCI.2018.8628777
[64] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* 86, 11 (1998), 2278–2324.
[65] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, et al. 2017. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the 30th IEEE Conference on Computer Vision and Pattern Recognition (CVPR'17)*. 105–114. https://doi.org/10.1109/CVPR.2017.19
[66] Dan Li, Dacheng Chen, Baihong Jin, Lei Shi, Jonathan Goh, and See-Kiong Ng. 2019. MAD-GAN: Multivariate anomaly detection for time series data with generative adversarial networks. In *Artificial Neural Networks and Machine Learning—ICANN 2019: Text and Time Series*. Lecture Notes in Computer Science, Vol. 11730. Springer, 703–716. https://doi.org/10.1007/978-3-030-30490-4_56
[67] Qingliang Li, Huibowen Hao, Yang Zhao, Qingtian Geng, Guangjie Liu, Yu Zhang, and Fanhua Yu. 2020. GANs-LSTM model for soil temperature estimation from meteorological: A new approach. *IEEE Access* 8 (2020), 59427–59443. https://doi.org/10.1109/ACCESS.2020.2982996
[68] Yujia Li, Kevin Swersky, and Richard Zemel. 2015. Generative moment matching networks. *arXiv:1502.02761* [cs.LG].
[69] Fiete Luer, Dominik Mautz, and Christian Bohm. 2019. Anomaly detection in time series using generative adversarial networks. In *Proceedings of the 2019 International Conference on Data Mining Workshops (ICDMW'19)*. IEEE, Los Alamitos, CA, 1047–1048. https://doi.org/10.1109/ICDMW.2019.00152
[70] Yonghong Luo, Xiangrui Cai, Ying Zhang, Jun Xu, and Yuan Xiaojie. 2018. Multivariate time series imputation with generative adversarial networks. In *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Eds.), Vol. 31. Curran Associates, Montréal, Canada, 1596–1607. https://proceedings.neurips.cc/paper/2018/file/96b9bff013acedfb1d140579e2fbeb63-Paper.pdf.
[71] Yonghong Luo, Ying Zhang, Xiangrui Cai, and Xiaojie Yuan. 2019. E^2GAN: End-to-end generative adversarial network for multivariate time series imputation. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*. 3094–3100. https://doi.org/10.24963/ijcai.2019/429
[72] B. Malin and L. Sweeney. 2001. Re-identification of DNA through an automated linkage process. In *Proceedings of the AMIA Symposium*. 423–427. https://pubmed.ncbi.nlm.nih.gov/11825223.
[73] H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. *arXiv:1602.05629* [cs.LG].
[74] Olof Mogren. 2016. C-RNN-GAN: Continuous recurrent neural networks with adversarial training. *arXiv:1611.09904* [cs.AI].
[75] G. B. Moody and R. G. Mark. 2001. The impact of the MIT-BIH arrhythmia database. *IEEE Engineering in Medicine and Biology Magazine* 20, 3 (2001), 45–50. https://doi.org/10.1109/51.932724
[76] Hao Ni, Lukasz Szpruch, Magnus Wiese, Shujian Liao, and Baoren Xiao. 2020. Conditional Sig-Wasserstein GANs for time series generation. *arXiv:2006.05421* [cs.LG].
[77] Konstantinos Nikolaidis, Stein Kristiansen, Vera Goebel, Thomas Plagemann, Knut Liestøl, and Mohan Kankanhalli. 2019. Augmenting physiological time series data: A case study for sleep apnea detection. *arXiv:1905.09068* [cs.LG].
[78] S. J. Pan and Q. Yang. 2010. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering* 22, 10 (2010), 1345–1359. https://doi.org/10.1109/TKDE.2009.191
[79] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. 311–318. https://doi.org/10.3115/1073083.1073135
[80] Dhasarathy Parthasarathy, Karl Bäckström, Jens Henriksson, and Sólrún Einarsdóttir. 2020. Controlled time series generation for automotive software-in-the-loop testing using GANs. *arXiv:2002.06611* [cs.LG].
[81] Damian Pascual, Alireza Amirshahi, Amir Aminifar, David Atienza, Philippe Ryvlin, and Roger Wattenhofer. 2020. EpilepsyGAN: Synthetic epileptic brain activities with privacy preservation. *IEEE Transactions on Biomedical Engineering* 67 (2020), 1. https://doi.org/10.1109/TBME.2020.3042574
[82] Marco A. F. Pimentel, Alistair E. W. Johnson, Peter H. Charlton, Drew Birrenkott, Peter J. Watkinson, Lionel Tarassenko, and David A. Clifton. 2017. Toward a robust estimation of respiratory rate from pulse oximeters. *IEEE Transactions on Biomedical Engineering* 64, 8 (2017), 1914–1923. https://doi.org/10.1109/TBME.2016.2613124
[83] Tom J. Pollard, Alistair E. W. Johnson, Jesse D. Raffa, Leo A. Celi, Roger G. Mark, and Omar Badawi. 2018. The eICU Collaborative Research Database, a freely available multi-center database for critical care research. *Scientific Data* 5, 1 (Sept. 2018), 180178. https://doi.org/10.1038/sdata.2018.178

