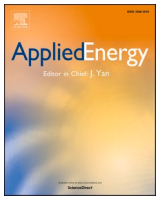
Applied Energy 393 (2025) 126107

Contents lists available at ScienceDirect

Applied Energy

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



Unsupervised and controllable synthesizing for imbalanced energy dataset based on AC-InfoGAN☆

Zhenghao Zhou [a,c], Yiyan Li [a,c,*], Runlong Liu [a,c], Xiaoyuan Xu [b,c], Zheng Yan [b,c]

[a] College of Smart Energy, Shanghai Jiao Tong University, Shanghai 200240, China
[b] The Key Laboratory of Control of Power Transmission and Conversion, Ministry of Education, Shanghai Jiao Tong University, Shanghai 200240, China
[c] Shanghai Non-Carbon Energy Conversion and Utilization Institute, Shanghai Jiao Tong University, Shanghai 200240, China

HIGHLIGHTS

- We propose an adaptive and contrastive InfoGAN model for unlabeled and unbalanced electrical time series dataset.
- We propose frequency-domain methods for creating positive samples and enhancing the neural network performances.
- We extract both discrete and continuous features with certain physical meanings.

ARTICLE INFO

Keywords:
Information maximizing generative adversarial nets
Unsupervised learning
Unlabeled and imbalanced dataset
Synthetic data generation
Feature extraction

ABSTRACT

Generating synthetic data has become a popular alternative solution to deal with the difficulties in accessing and sharing field measurement data in power systems. However, to make the generation results controllable, existing methods (e.g., Conditional Generative Adversarial Nets, cGAN) require labeled dataset to train the model, which is demanding in practice because many field measurement data lack descriptive labels. Meanwhile, real-world datasets are naturally imbalanced, causing bias in neural network training. In this paper, we introduce the Adaptive and Contrastive Information Maximizing Generative Adversarial Nets (AC-InfoGAN) to achieve controllable synthesizing for the unlabeled and imbalanced energy dataset. Features with physical meanings can be automatically extracted by maximizing the mutual information between the input latent code and the classifier output. Then the extracted features are used to control the generation results similar to a vanilla cGAN framework. We employ the Gumbel-Softmax distribution and frequency-based contrastive learning techniques to dynamically adapt to the imbalanced dataset to avoid the model training bias. Meanwhile, frequency-domain neural network modules are introduced to the AC-InfoGAN framework to enhance the model performances. Case study is based on the unlabeled and imbalanced energy datasets of power load and renewable energy output. Results demonstrate that AC-InfoGAN can extract both discrete and continuous features with certain physical meanings, as well as generating realistic synthetic energy data that satisfy given features

1. Introduction

Adequate and high-quality dataset is the precondition for data-driven studies in power systems. However, due to concerns of energy security and user privacy, field measurement data of power systems cannot be easily obtained and shared by the academic community, which is adverse to the development of data-driven technologies. In this case, generating and utilizing synthetic data becomes an alternative solution. Synthetic data is defined as the imitated data derived from the field measurements, which has similar characteristics with the field measurements but does not correspond to any real-world individuals or objects. As a result, synthetic dataset can be accessed and shared by the academic community without additional concerns.

In general, there are two main methods of generating synthetic data: simulation-based methods and data-driven methods. Simulation-based methods create synthetic data based on the simulation process of physical models [1], such as generating synthetic load profiles by load model simulation [2,3]. The advantages are that the results are

☆ This article is part of a Special issue entitled: 'Next-Gen AI in Power(Yanli Liu)' published in Applied Energy.

* Corresponding author at: College of Smart Energy, Shanghai Jiao Tong University, Shanghai 200240, China.
E-mail address: yiyan.li@sjtu.edu.cn (Y. Li).

https://doi.org/10.1016/j.apenergy.2025.126107
Received 28 October 2024; Received in revised form 11 April 2025; Accepted 10 May 2025
Available online 20 May 2025

explainable and controllable. However, the realisticness of the generated synthetic data relies heavily on the modeling accuracy and diversity, which is labor-intensive and inflexible. Furthermore, simulation-based methods often employ approximate simplified models, which can lead to deviations in the generated synthetic data. Data-driven methods generate synthetic data by learning the patterns from limited actual data, which is an end-to-end process and draws more attention due to its convenience and flexibility.

Data-driven methods can be further divided into three categories: supervised learning methods, unsupervised learning methods, and deep generative methods. The supervised learning methods learn the patterns from historical data, and then generate synthetic data by projecting into the future. [4] implemented Long Short-Term Memory (LSTM) to learn patterns from historical energy consumption data, and then synthesized more data by utilizing the model's forecasting ability. Supervised learning methods are suitable for synthesizing time-series data, but the results lack diversity due to the supervised-learning nature. Unsupervised learning methods are the mainstream in synthetic data generation, because they can better consider the uncertainty. [5] employed a bottom-up analysis based on behavior clustering and Markov chains to model residential building energy consumption. [6] proposed a generation framework utilizing D-vine copulas for transport network expansion planning. This framework includes steps such as hierarchical clustering, feature extraction, and D-vine copulas modeling.

As deep learning advances rapidly, deep generative methods, recognized as a specialized subset, have gained considerable attention in the realm of synthetic data generation, such as Generative Adversarial Networks (GAN) [7], Variational Autoencoders (VAE) [8], and diffusion models [9]. GAN can learn the implicit distribution of a real-world dataset by unsupervised dynamic adversarial training. VAE learns the latent distribution of the data through variational inference. The diffusion model gradually adds and removes noise based on the Markov chain, learns the reversal process of data distribution. [10] utilized GAN to simultaneously generate a set of synthetic load profiles with spatial-temporal correlations. [11] combined GAN and neural ordinary differential equations to generate high-fidelity synthetic phasor measurement unit data under varying load conditions, without needing to understand the underlying nonlinear dynamic equations. Deep generative models can generate various types of data, not just load or sensor data. For instance, [12] explored the use of GAN to generate synthetic electricity price scenarios. [13] introduced a GAN-based method to generate high-fidelity synthetic appliance power data. Some researchers combined VAE and GAN to synthesize data, such as energy consumption data [14] and renewable energy data [15]. There are also many studies focusing on scenario generation. Deep generative models can capture the uncertainty and spatiotemporal correlations of renewable energy plants without explicitly modeling the distributions and can generate diverse high-fidelity scenarios [16–18]. Considering privacy issues, [19,20] proposed new frameworks, which combine federated learning to generate high-quality scenarios while preserving privacy. Huang et al. introduced a model combined the powerful generative power of GAN with the strict privacy preservation of differential privacy [21].

To make the generation results controllable, conditional deep generative frameworks are further proposed to generate synthetic data upon certain conditions [22–24]. [25] encoded seasons and load types as labels to generate unique synthetic load profiles that match the labels. Yuan et al. proposed a method called C-StyleGAN2-SE for generating intraday renewable energy scenarios [26]. This method combines style-based generative adversarial networks with conditional generative adversarial networks, using meteorological variables as conditional inputs to generate high-fidelity renewable energy scenario data with complex spatiotemporal correlations. Wang et al. proposed a multivariate load state generation model based on conditional variational autoencoder [27]. By jointly optimizing the output noise parameters and sample-specific noise parameters, the multivariate load data with similar distribution to historical data was generated. Zhao et al. proposed the conditional diffusion model to generate source-charge scenes, and processed the condition information through the noise prediction network and the spatio-temporal fusion module to generate high-fidelity source-charge scene data [28]. Based on the conditional diffusion model, Wang et al. designed the noise estimation model and attention mechanism to achieve customized load characteristic synthesis to meet the data needs of different power customers [29].

Note that the training of conditional deep generative models requires labeled data (e.g., the user type, location, appliance name of a load profile). However, actual labels are even harder to acquire than measurement data due to user privacy and energy security concerns, while manual-labeling is costly and labor-intensive. Lacking labels impairs the applicability of conditional deep generative model in the energy sector. A commonly-used alternative solution is to first cluster the unlabeled data into different clusters, and then train vanilla deep generative model for each cluster respectively to enhance the generation results [30,31].

But the results still lack physical meanings and therefore are unexplainable. To address this issue, Information Maximizing Generative Adversarial Network (InfoGAN) is proposed by Chen et al. [32], which is an unsupervised learning method to achieve controllable synthetic data generation. [33] leverages the discrete latent feature extraction of InfoGAN as a clustering method, applying it to load data generation.

Although InfoGAN is promising in learning and synthesizing unlabeled dataset, the challenge of data imbalance has not been well addressed. Real-world data often exhibits a long-tail distribution [34], which is naturally imbalanced and can bias the model training and synthetic data generation results. To address this issue, in this paper we introduce an adaptive and contrastive InfoGAN (AC-InfoGAN) framework to achieve controllable synthesizing for the unlabeled and imbalanced dataset.

The major contributions of the paper are considered three folds:

(1) AC-InfoGAN framework is introduced to synthesize the highly-imbalanced and unlabeled energy time series datasets. This framework can extract both continuous and discrete features with clear physical meanings from the dataset as well as generating high-fidelity synthetic samples, which is efficient and is helpful in improving the dataset interpretability and proportionality.
(2) A novel frequency-based contrastive learning method is proposed to create positive samples, which can further enhance the AC-InfoGAN model training.
(3) A frequency-domain Multi-layer Perceptron (FreMLP) is introduced to formulate the AC-InfoGAN model, which can better capture the global characteristics of time series data and improve the model performance.

