Synthetic Time-series Data Generation for Smart Grids using 3D Autoencoder GAN

Guihai Zhang and Biplab Sikdar, *Fellow, IEEE*

***Abstract*—Given the growing significance of data-driven approaches in analysis and decision-making in smart grid, the availability of diverse and representative datasets is paramount. However, challenges such as privacy concerns, data size limitations, and data quality issues have constrained the usage of real-world data. In this paper, we introduce the 3D Autoencoder Generative Adversarial Network (3DAE GAN) as a solution to generate high-resolution and multivariate synthetic time-series data capable of representing various real power consumption patterns across different households and driving data for Electric Vehicles (EVs). Beyond the conventional GAN structure, the incorporation of both the Autoencoder and 3D-convolution processes enables a more comprehensive extraction of patterns in data, thereby addressing limitations present in existing data generation methods. Evaluation results using the Pecan Street and Emobpy simulated EV dataset demonstrate that the proposed method generates synthetic data with higher similarity scores compared to existing approaches. Furthermore, downstream prediction tasks are conducted to establish the comparability between using the original data and the synthetic data, revealing no significant differences. Moreover, the risk of possible information leakage from synthetic data about original data is evaluated by performing membership inference attacks and population attacks on the prediction models that are trained with synthetic data. The robustness of the synthetic data are examined when facing FGSM attacks.**

***Index Terms*—Power Grid Data, Synthetic Data, Time Series Generation, Deep Learning, GANs, 3D Convolution**

I. Introduction

The evolution of smart grids has led to a new era of efficiency, reliability and sustainability in power systems. Central to the success of smart grids is the availability of high-quality data, particularly in tasks such as power consumption analysis. Unfortunately, data scarcity, breaches, and privacy concerns present significant challenges in acquiring and utilizing diverse real-world data. Synthetic data offers the advantage of preserving the underlying patterns and characteristics of the original, real data. The release of synthetic data mitigates privacy concerns, as the confidentiality of the original sensitive data remains safeguarded. Even in the event of a data breach involving synthetic data, the private information of users is not disclosed.

Existing synthetic data generation methods exhibit limited attention to the time-series characteristics of prolonged data durations. Some approaches primarily focus on short-term periods, resulting in inadequate capture of information and interconnections among data distributed over larger time intervals [1]. Consequently, the synthetic data fails to entirely encapsulate the inherent characteristics of the original data. Moreover, the generated synthetic data tend to be univariate, representing only a singular type of data (e.g., one house or one appliance) [2], [3]. The models are only able to handle a single feature. Therefore, repeated running of the same model are needed to generate various desired data.

To address these limitations, this paper proposes the 3D Autoencoder Generative Adversarial Network (3DAE GAN) to generate multivariate time-series data with a high resolution (i.e., per-minute granularity) without prior analysis on the data. This is achieved by implementing a 3D convolutional process to effectively handle multivariate data along the third temporal axis. Initially, the multivariate data is transformed into a 2D matrix, capturing the correlation between features. These 2D matrices are then stacked along the temporal axis (timestamps) to form a 3D structure, preparing the data for the subsequent 3D convolutional processing in the model to learn complex and underlying patterns. Based on the learnt patterns, synthetic data are generated. The synthetic data generated from the proposed method holds potential for applications in subsequent tasks such as power consumption analysis and prediction tasks. The proposed method integrates the conventional GAN with an Autoencoder. Additionally, the 3D convolution process aids in examining and storing time-series features by treating time as the third axis. To demonstrate its performance, random Gaussian noise is input to the model to produce synthetic data that encompasses the power consumption profiles for distinct houses. In addition, the privacy analysis is conducted to the synthetic data by performing membership inference attacks (MIA) on the prediction models trained with synthetic data. This is to examine whether an attacker can determine if a data record was used for the model training or not.

The primary contributions of this paper are:

- We introduce 3DAE GAN, incorporating time-series and multivariate factors to generate synthetic data, addressing challenges of data scarcity and privacy in power grid data.
- Compared to existing methods, our model generates synthetic data that more closely resembles the original data.
- We analyze the synthetic data's effectiveness for prediction tasks and find that its accuracy is comparable to the results obtained when utilizing the original data.
- We examine the privacy risk in the generated synthetic data by checking if the synthetic data can cause any information leakage in itself or in the original dataset.
- We evaluate the robustness of the synthetic against adversarial attacks.

The subsequent sections of the paper are as follows. Section II provides the literature review of related works. Section III describes the proposed model in detail. Section IV presents the evaluation results. Finally, Section V concludes our work.

II. Related Works

Traditionally, statistical methods have been used to analyze linear correlations among data and model these interconnections

through mathematical equations. In [4], a bottom-up model is adopted for analyzing occupant behavior to simulate energy consumption in residential buildings, while [5] integrates Markov Chains with non-intrusive load decomposition to replicate home energy consumption. In [6], the Markov Chain method is applied to generate synthetic data for monitoring the health of electric vehicle (EV) batteries. The synthetic data generator in [7] is based on temporal modeling of EV sessions with Gaussian mixture models. Additionally, [8] introduces a data generation method involving the simulation of power system operation based on geographical correlations and actual operational characteristics. While they provide straightforward extrapolation within linear space, these models frequently overlook the complexity and diversity inherent in various data types.

Compared to statistical models, neural network-based approaches are better at capturing complex and non-linear correlations within datasets. Notably, GANs have found extensive application in synthetic data generation tasks, such as generating load patterns and energy consumption data [9], [10]. Similarly, [11], [12] modified GANs for data augmentation in industrial piston and wind turbine data. In [13], GANs are employed to generate wind and photovoltaic power profiles. While these networks can explore and discern underlying patterns and characteristics without prior statistical analysis, challenges related to training instability and model collapse persist. The self-attention mechanism to acquire knowledge of internal time-series properties is implemented in GANs [3], facilitating the generation of data. However, it only considers data with a single feature and it cannot support more features from multivariate data. The Wasserstein GAN (WGAN) can improve training convergence and generate diverse data, utilizing the Wasserstein distance [14]. Furthermore, [15] improved model training stability and enhanced generation diversity by using added gradient penalty. In [16], WGAN-GP is used to generate synthetic attack data for Intrusion Detection Systems and demonstrate the advantages in training stability and generating diverse data. However, there is a trade-off as a WGAN-GP model may unintentionally compromise the fidelity of the generated data in terms of its similarity to the original dataset. For smart home data, [17] proposed VAE-GAN that integrates Variational Autoencoder (VAE) with GAN for generating electrical load data. However, the VAE compresses the input sequence into a Gaussian distribution, a condition that is not universally valid and may result in the loss of essential information within the data.

