International Journal of Data Science and Analytics (2023) 16:421–434
https://doi.org/10.1007/s41060-023-00416-6

REGULAR PAPER

Sparse self-attention guided generative adversarial networks for time-series generation

Nourhan Ahmed[1] · **Lars Schmidt-Thieme**[1]

Received: 26 November 2022 / Accepted: 15 June 2023 / Published online: 5 July 2023
© The Author(s) 2023

Abstract
Remarkable progress has been achieved in generative modeling for time-series data, where the dominating models are generally generative adversarial networks (GANs) based on deep recurrent or convolutional neural networks. Most existing GANs for time-series generation focus on preserving correlations across time. Although these models may help in capturing long-term dependencies, their capacity to pay varying degrees of attention over different time steps is inadequate. In this paper, we propose *SparseGAN*, a novel sparse self-attention-based GANs that allows for attention-driven, long-memory modeling for regular and irregular time-series generation through learned embedding space. This way, it can yield a more informative representation for time-series generation while using original data for supervision. We evaluate the effectiveness of the proposed model using synthetic and real-world datasets. The experimental findings indicate that forecasting models trained on SparseGAN-generated data perform comparably to forecasting models trained on real data for both regularly and irregularly sampled time series. Moreover, the results demonstrate that our proposed generative model is superior to the current state-of-the-art models for data augmentation in the low-resource regime and introduces a novel method for generating realistic synthetic time-series data by leveraging long-term structural and temporal information.

Keywords Time series generation · Generative adversarial networks · Sparse self-attention

1 Introduction

Time-series Generation is a crucial task in many notable disciplines, including finance [1], machine fault diagnosis [2], medicine [3], and music [4]. Significant progress in the area of time-series generation has been achieved with the introduction of Generative Adversarial Networks (GANs). GANs are neural networks that are meant to generate synthetic instances of data utilizing two neural networks, a generator and a discriminator, that operate against each other at the same time [5]. The generator learns to generate fake data to get the discriminator to classify its generated samples as authentic. The discriminator, on the other hand, attempts to distinguish between authentic and produced data. Finally, the generator could generate realistic data.

GANs have demonstrated their ability to generate realistic data and have made remarkable progress in various tasks, such as the generation of time-series [3, 4, 6, 7], images [8, 9], and videos [10]. Particularly, a significant amount of work has utilized GANs based on Recurrent Neural Networks (RNNs) for time-series generation [3, 4, 6, 7]. However, by carefully examining the generated samples from these models, we can observe that RNN-based GANs, such as LSTM GANs and gated recurrent GANs, cannot handle long sequences. Although RNN-based GANs can generate many realistic samples, there is still a difficulty in training them due to exploding vanishing gradients and mode collapse that limits their generation capability. In addition, the majority of time series generation models assume that time series data are collected at regularly spaced intervals of time, and so are referred to as "regular time series". In real-world contexts, however, unevenly spaced time is a major issue for time series data [6]. Several factors, such as technological flaws in sensing equipment and imprecision in the sensors, might lead to irregular sampling [11]. Accordingly, since RNN-based GANs are typically designed for regular time-series data,

✉ Nourhan Ahmed
ahmedn@ismll.unihildesheim.de

Lars Schmidt-Thieme
schmidt-thieme@ismll.uni-hildesheim.de

[1] Information Systems and Machine Learning Lab, University of Hildesheim, 31141 Hildesheim, Lower Saxony, Germany

they cannot maintain informative varying intervals properly, which is a major concern for generating time-series data.

Although there is no single solution to these drawbacks, GANs combined with self-attention mechanisms make a convincing argument. Self-attention mechanisms [12] allow faster and higher-quality learning by paying more attention to the most important parts with relatively low computational cost. The incorporation of the self-attention module has proved its success in many deep learning-based applications, such as text translation [13], speech recognition [14], and image generation [9]. The softmax transformation is commonly used in self-attention to construct the attention distribution that reflects the relative significance of all positions in a sequence of inputs [15]. However, the memory and computational complexity of self-attention increase exponentially with sequence length.

To address the aforementioned issues, faster and more efficient alternatives like sparsity-based self-attention can be used instead. In this work, we introduce SparseGAN, a novel end-to-end generative model for regular and irregular multivariate time series. We use sparse self-attention to limit the selection to only relevant time steps in order to encode the similarity information of temporal behavior more effectively. Our primary innovation is the inclusion of a joint learning mechanism coupled with a sparse self-attention mechanism that replaces the use of a full fixed-size representation derived from continuous-time inputs. This gives SparseGAN more representational flexibility compared to previous RNN-based GANs. The key contributions of this paper can be presented as follows:

- To the best of our knowledge, we propose the first time-series generation model that relies on self-attention to generate realistic synthetic data for time-series forecasting. Precisely, we leverage the sparse self-attention layer to build a fully attentional generative model that is not only capable of accessing all historical input steps regardless of the time-series length but is also supported by supervision from the original data.
- We design a joint framework for time-series generation. Our approach is more generic than any sequential generative model, as it can handle regular and irregular time-series data.
- Experimental results prove that the SparseGAN model consistently outperforms all baseline models with an error reduction of around 15% over the former best model for both regular and irregular time-series data.
- We further show that the time series synthesized from our model can be applied to other tasks, such as data augmentation and data substitution in low-data regimes for training better time-series forecasting models.

2 Related work

Generative adversarial networks GANs are generative models comprised of two networks: a generator and a discriminator. They are capable of generating new data points that are similar to the original dataset [5]. Several works have used GANs for one-dimensional time-series generation. For instance, Lou et al. [16] introduced two distinct approaches for one-dimensional data augmentation using fully connected layers: *Wasserstein Generative Adversarial Network* (WGAN) and WGAN along with autoencoder for more efficient data representation. Ramponi et al. [6] are the first to introduce one-dimensional convolutional layers as an alternative to fully connected networks to handle regularly and irregularly sampled time-series. In the recent work of Shao et al. [2], Wiese et al. [1] and Dogariu et al. [17, 18], the authors proposed models similar to [6] with an emphasis on machine fault diagnosis and financial time-series, respectively.

In addition, other scholars have considered using GANs for multivariate time-series generation. Mogren [4] introduced continuous RNN-based GANs to model the entire conditional distribution of data sequences for classical music generation purposes. In his approach, unidirectional LSTM layers are used in the generator, while bidirectional LSTM layers are used in the discriminator. Similarly, Esteban et al. [3] proposed two different approaches for time-series generation with an emphasis on medical data: *Recurrent GAN* (RGAN) and *Recurrent Conditional* GAN (RCGAN). These models follow a similar architecture to that of the standard GAN, but LSTM layers were used to replace the fully-connected layers in both the generator and the discriminator. Later, Bandara et al. [19] proposed employing standard generation models such as dynamic time warping averaging with an emphasis on to produce time series synthetically in order to enhance forecasting models. Yoon et al. [7] have introduced a joint learning mechanism for training GANs. Particularly, the authors proposed an RNN-based GAN model that is trained jointly with an encoder-decoder network. Recently, Arnout et al. [20] introduced *Class-specific Recurrent GAN* (CLaRe-GAN) by conditioning the generator on auxiliary input comprising class-specific and class-independent properties. Specifically, the model consists of two encoders, one for each kind of information (inter- and intra-class characteristics), a shared-latent-space assumption, and a class discriminator that discriminates across latent vectors to extract class-specific features. Despite their promising performance, these models are still inadequate for generating time-series data that take into account all the relevant long dependencies and information in time-series data.

Attention mechanisms Sequential problems have incorporated attention mechanisms such as time-series forecasting [21–24], time-series classification [25, 26], time-series

anomaly detection [27, 28] and neural machine translation [29, 30]. The first is Transformer model [12] which is an encoder-decoder architecture based on the attention mechanism designed to handle sequential data effectively without RNN layers. Recently, a variety of transformer-based methods have been proposed, such as Transformer-XL [29], Reformer [31] and Universal Transformer [32]. For example, Transformer-XL [29] can be considered an extension of the transformer model that tackles the issue of fixed-span attention that allows information to flow across segments. Reformer [31] is another transformer model extension that replaces the dot-product attention with locality-sensitive hashing attention to reduce the computational complexity and the standard residual blocks with reversible residual layers. The Universal Transformer [32] is another transformer architecture that combines the transformer model's parallelizability with RNNs' inductive bias. Similarly, Hu et al. [33] proposed self-attention along with RNN to mine more information of time series for forecasting task. Another work by Wan et al. [34] proposed using CNN-based model using 1D dilated convolution, followed by the feature extraction utilizing a self-attention mechanism for time-series prediction.

