Energy and AI 9 (2022) 100161

Contents lists available at ScienceDirect

Energy and AI

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

Synthetic demand data generation for individual electricity consumers : Generative Adversarial Networks (GANs)

Bilgi Yilmaz [a,b,*], Ralf Korn [a,c]

[a] TU Kaiserslautern, Kaiserslautern, Germany
[b] Kahramanmaras Sutcu Imam University, Kahramanmaras, Turkey
[c] Fraunhofer ITWM, Kaiserslautern, Germany

ARTICLE INFO

MSC:
00-01
99-00

Keywords:
Electricity consumption
Generative adversarial networks
Synthetic data generation
Unsupervised learning
RCGAN
TimeGAN
CWGAN
RCWGAN

ABSTRACT

Load modeling is one of the crucial tasks for improving smart grids' energy efficiency. Among many alternatives, machine learning-based load models have become popular in applications and have shown outstanding performance in recent years. The performance of these models highly relies on data quality and quantity available for training. However, gathering a sufficient amount of high-quality data is time-consuming and extremely expensive. In the last decade, Generative Adversarial Networks (GANs) have demonstrated their potential to solve the data shortage problem by generating synthetic data by learning from recorded/empirical data. Educated synthetic datasets can reduce prediction error of electricity consumption when combined with empirical data. Further, they can be used to enhance risk management calculations. Therefore, we propose RCGAN, TimeGAN, CWGAN, and RCWGAN which take individual electricity consumption data as input to provide synthetic data in this study. Our work focuses on one dimensional times series, and numerical experiments on an empirical dataset show that GANs are indeed able to generate synthetic data with realistic appearance.

1. Introduction

A smart grid is an electrical power grid that incorporates information and communication technologies with the power grid. Smart grids aim to improve electricity efficiency by exchanging information in real-time between suppliers and consumers. They enable suppliers to forecast electricity demand (e.g., current energy consumption and user profiles). Hence, energy suppliers can maximize electricity efficiency by providing an accurate electricity load [1].

If energy suppliers deliver more electricity than required due to insufficient predictions, they may suffer high financial losses. Conversely, energy suppliers experience severe crises, such as frequency drops and blackouts, if they provide less electricity to the system than required. An efficient prediction needs a high amount and quality of data. However, gathering a sufficient amount of high-quality electricity data is incredibly costly and time-consuming. More importantly, while electricity grid models of all kinds are publicly available, data is a much more scarce resource. Due to high data privacy concerns, the research community must rely on rare and limited datasets that are publicly available, which limits the research and application of new load prediction methods.

Datasets containing sensitive information tend to be strictly protected and, hence, out of reach of a third company. If we can generate synthetic data that preserves the statistical behavior and stylized facts of empirical data, we can use the synthetic data to remove the privacy barrier to data disclosure. In this regard, high-quality synthetic datasets have invaluable applications, such as understanding the distribution of empirical data, compression, efficient storing, data augmentation, system testing, and data disclosure or avoiding data privacy [2]. Synthetic data generation is an effective method to address such quantity, quality, and privacy issues.

There are many methods to generate synthetic data but, Generative Adversarial Networks (GANs) are standing out due to their performance and the flexibility they show in representing the empirical data, as evidenced by their success in generating and manipulating images and natural languages [3–6]. Therefore, this study focuses its attention only on GANs.

A standard GAN consists of two neural networks: a generator and a discriminator. By feeding GANs with a vector sampled from a standard multivariate normal distribution $\mathcal{N}(0, I)$ and utilizing the generator and discriminator (critic), we find a deterministic transformation that maps $\mathcal{N}(0, I)$ onto the distribution of the empirical data [2]. This way,

* Corresponding author at: TU Kaiserslautern, Kaiserslautern, Germany.
E-mail addresses: yilmaz@mathematik.uni-kl.de (B. Yilmaz), korn@mathematik.uni-kl.de (R. Korn).

https://doi.org/10.1016/j.egyai.2022.100161
Received 14 February 2022; Received in revised form 9 April 2022; Accepted 9 April 2022
Available online 20 April 2022

the generator attempts to produce fake samples while the discriminator focuses on distinguishing between real and fake/artificial samples [7].

GANs are generative models qualified enough to generate new samples never seen before, but they also should have similar distributional properties as empirical data [8]. Thus, they can be used for data augmentation. Although GANs have been originally developed for image processing and computer vision in the first place, achieving excellent results for the generation of realistic images, has resulted in an unparalleled surge of interest, and significant advances have been made in many different research fields [7]. Their performance also captivates researchers who work on sequential data like music generation, medical data, and finance. Our study also considers the sequential data applications of GANs, more specifically, focusing on individual electricity consumption.

The original GANs are notoriously tough to train and generally suffer from the problem of missing modes (lack of variety), where the discriminator cannot assemble samples in some regions. GANs also highly suffer from mode collapse and vanishing gradient. Therefore, training stabilization and convergence of GANs still have been intensively studied, both theoretically and experimentally. We present detailed literature on the evaluation of GANs in Section 2.

There are two principal strategies in the application of GANs to load data forecasting: (i) Using a standard GAN architecture to generate synthetic load data where its performance is evaluated by comparing the divergence/convergence behavior of the synthetic data compared with regard to the empirical data. (ii) More complex GAN architectures are used in generating synthetic load data and added to recorded load data to extend empirical data. Then, a learning algorithm is utilized for extending the training data. The first strategy provides only a scenario generation solution and generates load profiles that lack precision. On the other hand, the second and the most commonly used strategy is data augmentation, despite being exceptionally influential and lacking in illustrating the full capacities of GANs [8]. Thus, in this study, we use the first strategy and focus on the synthetic data generation of 15 min, individual electricity consumption records using GANs.

The electricity market is a domain that particularly looks for synthetic data generation technology. We can remove barriers of costs and data protection by using synthetic data generation methods like GANs to generate fully-synthetic individual electricity consumption records.

Most of the studies commonly offer using long-short-term memory (LSTM) in applications of GANs to time series data. Although the studies suggest LSTM, we avoid using it to reduce the computational cost of the training of GANs and overfitting issues. Instead, we generate synthetic data by using Recurrent Conditional GAN (RCGAN) [9], Time-Series GAN (TimeGAN) [10], Conditional Wasserstein GAN (CWGAN) [3], and Recurrent Conditional Wasserstein GAN (RCWGAN) in this article. We compare the results of each GAN with empirical data, and they show very accurate synthetic data in the sense of distributional properties and similarity in evolution over time.

We offer GANs so that energy suppliers may share their electricity consumption records without any privacy concerns. Energy suppliers can model the grid load more accurately and increase their prediction efficiency. In this aspect, energy suppliers only need to share their electricity consumption records to construct appropriate GANs for generating data similar to the recorded data in the sense that is required.

The primary contributions of our study are:

- We propose to use GANs to generate data for 15 min individual electricity consumption.
- We compare a variety of GANs (RCGAN, TimeGAN, CWGAN, and RCWGAN) to show their suitability for different tasks.
- We offer to use recurrent neural networks instead of LSTM neural networks for time series.

Our work is structured as follows: Section 2 contains a short literature review on GANs. Section 3 serves to illustrate some aspects of GANs we use without any details. Section 4 introduces the individual electricity consumption data besides the explanatory data analysis and synthetic data generation concerning the selected GANs. Finally, Section 5 concludes the study, and Appendix A illustrates the evaluation metrics.

2. A short review of literature on GANs

The framework of GANs has first been proposed in [11] and rapidly gained popularity in the machine learning field. It is based on a two-player game, where the players are called generator and discriminator. The generator is trained to generate fake/artificial data from a given distribution to deceive the discriminator while the discriminator is trained to distinguish between real and artificial samples. [12] extends the framework to a conditional model by conditioning the actions of both players on additional information (such as labels, tags or attributes). This additional information is fed into the GAN framework by an additional input layer.

As GANs should generate realistic data out of noise on the basis of a real data sample, their common objective is to minimize the distance between the empirical and synthetic/artificial data distributions. Depending on how this distance is measured, the objectives that are used to train GANs will differ. Popular examples of distance measure are the Wasserstein distance and f-divergence.

GANs are notoriously tough to train and can suffer from the problem of missing modes or mode collapse (lack of variety), where the model cannot assemble samples in some regions of the space and from vanishing gradients which essentially stops the training process in a non-optimal state. To deal with these issues gained a lot of attention in the GAN research. It is beyond the scope of our study to fully report on this. Comparison of various distance measures in [3] suggests to use the Wasserstein-1-metric which leads to WGAN, one of the approaches we will compare below. An alternative to overcome the training problems is suggested in [13]. They propose the Least Squares GANs (LSGANs), which adopts a least-squares loss function for the discriminator to overcome the vanishing gradients issue.

For our task of generating data in the form of time series, we take a short look at their application on the financial markets and at the electricity markets. [14] introduces a new GAN called QuantGANs for financial markets. It is among the first works considering the mathematical foundations when applying GANs to generate financial time series. It consists of a GAN variation utilizing temporal convolutional networks to capture long-range dependencies like volatility clusters. Its objective is to approximate a realistic asset price simulator using a neural network, data-driven concept. QuantGANs is good at capturing the temporal dependence of financial time series (e.g., volatility clustering). [15] also proposes a variant of CGAN to generate order flow in the limit order book. CGANs have also been considered in [9,16–18] for univariate and in [19–21] for multivariate time-series generation. In general, modeling financial time series is more challenging than other time series due to their high volatility and unexpected market behavior. Hence, as alternative modeling, [22–24] offers a variety of GANs for modeling financial time series.