The contributions of similar existing work that use WGAN and VAE GAN are summarized in Table I. While these methods offer alternatives for generating synthetic data, they exhibit limitations in handling multivariate features and producing long-duration time-series data. Typically, the data is processed in a 1D manner, focusing solely on values along the temporal axis, without accounting for the correlation between features or the evolution of these correlations over time. This paper addresses these gaps by proposing a novel 3D Convolutional approach to effectively capture underlying patterns in multivariate time-series data, such as power consumption. Unlike prior work, which has not explored 3D convolution for time-series data, the proposed method leverages this technique to model feature correlations and their temporal dynamics. Furthermore, the proposed method also incorporates privacy assessments and robustness evaluations against adversarial attacks.

TABLE I
CAPABILITIES OF EXISTING SIMILAR METHODS

Method	Multi-variant	Long duration
[3]	✗	✗
[9]	✓	✗
[10]	✓	✗
[11]	✗	NA
[13]	✗	✓
[16]	✓	NA
[17]	✗	✓

III. METHODOLOGY

A. Dataset

The datasets used in this paper are individual house's daily energy consumption from Pecan Street [18] and the electric vehicle (EV) driving data from Emobpy simulation tool. For Pecan Street datasets, two different datasets are used: Pecan-A and Pecan-B. Pecan-A contains the electricity usages of random households from different cities within the time duration from July 1, 2022 to October 30, 2022. The total number of data points is 175680 with a training part of 150000 and a testing part 25680. Pecan-B only contains households consumptions from New York within the time duration from June 24, 2019 to October 31, 2019. The total number of data points is 264960 with a training part of 210000 and a testing part 54960. Both Pecan-A and Pecan-B have the granularity of one minute. For illustration purpose, the utilization of 16 households' power consumption serves as a demonstration of the proposed model's ability to effectively manage diverse features concurrently. The EV dataset, originally simulated using Emobpy, incorporates a range of driver categories and travel habits to accurately model driving behaviors for individual journeys. Driving activity data are then collected at a resolution of one second, ensuring a high level of temporal granularity. The dataset encompasses 35 features, each intricately associated with specific driving conditions, including instantaneous power consumption, road type, vehicle speed, and passenger count. The total number of data points is 180000 with a training part of 162000 and a testing part 18000. For Pecan Street dataset the input feature size is 16, and the input feature size of EV data is 35. Thus, the proposed method's generalizability can be verified, demonstrating its capability to effectively manage varying dimensions of multivariate data.

B. Data Transformation

This step is to transform the original data into 3D time-series structure for further 3D Convolutional processing. Normalization of data is done by min-max scaling for each feature. This scaling method helps to maintain consistency across features and to provide model training stability. Building on the methodology outlined in [19], Gramian Angular Fields (GAF) are employed to transform 1D feature data into images, with the aim of enhancing inter-feature correlations. GAF encodes the relationships between features at a specific timestamp by

mapping them into a matrix that captures their magnitude and angular direction within a normalized range. By converting time-series into 2D images, GAF enables the application of image-based analysis techniques, improving both computational efficiency and the model's ability to detect subtle inter-feature dependencies.

Consider a time-series data with n features with single scaled data vector $X = [x_1, x_2, ...x_n]$. It can be encoded into polar coordinates by using the angular cosine and time stamp:

$$\begin{cases} \phi = arccos(x_i), & -1 \leq x_i \leq 1, x_i \in X \\ r = \frac{s_i}{T}, & s_i \in T \end{cases} \quad (1)$$

where s_i is the time step position of each feature and T is the defined radial distance for the polar coordinates. Then, the GAF matrix is obtained by considering the trigonometric sum between points to identify the temporal correlations:

$$\text{GAF} = [cos(\phi_i + \phi_j)] = X' \cdot X - \sqrt{I - X^2}' \cdot \sqrt{I - X^2} \quad (2)$$

where the I is an unit vector. For Pecan Street datasets, the result of the GAF encoding is a square matrix of size 16×16. This matrix captures the time-series relationships among the chosen features, offering a thorough depiction of their dynamic interactions. Subsequently, the GAF square matrices from 60-minute intervals span among the third dimension to form the 3D matrix. This 3D representation facilitates 3D convolution operations, enabling the model to assimilate correlations between features and time concurrently. Therefore, the training dataset (150000×16) is divided into individual 3D matrices with a shape of $60 \times 16 \times 16$ for model training. As for the EV dataset, each data block is structured into shape of $60 \times 35 \times 35$.

The subsequent 3D convolutional process can effectively capture the spatial and temporal interconnections within the data. This approach allows the model to extract meaningful features by convolving across multiple dimensions, such as height, width, and depth (time). The 3D convolutional layers leverage these multi-dimensional structures to learn complex patterns that are not evident in normal 2D representations. In the context of time-series data, the 3D convolutional process enables the model to understand relationships across sequential time frames or layered data points, thereby enhancing its capacity to recognize subtle interactions and dependencies, to handle the temporal relationships and the interconnections between various features.

C. Model Structure

The structure of the proposed 3DAE GAN to generate synthetic data is illustrated in Figure 1. The architecture comprises of four primary components: encoder (E), decoder (D), generator (G), and discriminator (Di). The encoder transforms input data x into a latent representation h, while the decoder utilizes the latent representation h to reconstruct the original data as x'. The generator takes Gaussian noise z as input and transforms it into a latent representation h' that ideally mirrors samples from h. The discriminator assesses the authenticity of a given data point by classifying the input representations h and h' as real or fake. This training process is indicated by the black arrow. During synthetic data generation, random noise is fed into the generator, and the decoder processes the output

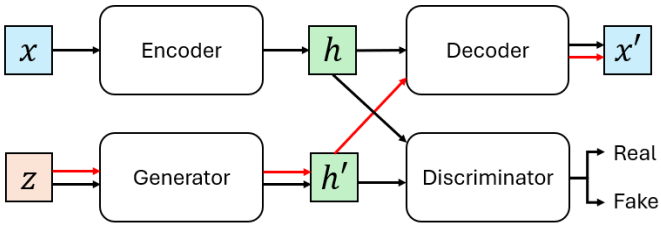


Fig. 1. 3DAE-GAN model structure. Black arrow: training flow; Red arrow: generating flow.

from the generator to produce synthetic data. This process is illustrated by the red arrow. Unlike the VAE-GAN [17], the latent representation in our method is defined as a vector of length 1024 instead of just mean and variance. This length is chosen because it is a power of 2 and is the nearest to $(60 \times 16) = 960$. This is intended to ensure the comprehensive retention of data knowledge without any tolerance for loss for the data block of 16 features along 60 minutes. Although the EV dataset used in this paper has 35 features, the same length of 1024 is applied for the hidden representation layer. Evaluation results indicate that this value serves as an effective default setting. Nevertheless, it remains adaptable and can be adjusted to suit varying circumstances when necessary.