Recently, several papers have proposed sparsity-based architectures that rely on sparse attention mechanisms, which are typically used in neural machine translation [35, 36]. For instance, Sparse Transformer is a transformer-based architecture that introduced self-attention with sparse factorization, to reduce the computational cost, making it possible to train attention networks with hundreds of layers on long sequences [36]. Adversarial Sparse Transformer is inspired by the sparse transformer and GANs for time-series forecasting [22]. However, none of these studies have investigated adapting self-attention, especially sparse self-attention, into GANs for time-series generation tasks. Unlike all the existing models, we propose a novel model for time-series generation that employs sparse self-attention in both parts of the GANs in conjunction with joint learning to increase the quality of generated time-series data.

3 Problem formulation

Let $\mathcal{X}, \mathcal{M}$ be any sets called *data space* and *model space*, respectively, e.g., $\mathcal{X} := \mathbb{R}$, and let $\mathcal{X}^* := \bigcup_{i\in\mathbb{N}} \mathcal{X}^i$ denote the set of finite sequences in $\mathcal{X}$. We model a *learning algorithm* as a map $a : \mathcal{X}^* \to \mathcal{M}$ from datasets $\mathcal{D}^{\text{train}} \in \mathcal{X}^*$ to models $\hat{y} \in \mathcal{M}$. Let $\ell : \mathcal{M} \times \mathcal{X}^* \to \mathbb{R}$ be an *evaluation measure*[1] where $\ell(\hat{y}, \mathcal{D}^{\text{test}})$ with a fresh test sample from the same data generating distribution as $\mathcal{D}^{\text{train}}$ yields an estimate of the expected error of the learned model. Our goal is to use training data $\mathcal{D}^{\text{train}}$ to generate some synthetic data V that best approximates real data distribution. We distinguish two variants of the synthetic data generation problem:

Data augmentation problem: Given a real dataset $\mathcal{D}^{\text{train}}$, a learning algorithm a, and an evaluation measure ℓ, find a synthetic dataset generator G such that the model trained on both, the training data and the data synthesized by it, V, has a minimal error, e.g., minimize:

$$\ell^{\text{augm}}(V; \mathcal{D}^{\text{train}}, a, \ell; \mathcal{D}^{\text{test}}) := \ell(a(\mathcal{D}^{\text{train}} \oplus V), \mathcal{D}^{\text{test}})$$

where $\oplus$ denotes the concatenation of sequences. The application idea behind the data augmentation problem is to add more synthetic training data to a dataset and fill unexplored input space to be able to learn more powerful models [2, 6, 37]. Consequently, it is a very successful method for expanding and improving the quality of training data. Thus, the better the synthetic data generator, the further the loss should get reduced when compared to train on the real data only.

Data substitution problem: Given the same as above, find a synthetic dataset generator G such that the model trained only on data synthesized by it, V, can be used to replace the training data such that the model trained only on data synthesized by G has a minimal error, e.g., minimize:

$$\ell^{\text{subst}}(V; a, \ell; \mathcal{D}^{\text{test}}) := \ell(a(V), \mathcal{D}^{\text{test}})$$

This problem is relatively new where the application idea behind the data substitution problem usually is, that models for sensitive data should be learned in potentially insecure environments such as the cloud or by a service provider, and thus the real data should not be shared, but just synthetic data fulfilling the same purpose [38, 39]. The data substitution problem silently assumes that the data generator does not have the capacity to simply reproduce the real training data, which arguably would yield the best data fundament for learning a model, but also would invalidate its purpose of not sharing the real data.

Here, we deal specifically with data augmentation and data substitution problems when learning forecasting models for time-series data.

4 The proposed model: SparseGAN

This section discusses our proposed model, SparseGAN, in detail as illustrated in Fig. 1. To begin, our model consists of two sub-nets namely, *supervision network* and *generation network* respectively. *Supervision network* can provide supervision information for the generation network in the

[1] The evaluation measure could be any evaluation measure, however in this paper, the evaluation measures used are described in detail in Sect. 5.3.

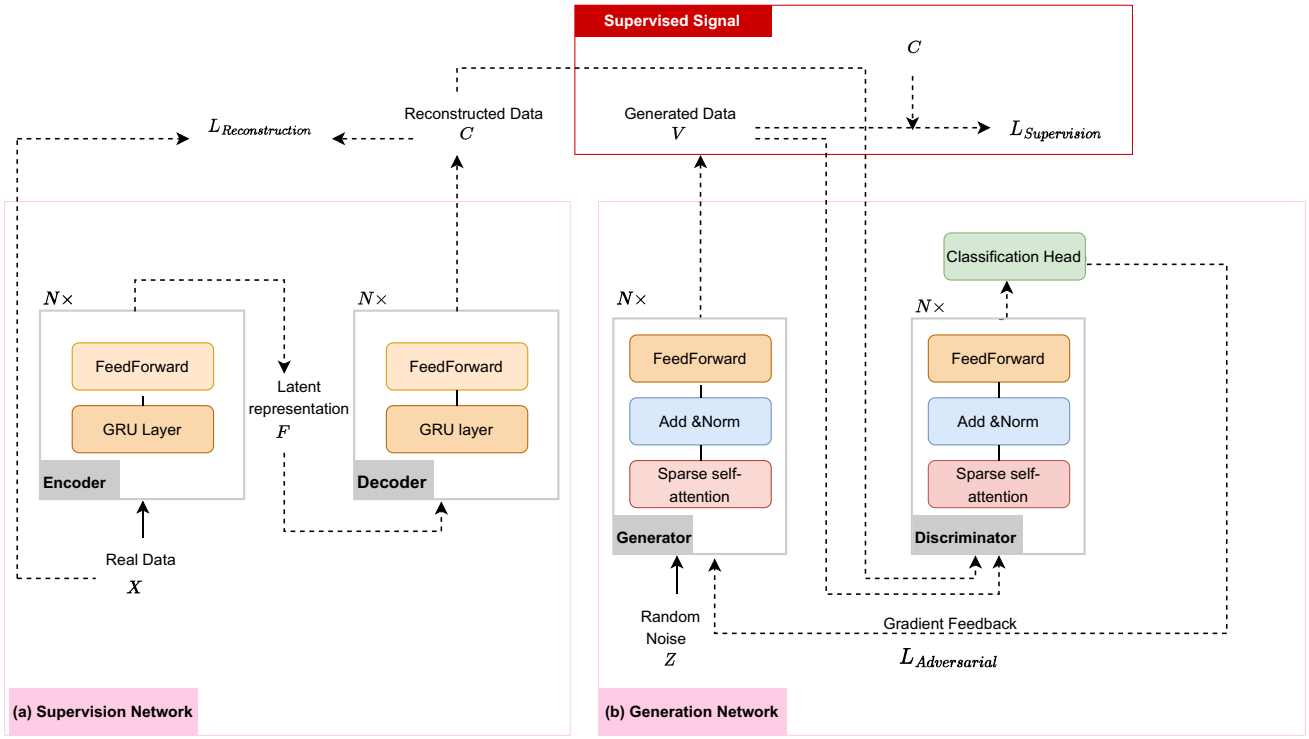

Fig. 1 The overall architecture of the SparseGAN model. The left part of the figure is our supervision network and the right part of the figure is the generation network. The whole network is trained end-to-end manner and the supervision network is used to supervise the generation network in the training phase

training stage. *Generation network* is used to generate time-series that are as realistic as possible. Accordingly, the supervision network exploits the fact that the training data include more information than merely whether produced data is genuine or synthetic; we may explicitly learn from actual data. Therefore, the supervision network reduces the data's complexity to enable the generation network to produce more realistic data and does not depend only on discriminator feedback to acquire more realistic data. The two sub-nets of SparseGAN are trained in an end-to-end manner. We elaborate on each of the two sub-nets below.