[84] Fuming Qu, Jinhai Liu, Yanjuan Ma, Dong Zang, and Mingrui Fu. 2020. A novel wind turbine data imputation method with multiple optimizations based on GANs. *Mechanical Systems and Signal Processing* 139 (May 2020), 106610. https://doi.org/10.1016/j.ymssp.2019.106610

[85] Alec Radford, Luke Metz, and Soumith Chintala. 2015. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv:1511.06434.*

[86] Mohammad Rasouli, Tao Sun, and Ram Rajagopal. 2020. FedGAN: Federated generative adversarial networks for distributed data. *arXiv:2006.07228* [cs.LG].

[87] Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. 2016. Generative adversarial text to image synthesis. In *Proceedings of the 33rd International Conference on Machine Learning*, M. F. Balcan and K. Q. Weinberger (Eds.). PMLR, New York, NY, 1681–1690.

[88] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen, and Xi Chen. 2016. Improved techniques for training GANs. In *Advances in Neural Information Processing Systems*, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett (Eds.), Vol. 29. Curran Associates, Barcelona, Spain, 2234–2242. https://proceedings.neurips.cc/paper/2016/file/8a3363abe792db2d8761d6403605aeb7-Paper.pdf.

[89] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. 2017. Membership inference attacks against machine learning models. *arXiv:1610.05820* [cs.CR].

[90] Yuki Sumiya, Kazumasa Horie, Hiroaki Shiokawa, and Hiroyuki Kitagawa. 2019. NR-GAN: Noise reduction GAN for mice electroencephalogram signals. In *Proceedings of the 2019 4th International Conference on Biomedical Imaging, Signal Processing*. ACM, New York, NY, 94–101. https://doi.org/10.1145/3366174.3366186

[91] He Sun, Zhun Deng, Hui Chen, and David C. Parkes. 2020. Decision-aware conditional GANs for time series data. *arXiv:2009.12682* [cs.LG].

[92] Yuqiang Sun, Lei Peng, Huiyun Li, and Min Sun. 2018. Exploration on spatiotemporal data repairing of parking lots based on recurrent GANs. In *Proceedings of the 2018 21st International Conference on Intelligent Transportation Systems (ITSC'18)*. IEEE, Los Alamitos, CA, 467–472. https://doi.org/10.1109/ITSC.2018.8569319

[93] Dougal J. Sutherland, Hsiao-Yu Tung, Heiko Strathmann, Soumyajit De, Aaditya Ramdas, Alex Smola, and Arthur Gretton. 2016. Generative models and model criticism via optimized maximum mean discrepancy. *arXiv:1611.04488.*

[94] Mark Treveil and Dataiku Team. 2020. *Introducing MLOps*. O'Reilly.

[95] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research* 9, 86 (2008), 2579–2605. http://jmlr.org/papers/v9/vandermaaten08a.html.

[96] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17)*. Curran Associates Inc., Red Hook, NY, USA, 6000–6010.

[97] Lu Wang, Wei Zhang, and Xiaofeng He. 2019. Continuous patient-centric sequence generation via sequentially coupled adversarial learning. In *Database Systems for Advanced Applications*, Guoliang Li, Jun Yang, Joao Gama, Juggapong Natwichai, and Yongxin Tong (Eds.). Springer International, Cham, Switzerland, 36–52.