Autoencoder is used because it is a powerful tool for learning efficient, low-dimensional latent representations of high-dimensional data. At the same time, GANs are highly effective at generating data that closely resembles the real-world distribution by leveraging a generator-discriminator framework. The generator creates synthetic samples, while the discriminator evaluates their realism. This adversarial training pushes the generator to produce highly realistic data with natural variability, enhancing the quality of synthetic samples.

Furthermore, this paper introduces a novel approach to synthetic data generation. Unlike existing VAE-GAN method, which use Gaussian noise as input to the generator to replicate the distribution of learned patterns and produce synthetic data, this approach mitigates potential privacy risks. Specifically, the reliance on Gaussian noise in traditional methods can inadvertently link the generated synthetic data to the original data, raising concerns about data confidentiality and privacy. Thus, to reduce the privacy concerns, we use the generator to produce the intermediate hidden representation and let the decoder reconstruct the data. This approach limits the direct link between the synthetic data and the original data. The generator now works on abstract, intermediate representations rather than directly on raw data, reducing the likelihood of revealing sensitive information that could be traced back to the original dataset.

The detailed layers of each component in the proposed 3DAE GAN are shown in Table II. Both encoder and decoder use 3D convolutional layers because the data has been transformed into 3D representations. Both generator and discriminator use MLP denser layers because they only need to process 1D data.

TABLE II
LAYERS OF EACH COMPONENT IN 3DAE GAN

Encoder	Decoder	Generator	Discriminator
Cov3D(64)	Dense(1920)	Dense(128)	Dense(512)
Cov3D(32)	Reshape	Dense(128)	Dense(256)
Cov3D(16)	Conv3DTranspose(16)	Dense(1024)	Dense(128)
Cov3D(8)	Conv3DTranspose(32)	Dense(2048)	Dense(128)
Flatten	Conv3DTranspose(64)	Dense(1024)	Dense(1)
Dense(1024)	Cov3D(1)	-	-

TABLE III
MODEL COMPLEXITY OF DIFFERENT METHODS

Method	WGAN-GP	VAE GAN	3DAE GAN
Generator	✓	✓	✓
Discriminator	✓	✓	✓
Encoder	-	✓	✓
Decoder	-	-	✓
Training complexity	$O(n \cdot L \cdot H^2)$	$O(n \cdot L \cdot H^2)$	$O(n \cdot L \cdot H^2)$

D. Training Losses

The custom losses based on the function of each part of the model structure are defined below for the training process.

1) Reconstruction: The input x is fed into the encoder, where it undergoes complex learning to generate hidden representations. These hidden representations are then passed through the decoder to reconstruct x as x'. The reconstruction loss L_r is obtained by checking the distance between the reconstructed x' from decoder output and input x of encoder. It allows the encoder and decoder to converge to an optimal point where the reconstructed x' is the same as input x. The loss is the mean squared error:

$$L_r = \|x - x'\|^2 . \tag{3}$$

This is the loss to be led back to the encoder and the decoder to ensure that both of them cooperate to learn hidden representations and to reconstruct the output x' that is close to input x. Similarly, when there is input z into the generator to produce fake hidden representation h', the distance between the real latent feature h from real input x and the generated h' is the divergence loss L_d. This loss prompts the generator to produce the same latent representation as the real input. This loss is also the squared mean error:

$$L_d = \|h - h'\|^2 . \tag{4}$$

This loss helps the encoder and generator to move toward a space where both of real and fake hidden representations are similar to each other.

2) GAN classification: Besides computing the losses above, the ideal generator generates fake h' from input z and this h' should be classified as a real sequence by the discriminator. The classification of predicted real sequence is "1" and predicted false sequence is "0". The generator should produce outputs that minimize the L_s. The objective is to let the discriminator think that the generated h' actually comes from the real data x. Hence, we have the synthetic loss using binary cross-entropy loss that prompts the optimizing of the generator:

$$L_s = E_h[log(D_i(h'))]. \tag{5}$$

The ideal discriminator needs to classify the real input sequence h from x as real and thus has real prediction loss L_{real}. At the same time, it should classify the fake sequence h' from z as fake and thus has fake loss L_{fake}. The two losses also use binary cross-entropy loss. The discriminator should minimize both the real loss and fake loss. Those two losses prompt the discriminator to converge:

$$L_{real} = E_h[log(D_i(h))], \tag{6}$$

$$L_{fake} = E_{h'}[1 - log(D_i(h'))]. \tag{7}$$

3) Simultaneous Training: All four components are connected with other and thus they are updated at the same time. Each loss plays a critical role during training. As described in the previous section, in the training phase, there are two types of inputs to models: the real input data x and the random noise z. Real input data x goes into the encoder to get real hidden h. Random noise z goes into the generator to produce fake hidden h'. Thus the L_d is obtained. At the same time, the decoder reconstructs x' from the real hidden h. Thus, the reconstruction loss L_r is computed. Then, for the produced real hidden h and fake hidden h', the loss L_s for generator, L_{real} and L_{fake} for discriminator are obtained. When all the losses are ready in a single round, backpropagation is initiated to update the model parameters.

The crucial objective is to ensure the active involvement of both the encoder and generator in the divergence loss L_d. Consequently, the following are the losses assigned to each component during the backpropagation process:

$$E \quad loss = \alpha \cdot L_d + L_r, \tag{8}$$

$$D \quad loss = L_r, \tag{9}$$

$$G \quad loss = \beta \cdot L_d + L_s, \tag{10}$$

$$Di \quad loss = L_{real} + L_{fake}. \tag{11}$$

The variables α and β serve as factors to regulate the significance of aligning the latent representations for the generator and the encoder. In this paper, both α and β are set to 0.1. At the same time, the decoder and the discriminator should work together to reconstruct data and to classify hidden representations to provide real time feedback to the encoder and the generator. Thus, all the four parts are trained at the same time to allow the dynamic training and interactions among all of them. After all the losses are computed, the optimizer updates each component at the same time to achieve simultaneous training

E. Model Complexity

It is noted that the proposed framework is more complex than the WGAN-GP [15] and VAE-GAN [17]. The details are listed in Table III. The proposed 3DAE-GAN has the most number of parts: encoder, decoder, generator and discriminator. Although the training complexity of neural networks are not straightforward to obtain, it can be represented by a general big-O notation $O(L \cdot H^2)$ where L represents the number of layers and H represents the number hidden neurons in a layer. For a simple comparison, let the number of layers and neurons be the same for all the encoders, decoders, generators and discriminators. The overall training complexity of the proposed method is $O(n \cdot L \cdot H^2)$, where n is the number of components.