4.1 Supervision network

SparseGAN adopts a few existing studies as its building blocks. First, the supervision network is an encoder-decoder network. Consider the input data, $\mathcal{X} = \{x_1^i, x_2^i, \ldots, x_t^i\}$, where x_t^i denotes the value of time series i at time t. The encoder consists of a series of Gated Recurrent Unit (GRU) cells which takes a time-series as input and produces latent feature representation, $F = \{f_1^i, f_2^i, \ldots, f_t^i\}$, of input time-series data [40, 41]. The decoder follows a formula similar to the encoder, which takes the latent feature representation from the encoder and outputs reconstructed time-series data, $C = \{c_1^i, c_2^i, \ldots, c_t^i\}$, which is similar to the input time-series that leverages the entire data structure [42]; this is crucial to facilitate the generation task. Accordingly, in order to reconstruct a high-dimensional time-series, the supervision network offers an effective and reliable solution. Utilizing the reconstructed data rather than the actual data enables the generation network to learn the underlying dynamics of the data through lower-dimensional representations [7]. In addition, this phase prevents the discriminator from getting stuck in a local minimum [6, 7, 43]. For the supervision network, we take x_t^i as input, and the objective is to reconstruct x_t^i. We use $L_{\text{reconstruct}}$ as our reconstruction loss to yield efficient and robust reconstruction for a high-dimensional time-series.

Let $A(\cdot)$ be the supervision network, the objective of the supervision network can be expressed as:

$$L_{\text{reconstruct}} = \mathbb{E}_{x \sim p_{\text{data}}(x)}[||\mathcal{X} - A(\mathcal{X})||_2] \quad (1)$$

4.2 Generation network

The generation network employs two independent networks: a generator G and a discriminator D, which follow the general GANs architecture. The seed for the generator is the random noise vector Z. Here, Z is a sequence of T points $\{z_t\}_{t=1}^T$ sampled independently from Gaussian distribution with mean equals to 0 and standard deviation equals to 1. G attempts to transform the random noise vector Z into realistic time series. On the other hand, D seeks to distin-

guish if the time series generated by G is realistic or not. For the generation network, let C denote the reconstructed time series yielded by the supervision network's decoder. Here, the discriminator evaluates the degree of authenticity of the generated time series $V = \{v_1^i, v_2^i, \ldots, v_t^i\}$ using the adversarial loss $L_{\text{adversarial}}$,

$$L_{\text{adversarial}} = \mathbb{E}_{c\sim p_{\text{data}}(c)}[log\ D(C)] + \mathbb{E}_{z\sim p_z(z)}[log\ (1 - D(V)] \tag{2}$$

We utilize the noise vector Z as a seed to G, then we receive the generated time-series v_t^i, afterwards fed v_t^i into D, and ultimately the adversarial loss $L_{\text{adversarial}}$.

While RNN-based GANs have been widely used to generate time-series data thanks to their ability to handle data sequences, they assume that the observations are sampled regularly and are fully observed at each sampling period. These assumptions often do not hold for real-world multidimensional time-series that can be sparse and irregular. Additionally, they scale inadequately with long input sequences. Taken together, these limitations make the existing RNN-based GANs unsuitable for our scenario [44, 45]. To properly deal with irregular time intervals and learn the implicit long-range dependencies, we propose the SparseGAN model based on the sparse self-attention module.

Sparse self-attention module In general, the canonical self-attention scores are computed by employing the classical softmax transformation to find the weighted sum transformations $y = (y_1, \ldots, y_L)$ of the input sequence $h = (h_1, \ldots, h_L)$ based on relevance [12]. The i-th output y_i of the input sequence h is determined as follows:

$$y_i = \sum_{j=1}^{L} k_{ij}(h_j W_v) \tag{3}$$

$$e_{ij} = \frac{(h_i W_q)(h_j W_k)^T}{\sqrt{d}} \tag{4}$$

$$k_{ij} = \text{softmax}(e_{ij}) \tag{5}$$

where $W_q \in R^{d\times d}$, $W_k \in R^{d\times d}$, and $W_v \in R^{d\times d}$ represent the parameter matrices. e_{ij} represents the attention scores and k_{ij} represents the relevance between the i-th and j-th input element.

This results in dense dependencies that capture the complete interactions between each pair of time steps and fails to assign zero probability to less significant relationships. This would result in less attention assigned to the relevant time steps, diminishing the performance. Besides, it does not scale well with long input sequences, requiring exponential computation complexity and storage capacity to generate all similarity scores according to Jain and Wallace [46] and Wu et al. [22].

To address this problem, we employ a transformation from the entmax family [47] to replace the softmax function in the self-attention module in both generator and discriminator parts. unlike softmax, they are capable of producing sparse probability distributions. Specifically, this is done by applying an α-entmax transformation of the attention scores e [47–49], defined as:

$$\alpha - \text{entmax}(e) = \arg\max_{p\in\Delta^d} \mathbf{p}^\top e + H_\alpha^T(p), \tag{6}$$

where $\Delta^d := \{p \in R^d : \sum_i p_i = 1\}$ is the probability simplex, and, for $\alpha \geq 1$, H_α^T is the Tsallis continuous family of entropies [50]:

$$H_\alpha^T(p) := \begin{cases} \frac{1}{\alpha(\alpha-1)}\sum_j (p_j - p_j^\alpha), & \alpha \neq 1 \\ -\sum_j p_j log\ p_j, & \alpha = 0. \end{cases} \tag{7}$$

Based on the definition of $H_\alpha^T(p)$, the entmax transformation can be defined as,

$$\alpha - \text{entmax}(e) = \text{ReLU}((\alpha - 1)e - \tau 1)^{1/\alpha} \tag{8}$$

where τ denotes the Lagrange multiplier which acts as a threshold, i.e., entries with score $e_i \leq \tau/\alpha-1$ get zero probability and 1 indicates a vector of all ones. In particular, we implement 1.5-entmax as a balance between softmax and sparsemax. By using sparsity to refine the attention weights, we can achieve a more expressive representation of the entire input.

Sparse self-attention GANs In this part, we propose sparse self-attention GANs. In particular, the generator architecture is first composed of a stack of sparse self-attention layers, where each layer learns a representation by taking the output from the previous layer that follows a setup close to the form of attention proposed by Vaswani et al. [12] using 1.5-entmax transformation [47]. Then, a fully connected feed-forward network is stacked on top of it. In the end, we employ a residual connection [51] around the stack of layers, followed by layer normalization [52]. The seed for the generator is the random noise vector Z drawn from the Gaussian distribution. To ensure that the discriminator can learn fake and real time-series, the discriminator is also composed of a stack of self-attention layers using 1.5-entmax transformation [12, 47] followed by a fully connected feed-forward network. Eventually, the proposed GANs become more capable of generating time-series data that fit the distribution of real data accurately.

4.3 End-to-end joint training

We train the two sub-nets jointly in an end-to-end manner. However, depending only on discriminator feedback has a

major flaw. The generated samples are based on the distribution from which the input random noise Z is sampled. As a result, there might be a significant gap between the generated samples and the actual data. Accordingly, We use C to supervise the generator of generation network to improve the quality of generated data and enable the generator to construct more realistic data sequences [7, 44, 53]. We define the supervision loss function as:

$$L_{\text{supervise}} = ||C - V||_2 \tag{9}$$

The final loss is made of reconstruction, adversarial and supervision losses, with λ being a hyper-parameter that controls the contribution of the supervision loss. So our final objective function is:

$$L^{\star} = arg \min_{G} \max_{D} L_{\text{adversarial}} + \lambda_s L_{\text{supervise}} + L_{\text{reconstruct}} \tag{10}$$

5 Experiments

In this section, we present our experiments to evaluate the proposed generative model, SparseGAN. The experiments seek to answer the following research questions: (i) *RQ1*: Can SparseGAN improve the quality of data generation for regular and irregular time series compared to the state-of-the-art generative models? (ii) *RQ2*: In low-resourced data scenarios, how effective is augmenting real-world datasets with SparseGAN generated data in boosting the accuracy of time-series forecasting models? (iii) *RQ3*: Can SparseGAN generate synthetic data that can be substituted for the existing real data? (iv) *RQ4*: How well SparseGAN generated data preserve original data's diversity and realism compared to the state-of-the-art generative models?