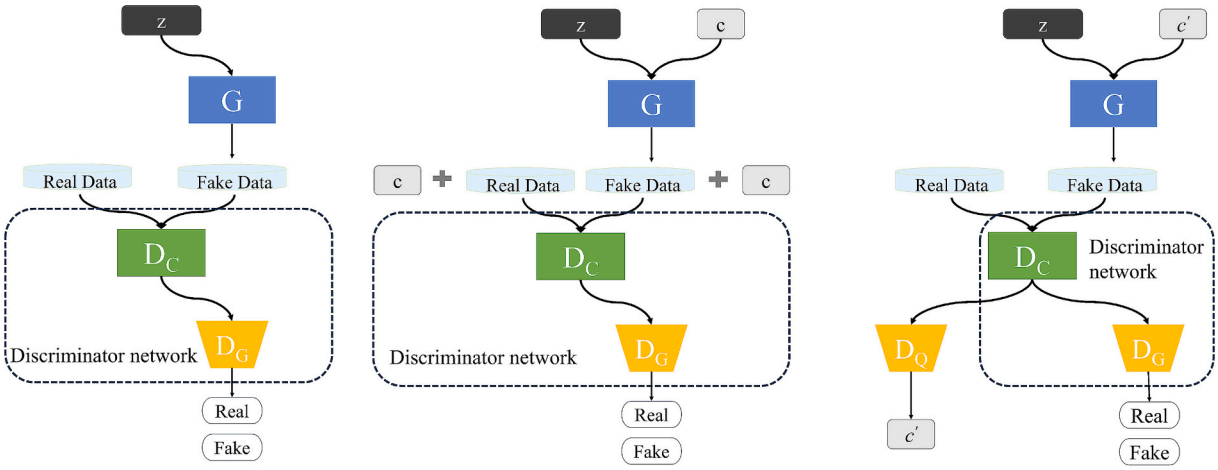
The rest of the paper are organized as follows: Section 2 introduces the methodology. Section 3 demonstrates the case study results, and Section 4 concludes this paper.

2. Methodology

In this section, we first introduce the vanilla GAN and cGAN to serve as the basis and benchmark of the AC-InfoGAN model. Then we introduce the AC-InfoGAN framework to achieve unsupervised and controllable feature extraction and synthetic data generation.

2.1. GAN and cGAN

GAN comprises a generator network (G) and a discriminator network (D). G aims to produce diverse synthetic data G($\mathbf{z}$) by learning the mapping from Gaussian noise $\mathbf{z}$ to real data $\mathbf{x}$, attempting to deceive D. Conversely, D evaluates and assigns higher scores to real data $\mathbf{x}$ and lower scores to G($\mathbf{z}$), aiming to distinguish between them. This setup frames GAN's training as an adversarial process between G and D, illustrated in Fig. 1(a) and (1).



(a) GAN (b) CGAN (c) InfoGAN

Fig. 1. The construction of the models, **z** represents gaussian noise, **c** represents label, **c**' represents latent code.

$$\min_G \max_D \left[\mathbb{E}_{\mathbf{x}\in P_r}[logD(\mathbf{x})] + \mathbb{E}_{\mathbf{x}\in P_g}[log(1-D(\hat{\mathbf{x}}))] \right] \tag{1}$$

where P_r represents the real data distribution, P_g represents the generated data. $\mathbb{E}$ is the expectation operator, and $\hat{\mathbf{x}}$=G(**z**). After GAN is well trained, the generator can work independently to create unlimited synthetic data G(**z**). To further stabilize the GAN's training process, Wasserstein GAN (WGAN) is proposed in [35] with a modified loss function:

$$\min_G \max_{D\in\omega} \left[\mathbb{E}_{\mathbf{x}\in P_r}[D(\mathbf{x})] - \mathbb{E}_{\hat{\mathbf{x}}\in P_g}[D(\hat{\mathbf{x}})] \right] \tag{2}$$

where ω is the set of 1-Lipschitz function. Besides, a gradient penalty method is further introduced to improve the performance of WGAN [36]. Then the final version of the loss function becomes:

$$L = \min_G \max_{D\in\omega} \left[\mathbb{E}_{\hat{\mathbf{x}}\in P_g}[D(\hat{\mathbf{x}})] - \mathbb{E}_{\mathbf{x}\in P_r}[D(\mathbf{x})] \right] + \lambda \mathbb{E}_{\tilde{\mathbf{x}}\in P_{\tilde{\mathbf{x}}}} \left[\left(\|\nabla_{\tilde{\mathbf{x}}} D(\tilde{\mathbf{x}})\|_2 - 1 \right)^2 \right] \tag{3}$$

To achieve controllable synthesis, cGAN is proposed to generate data that satisfy given conditions. Both the generator and the discriminator in cGAN have the condition vector c (i.e., labels of the data) as an additional input to help the generator learn the conditional distribution from the actual dataset, as shown in Fig. 1(b). The loss function of cGAN is

$$\min_G \max_D V(D,G) = \mathbb{E}_{\mathbf{x}\sim p_r(\mathbf{x})}[logD(\mathbf{x}|\mathbf{c})] + \mathbb{E}_{\mathbf{z}\sim p_z(\mathbf{z})}[log(1-D(G(\mathbf{z}|\mathbf{c})))] \tag{4}$$

As mentioned in Section 1, the training of cGAN requires labels of the actual dataset, which is demanding and costly. Next, we will introduce InfoGAN framework that can extract features with certain physics meaning from dataset to achieve convenient and controllable synthetic data generation.

2.2. InfoGAN

In the InfoGAN model, a latent code vector **c**' is introduced as an additional input to the generator to represent the certain physical meaning features to be extracted. If a stable correlation pattern between **c**' and the generated dataset G(**z**,**c**') can be established, then **c**' is considered to be features of G(**z**,**c**') with certain physical meanings. To achieve this, a mutual information term I(**c**', G(**z**,**c**')) is introduced to the GAN loss function to maximize their correlation:

$$\min_G \max_D V_1(D,G) = V(D,G) - \lambda I(\mathbf{c}', G(\mathbf{z},\mathbf{c}')) \tag{5}$$

where λ is the weighting factor. Note that in practice, I(**c**', G(**z**,**c**')) is hard to compute directly as it requires access to the posterior probability P(**c**'|**x**) that is difficult to estimate. As an approximate solution, a classifier network Q is further introduced to approximate P(**c**'|**x**), as shown in Fig. 1(c). We can define a variational lower bound, L_I(G,Q), of the mutual information, I(**c**', G(**z**,**c**')) [37]. If we set the common part as D_C, then we have D = D_C + D_G, Q = D_C + D_Q. D and Q share the same network structure and parameters except the last layer. The revised loss function becomes:

$$L_I(G,Q) \le I(\mathbf{c}', G(\mathbf{z},\mathbf{c}')) \tag{6}$$

$$\min_{G,Q} \max_D V_{InfoGAN}(D,G,Q) = V(D,G) - \lambda L_I(G,Q) \tag{7}$$

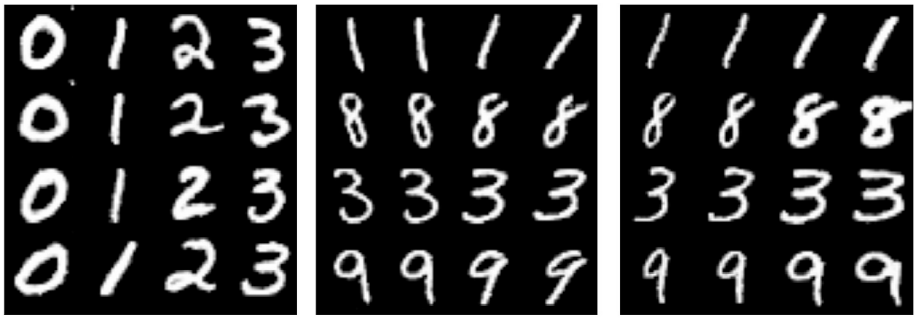
After the InfoGAN model is trained, we can correlate the extracted latent vector **c**' with physical characteristics of the unlabeled dataset to interpret the data, as well as controlling the generation results by setting up the digits in **c**'. The diversity of the generation results can be still achieved by varying Gaussian noise **z**. Meanwhile, we also apply Wasserstein distance and gradient penalty techniques to alleviate the mode collapse issue of GAN models, as mentioned in Section 2.1.

In Fig. 2, we show the results of applying the InfoGAN model to the MNIST dataset. The discrete code $\mathbf{c_1}$ captures significant changes in the shape of the digits. By changing the discrete code $\mathbf{c_1}$, the type of generated digit can be altered. The continuous codes capture continuous variations. $\mathbf{c_2}$ models the rotation of the digits, and $\mathbf{c_3}$ controls the width. InfoGAN extract interpretable features from MNIST, making the results controllable.

2.3. Adaptive distribution

In the original InfoGAN model, the discrete categorical control $\mathbf{c_1}$ follows a fixed uniform distribution, i.e., $\mathbf{c_1} \sim U(K=k, p=1/k)$, where k represents the number of discrete categories and p denotes the probability of each category. However, due to factors such as missing data and varying data sources, real-world datasets often exhibit imbalanced class distributions. The prior assumption of a uniform distribution for the discrete code does not align with the true class probabilities within the data, leading to impaired performances when applying InfoGAN to such imbalanced datasets.

To address this issue, inspired by the work of [38], this paper introduces the Gumbel-Softmax distribution to replace the fixed uniform probability discrete distribution (As shown in Fig. 3). The Gumbel-Softmax combines the Gumbel distribution with the Softmax function to enable the approximation of discrete variables during gradient descent. This allows the class probabilities to be updated during



(a) Shape (b) Rotation (c) Width

Fig. 2. Manipulating latent codes on MNIST: Each latent code changes from left to right while the other latent codes are fixed [32].

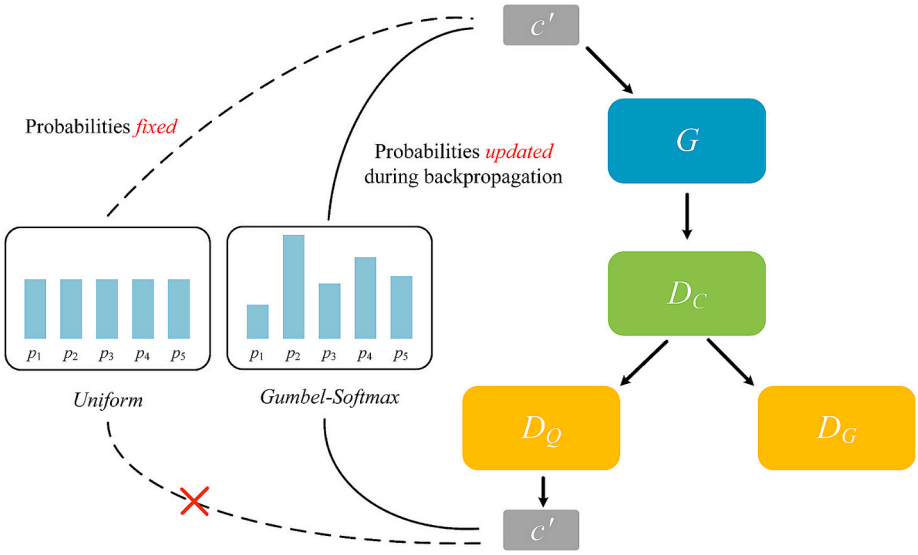


Fig. 3. Our model samples discrete categorical control code from a Gumbel-Softmax distribution that can update probabilities during backpropagation, replacing uniform distribution.

$$\mathbf{c}_1^i = \frac{\exp((\log(p_i) + g_i)/\tau)}{\sum_{j=1}^{k} \exp\left(\left(\log\left(p_j\right) + g_j\right)\Big/\tau\right)}, \quad i = 1, \ldots, k \tag{8}$$

backpropagation, thereby learning the prior distribution. Specifically, if $p_1, p_2, \ldots, p_k$ are the class probabilities, then sampling a k-dimensional vector $\mathbf{c}$ can be done in a differentiable manner.