TABLE IV
HYPERPARAMETERS EXPLORATION PRELIMINARY RESULTS

Hyperparameters	Cosine Similarity	Duration
Default $h = 1024$	0.8722	93s
$h = 512$	0.8714	93s
$h = 2048$	0.8710	94s
Default no. of neurons	0.8722	93s
halved neurons	0.8670	92s
doubled neurons	0.8714	95s
Default no. of layers	0.8722	93s
Reduced layers	0.8704	87s
Increased layers	0.8729	102s

F. Hyperparameters

Before conducting the final evaluation of each method, several settings were explored to determine the optimal configuration. The most critical hyperparameters include the size of the hidden latent representation (h), the number of neurons per layer, and the total number of layers. Preliminary experiments were performed to evaluate the impact of each hyperparameter individually while keeping all other factors constant at their default values. These initial results were used to guide the selection of the final values for the hyperparameters. All default settings are shown in Table II.

The default number of the latent representation h is selected to be 1024. We also check the results when the value is halved and doubled. The number of neurons in each layer of all components is also halved and doubled to examine the influences in this exploration. For the number of layers in each component, the layers are reduced by one and increased by one, respectively. The overall findings are shown in Table IV. The cosine similarity of synthetic data to original data and the time needed for each epoch is recorded during the experiments. From the comparison, we note that the choice of the hyperparameters does not hugely affect the results. The cosine similarity for all cases are around 0.87. It is noted that by decreasing the neurons in each layer and the number of layers, the performance decreases, but with the benefits of less training time. The best cosine similarity is obtained when the network layers are increased. However, the training time increases to 102 seconds. Therefore, for the remaining evaluations, the default settings are used. It does not cause any appreciable decrease in cosine similarity and keeps the training time to be acceptable. These settings may be adapted for future tasks and datasets as needed. For training the proposed 3DAE GAN, the batch-size is 4 and the optimizer is Adam with learning rate of 0.0001.

IV. RESULTS

The performance of the proposed 3DAE GAN model is evaluated using both similarity metrics and functionality checks. The proposed method is compared with existing WGAN-GP and VAEGAN to illustrate its advantages. Furthermore, privacy analysis and adversarial robustness of the generated synthetic data are evaluated to provide more comprehensive understanding. All experiments were run on a NVIDIA A100-SXM4-80GB with CUDA version of 12.2 and cuDNN version 8. Programs and scripts are using TensorFlow version of 2.13.0.

The similarity scores used are cosine similarity, Jensen-Shannon distance, and Euclidean distance. Cosine similarity evaluates the alignment between points by measuring the angle between them, determining if they are oriented in the same direction. In contrast, Jensen-Shannon distance treats each point as a distinct distribution, enabling an evaluation of their resemblance in a symmetric way. Euclidean distance quantifies the shortest distance between two points, assessing their proximity. The cosine similarity and Euclidean distance are given by:

$$\text{Cosine Similarity}(P,Q) = \frac{Q \cdot Q}{\|P\|\|Q\|}, \tag{12}$$

$$\text{Euclidean distance}(P,Q) = \left[\sum_{i=1}^{n} (Q_i - P_i)^2\right]^{1/2}. \tag{13}$$

The Jensen-Shannon distance between the original data P and synthetic data Q is given by:

$$\text{Divergence}(P\|Q) = \frac{1}{2}(\text{KLD}(P\|M) + \text{KLD}(Q\|M)), \tag{14}$$

$$\text{Jensen-Shannon distance}(P,Q) = \sqrt{\text{Divergence}(P\|Q)}, \tag{15}$$

where KLD() represents the Kullback-Leibler divergence and $M = (P + Q)/2$.

For the functional assessment, we test how well the real and synthetic datasets perform in downstream tasks. For Pecan Street datasets, we perform the consumption prediction using a typical LSTM model. The original and synthetic power consumption (generated from our method and existing methods) of House ID 1 for the both Pecan-A and Pecan-B datasets are used. A sliding window of 30 minutes is used to predict the next consumption. LSTM is chosen because it is a popular deep learning model that has proven to be effective in capturing temporal dependencies and making accurate predictions [20]. The structure of the LSTM model used is shown in Table V. As the EV dataset is a type of time-series data with multiple features, we adopt a different prediction approach. The instantaneous power consumption feature is selected as the prediction target, while the remaining 33 features, such as EV speed, road conditions, and acceleration, serve as inputs. The prediction model is a 3D convolutional model from Alex-net as shown in Table VI.

TABLE V
LSTM PREDICTION MODEL LAYERS

Layer	Details
1	LSTM(64)
2	LSTM(128)
3	LSTM(64)
4	Dropout(0.2)
5	Dense(1)

The same prediction model settings are used for all the four datasets (original, 3DAE-GAN generated, WGAN-GP generated and VAE-GAN generated). Thus, there are 4 unique prediction models after training. Subsequently, these models are validated on the same test dataset (original data) prepared before synthetic data generation. The Mean Absolute Percentage Error (MAPE) of each method is used to compare the prediction performance:

$$\text{MAPE} = \frac{100\%}{N} \sum_{i=0}^{N-1} \frac{y_i - y_i'}{y_i}, \tag{16}$$

TABLE VI
3D CONVOLUTIONAL PREDICTION MODEL LAYERS

Layer	Details
1	Cov3D(256)
2	Cov3D(384)
3	Cov3D(384)
4	Cov3D(256)
5	Flatten()
6	Dense(6220)
7	Dropout(0.5)
8	Dense(1)

TABLE VII
SETTINGS TO PERFORM MEMBERSHIP INFERENCE ATTACK

Settings	$S1$	$S2$	$S3$
Target model dataset	Original	Synthetic	Synthetic
Shadow model dataset	Original	Synthetic	Synthetic
No. of shadow models	10	10	10
Training (Member) data size	10000	10000	10000
Testing (Non-member) data size	10000	10000	10000
Evaluation dataset	Original	Synthetic	Original

where y is the true value and y' is the predicted value.