5.1 Datasets

We assess the utility of SparseGAN using five large-scale datasets. Since this study aims to assess the utility of SparseGAN for regular and irregular time-series, we train our model on five datasets, three of which are regular and two are irregular time-series.[2] We list these datasets below (Table 1).

1. *Sine waves* We generate sine waves with nonlinear variations by varying the amplitudes, frequencies and phases. We generate waves with frequencies in [1.0, 5.0], amplitudes in [0.1, 0.9], and random phases between $[-\pi, \pi]$ and dimension in $\{1, \ldots, 5\}$.
2. *Google stocks*[3] We use the daily historical Google stocks data from 2004 to 2020 which includes five attributes: *open, close, high, low*, and *volume*.
3. *Appliances energy*[4] The appliances energy dataset contains 4.5 months of energy consumption of 10-min readings. This dataset includes 29 attributes, including temperature and humidity conditions in different areas of the house.
4. *Power consumption*[5] We consider a dataset for the electricity consumption containing one-minute readings for 4 years in one household. This dataset consists of 9 attributes, including active power, reactive power and voltage. The processing of this dataset is done by selecting only 40% of each time series' points to create irregularly sampled data.
5. *Air quality*[6] We consider the hourly responses of gas concentration from a certified analyzer, which are recorded between March 2004 and February 2005. This dataset consists of 15 attributes, such as humidity and temperature. We follow [54] to create an irregular version of this dataset, in which the authors select only 40% of each time series to generate an irregular dataset. We follow the same procedure here. To maintain a fair comparison, the eliminated observations are randomly selected and maintained constant across all experimental settings and baselines.

Table 1 Datasets statistics

Dataset	Sequences	Dimension	Average length
Sines waves	10,000	5	24 points
Google stocks	3686	6	22 days
Energy appliances	19,711	29	24 h
Power consumption	20,752	9	47 months
Air quality	935	15	One year

5.2 Baseline models

To assess the performance of SparseGAN, we compare SparseGAN with time-series generative models, namely TimeGAN [7], a related method to our proposed one that

[2] For irregular time series, it is worth noting that we do not generate missing data points in the time series nor regularize irregular time stamps. We aim at generating new irregularly-sampled time series which mimic the ones in the original dataset.

[3] https://www.kaggle.com/darthmanav/google-stock-prices-from-2004-to-2020.

[4] https://www.kaggle.com/loveall/appliances-energy-prediction.

[5] https://archive.ics.uci.edu/ml/datasets/individual+household+electric+power+consumption.

[6] https://archive.ics.uci.edu/ml/datasets/air+quality.

combines unsupervised adversarial learning with supervised training, and several latest deep learning-based models, including T-CGAN [6], which is a data generation approach designed for time-series with irregular sampling, RCGAN [3], which is a standard GAN but LSTM layers have been used in both the generator and the discriminator, C-RNN-GAN [4], which is a continuous RNN-GAN to model the entire conditional distribution of data sequences, T-Forcing [55], which is RNN model trained with Teacher Forcing technique, P-Forcing [56], which is RNN model trained with Professor Forcing technique to capture long-term dependencies, WaveNet [57], which is a model for sequential generation of raw-waveform audio that imitates human voice, and WaveGAN [58], which is a convolutional-based GAN for raw-waveform audio generation.

5.3 Evaluation

Two aspects of the generated time-series data are considered for assessment. The first aspect is fidelity, which refers to how effectively the synthesized data retains the original data properties [59, 60]. In order to measure this, we performed two evaluation tasks to measure how well is our approach compared to different baseline models in terms of fidelity [59, 60]: (1) *Data augmentation*: in a low-data regime, we investigate augmenting training data with generated data. We reduce the number of data points available to train and evaluate the performance on the test set to mimic the low-data regime setting, and (2) *Data substitution*: we consider only the generated examples for training and evaluate the performance on the test set.

In this paper, we assess the SparseGAN model for data augmentation and data substitution tasks using two time-series forecasting models: LSTM model following [7] and LSTnet [61] which utilizes CNN along with RNN.[7] In this paper, we reduce the problem to learning a one-step-ahead forecasting model with minimal expected error. The second aspect is diversity [59, 60], which is measured by how well the generated data preserves the original data's distribution. Inspired by the Frechet Inception Distance [62], we employ a 4-layer LSTM model to discriminate between actual and synthetic data as a standard supervised task. The classification accuracy on the held-out test set is then reported. A lower classification accuracy score indicates that the synthesized data closely resembles real-world time-series properties. Furthermore, it demonstrates that the generated time series are indistinguishable from real-world data.

[7] In the appendix, we elaborate on this evaluation in detail.

6 Results and discussion

6.1 Generated data fidelity

We train two time-series forecasting models for each dataset under two regimes as described in the Sect. 6: (i) data substitution scenario and (ii) data augmentation for low-resourced data scenarios.

Data augmentation for low-resourced data scenarios In this section, we consider the scenario where forecasting models are trained on real-world data augmented by SparseGAN-generated data. To be clear, the purpose of this experiment is to investigate whether augmenting original data with SparseGAN-generated data will improve the accuracy of our forecasting models in low-data regime settings (*RQ1* and *RQ2*). Simulating the low-data regime settings was introduced for NLP tasks such as text classification [63, 64]. Inspired by this line of research, we explore sub-sampling a small training set to mimic the low-data regime scenario for time series forecasting, where we train generative models with limited data.

This experiment goes as follows. We focus on experiments with 10 and 50 data points to simulate realistic low-resourced data settings where we frequently observe poor performance. For data augmentation, we add 100 SparseGAN synthetic data points to the original data as shown in Tables 2 and 3. This procedure allows us to evaluate the performance of SparseGAN in data augmentation tasks for time-series forecasting models. Results are reported as MAE on the test set. The findings in Tables 2 and 3 demonstrate that data augmentation using SparseGAN can drastically improve the accuracy of time-series forecasting models compared to other generative models. In addition, the results provide strong evidence supporting the significance of integrating a sparse self-attention mechanism and a supervised signal for enhancing the quality of generated data, particularly when compared to models that solely rely on convolutional and RNN layers. The supervision network acts as an auxiliary network, providing valuable feedback to the generator based on the properties and characteristics of the real data. This feedback enables the supervision network to offer more direct and informative signals to the generator, resulting in improved convergence and enhanced quality of the generated samples. By combining the sparse self-attention mechanism as a fundamental building block with the integration of the supervised signal obtained from a supervision network, SparseGAN outperforms existing state-of-the-art generation models in terms of accuracy. For instance, there is around 20% MAE reduction compared to the previous best model results, TimeGAN, in the power consumption dataset. Generally, it can be concluded that in all cases, the SparseGAN model yields improvements of around 11–17 %, manifested by a reduction in MAE, over the state-

Table 2 Performance summary of MAE scores for data augmentation in low-data regime