In the equation, c_1^i represents the i-th element of $\mathbf{c_1}$; g_i and g_j are samples drawn from a Gumbel(0,1) distribution; and $\tau \in (0, \infty)$ is a temperature coefficient, a hyperparameter used to control the similarity between the samples from the Gumbel-Softmax distribution and the uniform distribution. As τ approaches 0, the sampled $\mathbf{c_1}$ from the Gumbel-Softmax distribution becomes closer to a uniform distribution. During training, the learnable class probabilities p_i's are updated via gradients to match the true class imbalance, starting from an initial uniform distribution. Note that the Gumbel-Softmax distribution is specifically designed for discrete categorical latent vectors. In contrast, a fixed uniform distribution is employed for the continuous latent vectors to achieve smooth and continuous feature control.

2.4. Contrastive learning

The classification of the latent vector $\mathbf{c}$ depends on the features that Q focuses on. However, since the training process is unsupervised, the specific features that Q attends to are ambiguous, leading to variability in the basis for classification (e.g., volatility, maximum value, identity, etc.). To address this, we introduce the concept of contrastive learning to force the model to focus on the identity of the curve. Contrastive learning is a self-supervised representation learning paradigm that aims to learn discriminative feature embeddings by maximizing similarity among positive samples while minimizing similarity among negative ones within a latent space. Positive samples refer to augmented instances of the same sample or intra-class samples that share similar characteristics, serving to align feature representations; negative samples are dissimilar instances from different classes or irrelevant data points, which can be used to establish contrastive boundaries to prevent representation collapse. For example, in the image processing domain, by enforcing the prediction consistency for two different views of the same image (i.e. positive samples), the model is encouraged to learn high-level representations of the image that are independent from views, such as object identity and image category [39,40].

In this section, we propose a novel frequency-based method for time series data to construct positive and negative samples based on curve identity and ensure that Q outputs similar results for positive pairs and

dissimilar results for negative pairs (As shown in Fig. 4). Owing to the inherent characteristics of time series data, constructing positive samples cannot be achieved through simple transformations like image flipping. We developed a straightforward yet effective method, which first transforms the time series into the frequency domain, randomly masks a small portion of low-amplitude frequencies, and then converts the data back to the time domain to generate positive samples. This masking process preserves the key characteristics of the data, allowing the model to distinguish data types still accurately.

In the absence of class labels, we treat each sample as an independent class. For a batch of N real curves, we generate 2 N curves by applying a transformation (frequency masking), which preserves the content of the original curves, creating variants of each curve. For the i-th curve S_i in this batch, its transformed variant S_i' is considered the only positive sample S_{pos}, while the remaining 2(N-1) curves are treated as negative samples S_{neg}, which consist of the k-th curve S_k ($k \neq i$) and the corresponding transformed variant S_k'. For each curve S_i, we define a loss function that uses {S_i, S_{pos}} as the positive pair and {S_i, S_{neg}} as negative pairs. During the training process, for each curve, we aim to improve the similarity among the positive pairs and the dissimilarity among negative pairs. Inspired by [39], the loss function of contrastive learning for a positive pair (i, j) is defined as:

$$\ell_i = -\log \frac{\exp\left(\boldsymbol{sim}\left(\mathbf{c}_i, \mathbf{c}_j\right)/\tau'\right)}{\sum_{k=1}^{2N} \mathbf{1}_{[k\neq i]} \exp\left(\boldsymbol{sim}\left(\mathbf{c}_i, \mathbf{c}_k\right)/\tau'\right)} \tag{9}$$

$$\boldsymbol{sim}\left(\mathbf{c}_i, \mathbf{c}_j\right) = \frac{\mathbf{c}_i^{\mathrm{T}} \cdot \mathbf{c}_j}{\|\mathbf{c}_i\| \|\mathbf{c}_j\|} \tag{10}$$

where $\mathbf{1}_{[k\neq i]} \in \{0,1\}$ is judgement function for negative pairs, j represents the positive pair, $\mathbf{c}$ refers to the features extracted using Q, τ' denotes the softmax temperature, and sim(·) represents the cosine similarity. During actual training, apart from S_i and S_{pos}, any other two curves are treated as negative pairs, which may lead to false negatives when both curves belong to the same category but are treated as negatives. To mitigate this, we compute the similarity using the features extracted from the penultimate layer of Q, while the final layer ensures that results within the same category remain similar. The overall loss is defined as $L_{\text{ntxent}} = \sum_{i=1}^{N} \ell_i$.

Based on the two adjustments mentioned above, the training objective of AC-InfoGAN becomes:

$$\min_{G,Q,p_i} \max_{D} L = V_{InfoGAN}(D, G, Q, p_i) + \lambda_2 L_{\text{ntxent}}(Q) \tag{11}$$

2.5. Network structure

The realization of AC-InfoGAN framework in this paper is shown in Fig. 5. The input is a 100-dimension Gaussian noise concatenated with a binary latent vector representing discrete features of the data and a continuous latent vector representing continuous features. The essence of GAN is to map a random noise distribution onto the distribution of real data. The dimensionality of the noise vector should be sufficiently large to capture the diversity of the generated data, yet not excessively large so as to avoid overfitting. Given that each sample of the generated data is 96-dimensional, we set up a comparable Gaussian noise dimension as 100, which is also a commonly-used value in many related studies [10,19,20]. This choice balances the need for diversity in the generated samples with the risk of overfitting, thereby facilitating the effective generation of realistic data.

Considering the sequential nature of time series data, we incorporate the attention mechanism and the frequency Multi-Layer Perceptron (MLP) into generative adversarial network:

2.6. Attention

Self-attention is a computational mechanism that selectively highlights multiple positions within a single sequence to construct a context-aware representation [41]. It operates by mapping a query vector and a set of key-value pairs to an output vector, computed as a weighted aggregation of values:

$$Attention(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = softmax\left(\frac{\mathbf{Q}\mathbf{K}^{\mathrm{T}}}{\sqrt{d_k}}\right)\mathbf{V} \tag{12}$$

where $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$ are learnable parameters matrixes, d_k is the dimension of queries and keys. In order to consider the complex temporal relationships, multi-head attention is used which linearly projects the queries, keys, and values multiple times with different learned linear projections in parallel:

$$\begin{aligned} MultiHead(\mathtt{Q}, \mathtt{K}, \mathtt{V}) &= Concat(head_1, \ldots, head_h)\mathtt{W}^{\mathtt{O}} \\ head_i &= Attention\left(\mathtt{Q}\mathtt{W}_i^{\mathtt{Q}}, \mathtt{K}\mathtt{W}_i^{\mathtt{K}}, \mathtt{V}\mathtt{W}_i^{\mathtt{V}}\right) \end{aligned} \tag{13}$$

where $\mathbf{W}^{\mathbf{O}}$, $\mathrm{W}_i^{\mathrm{Q}}$, $\mathrm{W}_i^{\mathrm{K}}$ and $\mathrm{W}_i^{\mathrm{V}}$ are the projection parameter matrices.

2.7. Frequency-domain MLP

Vanilla MLP relies on point mapping to capture time mapping and cannot handle global dependencies of time series. To address this, Yi et al. purposed the Frequency-domain MLP (FreMLP) [42]. FreMLP can

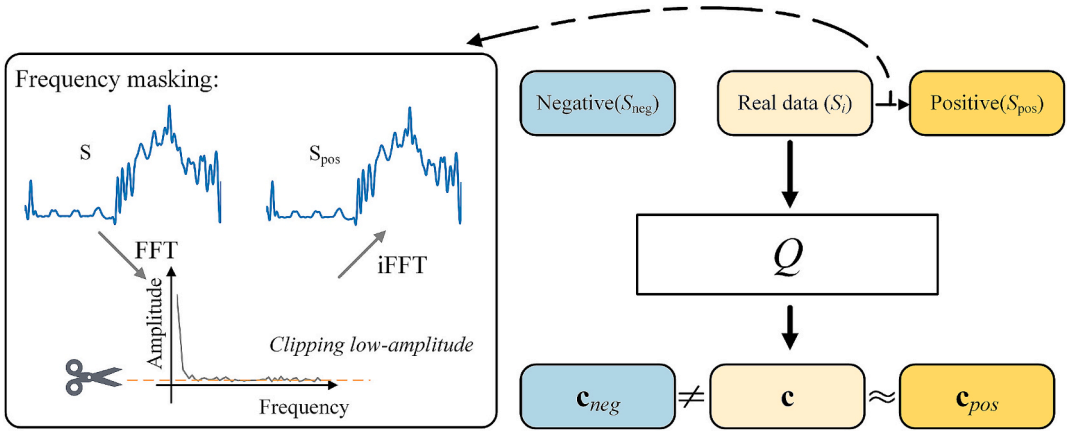


Fig. 4. The additional loss used to characterize contrastive learning. We employ frequency-masking data augmentation to obtain positive samples. Q is expected to capture the similarity between the features of positive samples and those of the true samples, as well as the dissimilarity between the features of negative samples and the true samples.