In parallel to these studies, the application of GANs to electricity markets has drawn accelerating attention in recent years. For instance, [25] uses DCGAN to generate wind and solar power plants' power profile scenarios. Both [8,26] investigate the application of DCGAN, LSGAN, and WGAN to forecast residential daily load profiles. [27] enters the stage with an ensemble method using auto-encoders and three types of GANs. Following [27–29] use GANs to generate synthetic load data. Both studies expand load data to increase sample and use the expanded data to build machine learning models to forecast load. [29] offers R-GAN, WGAN, and Metropolis–Hastings GAN (MH-GAN) to generate synthetic load data. [30] presents a bidirectional

GAN (BiGAN) to tackle the insufficient load data issue. [31] offers a CWGAN-GP for probabilistic load forecasting, where it uses GAN to assist a forecaster by generating residual scenarios. Additionally, the experimental study [32] reports PC-GAN is good at generating realistic wind power turbines scenarios. Furthermore, [33] uses vanilla GANs in generating realistic load profiles to improve building energy efficiency understanding, flexibility in electricity demand, and building-grid interactions.

As the studies we reviewed here reveal, GANs now constitute a broad literature in their own right focusing on theoretical issues and applications in various areas.

3. Methodology

As introduced in [11], GANs belong to the family of unsupervised learning algorithms. They are able to learn dense representations of input datasets and are utilized as generative models. The superiority of GANs is the ability to generate new samples having (nearly) the same distribution as the training dataset.

GANs consist of two neural networks which are competing with each other. In this respect, the training of GANs relies on game theory scenarios. The generator (G) directly produces samples from a well-known random distribution (e.g., normal and uniform distributions) as input (latent vector z) and its adversary, the discriminator (D), attempts to distinguish between samples drawn from training and generated data. The discriminator output ($D(x)$) represents the probability that a sample belongs to the distribution underlying the training data. On the other hand, the generator output ($G(z)$) is a sample from the distribution learned from the training dataset.

The competition between the generator and discriminator is mathematically formulated as a min–max game

$$\min_G \max_D V(D,G) = \min_G \max_D \Big(\mathbb{E}_{x\sim\mu}\Big[\log\Big(D(x)\Big)\Big] + \mathbb{E}_{z\sim\gamma}\Big[\log\Big(1-D(G(z))\Big)\Big] \Big),$$

where $D(x) : \mathbb{R}^n \mapsto [0,1]$ and $G(z) : \mathbb{R}^d \mapsto \mathbb{R}^n$. Here, the generator G transforms random samples $z \in \mathbb{R}^d$ from a predefined distribution γ (typically a Gaussian distribution) into generated samples $G(z)$ [34,35]. This representation is nothing but a linear function that heuristically displays the adversarial nature of the competition between the generator and discriminator. In this setting, the discriminator's output is a binary variable ($D(x) = 1$ and $D(x) = 0$ for real and fake samples, respectively) while the generator output is a vector.

Note that in generating or predicting time series data, identifying the (approximately) correct conditional distribution is more meaningful than learning the joint distribution. This is because in the studies of predictive modeling, we are interested in specifying conditional distributions of the future time series $x_{future} = x_{t+1:t+q}$ (i.e., the next q values) given the past p observations of the time series $x_{past} = x_{t-p+1:t}$ at time t (see [36]).

The exact choices of the loss functions of the discriminator and the generator will characterize the four different types of GAN that we are considering below.

3.1. Recurrent Conditional GAN (RCGAN)

A Recurrent Conditional GAN (RCGAN) follows the standard architecture of a regular GAN, but in this case, both generator and discriminator are replaced by recurrent neural networks (RNN). RCGAN can generate sequences of realistic data subject to some conditional inputs.

Let us denote the vector comprising T outputs from an RNN receiving a sequence of T vectors $\{x_t\}_{t=1}^T (\mathbf{x}_t \in \mathbb{R}^d)$ by $RNN(X)$ and the average cross-entropy between sequences $\mathbf{a}$ and $\mathbf{b}$ by $CE(\mathbf{a},\mathbf{b})$, respectively. Then, the discriminator and generator loss of $\mathbf{X}_n, \mathbf{y}_n$, where $X_n \in \mathbb{R}^{T\times d}$ and $\mathbf{y}_n \in \{0,1\}^T$, are given as in [9] by

$$D_{loss}(\mathbf{X}_n,\mathbf{y}_n) = -CE(RNN_D(\mathbf{X}_n),\mathbf{y}_n),$$
$$G_{loss}(Z_n) = D_{loss}(RNN_G(Z_n),\mathbf{1}) = -CE(RNN_D(RNN_G(Z_n)),\mathbf{1}),$$

where $\mathbf{y}_n$ is either a vector with all entries equal to one for empirical sequences or all entries equal to zero for fake/artificial sequences, and Z_n is a sequence of T points $\{z_t\}_{t=1}^T$ sampled independently from the latent/noise space $\mathbf{Z} = \mathbb{R}^m$. Hence, we have $Z_n \in \mathbb{R}^{T\times m}$. $\mathbf{1} = (1,\dots,1)$ indicates the decision that the discriminator accepts a given sequence as being generated from the distribution of the underlying data. The discriminator uses both empirical and fake sequences in each training step.

3.2. Time-series GAN (TimeGAN)

[10] comes on to the stage with the idea of TimeGAN. This framework considers a dataset, where each instance consists of static and temporal features. Static features remain the same without changing over time (e.g., gender), while temporal features are updated over time (e.g., electricity consumption).

Let us denote the vector of static and temporal features by $\mathcal{S}$ and $\mathcal{X}$, respectively. Also, let $\mathbf{S} \in \mathcal{S}$ and $\mathbf{X} \in \mathcal{X}$ be random vectors instantiated with specific values denoted as s and x. Now, consider tuples $\mathbf{S}, \mathbf{X}_{1:T}$ having a joint distribution p. In this setting, the length T of each sequence is also a random variable. In the training data, let individual samples be indexed by $n \in \{1,\dots,N\}$, so the training dataset is denoted by $\mathcal{D} = \{(s_n, x_{n,1:T_n})\}_{n=1}^N$.

The goal is to use $\mathcal{D}$ to find the best-approximating density $\hat{p}(\mathbf{S}, X_{1:T})$ to empirical data density $p(\mathbf{S}, X_{1:T})$. However, such a task may be cumbersome in the standard GAN framework. [10] offers using an autoregressive decomposition of $p((S), \mathbf{X}_{1:T}) = p(\mathbf{S})\Pi p(\mathbf{X}_t|\mathbf{S}, \mathbf{X}_{1:t-1})$ to focus specifically on the conditionals.

Unlike the classical GANs, TimeGAN contains four neural-network components: two autoencoding components (embedding and recovery functions) and two adversarial components (generator and discriminator). The fundamental insight is the autoencoding components are trained jointly with adversarial components. Hence, TimeGAN simultaneously learns to encode features, generate replicas, and iterate across time. The embedding network provides the latent space, the adversarial network performs within this space, and latent dynamics of both empirical and generated/artificial data are synchronized through a supervised loss.

3.3. Conditional Wasserstein GAN (CWGAN) and Recurrent Condition Wasserstein GAN (RCWGAN)

The concept of Wasserstein GAN (WGAN) is first introduced in [3], and it aims to address the mode collapse and vanishing gradient challenges in the standard GANs by optimizing the Wasserstein-1.

Calculating the Wasserstein-1 distance is intractable, but we can approximate it by changing the optimization objective to

$$\min_G \max_D \mathbb{E}_{x\sim p_{data}}\Big[D(x)\Big] - \mathbb{E}_{z\sim p_z}\Big[D(G(z))\Big]$$

as long as D is a k-Lipschitz function. This constraint can be satisfied by clipping the weight D so that it lies in a compact space $[-c,c]$, e.g., $c = 0.01$. However, weight-clipping constrains the discriminator capacity, pushes weights to the two extreme values of the permitted range $[-c,c]$, and can lead to exploding or vanishing gradients.

WGAN with gradient penalty (WGAN-GP) aims to remedy these problems by replacing the weight-clipping with a gradient penalty [4]. It makes use of the fact that a differentiable function is 1-Lipschitz if and only if it has gradients with the norm at most 1, everywhere. Consequently, we can enforce the Lipschitz constraint softly by penalizing

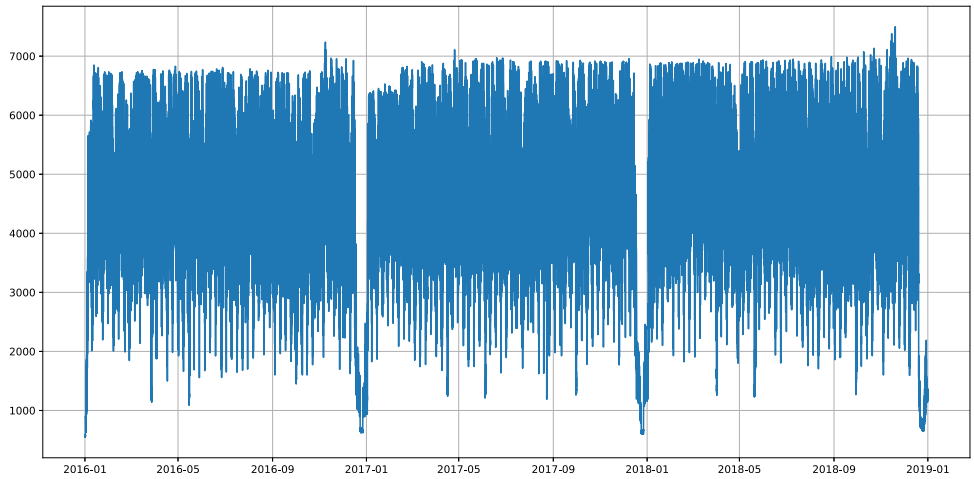


Fig. 1. Three years of 15 min electricity consumption data of an individual costumer.

the discriminator with a gradient penalty. Then, the objective of the GAN becomes

$$\min_G \max_D \mathbb{E}_{x\sim p_{data}}\Big[D(x)\Big] - \mathbb{E}_{z\sim p_z}\Big[D(G(z))\Big] - \lambda \mathbb{E}_{\hat{x}\sim p_{\hat{x}}}\Big[(\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1)^2\Big],$$

where λ is the penalty coefficient.