Forecasting model	Generative model	Datasets				
		Sine waves	Google stocks	Energy appliances	Air quality	Power consumption
LSTM	No augmentation	0.1431 ± 0.0018	0.3215 ± 0.0008	0.2897 ± 0.0016	0.3282 ± 0.0008	0.2965 ± 0.0004
	SparseGAN	**0.0181 ± 0.0008**	**0.1412 ± 0.0006**	**0.1721 ± 0.0008**	**0.1821 ± 0.0008**	**0.1642 ± 0.0008**
	TimeGAN	0.0202 ± 0.0008	0.1573 ± 0.0008	0.1832 ± 0.0008	0.2077 ± 0.0008	0.1891 ± 0.0004
	T-CGAN	0.0249 ± 0.0012	0.2192 ± 0.0006	0.1965 ± 0.0006	0.2379 ± 0.0008	0.2094 ± 0.0006
	RCGAN	0.0410 ± 0.0008	0.2598 ± 0.0012	0.2039 ± 0.0006	0.2588 ± 0.0006	0.2296 ± 0.0008
	C-RNN-GAN	0.0487 ± 0.0006	0.2881 ± 0.0012	0.2441 ± 0.0012	0.2778 ± 0.0008	0.2442 ± 0.0012
	T-forcing	0.0508 ± 0.0008	0.2605 ± 0.0008	0.2189 ± 0.0006	0.2390 ± 0. 0012	0.2194 ± 0.0014
	P-forcing	0.0497 ± 0.0008	0.3092 ± 0.0006	0.2935 ± 0.0006	0.2779 ± 0.0006	0.2575 ± 0.0014
	WaveNet	0.0509 ± 0.0011	0.2912 ± 0.0012	0.2878 ± 0.0006	0.2714 ± 0.0008	0.2506 ± 0.0014
	WaveGAN	0.0526 ± 0.0006	0.2592 ± 0.0006	0.2738 ± 0.0008	0.2648 ± 0.0008	0.2494 ± 0.0004
LSTnet	No augmentation	0.1429 ± 0.0018	0.3688 ± 0.0008	0.2884 ± 0.0016	0.2977 ± 0.0008	0.2862 ± 0.0004
	SparseGAN	**0.0154 ± 0.0008**	**0.1282 ± 0.0006**	**0.1715 ± 0.0008**	**0.1687 ± 0.0008**	**0.1634 ± 0.0008**
	TimeGAN	0.0172 ± 0.0008	0.1472 ± 0.0006	0.1827 ± 0.0008	0.1892 ± 0.0008	0.1882 ± 0.0004
	T-CGAN	0.0244 ± 0.0008	0.2078 ± 0.0008	0.1935 ± 0.0008	0.2098 ± 0.0008	0.2014 ± 0.0004
	RCGAN	0.0477 ± 0.0006	0.2504 ± 0.0008	0.2019 ± 0.0008	0.2302 ± 0.0008	0.2219 ± 0.0006
	C-RNN-GAN	0.0678 ± 0.0008	0.2717 ± 0.0006	0.2371 ± 0.0008	0.2586 ± 0.0008	0.2403 ± 0.0004
	T-forcing	0.0477 ± 0.0012	0.2491 ± 0.0012	0.2079 ± 0.0006	0.2294 ± 0.0008	0.2094 ± 0.0004
	P-forcing	0.0475 ± 0.0008	0.2982 ± 0.0008	0.2765 ± 0.0008	0.2704 ± 0.0008	0.2414 ± 0.0004
	WaveNet	0.0578 ± 0.0008	0.2795 ± 0.0006	0.2691 ± 0.0008	0.2696 ± 0.0008	0.2456 ± 0.0004
	WaveGAN	0.0577 ± 0.0014	0.2281 ± 0.0006	0.2545 ± 0.0008	0.2516 ± 0.0008	0.2424 ± 0.0004

Bold font indicates the best performing model, while underlined font represents the second best performing model
We sample 10 training data points and then use 100 synthetic data points for augmenting our training sets

Table 3 Performance summary of MAE scores for data augmentation in low-data regime

Forecasting model	Generative model	Datasets				
		Sine waves	Google stocks	Energy appliances	Air quality	Power consumption
LSTM	No augmentation	0.1352 ± 0.0018	0.2652 ± 0.0008	0.2859 ± 0.0016	0.3175 ± 0.0008	0.1942 ± 0.0004
	SparseGAN	**0.0177 ± 0.0006**	**0.1311 ± 0.0006**	**0.1719 ± 0.0008**	**0.1858 ± 0.0008**	**0.0684 ± 0.0008**
	TimeGAN	0.0198 ± 0.0014	0.1556 ± 0.0006	0.1816 ± 0.0008	0.2214 ± 0.0008	0.0882 ± 0.0006
	T-CGAN	0.0206 ± 0.0012	0.2071 ± 0.0006	0.1932 ± 0.0012	0.2276 ± 0.0006	0.0984 ± 0.0006
	RCGAN	0.0329 ± 0.0014	0.2366 ± 0.0006	0.1989 ± 0.0008	0.2374 ± 0.0006	0.1094 ± 0.0012
	C-RNN-GAN	0.0356 ± 0.0008	0.2573 ± 0.0006	0.2288 ± 0.0014	0.2346 ± 0.0006	0.1105 ± 0.0014
	T-forcing	0.0485 ± 0.0008	0.2583 ± 0.0006	0.2529 ± 0.0006	0.2464 ± 0.0008	0.1291 ± 0.0006
	P-forcing	0.0465 ± 0.0006	0.2415 ± 0.0006	0.2489 ± 0.0014	0.2374 ± 0.0006	0.1247 ± 0.0012
	WaveNet	0.0473 ± 0.0008	0.2681 ± 0.0006	0.2429 ± 0.0012	0.2377 ± 0.0008	0.1307 ± 0.0012
	WaveGAN	0.0507 ± 0.0012	0.2792 ± 0.0006	0.2469 ± 0.0008	0.2476 ± 0.0006	0.1348 ± 0.0006
LSTnet	No augmentation	0.2344 ± 0.0018	0.2625 ± 0.0014	0.2747 ± 0.0016	0.2669 ± 0.0008	0.1837 ± 0.0004
	SparseGAN	**0.0149 ± 0.0006**	**0.1293 ± 0.0006**	0.**1719 ± 0.0008**	**0.1668 ± 0.0008**	**0.0515 ± 0.0008**
	TimeGAN	0.0168 ± 0.0008	0.1451 ± 0.0008	0.1808 ± 0.0008	0.1887 ± 0.0006	0.0778 ± 0.0006
	T-CGAN	0.0192 ± 0.0006	0.1973 ± 0.0008	0.1942 ± 0.0008	0.2091 ± 0.0008	0.0914 ± 0.0006
	RCGAN	0.0289 ± 0.0006	0.2281 ± 0.0006	0.2042 ± 0.0008	0.2124 ± 0.0014	0.1029 ± 0.0008
	C-RNN-GAN	0.0295 ± 0.0008	0.2473 ± 0.0008	0.2139 ± 0.0008	0.2176 ± 0.0012	0.1090 ± 0.0006
	T-forcing	0.0435 ± 0.0014	0.2496 ± 0.0006	0.2335 ± 0.0006	0.2432 ± 0.0008	0.1114 ± 0.0012
	P-forcing	0.0408 ± 0.0014	0.2631 ± 0.0008	0.2439 ± 0.0012	0.2392 ± 0.0006	0.1261 ± 0.0014
	WaveNet	0.0412 ± 0.0012	0.2614 ± 0.0006	0.2405 ± 0.0006	0.2365 ± 0.0012	0.1282 ± 0.0008
	WaveGAN	0.0487 ± 0.0012	0.2666 ± 0.0006	0.2491 ± 0.0008	0.2456 ± 0.0012	0.1317 ± 0.0014

Bold font indicates the best performing model, while underlined font represents the second best performing model
We sample 50 training data points and then use 100 synthetic data points for augmenting our training sets

Table 4 Performance summary of MAE scores for data augmentation in low-data regime