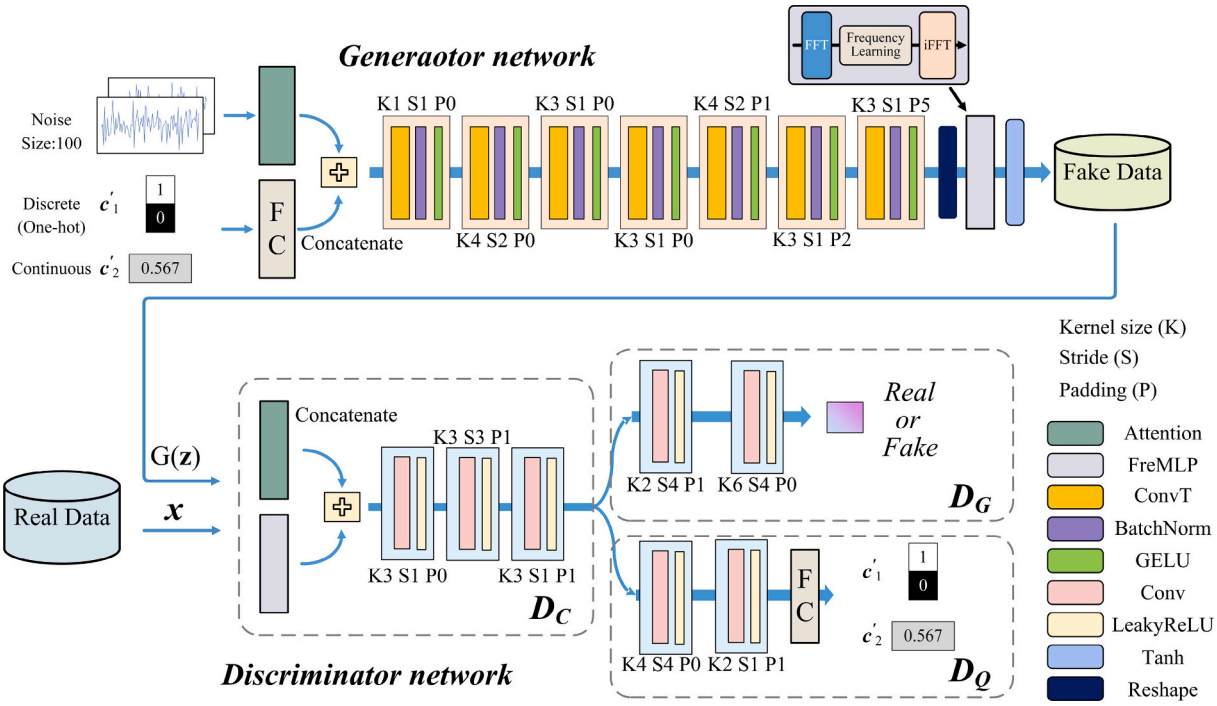


Fig. 5. Realization of AC-InfoGAN with corresponding kernel size (K), stride (S), padding (P) for each convolutional layer.

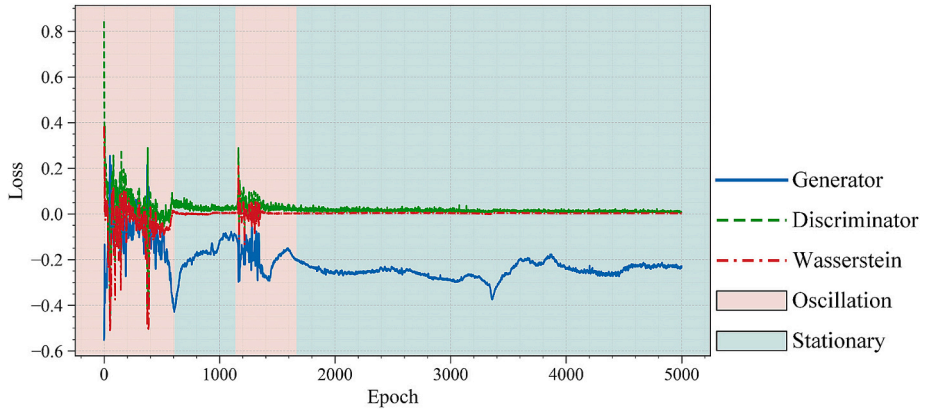


Fig. 6. The training process of AC-InfoGAN. Genrerator loss, discriminator loss and Wasserstein distance.

capture a global view of time series and concentrate on the key part by operating on spectral components acquired from series transformation. So, the ability of FreMLP is powerful to identify most important features and key patterns.

FreMLP encompasses two primary stages: transforming time-domain signals into frequency-domain spectra via Discrete Fourier Transform (DFT), followed by frequency learning through separate processing of real and imaginary coefficients using redesigned MLPs for each component. This dual-stage process, applied at both inter-series and intra-series levels, enhances model accuracy by capturing channel-wise and time-wise dependencies in the frequency domain.

Owing to channel independence, we only apply the frequency temporal learner, which aims to learn the temporal patterns in the frequency domain.

Assume a time series $\mathbf{X} = [X_1, X_2, \ldots, X_T] \in \mathbb{R}^{N \times T}$, consisting of N series and T timestamps, the frequency temporal learner can be formulated by:

$$Z_{temp} = \mathrm{DFT}(\mathbf{X}) = \mathrm{Re}(Z_{temp}) + j\mathrm{Im}(Z_{temp}) \tag{14}$$

$$S_{temp} = ReLu(\mathrm{Re}(Z_{temp})W_k - \mathrm{Im}(Z_{temp})W_j + B_k) + jReLu(\mathrm{Re}(Z_{temp})W_j + \mathrm{Im}(Z_{temp})W_k + B_j) \tag{15}$$

$$\mathbf{S} = \mathrm{IDFT}(S_{temp}) \tag{16}$$

$$\mathbf{Y} = LeakyRelu(\mathbf{S}w_1 + b_1)w_2 + b_2 \tag{17}$$

where W_k, W_j, B_k, B_j are the weight matrix and the bias in the frequency temporal learner, and w_1, w_2, b_1, b_2 are the weight matrix and the bias in the feedforward neural network. $\mathbf{Y}$ is the final output of FreMLP.

The generator network is a deep Convolutional Neural Network (CNN) with attention and FreMLP. The input latent code first goes through a fully connected layer, and then concatenates with the output of the attention layer. Transpose convolution layers with decreasing number of kernels are implemented to convert the input vector to the generated synthetic data. Batch normalization layers [43] and Gaussian Error Linear Units (GELU) activation functions [44] are attached after each convolution layer to stabilize the training process. After the

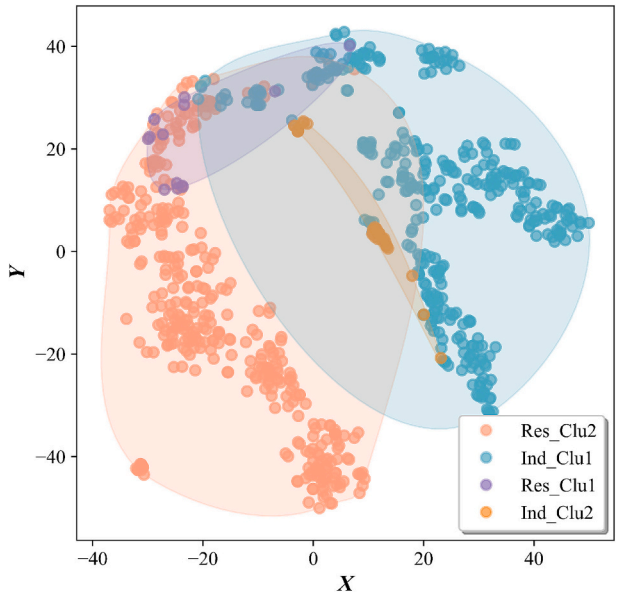


Fig. 7. 2-D t-SNE visualization of AC-InfoGAN results.

convolution layers, a FreMLP layer is connected to polish the generated data from a global view.

In discriminator, the real and fake data are processed separately through an attention layer and a FreMLP layer. Then the outputs from these layers are concatenated and fed into a set of convolution layers with increasing number of kernels and LeakyReLU activation functions. The classifier network shares the same structure and parameter with the discriminator network except the last layer to output the latent vector instead of the scores. Because Wasserstein distance and gradient penalty is implemented, there is neither batch normalization layer nor Sigmoid activation function in the discriminator network.

Note that we set up the hyper-parameters (K, S, P) of each convolutional layers based on two major principles: first, we set up small kernel sizes and small strides to make sure the input data is carefully learned without significant information losses. Meanwhile, we also set up zero paddings to keep a proper size of the output feature map, so that the marginal information can also be adequately extracted [45,46]. Second, considering the configuration of a neural network has inherent subjectivity and randomness, based on the first principle, we have conducted trial and error experiments on different combinations of the hyper-parameters and select the one with the best model performance.

3. Case study

In this section, we evaluate the performance of the AC-InfoGAN model in two aspects: 1) the ability to extract certain physical meaning features from unlabeled dataset, 2) the realisticness of the generation results. Two datasets are selected to set up the test case. The first dataset is a load profile dataset including residential and industrial users from 2017 to 2019, collected from utilities in North Carolina in the U.S. This dataset includes 1400 daily load profiles with 15-min granularity. We expect the AC-InfoGAN model to extract a binary feature named "load type" to correctly label these two types of load profiles, and then generate realistic synthetic load profiles for each type. The second dataset includes power output profiles of photovoltaic power plants starting from 2018 to 2020, collected from a PV operator in North Carolina in the U.S. This dataset also has 15-min granularity with 700 daily profiles in total. In this dataset, we will also try to extract continuous features to characterize the shape (such as magnitude, volatility, etc.) of the power output profiles, thus achieving further controllability of the generation process. The model is built in PyTorch environment and trained on a single NVIDIA GeForce 3090 GPU.