CWGAN extends WGAN-GP by adding additional information to the model and then, the optimization problem evolves to

$$\min_G \max_D \mathbb{E}_{x\sim p_{data}}\Big[D(x|y)\Big] - \mathbb{E}_{z\sim p_z}\Big[D(G(z|y))\Big] - \lambda \mathbb{E}_{\hat{x}\sim p_{\hat{x}}}\Big[(\|\nabla_{\hat{x}} D(\hat{x}|y)\|_2 - 1)^2\Big],$$

where y corresponds to the vector of additional information.

Recurrent Conditional Wasserstein GAN (RCWGAN) follows the standard architecture of the CWGAN. However, the generator and discriminator are replaced by Recurrent Neural Networks (RNNs).

4. Data and empirical analysis

4.1. Data analysis and stylized facts

We use individual electricity consumption data provided by a German electricity provider in our empirical data analysis. Our empirical data consists of three years fifteen minutes electricity consumption of a customer. We visualize the electricity consumption amount of the customer in Fig. 1. The electricity consumption is decreasing significantly in the end of the years. Such a significant decrease is most probably due to Christmas. The variability at the lower part is higher than at the upper boundary of the data as a result of weekends, holidays, and maintenance periods of the consumer. Fig. 1 also reveals that the electricity consumption is consistent with previous years in a classical calendar year.

Fig. 2 is a three-dimensional illustration of electricity consumption data: Illustrates the daily, weekly, monthly, and yearly electricity consumption behavior of the consumer. It separates years and graphs the empirical data within the corresponding years independently. More specifically, from top to down, it illustrates 2016, 2017, and 2018, respectively. It visualizes each month, week, and day of the week independently. Months are separated by a line and starting from January each month printed on x-axes to follow the monthly electricity consumption. The weekdays are printed on the right axis of each figure. Hence, we can observe the yearly, monthly, weekly, and daily electricity consumption. In Fig. 2, each square represents a day, and colors represent the corresponding average electricity consumption to that day. As the color gets darker, the electricity consumption decreases, and the color gets bright as the electricity consumption increases. Consequently, we can identify holidays from the dark squares. For instance, the dark squares in December represent Christmas, or the dark squares in May represent the national holidays in Germany. Additionally, we observe the electricity consumption is decreasing during weekends. Furthermore, electricity consumption increased slightly in 2017 and 2018, also observable in Fig. 3.

Table 1
Descriptive statistics of the data.

	Classical calendar	Only business days	Preprocessed data
Count	105 216.00	75 072.00	11 905.00
Mean	4579.89	4855.05	−0.01
Std	1218.55	1125.32	0.73
Min	546.58	546.58	−3.24
25/%	3988.20	4335.31	−0.51
50/%	4725.66	4967.84	0.00
75/%	5399.06	5582.03	0.49
Max	7497.90	7497.90	1.60

We summarize the distributional properties of the empirical data in Fig. 3. The top graph illustrates distributions of electricity consumption in each year. The graph in the middle introduces distributions of electricity consumption on each day of the week. The bottom graph shows distributions of electricity consumption in each month. As we mentioned earlier, electricity consumption increases slightly from 2016 to 2018. On the other hand, electricity consumption is quite regular on business days while it decreases during weekends. Additionally, the electricity consumption is standard and almost equal in each month except December. In December, electricity consumption decreases due to Christmas. More importantly, the figure shows electricity consumption is close to the normal distribution, but unfortunately, it includes outliers, and the outliers are generally at the bottom. It is an expected result since the electricity consumption decreases during the night, on weekends and holidays.

We introduce the descriptive statistics of the empirical data in Table 1. We introduce the preliminary statistics of the original data and electricity consumption data only for business days in the second and third columns, respectively. In the empirical analysis, we prefer to work on business days since the electricity consumption behavior is quite regular during holidays. Since we exclude holidays from the data, the number of observations decreased from 105 216 to 75 072. Consequently, the standard deviation of the data decreased from 1218.55 to 1125.32. On the other hand, the quantiles increased however, minimum and maximum values of the data are both preserved. Probably, the minimum is not changing because the customer is having maintenance, which is during the business day.

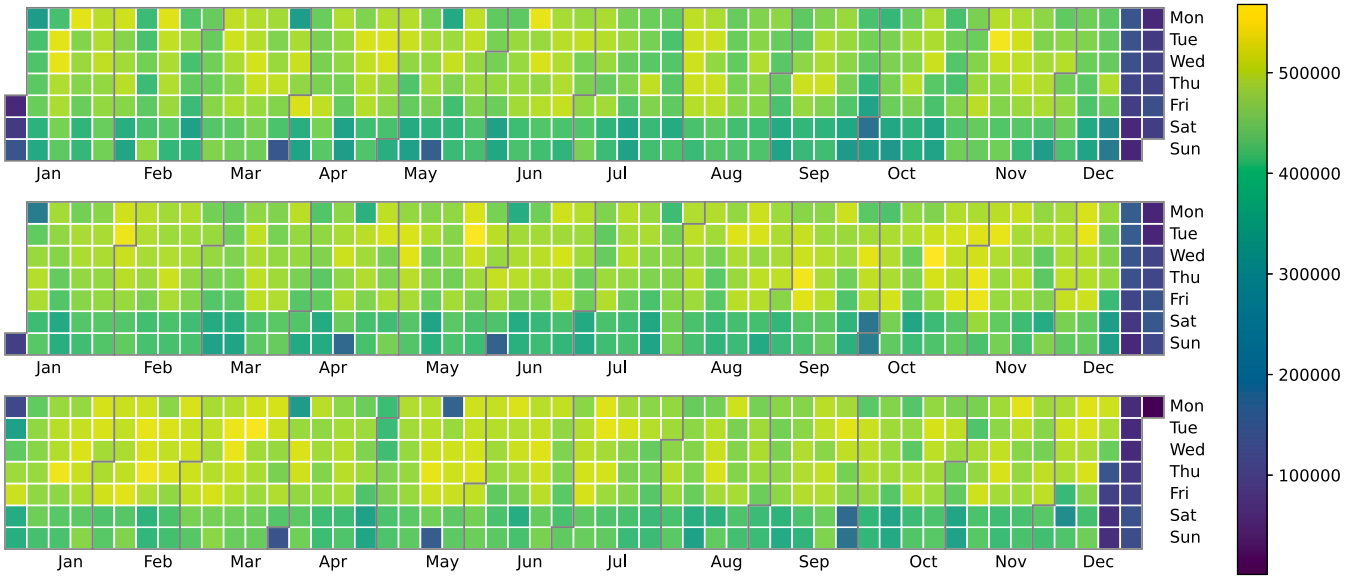


Fig. 2. Three years of daily electricity consumption behavior of the consumer. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

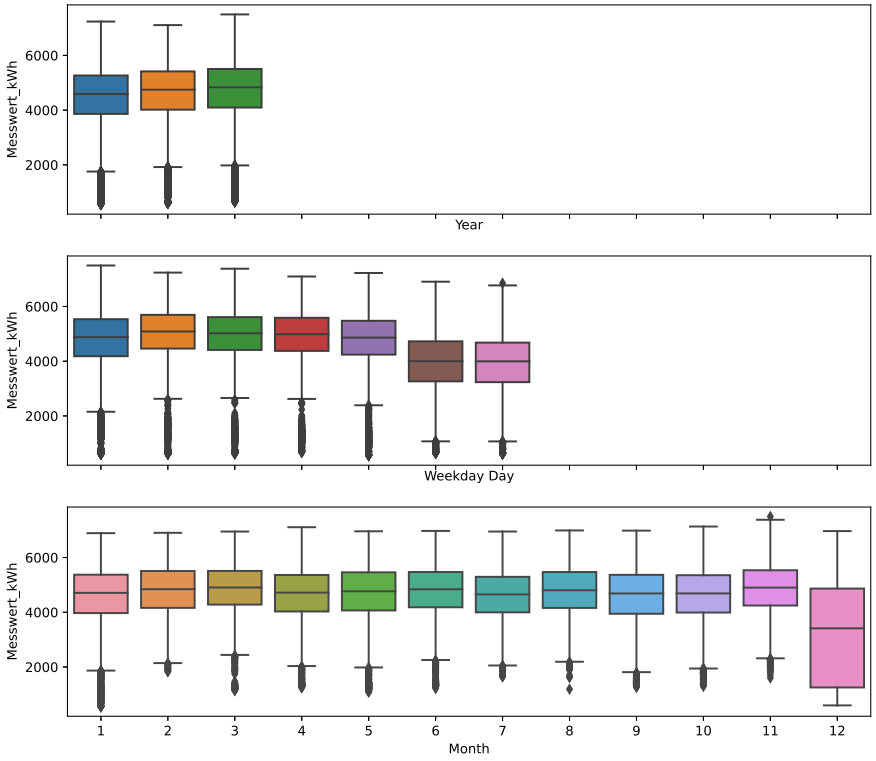


Fig. 3. The electricity consumption distributional properties.

4.2. Empirical analysis

As we observe from Section 4.1, the electricity consumption is quite regular in classical calendar years. Therefore, we sub-sampled the data to avoid the high computational cost of GANs training process. In the empirical analysis, we worked on the sub-sample given within the period '2016-01-08 00:00:00' - '2016-06-30 00:00:00' having a length of 11 905. Notice that although we sub-sampled the data, the sub-sample still has enough data points to train GANs.

Properly representing empirical datasets is crucial for training neural networks. Therefore, before starting our empirical analysis, we exclude holidays from the electricity consumption data since the consumption is stable on the base load during holidays and then scale the data by using the robust scaling algorithm (see Fig. 4). By excluding holidays, we remove outliers related to weekends. After preprocessing, the empirical data becomes more regular. We summarize the descriptive statistics of the preprocessed data in Table 1 to compare with the original data.