Forecasting model	Generative model	Datasets				
		Sine waves	Google stocks	Energy appliances	Air quality	Power consumption
LSTM	SparseGAN	**0.0813 ± 0.0026**	**0.0362 ± 0.012**	**0.2453 ± 0.0007**	**0.4576 ± 0.006**	**0.2932 ± 0.016**
	TimeGAN	0.0937 ± 0.0016	0.0384 ± 0.0002	0.2691 ± 0.0004	0.4686 ± 0.0018	0.3141 ± 0.0016
	T-CGAN	0.0948 ± 0.0014	0.0388 ± 0.0012	0.2736 ± 0.0016	0.4776 ± 0.0012	0.3172 ± 0.0012
	RCGAN	0.0973 ± 0.0019	0.0404 ± 0.0014	0.2926 ± 0.0005	0.5171 ± 0.0014	0.3232 ± 0.0006
	C-RNN-GAN	0.1272 ± 0.0014	0.0386 ± 0.0012	0.4831 ± 0.0015	0.5148 ± 0.0008	0.3335 ± 0.0008
	T-forcing	0.1507 ± 0.0004	0.0388 ± 0.0016	0.3159 ± 0.0015	0.5576 ± 0.0018	0.3432 ± 0.0012
	P-forcing	0.1164 ± 0.0008	0.0436 ± 0.0012	0.3036 ± 0.0006	0.5436 ± 0.0008	0.3515 ± 0.0014
	WaveNet	0.1174 ± 0.0016	0.0428 ± 0.0014	0.3118 ± 0.0015	0.5561 ± 0.0018	0.3532 ± 0.0014
	WaveGAN	0.1344 ± 0.0013	0.0412 ± 0.0011	0.3078 ± 0.0007	0.5496 ± 0.0006	0.3572 ± 0.0012
	Real data	0.0772 ± 0.0014	0.0352 ± 0.0018	0.2426 ± 0.0014	0.4535 ± 0.0018	0.2916 ± 0.0018
LSTnet	SparseGAN	**0.0812 ± 0.0016**	**0.0344 ± 0.0012**	**0.2284 ± 0.0017**	**0.4536 ± 0.0006**	**0.2769 ± 0.0016**
	TimeGAN	0.0937 ± 0.0006	0.0352 ± 0.0014	0.2366 ± 0.0012	0.4589 ± 0.0008	0.2898 ± 0.0014
	T-CGAN	0.0948 ± 0.0014	0.0358 ± 0.0016	0.2874 ± 0.0018	0.4618 ± 0.0008	0.2977 ± 0.0012
	RCGAN	00.0973 ± 00.0019	00.0368 ± 00.0014	00.4672 ± 00.0014	00.4629 ± 00.0008	00.3112 ± 0.0012
	C-RNN-GAN	0.1274 ± 0.0004	0.0368 ± 0.0004	0.3092 ± 0.0016	0.4621 ± 0.0014	0.3158 ± 0.0008
	T-forcing	0.1501 ± 0.0004	0.0368 ± 0.0018	0.3024 ± 0.0008	0.4632 ± 0.0008	0.3262 ± 0.0009
	P-forcing	0.1162 ± 0.0008	0.0365 ± 0.0014	0.3039 ± 0.0018	0.4638 ± 0.0018	0.3369 ± 0.0011
	WaveNet	0.1174 ± 0.0006	0.0368 ± 0.0016	0.3054 ± 0.0007	0.4643 ± 0.0014	0.3412 ± 0.0014
	WaveGAN	0.1342 ± 0.0013	0.0386 ± 0.0001	0.3026 ± 0.0012	0.4648 ± 0.0008	0.3437 ± 0.0026
	Real data	0.0743 ± 0.0016	0.0342 ± 0.0008	0.2191 ± 0.0008	0.4508 ± 0.0008	0.2748 ± 0.0006

Bold font indicates the best performing model, while underlined font represents the second best performing model
We sample 50 training data points and then use 100 synthetic data points for augmenting our training sets

Table 5 Performance summary of the distance between the generated time series of different methods and original data in terms of classification accuracy score

Generative model	Datasets				
	Sine waves	Google stocks	Energy appliances	Air quality	Power consumption
SparseGAN	**0.0091 ± 0.0006**	**0.0822 ± 0.0026**	**0.1624 ± 0.0008**	**0.3862 ± 0.0016**	**0.2926 ± 0.0024**
TimeGAN	0.0118 ± 0.0008	0.1026 ± 0.0021	0.2764 ± 0.0012	0.4026 ± 0.0012	0.3066 ± 0.0018
T-CGAN	0.2292 ± 0.0006	0.1824 ± 0.0014	0.2744 ± 0.0004	0.418 ± 0.0012	0.3622 ± 0.0008
RCGAN	0.0226 ± 0.0008	0.1962 ± 0.0027	0.3362 ± 0.0017	0.4424 ± 0.0012	0.4926 ± 0.0005
C-RNN-GAN	0.2296 ± 0.0040	0.3997 ± 0.0028	0.4992 ± 0.0008	0.4412 ± 0.0014	0.4894 ± 0.0005
T-forcing	0.4955 ± 0.0001	0.2264 ± 0.0035	0.4834 ± 0.0004	0.4324 ± 0.0016	0.4917 ± 0.0005
P-forcing	0.4308 ± 0.0027	0.2575 ± 0.0026	0.4125 ± 0.0006	0.4465 ± 0.0012	0.4725 ± 0.0014
WaveNet	0.1586 ± 0.0011	0.2323 ± 0.0028	0.3973 ± 0.0010	0.4266 ± 0.0012	0.4866 ± 0.0006
WaveGAN	0.2774 ± 0.0013	0.2175 ± 0.0022	0.3634 ± 0.0012	0.4534 ± 0.0015	0.4816 ± 0.0005

Bold font indicates the best performing model, while underlined font represents the second best performing model
Please refer to Sect. 5.3 for details

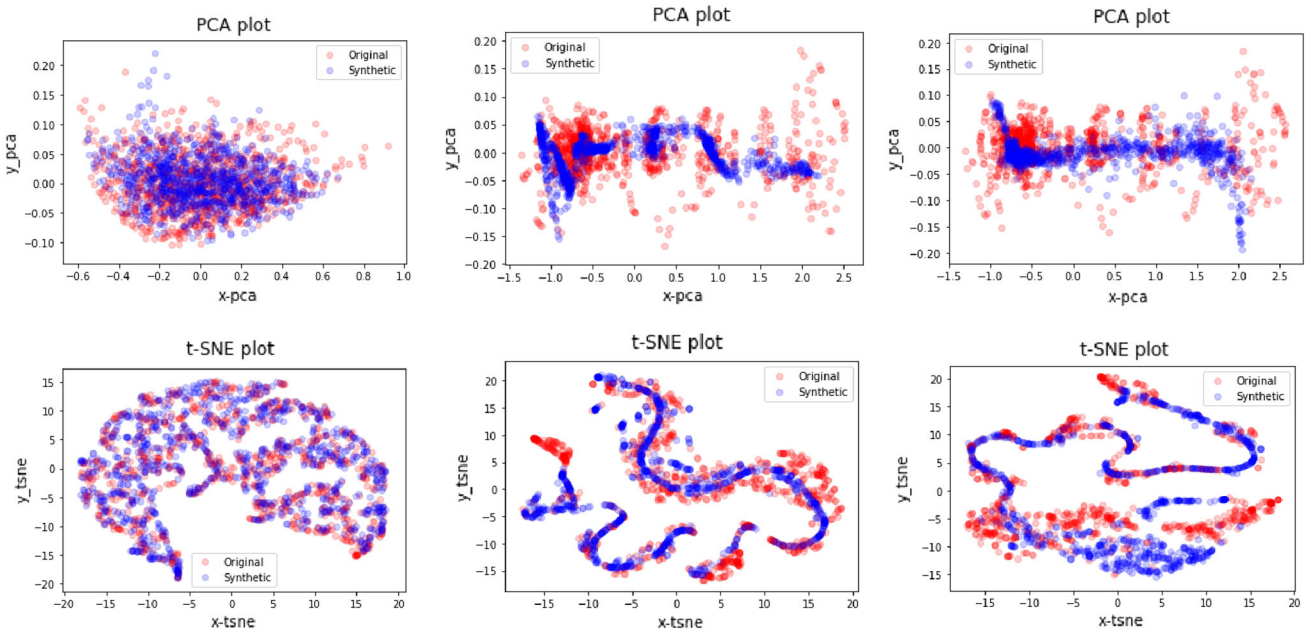

Fig. 2 A visual comparison of real data and its generated counterpart. The first row of the figure depicts PCA visualizations of Sines Waves, Google Stocks, and Energy Appliances, whereas the second row depicts t-SNE visualizations of Sines Waves, Google Stocks, and Energy Appliances, respectively. Here, red dots represent authentic data while blue ones represent synthetic data (colour figure online)

Table 6 Analysis of the hyper-parameter λ using Energy dataset based on LSTM forecasting model for data substitution task

λ	0	0.1	1	10	100
MAE	0.2831 ± 0.0018	0.2752 ± 0.0016	0.2653 ± 0.0012	0.2453 ± 0.0007	0.2572 ± 0.0012

Table 7 Analysis of the impact of sparse attention based on LSTM forecasting model for data substitution task using MAE scores

Attention mechanism	Energy appliances	Power consumption
1.5 Entmax	0.1624 ± 0.0008	0.2926 ± 0.0024
Softmax	0.1831 ± 0.0018	0.3758 ± 0.0016

of-the-art baseline models in the five datasets as shown in Tables 2 and 3.