3.1. Case 1: Residential and Industrial Load Profiles

We randomly select 700 residential daily load profiles and 700 industrial daily load profiles from the load profile dataset to set up the test case. We use the 1400 daily load profiles in total to train and evaluate the AC-InfoGAN model. The minimum-maximum normalization is implemented to preprocess the load profiles to constrain the data range to [0,1]:

$$x^* = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{18}$$

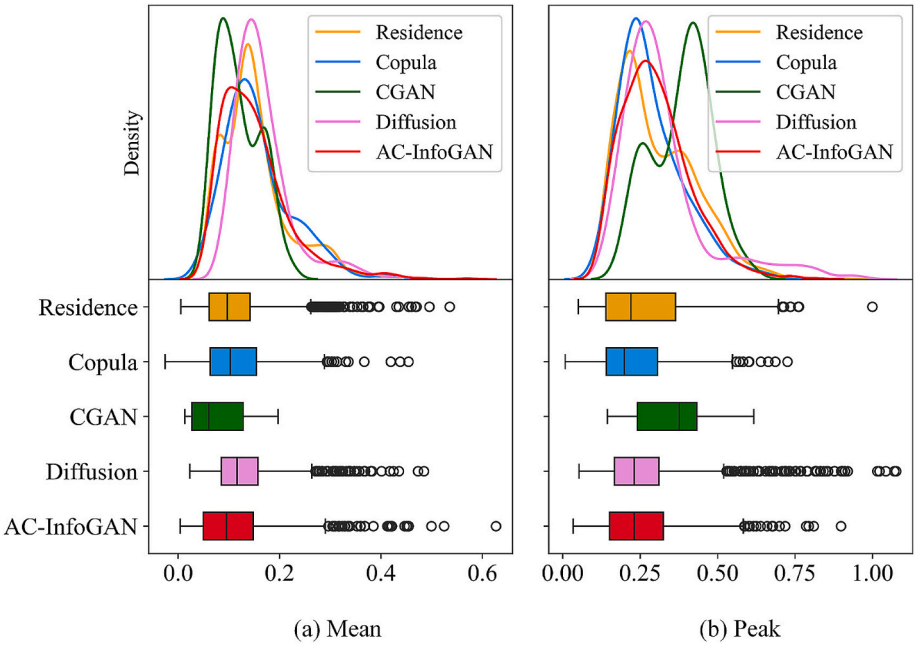


Fig. 8. (a) Residential mean load distribution curves and boxplots, and (b) Residential peak load distributions and boxplots.

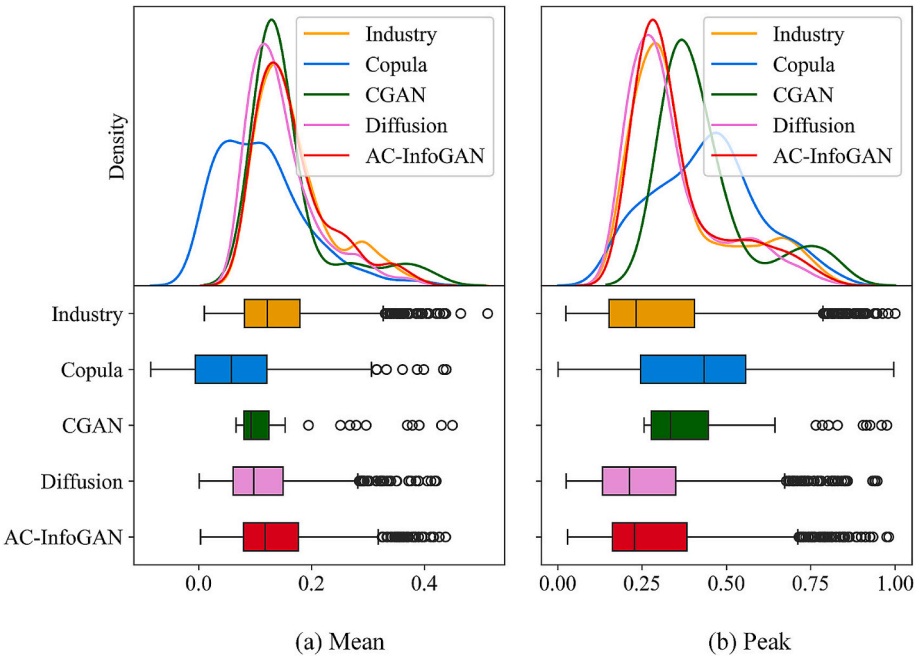


Fig. 9. (a) Industrial mean load distribution curves and boxplots, and (b) Industrial peak load distributions and boxplots.

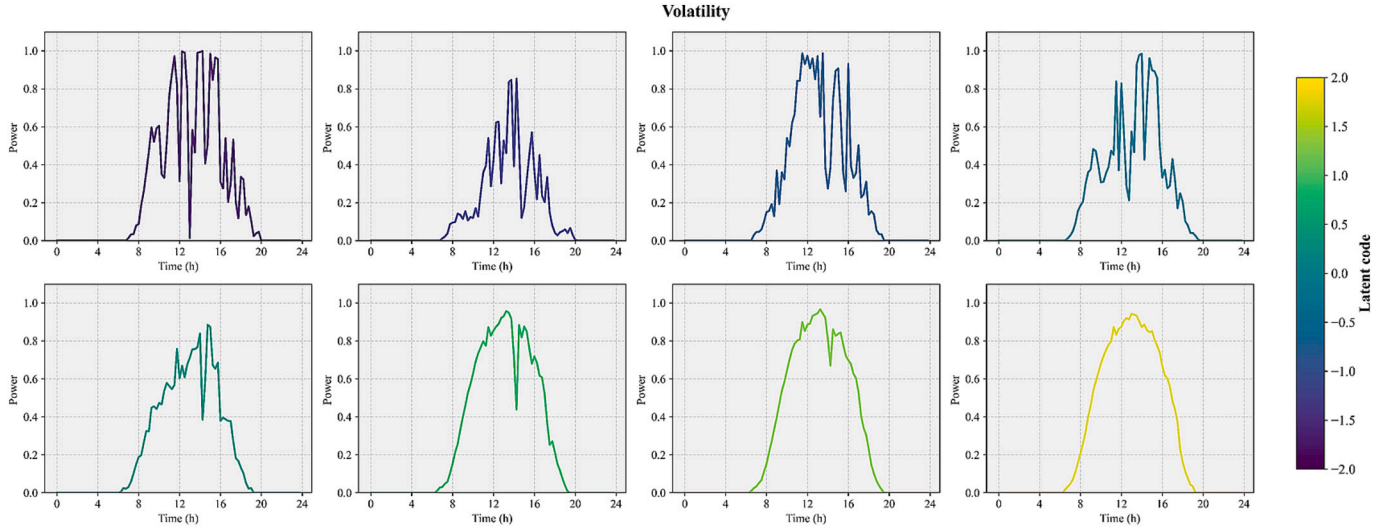


Fig. 10. Schematic diagram of PV output volatility varying with continuous latent code. The sampling results of the continuous latent code generation. The continuous latent code increases sequentially from left to right and from top to bottom.

where x^* is the normalized value, and x_{max} and x_{min} are the maximum and minimum values of the data samples, respectively. To extract a binary feature describing the load type, we set both the latent code c' and the output of the Q network as a two-digit vector in the one-hot encoding format. As a result, (0,1) will represent one type of load while (1,0) will represent the other. Note that due to the unsupervised learning nature of InfoGAN, there is no fixed mapping between the latent code and the load type. For simplicity, we mark the load profiles labeled by (1,0) as the first class, while (0,1) as the second.

3.2. Training process of AC-InfoGAN

The training process of the AC-InfoGAN model is summarized in Fig. 6, including the Wasserstein loss, generator loss and discriminator loss curves. The model is considered converged when the Wasserstein loss stabilizes around 0. Besides, after every training epoch, we save the network parameters and use the classifier network Q to assign labels to each load profile sample.

From Fig. 6, we find that the training process does not converge smoothly. Instead, there are multiple oscillation and stationary periods. During each oscillation period, the Wasserstein distance fluctuates significantly, and so do the labeling results for the load profile samples. This means the model is adjusting its parameters drastically to find a better way to label the load profiles to minimize the loss function, even if the labeling results need to be completely reversed compared to the previous epoch. After 2 oscillation and 1 stationary periods, the model tends to be stabilized and enters the final stationary period, in which most residential profiles are assigned to class 1 and most industrial profiles are assigned to class 2. The training takes approximately 1 h to converge on personal desktop. In this case, once $\mathbf{c}'$ is set to (1,0), the AC-InfoGAN model will generate synthetic load profiles that has residential characteristics.

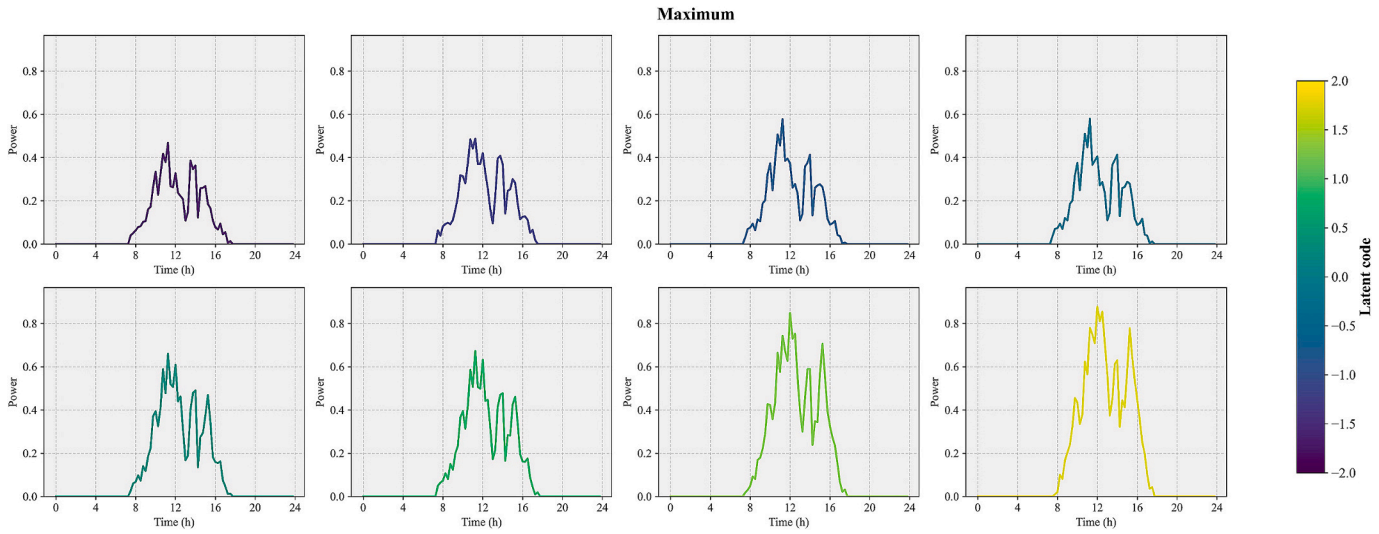


Fig. 11. Schematic diagram of PV output maximum varying with continuous latent code. The sampling results of the continuous latent code generation. The continuous latent code increases sequentially from left to right and from top to bottom.