The robust scaler algorithms scale the data that are robust to data outliers. This scaling method follows a similar method to the Min–Max scaler but uses the interquartile range (rather than the min–max used in Min–Max Scaler). The median and scales of the data are removed with

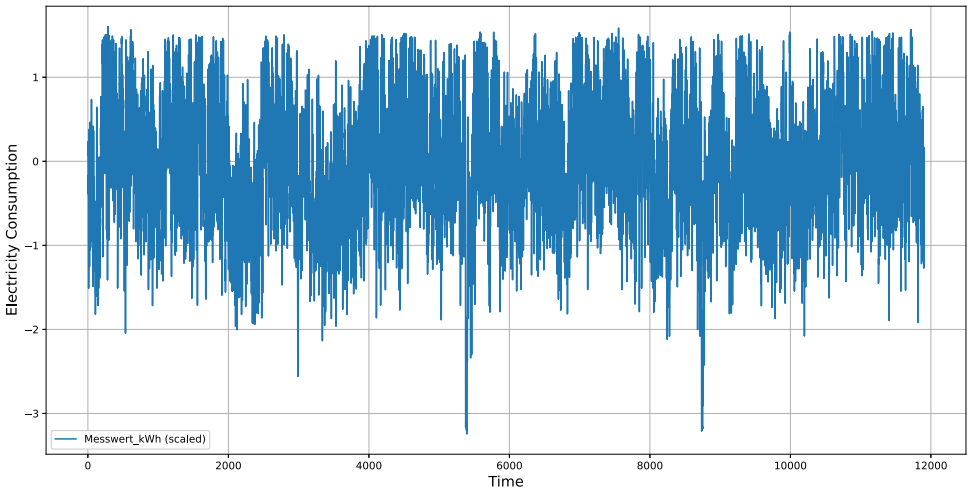


Fig. 4. Sub sample (January 8 to June 30 2016) of transformed 15 min consumption data used in the empirical analysis.

this scaling algorithm according to the quantile range. The robust scaler algorithm uses the interquartile range. So, the data becomes robust to outliers. The mathematical formula of the robust scaler is

$$x_i = \frac{x_i - Q_1(x)}{Q_3(x) - Q_1(x)},$$

where Q_1 and Q_3 denote first and third quantiles, respectively.

Now, we can present the results of the selected GANs that we use in our experiments. In the numerical analysis, we have specified both time series parameters p and q as 3 to find the conditional distribution. Hence, the discriminators use as inputs the conditioning time series $X_{t-p+1:t}$ to generate the part of the time series $X_{t+1:t+q}$, i.e., we use a rolling window size $p + q = 6$. We optimize algorithms of the GANs for a total of 1000 generator weight updates. Further, we utilize the Adam optimizer [37] with parameters $\beta_1 = 0$ and $\beta_2 = 0.9$ in the optimization procedure of the neural network weights and set the learning rates to 0.001. In the RCGAN and TimeGAN cases, we apply two time-scale updates (TTUR) [38] and set the learning rate to 0.003. Additionally, we update discriminator weights two times per generator weight update to improve the GANs stability. The number of epochs used in all GANs is 1.000, with a batch size of 200 for all GANs. In our numerical implementations, we used Pytorch [39] to train the GANs. Pytorch is a model-level library bases on Python that provides high-level building blocks for developing deep learning models. It is an open-source symbolic tensor manipulation framework and used commonly as an alternative to Tensorflow.

Table 2 illustrates the key statistical properties of the historical and synthetic datasets that we have generated. Note that we compare the key statistics of the transformed data and the generated datasets only for a single path each. Thus, the comparison can vary for different generated datasets (see for instance the illustrations in Fig. 5). The table reveals that the synthetic datasets have relatively similar means and standard deviations as the historical dataset. Even though we have variations among the statistical properties of historical and synthetic datasets, the variations are relatively small. The table, reveals that TimeGAN has the lowest standard deviation while CWGAN has the highest standard deviation. Additionally, the table reveals that the selected GANs are struggling to mimic the extreme data values which is also observable from Fig. 7. Here, CWGAN has the largest maximum value while RCGAN has the lowest minimum value. We analyze the synthetic datasets in more detail below.

As we already mentioned in Section 1, GANs have been originally developed for image processing and computer vision. In image processing, we can distinguish between real and fake images easily. On the other hand, in time series applications, it is not clear to determine whether the outputs of the GANs are realistic or non-realistic. In time series applications, we only have paths generated by GANs which can be compared with one existing real time series. Here, we can compare marginal distributions of 15 min consumption data as in Table 2 or in Fig. 7 or compare the autocorrelation structures of the time series. Therefore, we illustrate simulations to observe the behavior of GANs. To this end, first, we generate 500 paths for each GAN and use these paths to compare the distributional behaviors of the output of the GANs and the empirical data.

Table 2
Key statistics for historical and generated transformed 15 min electricity consumption data.

	Historical	RCGAN	TimeGAN	CWGAN	RCWGAN
Mean	−0.0130	0.0359	0.0321	0.1444	−0.0570
Std	0.7294	0.7071	0.6093	0.7355	0.7135
Max	1.6031	2.3072	2.0599	2.5754	1.9045
Min	−3.2427	−2.4751	−1.3922	−1.9374	−1.7158

Fig. 5 illustrates simulations of four GANs types: RCGAN, TimeGAN, CWGAN, and RCWGAN. For each GAN type, the generated 500 paths are graphed with gray lines. We also graph one of the simulated paths (orange line) besides the empirical data (blue line) to observe the synthetic data's similarity in behavior compared to the original one. The figure shows the generated data paths are bounded from above and below, and none is exploding. The figure also reveals GANs are generating some synthetic datasets having larger maximum and minimum data points than the empirical data. This is particularly interesting when the GAN simulations are performed for risk management purposes such as for checking if the provider can cope with extreme consumption demand. Further, we can claim that the maximum and minimum electricity consumption points are close to CWGAN simulations compared to the other three GANs. Also, the behavior of the highlighted path almost has the same behavior as the empirical dataset.

The daily electricity consumption behavior is critical since the electricity consumption changes according to the state of the day (morning, noon, evening, and night). However, it is difficult to follow the generated electricity consumption paths behavior daily from Fig. 5. Therefore, we graph trajectories of a random business day (10 March 2016 (Tuesday)) in Fig. 6 to illustrate the daily behaviors of the synthetic datasets. In this figure, blue lines represent the generated datasets, and orange dashed-lines represent the empirical datasets. For these figures, we use the non-transformed synthetic and empirical datasets. Thus, they illustrate real values of electricity consumption. They reveal that all four GANs types are successively mitigating the daily electricity consumption behavior. Also, notice the synthetic datasets patterns are

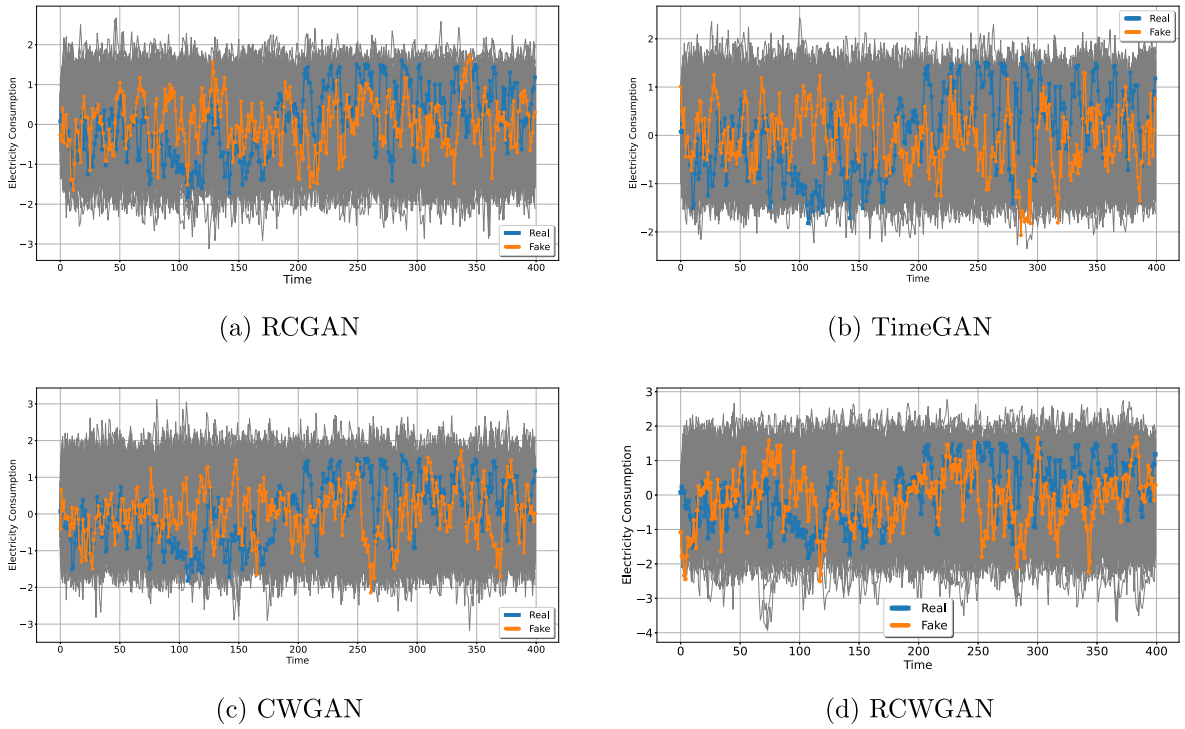


Fig. 5. Synthetic individual electricity consumption trajectories data (orange and gray) as generated with GANs and observed electricity consumption of the individual customer (blue). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