Data substitution scenario We now consider the scenario where we fully train our forecasting models on synthetic data. To answer *RQ3*, we compare the performance of synthetic-data-trained forecasting models when using SparseGAN against other baseline models. We present the findings of this experiment in Table 4. It is interesting to see that the SparseGAN model consistently outperforms all baseline models. For instance, there is around 6% improvement over the previous best model results, TimeGAN assuming that real-data performance can be achieved. In addition, the findings corroborate that SparseGAN can better handle the long-range dependencies between distant time stamps compared to other baseline models. SparseGAN also outperforms other baseline models in mimicking the properties of real data, both regular and irregular, emphasizing its utility in generating synthetic data that can then substitute real-world data.

6.2 Generated data diversity

To further analyze the generated data, we explore how well Sparse-GAN generated data preserves the diversity and patterns of original data as described in Sect. 5.3 (*RQ4*). Experimental results on different datasets are illustrated in Table 5. As illustrated in Table 5, SparseGAN consistently generates synthesized data that closely resembles real-world time-series' diverse patterns. Furthermore, we observe in Table 5 that SparseGAN generated time-series are indistinguishable from real-world data. The findings in Table 5 demonstrate that our model achieves the least classification accuracy with a big gap compared to other generative models. To further highlight the similarities between the actual data and synthetic data, Fig. 2 is a visualization graph of data distributions transferred to two dimensions using PCA and t-SNE. Again, we observe that synthesized data closely mimics the various patterns of real-world time series, demonstrating the efficiency of the suggested approach.

7 Sensitivity analysis

We finally conducted an ablation study to test the robustness of our findings. First, we investigated the sensitivity of the parameter λ, which balances the generation loss and the adversarial training part. We alter the value of λ among $\{0, 0.1, 1, 10, 100\}$ and report the performance of SparseGAN on the Energy Appliances dataset for data substitution. Here, using λ is equivalent to training our model without the supervision network. As shown in Table 6, SparseGAN performs better as λ increases. It is evident that our proposed approach benefits from the supervision network and enhances performance, but excessively high values of λ which give huge weight to the supervision network, have a negative impact on SparseGAN. Second, we investigated the effect of various attention mechanisms on SparseGAN performance. Table 7 indicates that, as compared to softmax, 1.5 entmax is more sparse and assigns greater scores to significant information that helps the model's performance.

8 Conclusion

In this paper, we have proposed a novel generative adversarial network, SparseGAN, which addresses the limitations of previous time-series generation models with regard to long-term dependencies. While previous research has built time-series generation on RNN and convolutional layers, we based the SparseGAN on sparse self-attention mechanism. In addition, our proposed model utilizes original data for supervision purposes. We show that SparseGAN can capture the long dependencies in time-series data efficiently. It is also capable of maintaining the original distribution of the data based on internal characteristics. The experimental findings substantiate that SparseGAN-generated data outperformed generative baseline models in the regular and irregular time-series data. In particular, the forecasting models which were trained on SparseGAN-generated data perform similarly to models trained on real-world data. In addition, SparseGAN provides an effective way to augment training data in low-resourced data settings.

Author contributions Nourhan Ahmed conceived of the presented idea. Nourhan Ahmed developed the theory and the methodology. Nourhan Ahmed performed the experiments and Nourhan Ahmed and Lars Schmidt-Thieme verified the proposed methodology. Both authors discussed the findings and contributed to the final manuscript.

Funding Open Access funding enabled and organized by Projekt DEAL.

Declarations

Conflict of interest The authors declare that there are no conflicts of interest to disclose.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

1. Wiese, M., Knobloch, R., Korn, R., Kretschmer, P.: Quant gans: deep generation of financial time series. Quant. Financ. **20**(9), 1419–1440 (2020)
2. Shao, S., Wang, P., Yan, R.: Generative adversarial networks for data augmentation in machine fault diagnosis. Comput. Ind. **106**, 85–93 (2019)
3. Esteban, C., Hyland, S.L., Rätsch, G.: Real-valued (medical) time series generation with recurrent conditional gans. arXiv preprint arXiv:1706.02633 (2017)
4. Mogren, O.: C-RNN-gan: Continuous recurrent neural networks with adversarial training. arXiv preprint arXiv:1611.09904 (2016)
5. Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A.C., Bengio, Y.: Generative adversarial nets. In: NIPS (2014)
6. Ramponi, G., Protopapas, P., Brambilla, M., Janssen, R.: T-cgan: Conditional generative adversarial network for data augmentation in noisy time series with irregular sampling. arXiv preprint arXiv:1811.08295 (2018)
7. Yoon, J., Jarrett, D., van der Schaar, M.: Time-series generative adversarial networks. In: Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8–14, 2019, Vancouver, BC, Canada, pp. 5509–5519 (2019)
8. Liang, J., Yang, J., Lee, H.-Y., Wang, K., Yang, M.-H.: Sub-gan: an unsupervised generative model via subspaces. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 698–714 (2018)
9. Zhang, H., Goodfellow, I., Metaxas, D., Odena, A.: Self-attention generative adversarial networks. In: International Conference on Machine Learning, pp. 7354–7363. PMLR (2019)
10. Wang, Y., Bilinski, P., Bremond, F., Dantcheva, A.: Imaginator: Conditional spatio-temporal gan for video generation. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 1160–1169 (2020)
11. Sun, C., Hong, S., Song, M., Li, H.: A review of deep learning methods for irregularly sampled medical time series data. arXiv preprint arXiv:2010.12493 (2020)
12. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: Proceedings of the 31st International Conference on Neural Information Processing Systems, pp. 6000–6010 (2017)
13. Tang, G., Müller, M., Rios, A., Sennrich, R.: Why self-attention?: A targeted evaluation of neural machine translation architectures. In: Conference on Empirical Methods in Natural Language Processing, October 31–November 4 Brussels, Belgium, 2018, pp. 4263–4272 (2018)
14. Salazar, J., Kirchhoff, K., Huang, Z.: Self-attention networks for connectionist temporal classification in speech recognition. In: ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 7115–7119. IEEE (2019)
15. Ahmed, N., Rashed, A., Schmidt-Thieme, L.: Learning attentive attribute-aware node embeddings in dynamic environments . Int. J. Data Sci. Analytics 1–13(2022)
16. Lou, H., Qi, Z., Li, J.: One-dimensional data augmentation using a Wasserstein generative adversarial network with supervised signal. In: 2018 Chinese Control And Decision Conference (CCDC), pp. 1896–1901. IEEE (2018)
17. Dogariu, M., Ştefan, L.-D., Boteanu, B.A., Lamba, C., Ionescu, B.: Towards realistic financial time series generation via generative adversarial learning. In: 2021 29th European Signal Processing Conference (EUSIPCO), pp. 1341–1345. IEEE (2021)
18. Dogariu, M., Ştefan, L.-D., Boteanu, B.A., Lamba, C., Kim, B., Ionescu, B.: Generation of realistic synthetic financial time-series. ACM Trans. Multimed. Comput., Commun., Appl. (TOMM) **18**(4), 1–27 (2022)
19. Bandara, K., Hewamalage, H., Liu, Y.-H., Kang, Y., Bergmeir, C.: Improving the accuracy of global forecasting models using time series data augmentation. Pattern Recogn. **120**, 108148 (2021)
20. Arnout, H., Bronner, J., Runkler, T.: Clare-gan: Generation of class-specific time series. In: 2021 IEEE Symposium Series on Computational Intelligence (SSCI), pp. 01–08. IEEE (2021)
21. Shih, S.-Y., Sun, F.-K., Lee, H.-Y.: Temporal pattern attention for multivariate time series forecasting. Mach. Learn. **108**(8), 1421–1441 (2019)
22. Wu, S., Xiao, X., Ding, Q., Zhao, P., Wei, Y., Huang, J.: Adversarial sparse transformer for time series forecasting. In: Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6–12, 2020, Virtual (2020)
23. Yin, X., Han, Y., Sun, H., Xu, Z., Yu, H., Duan, X.: Multi-attention generative adversarial network for multivariate time series prediction. IEEE Access **9**, 57351–57363 (2021)
24. Gao, C., Zhang, N., Li, Y., Bian, F., Wan, H.: Self-attention-based time-variant neural networks for multi-step time series forecasting. Neural Comput. Appl. **34**(11), 8737–8754 (2022)
25. Tripathi, A.M., Baruah, R.D.: Multivariate time series classification with an attention-based multivariate convolutional neural network. In: 2020 International Joint Conference on Neural Networks (IJCNN), pp. 1–8. IEEE (2020)
26. Chen, W., Shi, K.: Multi-scale attention convolutional neural network for time series classification. Neural Netw. **136**, 126–140 (2021)
27. Liu, S., Zhou, B., Ding, Q., Hooi, B., Bo Zhang, Z., Shen, H., Cheng, X.: Time series anomaly detection with adversarial reconstruction networks. IEEE Trans. Knowl. Data Eng. **35**(4), 4293–4306(2022)
28. Ding, C., Sun, S., Zhao, J.: Mst-gat: a multimodal spatial-temporal graph attention network for time series anomaly detection. Inf. Fus. **89**, 527–536 (2023)
29. Dai, Z., Yang, Z., Yang, Y., Carbonell, J.G., Le, Q., Salakhutdinov, R.: Transformer-xl: attentive language models beyond a fixed-length context. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 2978–2988 (2019)
30. Song, K., Wang, K., Yu, H., Zhang, Y., Huang, Z., Luo, W., Duan, X., Zhang, M.: Alignment-enhanced transformer for constraining NMT with pre-specified translations. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 8886–8893 (2020)
31. Kitaev, N., Kaiser, L., Levskaya, A.: Reformer: the efficient transformer. In: International Conference on Learning Representations (2019)
32. Dehghani, M., Gouws, S., Vinyals, O., Uszkoreit, J., Kaiser, L.: Universal transformers. In: International Conference on Learning Representations (2018)
33. Hu, Y., Xiao, F.: Network self attention for forecasting time series. Appl. Soft Comput. **124**, 109092 (2022)
34. Wan, R., Tian, C., Zhang, W., Deng, W., Yang, F.: A multivariate temporal convolutional attention network for time-series forecasting. Electronics **11**(10), 1516 (2022)
35. Zhang, J., Zhao, Y., Li, H., Zong, C.: Attention with sparsity regularization for neural machine translation and summarization. IEEE/ACM Trans. Audio, Speech, Lang. Process. **27**(3), 507–518 (2018)
36. Child, R., Gray, S., Radford, A., Sutskever, I.: Generating long sequences with sparse transformers. arXiv preprint arXiv:1904.10509 (2019)