To evaluate the level of privacy protection provided by the generated synthetic data, we perform the membership inference attack (MIA) and population attack by using the open-source tool Privacy Meter [21]–[24]. For both membership inference attack and population attack evaluation, the target models are the prediction models using the same setting as described in Table V and Table VI. For membership inference attack, the known data records that are used in model training are member data and any other data records are non-member data. When an attacker can use the knowledge gained from shadow models to correctly classify the member data in the target model's training dataset, we define that the shadow dataset has the risk of information leakage. In this evaluation, three types of checks are designed: original data self-leakage ($S1$), synthetic data self-leakage ($S2$) and cross-leakage ($S3$). In $S1$, both the target and shadow models use the original dataset, and the evaluation dataset is also from original dataset. In $S2$, both the target and shadow models use the synthetic dataset, and the evaluation dataset is also from synthetic dataset. In $S3$, both the target and shadow models use the synthetic dataset, but the evaluation dataset is from original dataset. This cross-leakage ($S3$) check is to examine the extent of privacy leakage when attackers can use information about synthetic data to identify any obtained original data to be member or non-member. The settings of the checks are summarized in Table VII. The member set and the non-member set of each target and shadow model are randomly selected from the respective datasets and are disjoint. The equal numbers of member and non-member make the inference attack to have no bias. The target model data and the shadow model data have no overlap. The data used for each shadow model is allowed to have partial overlap. The evaluation dataset does not overlap with either target dataset nor shadow dataset. The privacy information leakage is examined by the success rate of the membership inference attack.

For population attack, shadow models and not needed and a single target model is trained using the training data. The attack

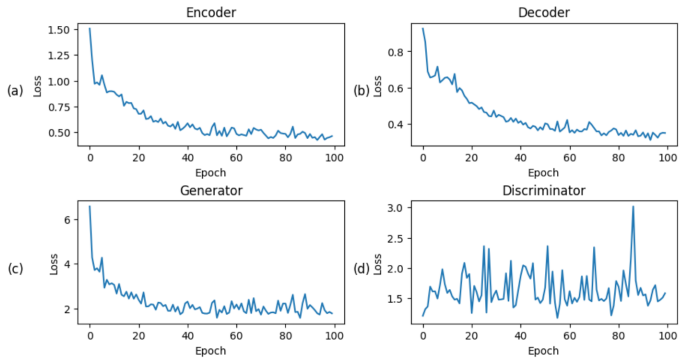


Fig. 2. Losses during the training of 3DAE GAN.

decision relies on statistical properties of the model outputs (e.g., average confidence, loss distribution) for a set of queries corresponding to the population. The attacker can query the model with a sufficiently large set of inputs representing the population of interest. Model outputs reveal patterns that can differentiate between trained and untrained populations.

The robustness of the models trained by synthetic data is evaluated using Fast Gradient Signed Method (FGSM) [25]. After the prediction model is trained using synthetic data, original testing data are modified by the FGSM method to obtain adversarial input. Then the new prediction results are collected to check how well does the prediction model handle inputs with adversarial perturbations. The adversarial data is:

$$x_{adv} = x + \epsilon \cdot \text{sign}(\nabla(L(x, y))), \tag{17}$$

where x in the normal input and y is the true value of x. $L(x, y)$ is the loss function used by the model during training, which quantifies the difference between the model's predictions for input x and true value y. This attack method is a single step to modify the input and we set ϵ to be 0.1 and 0.01.

A. Model Training

The loss trend of each component during the training process when using Pecan-A is shown in Figure 2 for the illustration of how each component is updated through the process. It can be seen that the generator's loss consistently decreases and stabilizes throughout the training process, indicating a convergence where the generated latent vector aligns with that of real input data. Conversely, the discriminator initially experiences a small loss that gradually increases. This behavior aligns with expectations, as the generator and discriminator are in a continuous feedback loop, each trying to outperform the other. Nevertheless, the discriminator loss oscillates at 1.6, indicating its relative stability throughout the ongoing adversarial training process. Overall, these plots help to confirm that the designed training process is achieved and each component has converged while interacting with each other.

B. Synthetic Data Generation

Due to space constraints, the synthetic consumption patterns of 4 households randomly selected from the 16 houses in Pecan-A are shown in Figure 3. An analysis of the average values for each house in both the Pecan-A dataset original and

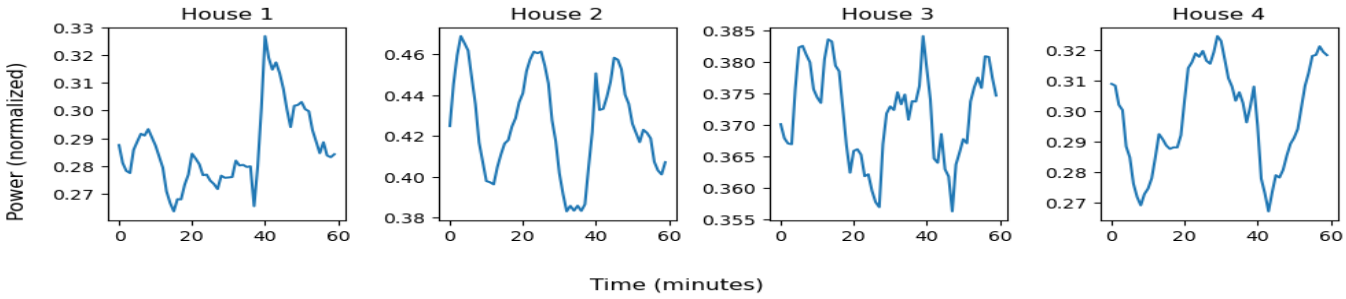


Fig. 3. 3DAE GAN generated synthetic power consumption for a house.

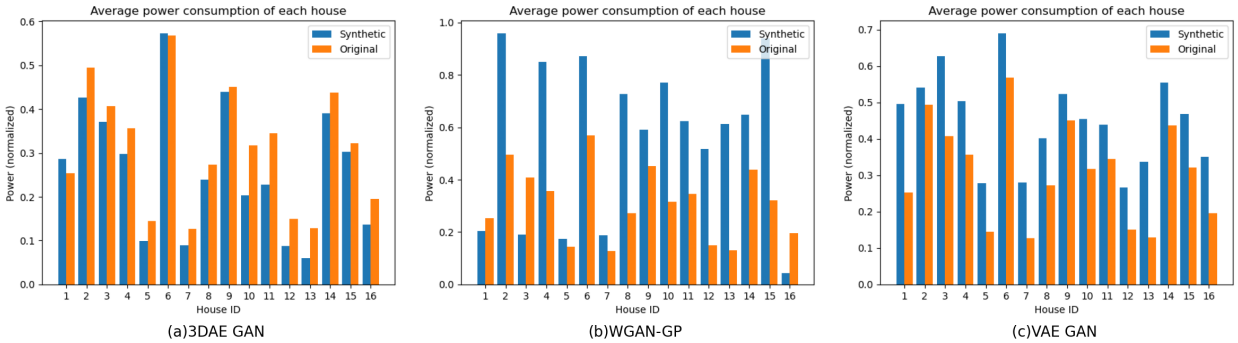


Fig. 4. Average consumption value from generated synthetic data for Pecan-A dataset.