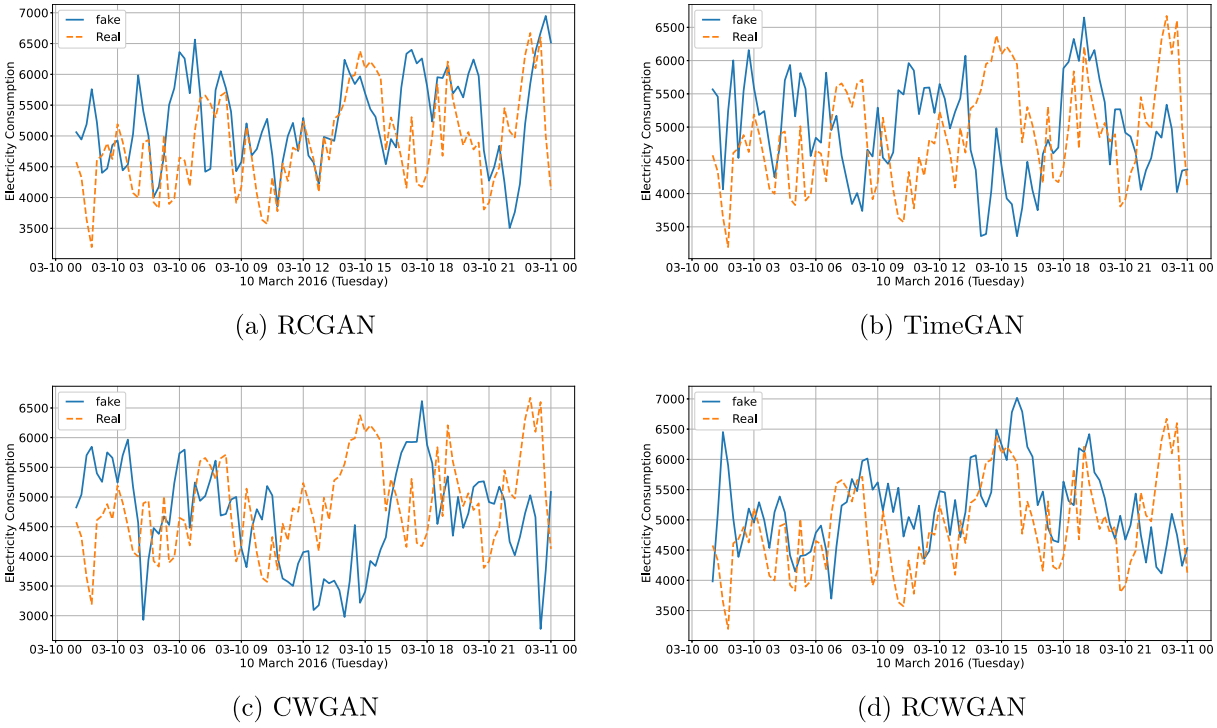


Fig. 6. Synthetic individual electricity consumption trajectories (blue) on 10 March 2016 (Tuesday) and observed data (orange).

similar but not identical to the empirical dataset. Hence, synthetic datasets look realistic without being a mere repetition of the training data patterns. At this point, we should remark the synthetic datasets that we illustrate in Fig. 6 are just a single result of corresponding GANs outputs. We can end up with more similar or less similar generated datasets if we recompile GANs training.

Fig. 7 illustrates the distributional properties of the GANs. In particular, it shows that the distributions of empirical and synthetic datasets

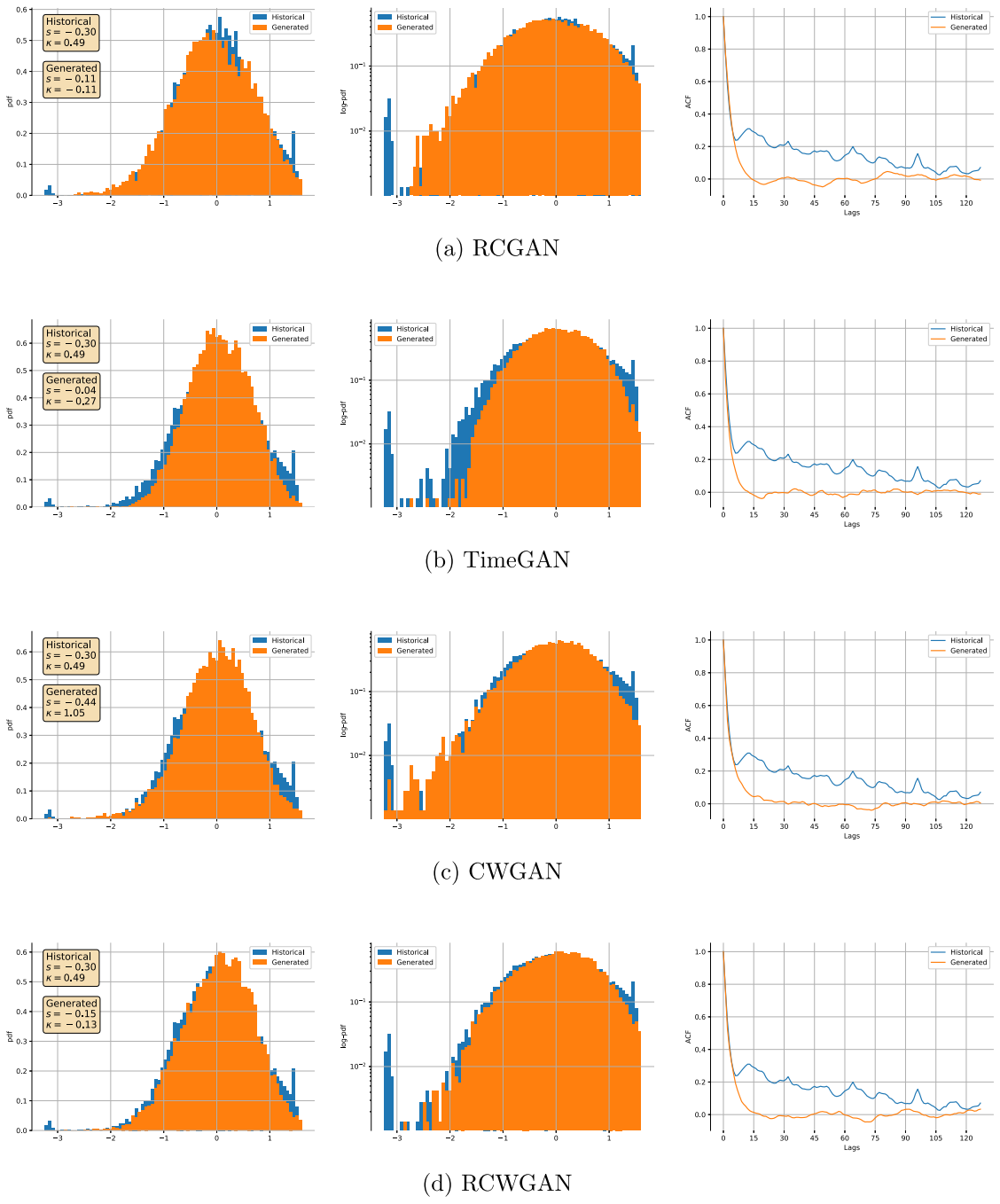
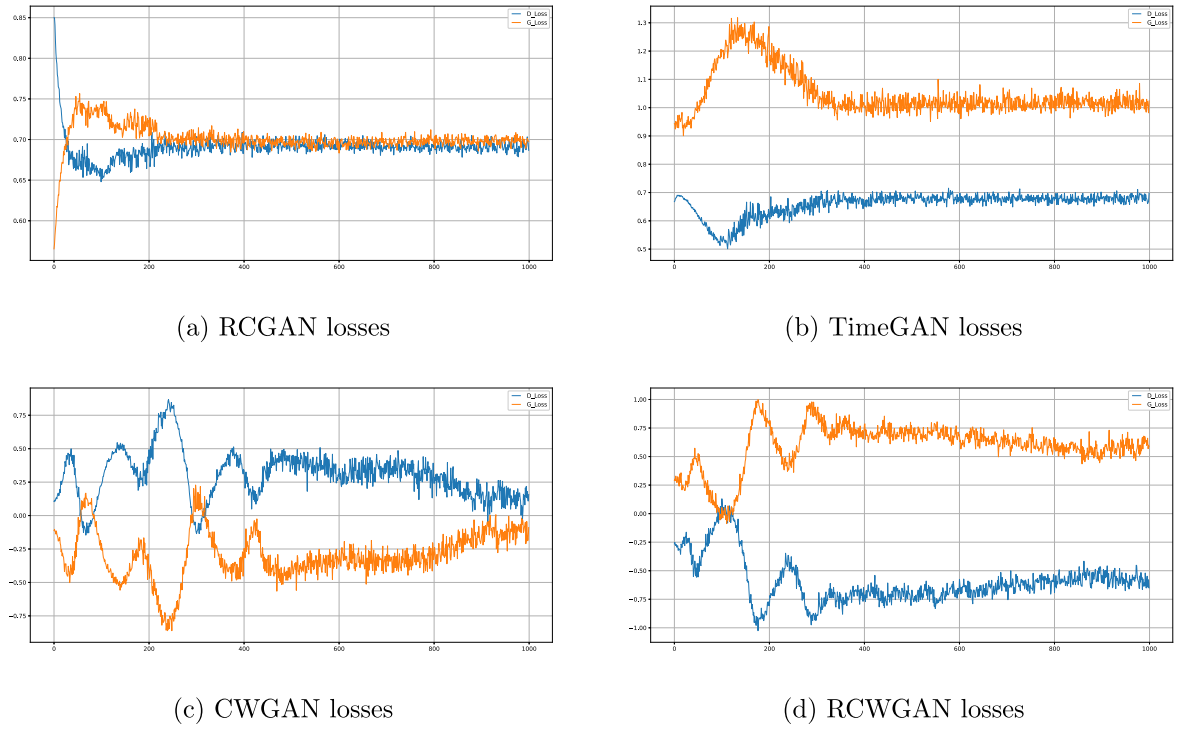


Fig. 7. Comparison of the marginal distributions of generated and original transformed 15-minutes electricity consumption data on the linear scale (1st column), log-plot (2nd column) and the auto-correlation fit with the real data. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

match closely. All GANs have approximately the same mean and skewness but, the kurtosis are different. The figure shows histograms of empirical and artificial datasets, along with their skewness and kurtosis statistics to measure symmetry and tail behaviors, and the auto-correlation changes. The histograms show that the distribution of 15 min electricity consumption from the synthetic data (orange) is almost identical to that of the empirical data (blue). However, the skewness and kurtosis statistics of the synthetic and empirical datasets are different. The skewness statistics have the same sign (negative) and are close to the skewness of the real sample, while the kurtosis statistics have different signs for RCGAN, TimeGAN, and RCWGAN. On the other hand, the kurtosis statistic of CWGAN is the only one having the same sign as the empirical data kurtosis statistics.