3.3. Evaluation of feature extraction results

The most distinct feature of AC-InfoGAN is that it can extract controllable features with certain physics meaning from unlabeled dataset. In our case, we expect the AC-InfoGAN model to extract a binary feature "load type" to distinguish between the residential load profiles and the industrial load profiles. If successful, the load profiles labeled (1,0) should be all residential (or industrial), while the load profiles labeled (0,1) should be all industrial (or residential).

Because we have the actual load type of each load profile, we calculate the labeling accuracy after the AC-InfoGAN model is well trained on balance dataset, shown in Table 1. We can see 96 % residential load profiles are labeled by cluster 1 with latent code (1,0), while 95 % industrial load profiles are labeled by cluster 2 with latent code (0,1). Fig. 7 is a 2-D visualization of the load profile samples and the AC-InfoGAN labeling results by t-distributed Stochastic Neighbor Embedding (t-SNE) [47] We can see that the labeling error happens in the overlapping area of residential and industrial samples, the samples of which are indistinguishable in nature.

As a comparison, Table 1 presents the performance of various clustering methods and the InfoGAN-based models in handling imbalanced datasets. The degree of imbalance in the table refers to the ratio of residential to industrial samples, such as 1:1 indicating an equal number of residential and industrial samples, and 1:2 indicating that the number of residential samples is half that of the industrial samples, and so on.

Regarding the AC-InfoGAN model, it can be observed that as the imbalance ratio increases, the classification accuracy shows only a slight decline. Compared to the original InfoGAN model, the improved version is less affected by data imbalance. Additionally, the table presents the performance of k-means, hierarchical clustering(HC), and Gaussian Mixture Models (GMM) under different levels of imbalance. These mathematically-based traditional clustering methods are generally less influenced by imbalanced datasets but lack the capability to effectively separate industrial from residential categories. Instead, InfoGAN-based methods focus on the high-dimensional shape characteristics of the load profiles, therefore leading to better classification accuracy. More importantly, InfoGAN-based methods can automatically extract a binary feature to label each load profile without human intervention. Such an auto-labeling capability is critical in processing the large amount of unlabeled data in power systems, and can be further used to achieve controllable synthetic data generation.

3.4. Evaluation of synthetic load profile generation results

In this section, we evaluate the realisticness of the synthetic load profiles generated by AC-InfoGAN. 320 synthetic load profiles are first generated by the trained AC-InfoGAN for both industrial and residential loads. Then we evaluate the realisticness of the generation results from two aspects: 1) we define 5 indexes to quantify and compare the shape characteristics between the real and the generated load profiles. 2) we calculate the distribution-to-distribution similarity between the real and the generated load profiles. As a benchmark, we also generate load profiles by the copula function and the cGAN model that shares a similar hyperparameter configuration with AC-InfoGAN.

The 5 indexes are defined as follows, including near-peak load, near-base load, high-load duration, rising duration and rising frequency [31,48].

Table 1
Classification percents.

Imbalance degree	Model	K-means		HC		GMM		InfoGAN		AC-InfoGAN	
	Type	Clu1	Clu2	Clu1	Clu2	Clu1	Clu2	Clu1	Clu2	Clu1	Clu2
1:1	Res	1	0	1	0	1	0	0.050	0.950	0.048	0.952
	Ind	0.364	0.636	0.274	0.726	0.253	0.747	0.949	0.051	0.957	0.043
1:2	Res	1	0	0.183	0.817	1	0	0.086	0.914	0.063	0.937
	Ind	0.365	0.635	0.718	0.282	0.252	0.748	0.871	0.129	0.931	0.069
1:5	Res	1	0	0.181	0.819	1	0	0.115	0.885	0.065	0.935
	Ind	0.421	0.579	0.715	0.285	0.288	0.712	0.742	0.258	0.955	0.045
1:10	Res	1	0	0.194	0.806	1	0	0.096	0.904	0.068	0.932
	Ind	0.324	0.676	0.763	0.237	0.323	0.677	0.716	0.284	0.926	0.074

- *Near-peak load* P_{peak} is defined as the 97.5 % percentile of a load profile. We exclude the maximum value of the load profile to avoid the influence of abnormal load spikes.
- *Near-base load* P_{base} is defined as the 2.5 % percentile of a load profile. Similarly, we exclude the minimum value to avoid the influence of zero values caused by factors such as communication failure, equipment fault, etc.
- *High-load duration* is defined as the maximum duration of a load profile segment that is continuously larger than $1/2(P_{peak} + P_{base})$, which is defined as high load.
- *Rising duration* measures the time it takes for the load to rise from the near-base load to the high load. This index reflects the dynamic characteristics of the load profile.
- *Rising frequency* measures how many times the load rises from the near-base load to the high load within a load profile. This index reflects the volatility of the load profile.

We denote the generation results of AC-InfoGAN with latent code (1,0) as GEN1, and (0,1) as GEN2. Table 2 shows the index calculation results for both the real and generated load profiles. We can see that GEN1 has similar characteristics with industrial loads, while GEN2 is similar with residential loads. For example, the near-peak load, near-base load and high-load duration of GEN1 is significantly larger than GEN2, showing typical characteristics of industrial loads. The rising frequency of GEN2 is higher than GEN1, indicating the volatility of residential loads. Based on Table 2, we can conclude that the generation results are statistically realistic compared with actual residential and industrial loads.

To draw a robust conclusion, we employ statistical methodologies to substantiate our findings. Given that the standard deviations are unequal, we utilize Welch's *t*-test to achieve a more precise result. The t-value and degree of freedom are calculated based on the means and standard deviations according to the following formulars:

$$t = \frac{\mu_1 - \mu_2}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}} \tag{19}$$

$$df = \frac{\left(\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}\right)^2}{\frac{\left(s_1^2/n_1\right)^2}{n_1 - 1} + \frac{\left(s_2^2/n_2\right)^2}{n_2 - 1}} \tag{20}$$

where μ is the mean, s is the standard deviation and n is the number of samples. If the degree of freedom is equal or greater than 30, the Gaussian distribution is used to obtain p-value:

$$p = 2 \cdot (1 - \Phi(|t|)) \tag{21}$$

where Φ is the Cumulative Distribution Function (CDF) of the standard normal distribution. If not, p-value is obtained by t-distribution:

$$p = 2 \cdot \left(1 - F_{t(df)}(|t|)\right) \tag{22}$$

where $F_{t(df)}$ is the CDF of a t-distribution with df degree of freedom.

We consider the null hypothesis H_0: $\mu_1 = \mu_2$ (H_1: $\mu_1 \neq \mu_2$), and μ is the mean. The null hypothesis H_0 is rejected if the p-value is less than the predetermined significance level α. We employ a small α value of 0.0001 in practice, ensuring generated data maintains sufficient diversity. *P*-values are summarized in Table 3. We can draw a conclusion that GEN1 learns the characteristics of the industrial loads, while GEN2 represents residential loads.

To further evaluate the model performance, we generate synthetic residential and industrial load profiles based on copula, cGAN and diffusion. The result distributions of mean and peak values of AC-InfoGAN, copula', cGAN', diffusion and the actual load profiles are compared in Figs. 8–9. We can see the AC-InfoGAN and diffusion have comparable and best performances. cGAN has better accuracy in approximating the main body of the actual load distribution, while copula has unstable performance in different types load profiles.

Additionally, we implement the permutation entropy and differential entropy to quantify the diversity and complexity of datasets:

$$H_p = -\sum_{i=1}^{n=m!} p_i \ln p_i \tag{23}$$

$$H_d = -\int_{-\infty}^{\infty} f(x) \ln f(x) dx \tag{24}$$

where H_p is the permutation entropy, H_d is the differential entropy, m is the time window size, n is the total number of possible permutation patterns, p_i is the probability that the i-th permutation pattern occurs, and $f(x)$ is the probability density function.

As shown in Table 4, for residential load profiles, the diffusion has the closest performance to real data. For industrial profiles, AC-InfoGAN is the closest one. Meanwhile, some useful metrics are implemented, such as Kullback-Leibler divergence (KL divergence), Fréchet Inception Distance (FID) and Dynamic Time Warping (DTW) [30,49,50], which can quantify the distribution-to-distribution similarity between the actual load profiles and the generation results. KL divergence measures the information loss between two probability distributions. FID can evaluate the statistical feature matching degree of the generative model. DTW aligns time series and measures time sequence similarity. The calculation formulas are as follows:

$$D_{KL}\mathbb{P}(\mathbb{P} \parallel \mathbb{Q}) = \sum \mathbb{P} log \frac{\mathbb{P}}{\mathbb{Q}} \tag{25}$$

$$\mathrm{FID} = \left\|\mu_{\mathrm{real}} - \mu_{\mathrm{gen}}\right\|_2^2 + \mathrm{Tr}\left(\Sigma_{\mathrm{real}} + \Sigma_{\mathrm{gen}} - 2\left(\Sigma_{\mathrm{real}}\Sigma_{\mathrm{gen}}\right)^{1/2}\right) \tag{26}$$

Table 3
P values of Welch's t-test.

Metrics	Near-peak load	Near-peak load	High-load duration	Rising duration	Rising frequency
Ind & GEN1	0.844	0.0004	0.686	0.0363	0.0159
Ind & GEN2	$<1 \times 10^{-100}$	$<1 \times 10^{-100}$	$<1 \times 10^{-100}$	$<1 \times 10^{-100}$	$<1 \times 10^{-100}$
Res & GEN1	$<1 \times 10^{-100}$	$<1 \times 10^{-100}$	$<1 \times 10^{-100}$	$<1 \times 10^{-100}$	$<1 \times 10^{-100}$
Res & GEN2	0.0376	0.5509	0.8123	0.0283	0.770

Table 2
Curve shape indexes of residential and industrial.