[98] Zhengwei Wang, Graham Healy, Alan F. Smeaton, and Tomas E. Ward. 2020. Use of neural signals to evaluate the quality of generative adversarial network performance in facial image generation. *Cognitive Computation* 12, 1 (2020), 13–24.

[99] Zhengwei Wang, Qi She, Alan F. Smeaton, Tomas E. Ward, and Graham Healy. 2020. Synthetic-neuroscore: Using a neuro-AI interface for evaluating generative adversarial networks. *Neurocomputing* 405 (2020), 26–36.

[100] Zhengwei Wang, Qi She, and Tomás E. Ward. 2021. Generative adversarial networks in computer vision: A survey and taxonomy. *ACM Computing Surveys* 54, 2 (2021), 1–38.

[101] Magnus Wiese, Robert Knobloch, Ralf Korn, and Peter Kretschmer. 2020. Quant GANs: Deep generation of financial time series. *Quantitative Finance* 20, 9 (Sept. 2020), 1419–1440. https://doi.org/10.1080/14697688.2020.1730426 arXiv: 1907.06673

[102] Samim Winiger. 2015. Obama Political Speech Generator—Recurrent Neural Network. Retrieved April 30, 2021 from https://github.com/samim23/obama-rnn.

[103] Liyang Xie, Kaixiang Lin, Shu Wang, Fei Wang, and Jiayu Zhou. 2018. Differentially private generative adversarial network. *arXiv:1802.06739.*

[104] L. Yi and M. Mak. 2019. Adversarial data augmentation network for speech emotion recognition. In *Proceedings of the 2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC'19)*. IEEE, Los Alamitos, CA, 529–534. https://doi.org/10.1109/APSIPAASC47483.2019.9023347

[105] Xin Yi, Ekta Walia, and Paul Babyn. 2019. Generative adversarial network in medical imaging: A review. *Medical Image Analysis* 58 (2019), 101552. https://doi.org/10.1016/j.media.2019.101552

[106] Chika Yinka-Banjo and Ogban-Asuquo Ugot. 2020. A review of generative adversarial networks and its application in cybersecurity. *Artificial Intelligence Review* 53, 3 (March 2020), 1721–1736. https://doi.org/10.1007/s10462-019-09717-4

[107] Jinsung Yoon, Daniel Jarrett, and Mihaela van der Schaar. 2019. Time-series generative adversarial networks. In *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), Vol. 32. Curran Associates, Vancouver, Canada, 5508–5518. https://proceedings.neurips.cc/paper/2019/file/c9efe5f26cd17ba6216bbe2a7d26d490-Paper.pdf.

[108] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. SeqGAN: Sequence generative adversarial nets with policy gradient. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence (AAAI'17)*. 2852–2858.

[109] Hui Zhang, Niannao Xiao, Peishun Liu, Zhicheng Wang, and Ruichun Tang. 2020. G-RNN-GAN for singing voice separation. In *Proceedings of the 2020 5th International Conference on Multimedia Systems and Signal Processing*. ACM, New York, NY, 69–73. https://doi.org/10.1145/3404716.3404718

[110] Zixing Zhang, Jing Han, Kun Qian, Christoph Janott, Yanan Guo, and Bjoern Schuller. 2019. Snore-GANs: Improving automatic snore sound classification with synthesized data. *arXiv:1903.12422* [cs.LG].

[111] Fei Zhu, Fei Ye, Yuchen Fu, Quan Liu, and Bairong Shen. 2019. Electrocardiogram generation with a bidirectional LSTM-CNN generative adversarial network. *Scientific Reports* 9, 1 (2019), Article 6734, 11 pages. https://doi.org/10.1038/s41598-019-42516-z

[112] Guangxuan Zhu, Hongbo Zhao, Haoqiang Liu, and Hua Sun. 2019. A novel LSTM-GAN algorithm for time series anomaly detection. In *Proceedings of the 2019 Prognostics and System Health Management Conference (PHM-Qingdao'19)*. IEEE, Los Alamitos, CA, 1–6.

Received 18 June 2021; revised 12 August 2022; accepted 21 August 2022