In Fig. 7, the log plots highlight some discrepancies in tails. They show that CWGAN is better than the remaining three GANs in generating low electricity consumption while TimeGAN is the worst in generating low electricity consumption. On the other hand, RCGAN is the best to generate high electricity consumption. The auto-correlations of all GANs and empirical datasets are relatively close. In summary, even though there are some differences in the skewness and kurtosis statistics, the distributional behaviors of the synthetic datasets are almost identical to the empirical data distribution. Comparisons with



(a) RCGAN losses

(b) TimeGAN losses

(c) CWGAN losses

(d) RCWGAN losses

Fig. A.8. Evolution of the generator and discriminator training loss functions of the GANs.

respect to further metrics that support our similarity claims are given in Appendix A.

5. Conclusion

Accurate simulation and predictions of individual electricity consumption have become vital for power grid regulations. Individual electricity consumption behavior has been widely studied via regression-based and artificial intelligence models. In the last decade, artificial intelligence models have dominated the research field since they do not require detailed building and environmental parameters. There are two main methods in artificial intelligence modeling: deep learning methods and traditional machine learning methods, which rely highly on historical record data. Undoubtedly, time-dependent record data is a crucial source of information for electricity markets.

The training data must be representative and contain sufficient variety to have reasonable artificial intelligence model performances. However, composing such sufficiently representative and wide-ranging record data is difficult, costly, and time-consuming. If we do not have sufficient data or sampling data deviate from the observed data distribution, the prediction accuracy of models is affected significantly.

This paper suggests synthetic data generation with GANs for individual electricity consumption data. To this end, we used RCGAN, TimeGAN, CWGAN, and RCWGAN in a real-world application. These four GANs achieve state-of-the-art results on synthetic electricity consumption data generation. The results reveal we can use GANs to avoid data privacy concerns and increase load modeling efficiency for grid modeling.

Fig. 7 reveals CWGAN is relatively best while TimeGAN is the worst GAN. This result is also consistent with the interpretation of Fig. A.8 which illustrates that the discriminator and generator loss functions of CWGAN both converge to zero while the loss functions of TimeGAN converge to the highest values. This is an indicator that CWGAN is the best performing GAN while TimeGAN is the worst.

We are aware that the efficiency of the GANs may be increased by increasing the information provided to the training of GANs. For instance, one may use the time steps as conditions on the generator, which increases the classification accuracy of the discriminator when the artificial time series is used to augment the training data of the training data. However, we stick to the one-dimensional times series and explore the selected GANs due to the unity of the paper. Hence, as future work, we are planning to extend the study by using the benefit of conditioning the generator with better information than just random noise space input as in Time-Conditioned GAN (T-CGAN) and Time Series GAN (TSGAN)introduced by [24,40], respectively.

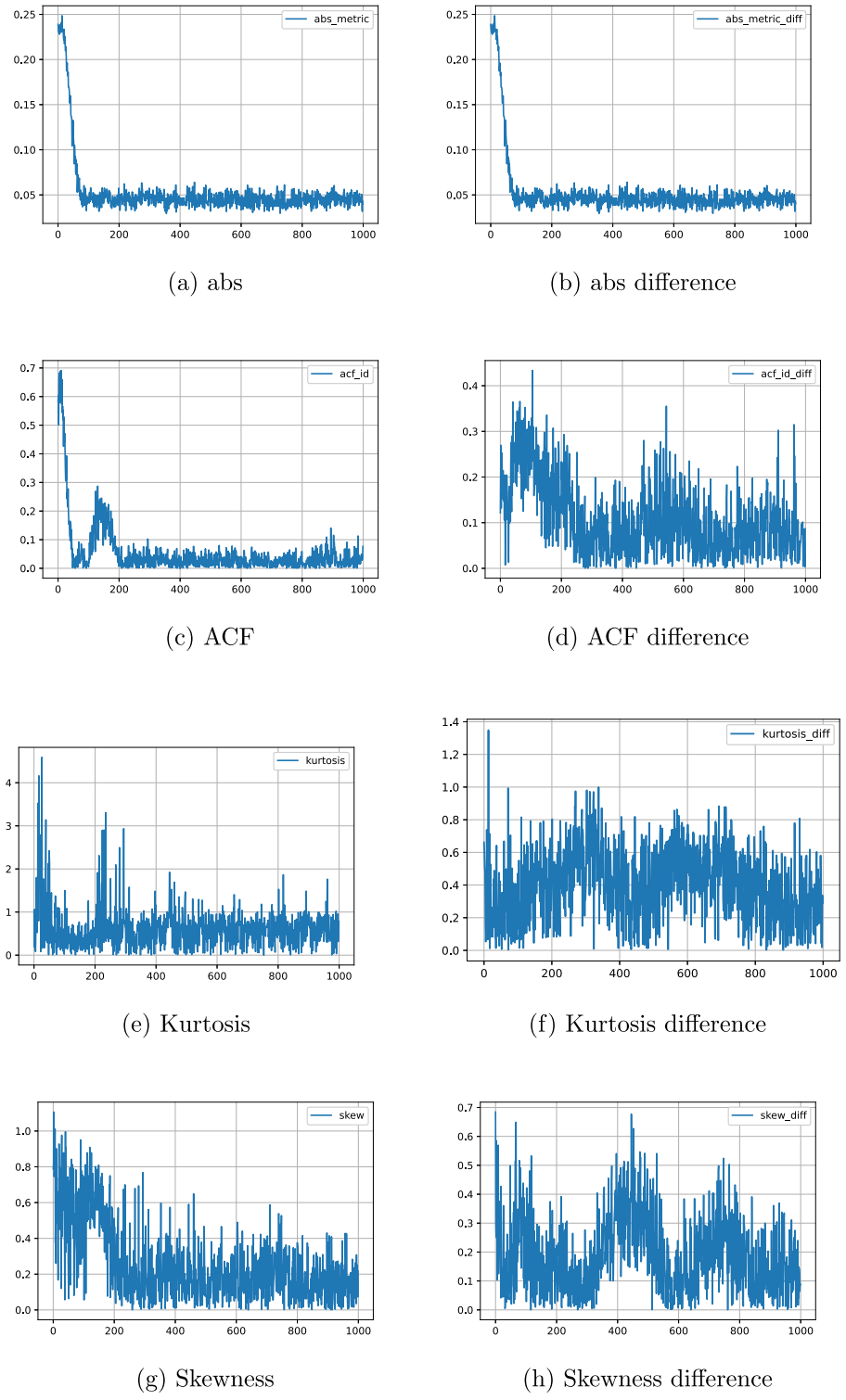
The already good performance of the GANs presented in this work show that they can be used for data augmentation for electricity demand. Thus, their synthetically generated data can be combined with the existing empirical demand data when risk management issues such as tests for extreme demand, the optimal timing of maintenance of wind wheels, the check for energy efficiency of buildings or the profitability of demand- or time-dependent price strategies should be examined.

Declaration of competing interest

One or more of the authors of this paper have disclosed potential or pertinent conflicts of interest, which may include receipt of payment, either direct or indirect, institutional support, or association with an entity in the biomedical field which may be perceived to have potential conflict of interest with this work. For full disclosure statements refer to https://doi.org/10.1016/j.egyai.2022.100161.Bilgi Yilmaz reports financial support was provided by German Ministry of Education and Research (BMBF). Bilgi Yilmaz reports a relationship with Federal Ministry of Education and Research Berlin Office that includes: funding grants. Co-author currently employed by Tu Kaiserslautern

Acknowledgments

The research of Bilgi Yilmaz has been funded by the German Ministry of Education and Research (BMBF) within the project *Analytisch-generative Netzwerke zur Systemidentifikation (AGENS)* (grant no: 05M20UKA).