37. Wen, Q., Sun, L., Yang, F., Song, X., Gao, J., Wang, X., Xu, H.: Time series data augmentation for deep learning: A survey. arXiv preprint arXiv:2002.12478 (2020)
38. Malekzadeh, M., Clegg, R.G., Haddadi, H.: Replacement autoencoder: a privacy-preserving algorithm for sensory data analysis. arXiv preprint arXiv:1710.06564 (2017)
39. Gupta, A.K., Shanker, U.: Mad-rappel: mobility aware data replacement and prefetching policy enrooted LBS. J. King Saud Univer.-Comput. Inf. Sci. **34**(6), 3454–3467 (2022)
40. Zhang, J., Du, J., Dai, L.: A GRU-based encoder-decoder approach with attention for online handwritten mathematical expression recognition. In: 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), vol. 1, pp. 902–907. IEEE (2017)
41. Peng, C., Li, Y., Yu, Y., Zhou, Y., Du, S.: Multi-step-ahead host load prediction with GRU based encoder-decoder in cloud computing. In: 2018 10th International Conference on Knowledge and Smart Technology (KST), pp. 186–191. IEEE (2018)
42. Tonekaboni, S., Eytan, D., Goldenberg, A.: Unsupervised representation learning for time series with temporal neighborhood coding. In: International Conference on Learning Representations (2021). https://openreview.net/forum?id=8qDwejCuCN
43. Ham, H., Jun, T.J., Kim, D.: Unbalanced gans: Pre-training the generator of generative adversarial network using variational autoencoder. arXiv preprint arXiv:2002.02112 (2020)
44. Luo, Y., Cai, X., Zhang, Y., Xu, J., Yuan, X.: Multivariate time series imputation with generative adversarial networks. In: Proceedings of the 32nd International Conference on Neural Information Processing Systems, pp. 1603–1614 (2018)
45. Shukla, S.N., Marlin, B.: Multi-time attention networks for irregularly sampled time series. In: International Conference on Learning Representations (2020)
46. Jain, S., Wallace, B.C.: Attention is not explanation. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, vol. 1, pp. 3543–3556 (2019)
47. Peters, B., Niculae, V., Martins, A.F.: Sparse sequence-to-sequence models. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 1504–1519 (2019)
48. Blondel, M., Martins, A., Niculae, V.: Learning classifiers with fenchel-young losses: generalized entropies, margins, and algorithms. In: The 22nd International Conference on Artificial Intelligence and Statistics, pp. 606–615. PMLR (2019)
49. Wu, S., Xiao, X., Ding, Q., Zhao, P., Wei, Y., Huang, J.: Adversarial sparse transformer for time series forecasting. Adv. Neural Inf. Process. Syst. **33**, 17105–17115 (2020)
50. Tsallis, C.: Possible generalization of Boltzmann–Gibbs statistics. J. Stat. Phys. **52**(1), 479–487 (1988)
51. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
52. Ba, J.L., Kiros, J.R., Hinton, G.E.: Layer normalization. arXiv preprint arXiv:1607.06450 (2016)
53. Schlegl, T., Seeböck, P., Waldstein, S.M., Schmidt-Erfurth, U., Langs, G.: Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In: International Conference on Information Processing in Medical Imaging, pp. 146–157. Springer (2017)
54. Shukla, S.N., Marlin, B.: Interpolation-prediction networks for irregularly sampled time series. In: International Conference on Learning Representations (2018)
55. Williams, R.J., Zipser, D.: A learning algorithm for continually running fully recurrent neural networks. Neural Comput. **1**(2), 270–280 (1989)
56. Goyal, A., Lamb, A., Zhang, Y., Zhang, S., Courville, A.C., Bengio, Y.: Professor forcing: a new algorithm for training recurrent networks. In: NIPS (2016)
57. van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., Kavukcuoglu, K.: WaveNet: a generative model for raw audio. arXiv preprint arXiv:1609.03499(2016)
58. Donahue, C., McAuley, J., Puckette, M.: Adversarial audio synthesis. arXiv preprint arXiv:1802.04208 (2018)
59. Sajjadi, M.S., Bachem, O., Lucic, M., Bousquet, O., Gelly, S.: Assessing generative models via precision and recall. In: Proceedings of the 32nd International Conference on Neural Information Processing Systems, pp. 5234–5243 (2018)
60. Dankar, F.K., Ibrahim, M., Castelli, M.: Fake it till you make it: guidelines for effective synthetic data generation. Appl. Sci. **11**(5), 2076–3417 (2021)
61. Lai, G., Chang, W.-C., Yang, Y., Liu, H.: Modeling long-and short-term temporal patterns with deep neural networks. In: The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, pp. 95–104 (2018)
62. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local Nash equilibrium. Adv. Neural Inf. Process. Syst. **30** (2017)
63. Hu, Z., Tan, B., Salakhutdinov, R.R., Mitchell, T.M., Xing, E.P.: Learning data manipulation for augmentation and weighting. In: Wallach, H., Larochelle, H., Beygelzimer, A., d' Alché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems (2019)
64. Kumar, V., Choudhary, A., Cho, E.: Data augmentation using pre-trained transformer models. arXiv preprint arXiv:2003.02245 (2020)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.