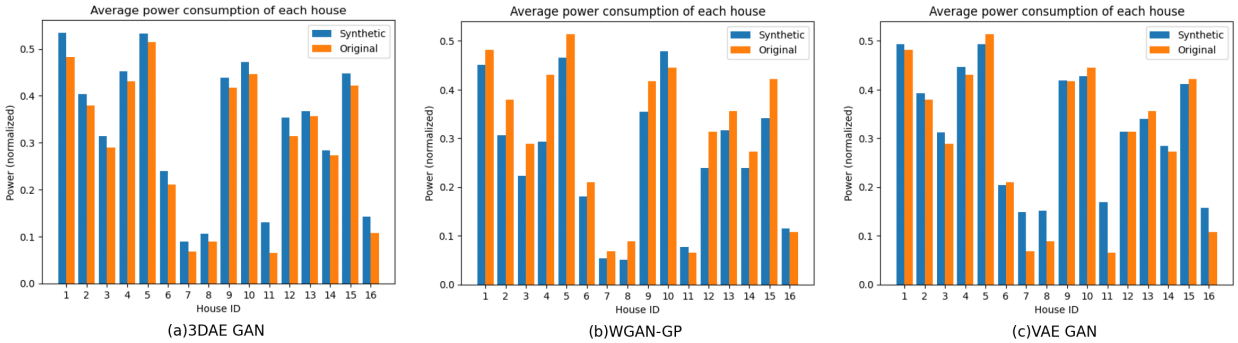


Fig. 5. Average consumption value from generated synthetic data for Pecan-B dataset.

TABLE VIII
AVERAGE MEAN ABSOLUTE ERROR IN AVERAGE POWER CONSUMPTIONS

Method	Pecan-A	Pecan-B	EV
WGAN-GP	0.2974	0.04878	0.04161
VAE GAN	0.1399	0.02778	**0.02938**
3DAE GAN (ours)	**0.0512**	**0.02772**	0.03550

synthetic datasets is presented in Figure 4 for the proposed 3DAE GAN, WGAN-GP and VAE GAN, respectively. Notably, the difference between our synthetic data and the original data for each house is considerably smaller than the differences between other synthetic datasets and the original data. Figure 5 shows the average values for Pecan-B original and synthetic data. It can be seen that our method performs well and has similar average value as original data. Notably, both WGAN-GP and VAE GAN methods also perform better for this Pecan-B dataset. There are no significant differences between original and synthetic means as compared to the results for Pecan-A.

To quantify the comparison between the average consumption, the average mean absolute error (MAE) is computed for each method for both Pecan-A and Pecan-B datasets. The results are shown in Table VIII. Our 3DAE GAN method produces synthetic data with the lowest average MAE for both datasets. This underscores our method's capability to generate synthetic data that closely aligns with the values of the original data across all households.

Despite not including the average value results of EV dataset in figures due to space constraints, we have summarized the average MAE in Table VIII. It is noted that the VAE GAN method performs better than our method in this dataset. However, this average MAE metric reflects only the magnitude comparison between the original and synthetic data, providing limited insight into pattern similarity. While the VAE GAN

TABLE IX
SIMILARITY SCORES BETWEEN REAL AND SYNTHETIC DATA FOR PECAN-A DATASET

Method	Cosine Similarity	Jensen-Shannon Distance	Euclidean Distance
WGAN-GP	0.723	0.322	171.758
VAE GAN	0.794	0.259	141.073
3DAE GAN (ours)	**0.859**	**0.204**	**62.169**

TABLE X
SIMILARITY SCORES BETWEEN REAL AND SYNTHETIC DATA FOR PECAN-B DATASET

Method	Cosine Similarity	Jensen-Shannon Distance	Euclidean Distance
WGAN-GP	0.819	0.236	76.472
VAE GAN	0.838	0.210	72.662
3DAE GAN (ours)	**0.878**	**0.175**	**55.502**

effectively reconstructs magnitudes close to the original EV data, it struggles to learn the hidden and complex trends. In contrast, our method excels at modeling these intricate patterns, making it more suitable for applications that require a deeper understanding of the underlying data relationships.

C. Similarity Scores

1) Pecan-A dataset: Table IX compares the similarity scores associated with synthetic data generated by the proposed method and existing approaches for the Pecan-A dataset. Notably, our method demonstrates superior performance across all evaluated metrics. For Jensen-Shannon and Euclidean distances, smaller values signify heightened similarity between vectors. Our approach yields the lowest Jensen-Shannon and Euclidian distances.

2) Pecan-B dataset: Table X shows the similarity scores of synthetic and original data of Pecan-B dataset. Our method achieves the best score for all the metrics. It is also noted that both WGAN-GP and VAE GAN have better performance on this dataset as compared to Pecan-A. This difference in performance can be explained by the nature of the data in Pecan-B, which comes from a single region. In contrast, Pecan-A includes data from multiple regions, introducing greater variability in household consumption patterns. Thus, the relatively homogenous nature of Pecan-B allows for more accurate modeling, contributing to the enhanced performance of these models compared to Pecan-A, where the increased complexity due to multi-region data may make it more difficult for the models to generalize effectively.

3) EV dataset: Table XI compares the similarity scores for EV dataset. Across all metrics, 3DAE GAN outperforms the other two models, achieving the highest cosine similarity, the lowest Jensen-Shannon distance, and the smallest Euclidean distance, indicating that its synthetic data is the most similar to the real data in both distribution and feature space.

Overall, these results demonstrate that 3DAE GAN is the most effective model in generating synthetic data that closely resembles the original data in multiple dimensions and in different datasets.