Class	Near-peak load		Near-peak load		High-load duration		Rising duration		Rising frequency	
	mean	std	mean	std	mean	std	mean	std	mean	std
Ind	9.682	5.783	2.979	0.708	2.323	3.286	1.585	1.208	1.091	0.829
GEN1	9.606	5.537	2.806	0.724	2.233	3.196	1.792	1.520	0.984	0.545
Res	1.155	0.690	0.434	0.361	0.627	1.264	2.345	0.965	3.212	2.264
GEN2	1.064	0.609	0.419	0.366	0.609	1.020	2.545	1.448	3.166	2.287

Table 4
The results of entropy calculation.

Metrics	Permutation entropy		Differential entropy	
Dataset	Res	Ind	Res	Ind
Real data	2.5613	2.5603	−1.2337	−1.0638
Copula	2.5344	2.5839	−0.8794	−2.9953
cGAN	2.5889	2.5821	−0.9204	−1.3014
Diffusion	**2.5686**	2.5685	**−1.3214**	−1.1606
AC-InfoGAN	2.5719	**2.5568**	−1.1157	**−1.0730**

$$DTW(R,G) = \min_{W} \sum_{s=1}^{k} d(w_s) \tag{27}$$

In eq. (25), $\mathbb{P}$ represents the target distribution, $\mathbb{Q}$ represents the reference distribution. In eq. (26), μ is the mean of feature vectors, $\sum$ is the covariance matrix and $\mathrm{Tr}(\cdot)$ is the trace of a matrix. In eq. (27), $d(w_s)$ is the distance between two points, and $w_s = (i, j)$ represents the correspondence between the i-th point of real data R and the j-th point of generated data G.

In this paper, we set $\mathbb{Q}$ as the distribution of actual load profiles, and $\mathbb{P}$ as the distribution of generated load profiles. Table 5 summarizes the calculation results of KL divergence, FID and DTW. Note that we calculate both $D_{KL}(\mathbb{P} \,||\, \mathbb{Q})$ and $D_{KL}(\mathbb{Q} \,||\, \mathbb{P})$due to the asymmetry. We can see that the AC-InfoGAN demonstrates comparable performance to diffusion on residential data, and outperforms all other models on industrial data.

3.5. Case 2: Continuous Feature Extraction Evaluation

In addition to case 1, in this case we select 700 daily photovoltaic power output profiles to further evaluate whether AC-InfoGAN can extract continuous features (such as magnitude and volatility of the profile) to better characterize the power output profiles, based on which we can achieve finer control of the follow-up synthetic data generation process. We select the photovoltaic power output profiles to formulate the test data, and concatenate a one-dimension continuous latent code to the original binary latent code to represent the continuous feature to be extracted. The continuous latent code is initialized by a uniform distribution within [−2,2]. After the AC-InfoGAN is trained, we will generate synthetic photovoltaic power output profiles constrained by different set up of the continuous latent code to demonstrate the effectiveness. The experiment is conducted twice on two different subsets of the photovoltaic power data set, one includes all weather conditions and the other only includes cloudy days. The generation results are shown in Fig. 10 and Fig. 11.

From Figs.10–11, we can see that the continuous latent code is highly correlated with a certain shape characteristic of the photovoltaic profiles. More specifically, in Fig. 10 when the continuous latent code increases from −2 to 2, the generated profiles share similar magnitude but become more and more smooth. As a result, we can conclude that the continuous latent code represents the volatility of the photovoltaic power profiles, which is granted by certain physical meanings. To better quantify the volatility of the profiles, we define 3 indexes including Sum of Absolute First-order Difference (SAFOD), Coefficient of Variation (CV) and Ratio of the Sum of Absolute First-order Difference to the Maximum (RSAFODM):

$$\mathrm{SAFOD} = \sum_{i=1}^{N-1} |x_{i+1} - x_i| \tag{28}$$

$$\mathrm{CV} = \frac{\sigma}{\mu} \tag{29}$$

$$\mathrm{RSAFODM} = \frac{\sum_{i=1}^{N-1} \left| x_{i+1} - x_i \right|}{\max(x)} \tag{30}$$

where N is the sequence length, x_i is the i-th element in the sequence, σ and μ are the standard deviation and mean value of the sequence, respectively. The index calculation results are shown in Table 6. we can see that all 3 indexes show clear decreasing trend along with the increase of the continuous latent code. Note that the indicators have a temporary increase when the latent code is −1.428. This is because the latent code is essentially a synthetic feature extracted by the AC-InfoGAN model trying to represent the profile volatility, which is influenced by the stochastic and nonlinear nature of the neural network. Therefore, the extracted latent code may be less effective in certain ranges, and is considered useful but not perfect. We further calculate 3 correlation indicators between the latent code and the 3 indexes: Pearson correlation coefficient, Spearman rank correlation coefficient and Kendall rank correlation coefficient, shown in Table 7. We can see that the latent code is highly correlated with the three volatility indexes. All the observations from Tables 6 and 7 demonstrate that the latent code successfully describes the profile volatility.

Similar conclusions can be obtained in Fig. 11. We can see that under cloudy days, the magnitude of the generated profiles increases along with the continuous latent code increasing from −2 to 2, while the profile volatility remains stable. This indicates that the continuous latent code represents the magnitude of the profiles. Also, the continuous latent code will be granted with different physical meanings when the AC-InfoGAN is trained on different datasets.

4. Conclusions

In this paper, we propose the AC-InfoGAN model to extract physical features and enable controllable synthetic data generation for unlabeled and imbalanced energy data. Compared to classical GAN-based methods such as cGAN, the most distinct characteristic of AC-InfoGAN is that an additional latent code is included in the model input to act as the data features to be extracted, and a classifier sharing the same parameters with the discriminator is introduced to output the latent code. Moreover, a mutual information term is included in the loss function to maximize the correlation between the input latent code and the classifier output. To better address the imbalance of the dataset in the dataset, we replace the uniform distribution with the Gumbel-Softmax distribution to

Table 5
Metrics between real data and generated data.

Metrics	KL			FID		DTW	
Model		Res	Ind	Res	Ind	Res	Ind
Copula	$\mathbb{Q} \,||\, \mathbb{P}$	0.263	0.329	0.00775	0.02934	360.11	1493.21
	$\mathbb{P} \,||\, \mathbb{Q}$	0.341	0.358				
cGAN	$\mathbb{Q} \,||\, \mathbb{P}$	0.165	0.042	0.00184	0.00129	257.72	605.91
	$\mathbb{P} \,||\, \mathbb{Q}$	0.123	0.038				
Diffusion	$\mathbb{Q} \,||\, \mathbb{P}$	**0.094**	0.013	**0.00021**	0.00061	**226.11**	517.30
	$\mathbb{P} \,||\, \mathbb{Q}$	**0.085**	0.020				
AC-InfoGAN	$\mathbb{Q} \,||\, \mathbb{P}$	0.100	**0.012**	0.00038	**0.00054**	237.08	**479.43**
	$\mathbb{P} \,||\, \mathbb{Q}$	0.107	**0.018**				

Table 6
Results of volatility indicator calculation.

Latent code	−2.000	−1.428	−0.857	−0.286	0.286	0.857	1.428	2.000
SAFOD	9.323	5.892	6.805	6.349	3.303	3.154	2.416	1.967
CV	0.720	0.809	0.719	0.714	0.629	0.613	0.602	0.568
RSAFODM	9.334	6.906	6.895	6.453	3.733	3.295	2.500	2.086

Table 7
Results of correlation indicators.

Variables	Pearson	Spearman	Kendall
SAFOD	−0.936	−1.000	−0.857
CV	−0.969	−1.000	−0.929
RSAFODM	−0.899	−0.905	−0.714

dynamically adapt to the real data distribution along with the model training process, and incorporate contrastive learning loss to guide the model training through frequency-domain analysis. The attention mechanism and frequency learning are utilized in the generator and discriminator for a global view to feature extraction and synthetic data. The architectures of the generator and discriminator are specifically designed to avoid overly narrow bottleneck layers and to facilitate maximal information throughput, encouraging the latent code to capture representative features of the unlabeled imbalanced dataset. These features, often tied to certain physical meanings, can further be used to control the generation process.

We have the following 3 key observations from the case study results: 1) The AC-InfoGAN training process has multiple oscillation and stationary periods, which is different from classical GAN-based methods. However, such a training process is considered reasonable because the InfoGAN-based model is seeking for an optimal way of labeling the data samples, which may lead to drastic changes of both the labeling results and the network losses. 2) AC-InfoGAN can extract both discrete features (load type) and continuous features (profile magnitude and volatility) of the energy data, without being influenced by imbalanced dataset. 3) The generation results of AC-InfoGAN has comparable realisticness with classical methods such as diffusion.

Future work may focus on increasing the dimension of the input latent code to extract multiple controllable features from the unlabeled dataset and evaluating their impact on data synthesis.

CRediT authorship contribution statement

Zhenghao Zhou: Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis. **Yiyan Li:** Writing – review & editing, Supervision, Resources, Project administration, Funding acquisition, Data curation, Conceptualization. **Runlong Liu:** Validation, Software, Methodology. **Xiaoyuan Xu:** Formal analysis, Conceptualization. **Zheng Yan:** Resources, Data curation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

This work was supported by National Natural Science Foundation of China under Grant 52307121, and also supported by Science and Technology Commission of Shanghai Municipality under Grant 23YF1419000.

Data availability

The authors do not have permission to share data.