(a) abs (b) abs difference

(c) ACF (d) ACF difference

(e) Kurtosis (f) Kurtosis difference

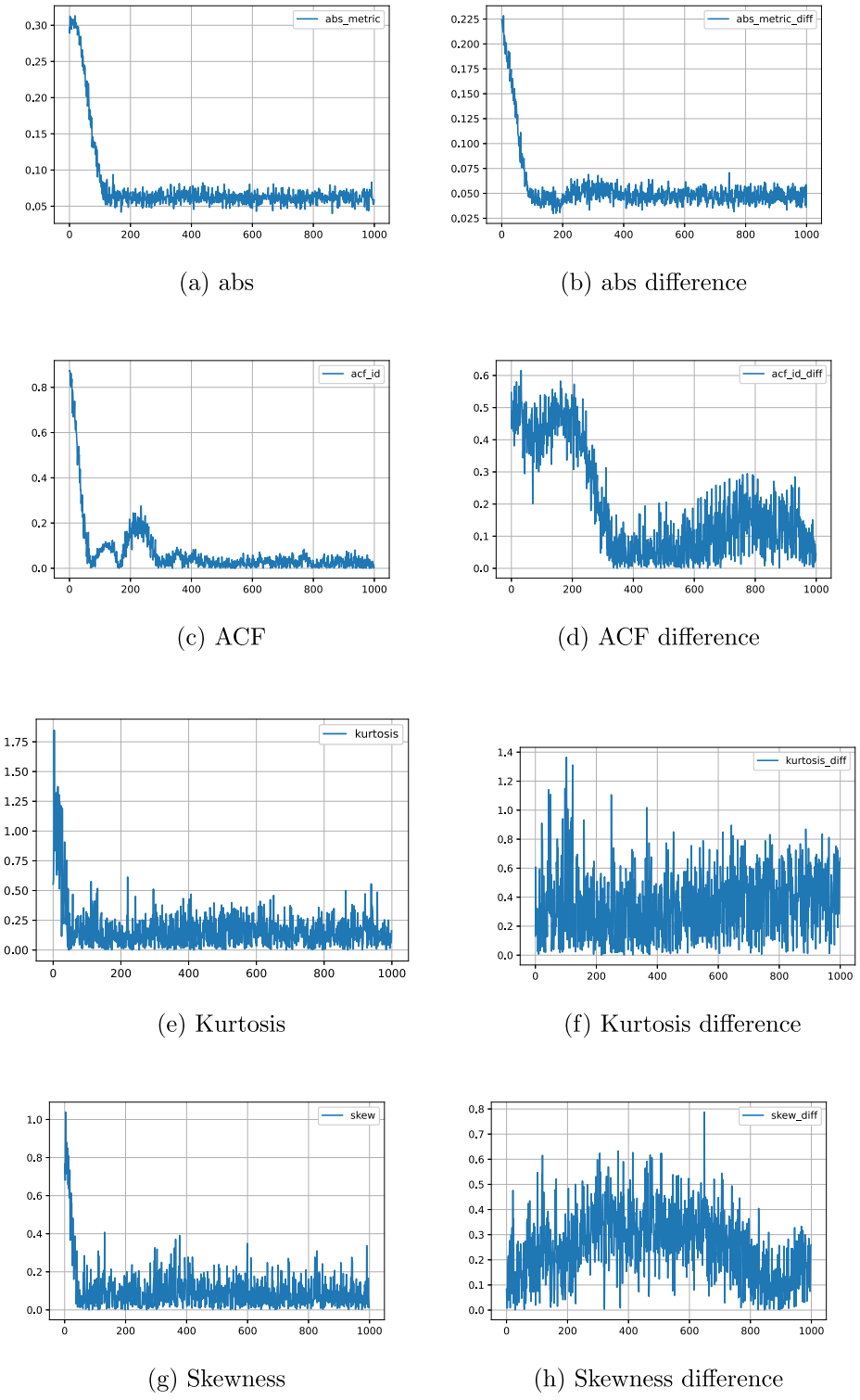
(g) Skewness (h) Skewness difference

Fig. A.9. Accuracy measures of RCGAN application.

Appendix A. Evaluation metrics for the GANs

For completeness, we here summarize the performance of our GANs with respect to further metrics. These metrics illustrate quantitatively how well GANs are generating realistic synthetic data. First, we start illustrating the generator and discriminator losses of the models. Then, we illustrate metrics on marginal distribution, the absolute difference of lag-1 auto-correlation, auto-correlation, the difference between auto-correlation, kurtosis and its difference, skewness, and its difference.

To compare the quality of generated scenarios, we plot the training curves of all types of GANs in Fig. A.8 in each iteration during the model training processes. The losses converge without showing any anomaly regarding the gradient for all GANs that we use. The convergence of the loss function of the discriminator to zero indicates the discriminator almost fails to distinguish between real and fake samples, which suggests the generated residual scenarios have nearly the same characteristics as empirical residuals.



(a) abs (b) abs difference

(c) ACF (d) ACF difference

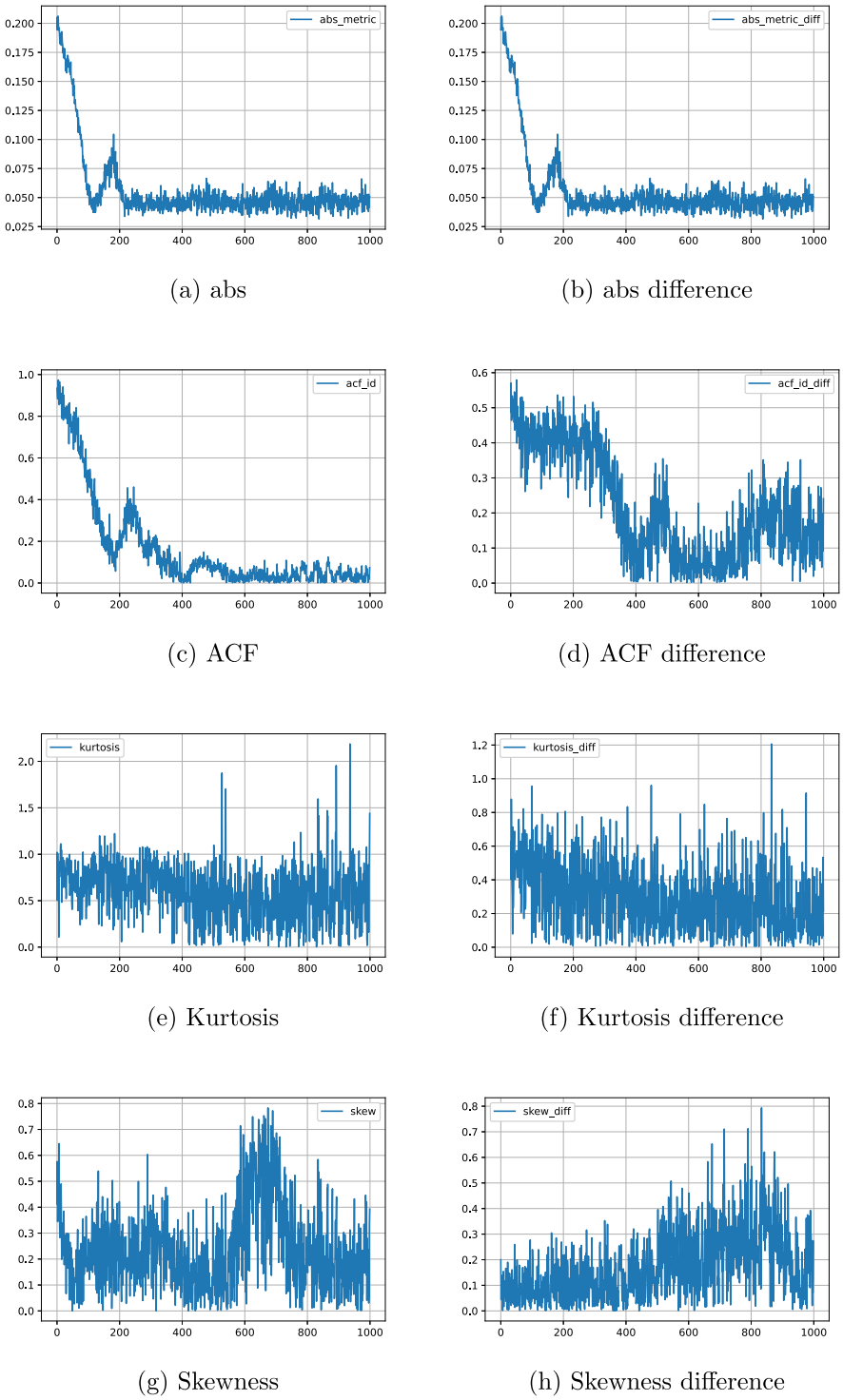
(e) Kurtosis (f) Kurtosis difference

(g) Skewness (h) Skewness difference

Fig. A.10. Accuracy measures of TimeGAN application.

Figs. A.9, A.10, A.11, and A.12 show the remaining metrics. For any of the GAN types, the discriminator losses of real and generated samples exhibit a large difference in the early stage of training since the generator has not learned the data distribution of the real data at the early stage. Then, the distances are converging for all GANs. When GANs are trained to converge, the accuracy measures converge to zero. Hence, our four GAN types are trained efficiently. These metrics show that all GANs that we used in this paper are good at generating synthetic data since all metrics converge nicely towards zero for all GANs.

Fig. A.8 shows the training progresses of the GANs as the epochs proceed. The figure shows initially a big difference between empirical and generated data distributions. In the beginning, the discriminator easily distinguishes between real and synthetic data, assigning high values to empirical data and low values to the generated data. As training progresses, the generator improves and the discriminator struggles to



(a) abs (b) abs difference

(c) ACF (d) ACF difference

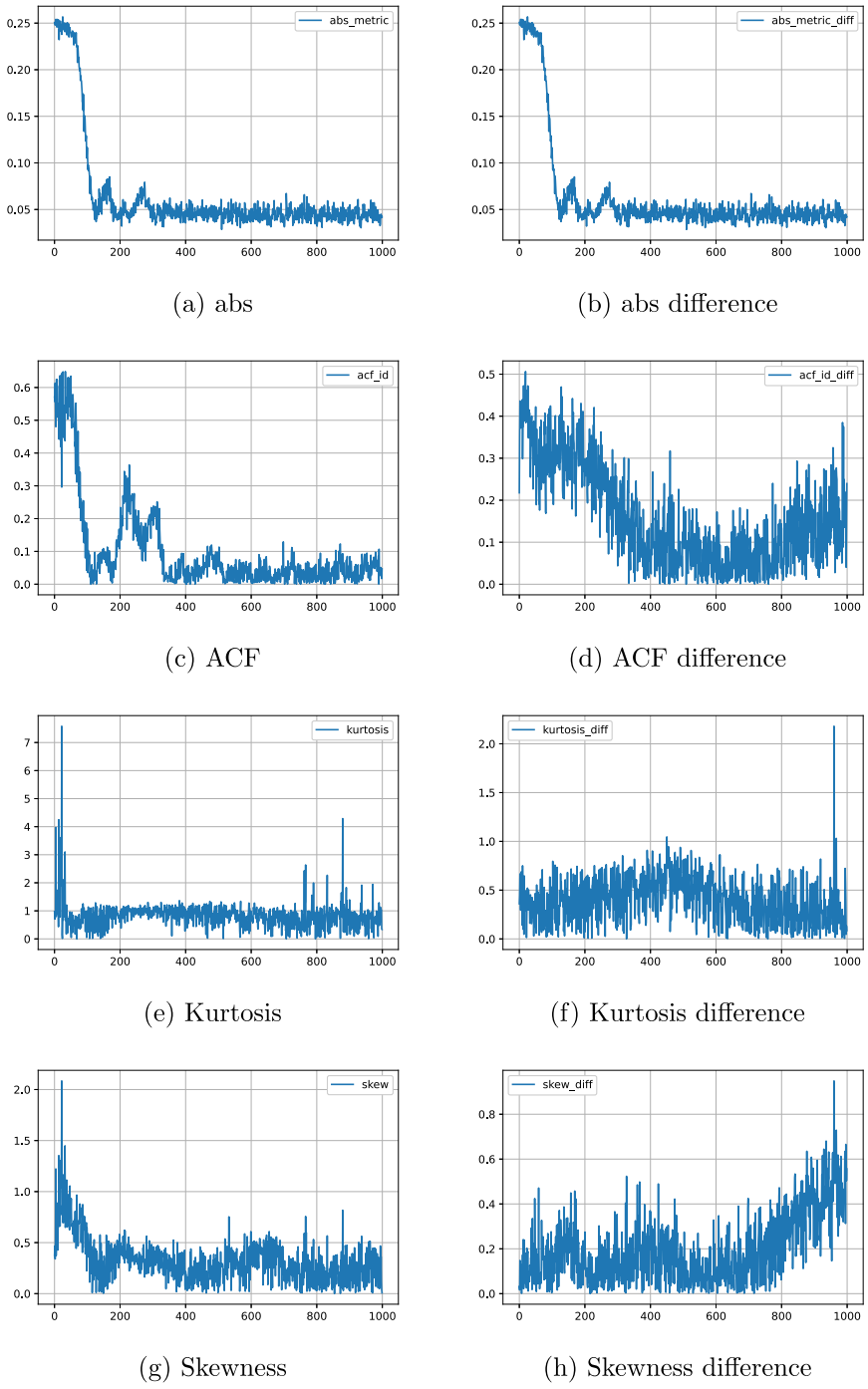
(e) Kurtosis (f) Kurtosis difference

(g) Skewness (h) Skewness difference

Fig. A.11. Accuracy measures of CWGAN application.