TABLE XI
SIMILARITY SCORES BETWEEN REAL AND SYNTHETIC DATA FOR EV DATASET

Method	Cosine Similarity	Jensen-Shannon Distance	Euclidean Distance
WGAN-GP	0.692	0.419	99.635
VAE GAN	0.810	0.349	84.929
3DAE GAN (ours)	**0.858**	**0.280**	**74.845**

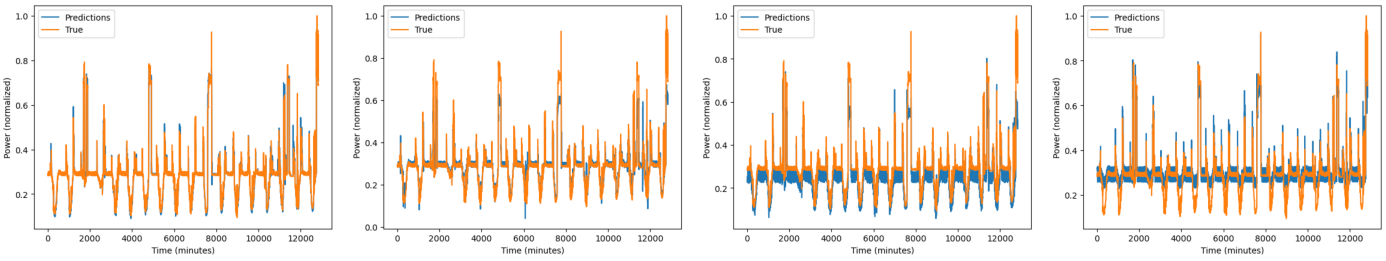
TABLE XII
PREDICTION MAPE USING DIFFERENT DATA FOR PECAN-A DATASET

Data	Normal	FGSM 0.01	FGSM 0.1
Original	5.121	10.539	52.568
WGAN-GP	14.638	28.938	54.373
VAE GAN	21.175	24.742	76.342
3DAE GAN (ours)	6.667	16.678	82.755

D. Functionality Performance

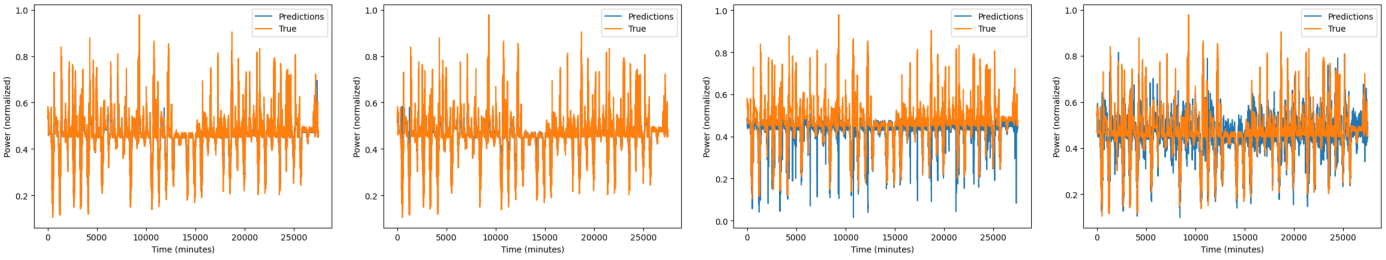
1) Pecan-A dataset: Table XII summarizes the MAPE of the prediction performance when using the Pecan-A dataset. When utilizing the original data, the MAPE is 5.121. Conversely, when using synthetic data generated from our 3DAE GAN method, the MAPE is 6.667, slightly larger than that of the original data. These results demonstrate that the synthetic data exhibits similar functionality to the original dataset in when applied to various tasks. On the other hand, the WGAN-GP and VAE GAN generated data produce large MAPE values of 14.638 and 21.175, respectively, indicating that these data would lead to larger errors in subsequent tasks. Additionally, plots of the real test data and the predicted values are depicted in Figure 6 for visualization. This visual comparison of the plots reveals that the prediction model based on our 3DAE GAN-generated dataset yields the smallest error when compared with the results from WGAN-GP and VAE GAN methods. Moreover, it can be seen that although our method has similar MAPE in normal scenarios, it performs a bit worse when there are FGSM attacks. This could be attributed to the fact that our method is well-trained to capture the patterns present in the original training data. When adversarial modifications are introduced to the original test dataset, the underlying patterns in the data are altered, creating a mismatch between the adversarial test data and the synthetic data generated by 3DAE GAN. Nonetheless, our method achieves the lowest MAPE compared to WGAN-GP and VAE GAN methods under smaller adversarial perturbations, specifically when $\epsilon = 0.01$.

2) Pecan-B dataset: Table XIII summarizes the MAPE of the prediction performance when using the Pecan-B dataset. For original data, the normal MAPE is 3.291. For our method's generated data, the normal MAPE is 5.653, which is slightly larger than that of original data. However, it is the smallest among the synthetic data. The plots of the prediction performance are shown in Figure 7. The synthetic Pecan-B data generated by 3DAE GAN demonstrates performance comparable to that of the original dataset, whereas the WGAN-GP and VAE GAN methods exhibit notably poorer results. When facing FGSM attacks, our method's generated data achieves comparable performance as the original data with lower MAPE in comparison



(a) Original (b)3DAE GAN (c)WGAN-GP (d)VAE GAN

Fig. 6. Prediction model based on different data for Pecan-A dataset.



(a) Original (b)3DAE GAN (c)WGAN-GP (d)VAE GAN

Fig. 7. Prediction model based on different data for Pecan-B dataset.

TABLE XIII
PREDICTION MAPE USING DIFFERENT DATA FOR PECAN-B DATASET

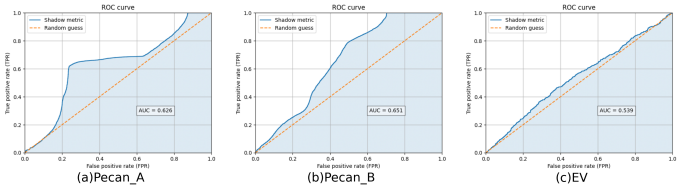
Data	Normal	FGSM 0.01	FGSM 0.1
Original	3.291	5.527	25.613
WGAN-GP	7.831	13.157	63.352
VAE GAN	7.552	19.563	52.879
3DAE GAN (ours)	5.653	8.532	28.619

TABLE XIV
PREDICTION MAPE USING DIFFERENT DATA FOR EV DATASET

Data	Normal	FGSM 0.01	FGSM 0.1
Original	9.117	12.454	50.335
WGAN-GP	18.211	22.492	42.547
VAE GAN	14.234	15.896	25.737
3DAE GAN (ours)	9.374	11.928	14.414

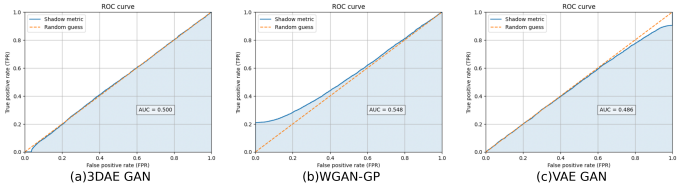
with existing WGAN-GP and VAE GAN methods. Previously, it was observed that WGAN-GP and VAE GAN methods generate more similar synthetic data for Pecan-B dataset than Pecan-A dataset. Thus their synthetic data used for this functionality evaluation also provides better prediction MAPE. However, they are more vulnerable to adversarial FGSM attacks. Since the smaller variations in household consumption within a single region in Pecan-B makes it easier for models like WGAN-GP and VAE-GAN to capture the underlying patterns, they are more prone to overfit to the original training data. Therefore, the data generated from those two methods are more vulnerable to the adversarial perturbations in test data.