References

[1] Lopez JMG, Pouresmaeil E, Canizares CA, Bhattacharya K, Mosaddegh A, Solanki BV. Smart residential load simulator for energy Management in Smart Grids. IEEE Trans Ind Electron Feb. 2019;66(2):1443–52. https://doi.org/10.1109/TIE.2018.2818666.
[2] Dickert J, Schegner P. A time series probabilistic synthetic load curve model for residential customers. In: 2011 IEEE Trondheim PowerTech. Trondheim: IEEE; Jun. 2011. p. 1–6. https://doi.org/10.1109/PTC.2011.6019365.
[3] J. K. Gruber and M. Prodanovic, "Residential Energy Load Profile Generation Using a Probabilistic Approach," in 2012 Sixth UKSim/AMSS European symposium on computer modeling and simulation, Nov. 2012, pp. 317–322. doi: https://doi.org/10.1109/EMS.2012.30.
[4] Sarochar J, et al. Synthesizing energy consumption data using a mixture density network integrated with long short term memory. In: 2019 IEEE green technologies conference(GreenTech). Lafayette, LA, USA: IEEE; Apr. 2019. p. 1–4. https://doi.org/10.1109/GreenTech.2019.8767148.
[5] Diao L, Sun Y, Chen Z, Chen J. Modeling energy consumption in residential buildings: a bottom-up analysis based on occupant behavior pattern clustering and stochastic simulation. Energ Build Jul. 2017;147:47–66. https://doi.org/10.1016/j.enbuild.2017.04.072.
[6] Sun M, Cremer J, Strbac G. A novel data-driven scenario generation framework for transmission expansion planning with high renewable energy penetration. Appl Energy Oct. 2018;228:546–55. https://doi.org/10.1016/j.apenergy.2018.06.095.
[7] Goodfellow I, et al. Generative adversarial nets. In: Advances in neural information processing systems. Curran Associates, Inc.; 2014.
[8] Doersch C. Tutorial on Variational Autoencoders. Jan. 03, 2021. https://doi.org/10.48550/arXiv.1606.05908. arXiv: arXiv:1606.05908.
[9] Ho J, Jain A, Abbeel P. Denoising diffusion probabilistic models. In: Advances in neural information processing systems. Curran Associates, Inc.; 2020. p. 6840–51.
[10] Hu Y, et al. MultiLoad-GAN: a GAN-based synthetic load group generation method considering spatial-temporal correlations. IEEE Trans Smart Grid Mar. 2024;15(2):2309–20. https://doi.org/10.1109/TSG.2023.3302192.
[11] X. Zheng, B. Wang, D. Kalathil, and L. Xie, "Generative Adversarial Networks-Based Synthetic PMU Data Creation for Improved Event Classification," in 2022 IEEE Power & Energy Society General Meeting (PESGM), Jul. 2022, pp. 1–1. doi: https://doi.org/10.1109/PESGM48719.2022.9917130.
[12] Walter V, Wagner A. Probabilistic simulation of electricity price scenarios using conditional generative adversarial networks. Energy AI Dec. 2024;18:100422. https://doi.org/10.1016/j.egyai.2024.100422.
[13] Harell A, Jones R, Makonin S, Bajić IV. TraceGAN: synthesizing appliance power signatures using generative adversarial networks. IEEE Trans Smart Grid Sep. 2021;12(5):4553–63. https://doi.org/10.1109/TSG.2021.3078695.
[14] Razghandi M, Zhou H, Erol-Kantarci M, Turgut D. Smart home energy management: VAE-GAN synthetic dataset generator and Q-learning. IEEE Trans Smart Grid Mar. 2024;15(2):1562–73. https://doi.org/10.1109/TSG.2023.3288824.
[15] Li Z, et al. A novel scenario generation method of renewable energy using improved VAEGAN with controllable interpretable features. Appl Energy Jun. 2024;363:122905. https://doi.org/10.1016/j.apenergy.2024.122905.
[16] Chen Y, Wang Y, Kirschen D, Zhang B. Model-free renewable scenario generation using generative adversarial networks. IEEE Trans Power Syst May 2018;33(3):3265–75. https://doi.org/10.1109/TPWRS.2018.2794541.
[17] Ye L, Peng Y, Li Y, Li Z. A novel informer-time-series generative adversarial networks for day-ahead scenario generation of wind power. Appl Energy Jun. 2024;364:123182. https://doi.org/10.1016/j.apenergy.2024.123182.
[18] Li S, Xiong H, Chen Y. DiffCharge: generating EV charging scenarios via a Denoising diffusion model. IEEE Trans Smart Grid Jul. 2024;15(4):3936–49. https://doi.org/10.1109/TSG.2024.3360874.
[19] Li Y, Li J, Wang Y. Privacy-preserving spatiotemporal scenario generation of renewable energies: a federated deep generative learning approach. IEEE Trans Industr Inform Apr. 2022;18(4):2310–20. https://doi.org/10.1109/TII.2021.3098259.
[20] Chen Z, Li J, Cheng L, Liu X. Federated-WDCGAN: a federated smart meter data sharing framework for privacy preservation. Appl Energy Mar. 2023;334:120711. https://doi.org/10.1016/j.apenergy.2023.120711.
[21] Huang J, Huang Q, Mou G, Wu C. DPWGAN: high-quality load profiles synthesis with differential privacy guarantees. IEEE Trans Smart Grid Jul. 2023;14(4):3283–95. https://doi.org/10.1109/TSG.2022.3230671.

[22] Mirza M, Osindero S. Conditional Generative Adversarial Nets. Nov. 06, 2014. arXiv: arXiv:1411.1784.
[23] Sohn K, Lee H, Yan X. Learning structured output representation using deep conditional generative models. Adv Neural Inf Proces Syst 2015;28.
[24] Dhariwal P, Nichol A. Diffusion models beat gans on image synthesis. Adv Neural Inf Proces Syst 2021;34:8780–94.
[25] A. Pinceti, L. Sankar, and O. Kosut, "Synthetic Time-Series Load Data via Conditional Generative Adversarial Networks," Jul. 07, 2021, arXiv: arXiv:2107.03545.
[26] Yuan R, Wang B, Sun Y, Song X, Watada J. Conditional style-based generative adversarial networks for renewable scenario generation. IEEE Trans Power Syst Mar. 2023;38(2):1281–96. https://doi.org/10.1109/TPWRS.2022.3170992.
[27] Wang C, Sharifnia E, Gao Z, Tindemans SH, Palensky P. Generating multivariate load states using a conditional variational autoencoder. Electr Power Syst Res Dec. 2022;213:108603. https://doi.org/10.1016/j.epsr.2022.108603.
[28] Zhao W, Shao Z, Yang S, Lu X. A novel conditional diffusion model for joint source-load scenario generation considering both diversity and controllability. Appl Energy Jan. 2025;377:124555. https://doi.org/10.1016/j.apenergy.2024.124555.
[29] Wang Z, Zhang H. Customized load profiles synthesis for electricity customers based on conditional diffusion models. IEEE Trans Smart Grid Jul. 2024;15(4):4259–70. https://doi.org/10.1109/TSG.2024.3366212.
[30] Silva WN, Bandória LHT, Dias BH, De Almeida MC, De Oliveira LW. Generating realistic load profiles in smart grids: an approach based on nonlinear independent component estimation (NICE) and convolutional layers. Appl Energy Dec. 2023;351:121902. https://doi.org/10.1016/j.apenergy.2023.121902.
[31] Wang Z, Hong T. Generating realistic building electrical load profiles through the generative adversarial network (GAN). Energ Build Oct. 2020;224:110299. https://doi.org/10.1016/j.enbuild.2020.110299.
[32] Chen X, Duan Y, Houthooft R, Schulman J, Sutskever I, Abbeel P. InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets. Jun. 11, 2016. arXiv: arXiv:1606.03657.
[33] Lan J, Zhou Y, Guo Q, Sun H. A data-driven approach for generating load profiles based on InfoGAN and MKDE. Front Energy Res Dec. 2023;11:1339543. https://doi.org/10.3389/fenrg.2023.1339543.
[34] Van Horn G, et al. The iNaturalist species classification and detection dataset. In: 2018 IEEE/CVF conference on computer vision and pattern recognition. Salt Lake City, UT: IEEE; Jun. 2018. p. 8769–78. https://doi.org/10.1109/CVPR.2018.00914.
[35] Arjovsky M, Chintala S, Bottou L. Wasserstein Generative Adversarial Networks. In: Proceedings of the 34th International Conference on Machine Learning. PMLR; Jul. 2017. p. 214–23.
[36] Gulrajani I, Ahmed F, Arjovsky M, Dumoulin V, Courville AC. Improved training of Wasserstein GANs. In: Advances in neural information processing systems. Curran Associates, Inc.; 2017.
[37] Barber D, Agakov F. The im algorithm: a variational approach to information maximization[J]. Adv Neural Inf Proces Syst 2004;16(320):201.
[38] Ojha U, Singh KK, Hsieh C-J, Lee YJ. Elastic-InfoGAN: Unsupervised Disentangled Representation Learning in Class-Imbalanced Data. 2025.
[39] Chen Ting, et al. A simple framework for contrastive learning of visual representations. In: International conference on machine learning. PmLR; 2020.
[40] Bachman P, Hjelm RD, Buchwalter W. Learning Representations by Maximizing Mutual Information Across Views. In: Philip Bachman, Hjelm R Devon, Buchwalter William, editors. Learning representations by maximizing mutual information across views. Advances in neural information processing systems; 2019. p. 32.
[41] Vaswani A, et al. Attention is All you Need. In: Advances in Neural Information Processing Systems. Curran Associates, Inc.; 2017.
[42] Yi Kun, et al. Frequency-domain mlps are more effective learners in time series forecasting. Adv Neural Inf Proces Syst 2023;36:76656–79.
[43] Ioffe S, Szegedy C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In: Proceedings of the 32nd International Conference on Machine Learning. PMLR; Jun. 2015. p. 448–56.
[44] Hendrycks D, Gimpel K. Gaussian Error Linear Units (GELUs). Jun. 05, 2023. arXiv: arXiv:1606.08415.
[45] Foggo B, Yu N, Shi J, Gao Y. Information losses in neural classifiers from sampling. IEEE Trans Neural Netw Learning Syst Oct. 2020;31(10):4073–83. https://doi.org/10.1109/TNNLS.2019.2952029.
[46] Foggo B, Yu N. On the Maximum Mutual Information Capacity of Neural Architectures. Jun. 10, 2020. https://doi.org/10.48550/arXiv.2006.06037. arXiv: arXiv:2006.06037.
[47] van der Maaten L, Hinton G. Visualizing Data using t-SNE. J Mach Learn Res 2008;9(86):2579–605.
[48] Price P. Methods for analyzing electric load shape and its variability. In: LBNL-3713E, 985909; May 2010. https://doi.org/10.2172/985909.
[49] Obukhov A, Krasnyanskiy M. Quality assessment method for GAN based on modified metrics inception score and fréchet inception distance. In: Silhavy R, Silhavy P, Prokopova Z, editors. Software Engineering Perspectives in Intelligent Systems. vol. 1294. Cham: Springer International Publishing; 2020. p. 102–14. https://doi.org/10.1007/978-3-030-63322-6_8. in Advances in Intelligent Systems and Computing, vol. 1294.
[50] Hssayeni MD. Deep learning regression models for limited biomedical time-series data. Florida Atlantic University; 2022.