distinguish between the two data. For the remainder of the training, the loss gradually lowers making the generated data distribution closer to the real data distribution, i.e., the one underlying the empirical data, indicating that GANs approach Nash Equilibrium points. Especially, the loss functions converge for both RCGAN and CWGAN. By monitoring the training, we can observe the models remain stable and keep improving. Both the generator and discriminator are largely stabilizing after around 200 and 400 epochs for RCGAN and the remaining three GANs, respectively. At this stage, the discriminator assigns very similar values to datasets.

It shows that the losses of both generator and discriminator networks converge without any anomaly. They show a stable behavior for all GANs. The discriminator loss converges to zero indicating that the discriminator nearly fails to distinguish the real and generated samples. Such a convergence behavior suggests that the generated samples have almost the same characteristics as the real samples [31].



(a) abs (b) abs difference

(c) ACF (d) ACF difference

(e) Kurtosis (f) Kurtosis difference

(g) Skewness (h) Skewness difference

Fig. A.12. Accuracy measures of RCWGAN application.

References

[1] Moon J, Jung S, Park S, Hwang E. Conditional tabular GAN-based two-stage data generation scheme for short-term load forecasting. IEEE Access 2020;8:205327–39. http://dx.doi.org/10.1109/ACCESS.2020.3037063.

[2] Xu L. Synthesizing tabular data using conditional GAN (Master's thesis), Massachusetts Institute of Technology; 2020.

[3] Arjovsky M, Chintala S, Bottou L. Wasserstein generative adversarial networks. In: International conference on machine learning. PMLR; 2017, p. 214–23.

[4] Gulrajani I, Ahmed F, Arjovsky M, Dumoulin V, Courville A. Improved training of wasserstein GANs. 2017, arXiv:1704.00028.

[5] Yu L, Zhang W, Wang J, Yu Y. SeqGAN: Sequence generative adversarial nets with policy gradient. 2017, arXiv:1609.05473.

[6] Zhu J-Y, Park T, Isola P, Efros AA. Unpaired image-to-image translation using cycle-consistent adversarial networks. 2020, arXiv:1703.10593.

[7] Silva VLS, Heaney CE, Li Y, Pain CC. Data assimilation predictive GAN (DA-PredGAN): applied to determine the spread of COVID-19. 2021, arXiv:2105.07729.

[8] Bendaoud NMM, Farah N, Ahmed SB. Comparing generative adversarial networks architectures for electricity demand forecasting. Energy Build 2021;247:111–52. http://dx.doi.org/10.1016/j.enbuild.2021.111152.

[9] Esteban C, Hyland SL, Rätsch G. Real-valued (medical) time series generation with recurrent conditional GANs. 2017, arXiv:1706.02633.

[10] Yoon J, Jarrett D, van der Schaar M. Time-series generative adversarial networks. In: Wallach H, Larochelle H, Beygelzimer A, d'Alché Buc F, Fox E, Garnett R, editors. Advances in neural information processing systems, Vol. 32. Curran Associates, Inc.; 2019.

[11] Goodfellow IJ, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y. Generative adversarial networks. 2014, arXiv:1406.2661.

[12] Mirza M, Osindero S. Conditional generative adversarial nets. 2014, arXiv:1411.1784.

[13] Mao X, Li Q, Xie H, Lau RYK, Wang Z, Smolley SP. Least squares generative adversarial networks. 2017, arXiv:1611.04076.
[14] Wiese M, Knobloch R, Korn R, Kretschmer P. Quant GANs: deep generation of financial time series. Quant Finance 2020;20(9):1419–40. http://dx.doi.org/10.1080/14697688.2020.1730426.
[15] Li J, Wang X, Lin Y, Sinha A, Wellman MP. Generating realistic stock market order streams. 2020, arXiv:2006.04212.
[16] Donahue C, McAuley J, Puckette M. Adversarial audio synthesis. 2019, arXiv:1802.04208.
[17] Engel J, Agrawal KK, Chen S, Gulrajani I, Donahue C, Roberts A. GANSynth: Adversarial neural audio synthesis. 2019, arXiv:1902.08710.
[18] Koshiyama A, Firoozye N, Treleaven P. Generative adversarial networks for financial trading strategies fine-tuning and combination. 2019, arXiv:1901.01751.
[19] Fu R, Chen J, Zeng S, Zhuang Y, Sudjianto A. Time series simulation by conditional generative adversarial net. 2019, arXiv:1904.11419.
[20] Koochali A, Dengel A, Ahmed S. If you like it, GAN it. Probabilistic multivariate times series forecast with GAN. 2020, arXiv:2005.01181.
[21] Wiese M, Bai L, Wood B, Buehler H. Deep hedging: Learning to simulate equity option markets. 2019, arXiv:1911.01700.
[22] Samuel R, Nico BD, Moritz P, Joerg O. Wasserstein GAN: Deep Generation applied on Bitcoins financial time series. 2021, arXiv:2107.06008.
[23] Efimov D, Xu D, Kong L, Nefedov A, Anandakrishnan A. Using generative adversarial networks to synthesize artificial financial datasets. 2020, arXiv:2002.02271.
[24] Smith KE, Smith AO. Conditional GAN for timeseries generation. 2020, CoRR abs/2006.16477.
[25] Chen Y, Wang Y, Kirschen D, Zhang B. Model-free renewable scenario generation using generative adversarial networks. 2018, arXiv:1707.09676.
[26] Gu Y, Chen Q, Liu K, Xie L, Kang C. GAN-based model for residential load generation considering typical consumption patterns. In: 2019 IEEE power energy society innovative smart grid technologies conference (ISGT). 2019, p. 1–5. http://dx.doi.org/10.1109/ISGT.2019.8791575.
[27] Zhang G, Guo J. A novel ensemble method for residential electricity demand forecasting based on a novel sample simulation strategy. Energy 2020;207:118265. http://dx.doi.org/10.1016/j.energy.2020.118265.
[28] Tian C, Li C, Zhang G, Lv Y. Data driven parallel prediction of building energy consumption using generative adversarial nets. Energy Build 2019;186:230–43. http://dx.doi.org/10.1016/j.enbuild.2019.01.034.
[29] Fekri MN, Ghosh AM, Grolinger K. Generating energy data for machine learning with recurrent generative adversarial networks. Energies 2020;13(1). http://dx.doi.org/10.3390/en13010130.
[30] Zhou D, Ma S, Hao J, Han D, Huang D, Yan S, Li T. An electricity load forecasting model for integrated energy system based on BiGAN and transfer learning. Energy Rep 2020;6:3446–61. http://dx.doi.org/10.1016/j.egyr.2020.12.010.
[31] Wang Y, Hug G, Liu Z, Zhang N. Modeling load forecast uncertainty using generative adversarial networks. Electr Power Syst Res 2020;189:106732. http://dx.doi.org/10.1016/j.epsr.2020.106732.
[32] Yuan R, Wang B, Mao Z, Watada J. Multi-objective wind power scenario forecasting based on PG-GAN. Energy 2021;226:120379. http://dx.doi.org/10.1016/j.energy.2021.120379.
[33] Wang Z, Hong T. Generating realistic building electrical load profiles through the Generative Adversarial Network (GAN). Energy Build 2020;224:110299.
[34] Wang Y. A mathematical introduction to generative adversarial nets (GAN). 2020, CoRR abs/2009.00169.
[35] Yilmaz B. Understanding the mathematical background of generative adversarial neural networks (GANs). 2021, Available at SSRN 3981773.
[36] Ni H, Szpruch L, Wiese M, Liao S, Xiao B. Conditional sig-wasserstein GANs for time series generation. 2020, arXiv:2006.05421.
[37] Kingma DP, Ba J. Adam: A method for stochastic optimization. 2017, arXiv:1412.6980.
[38] 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. 2018, arXiv:1706.08500.
[39] Paszke A, Gross S, Massa F, Lerer A, Bradbury J, Chanan G, Killeen T, Lin Z, Gimelshein N, Antiga L, Desmaison A, Kopf A, Yang E, DeVito Z, Raison M, Tejani A, Chilamkurthy S, Steiner B, Fang L, Bai J, Chintala S. PyTorch: An imperative style, high-performance deep learning library. In: Wallach H, Larochelle H, Beygelzimer A, d'Alché Buc F, Fox E, Garnett R, editors. Advances in neural information processing systems 32. Curran Associates, Inc.; 2019, p. 8024–35, URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.
[40] Ramponi G, Protopapas P, Brambilla M, Janssen R. T-CGAN: Conditional generative adversarial network for data augmentation in noisy time series with irregular sampling. 2019, arXiv:1811.08295.