3) EV dataset: MAPE when making predictions to get power consumption based on driving information using EV dataset is shown in Table XIV. Under the normal, unperturbed setting, 3DAE GAN synthetic data achieves a MAPE of 9.374, which is very close to the original data's MAPE of 9.117, indicating that



(a)Pecan_A (b)Pecan_B (c)EV

Fig. 8. Membership inference attack ROC curves of self-leakage using original data for generating datasets: (a) Pecan-A, (b) Pecan-B, (c) EV.



(a)3DAE GAN (b)WGAN-GP (c)VAE GAN

Fig. 9. Pecan-A: membership inference attack ROC curves of self-leakage using synthetic data from different generating method: (a) 3DAE GAN, (b) WGAN-GP, (c) VAE GAN.

it generates synthetic data that closely mirrors the real EV data. However, as adversarial perturbations increase, the performance of all models degrades, but 3DAE GAN shows the smallest increase in MAPE. The smallest MAPE using our method when under FGSM attacks provides an insight that in certain cases, synthetic data could play a valuable role in defending against adversarial FGSM attacks, potentially offering more robust predictions than real data for specific types of datasets.

The performance of the 3DAE GAN method is consistently strong across various datasets and conditions, demonstrating its effectiveness in generating synthetic data that closely resembles real-world data while maintaining the similar functionality as

TABLE XV
MEMBERSHIP INFERENCE ATTACK ACCURACIES OF ALL SETTINGS

Settings	Original Self-leakage ($S1$)	Synthetic Self-leakage($S2$)			Cross-leakage($S3$)		
Shadow data type	Original	3DAE GAN	WGAN-GP	VAE GAN	3DAE GAN	WGAN-GP	VAE GAN
Evaluation data type	Original	3DAE GAN	WGAN-GP	VAE GAN	Original	Original	Original
Pecan-A	0.5048	0.4998	0.4331	0.4988	0.5	0.5068	0.4959
Pecan-B	0.5468	0.4883	0.5031	0.4975	0.4	0.5143	0.5941
EV	0.5185	0.5065	0.552	0.5025	0.5015	0.5005	0.5

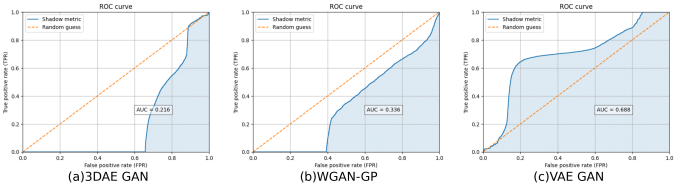


Fig. 10. Pecan-A: membership inference attack ROC curves of cross-leakage using synthetic data from different generating method: (a) 3DAE GAN, (b) WGAN-GP, (c) VAE GAN.

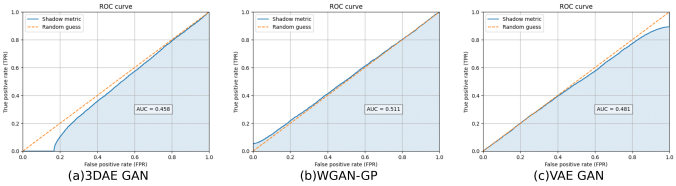


Fig. 11. Pecan-B: membership inference attack ROC curves of self-leakage using synthetic data from different generating method: (a) 3DAE GAN, (b) WGAN-GP, (c) VAE GAN.

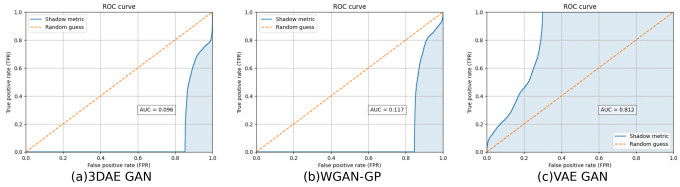


Fig. 12. Pecan-B: membership inference attack ROC curves of cross-leakage using synthetic data from different generating method: (a) 3DAE GAN, (b) WGAN-GP, (c) VAE GAN.

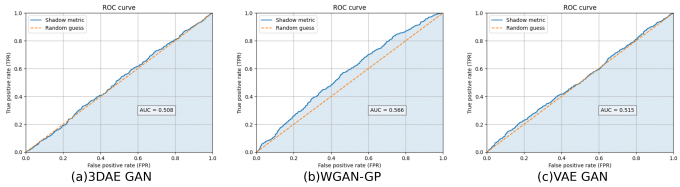


Fig. 13. EV: membership inference attack ROC curves of self-leakage using synthetic data from different generating method: (a) 3DAE GAN, (b) WGAN-GP, (c) VAE GAN.

the original dataset. The robustness against adversarial attack is task specific and depends on the real scenarios to deal with.

E. Membership Inference Attack

The classification accuracies of membership inference attacks (MIA) for all three datasets are shown in Table XV. It can be seen that all accuracies are around 0.5 and this shows that the membership inference attacks of all settings perform no better or just slightly better than random guessing. However, even if the accuracy of membership inference attacks is low, there could still be some residual risk of privacy leakage. The synthetic data might inadvertently reveal patterns or characteristics of the original data that could be exploited. Receiver Operating Characteristic (ROC) curves are plotted out to provide comprehensive analysis about the privacy link between original and synthetic data. The Area Under the Curve (AUC) is the area under the ROC curve and it measures the overall ability of the attacker to distinguish between the positive (member) and negative (non-member) classes. It represents the likelihood that a randomly chosen positive instance is ranked higher than a randomly chosen negative instance by the classifier. The plots in Figure 8 show the ROC curve and its AUC for self-leakage check using original data for Pecan-A, Pecan-B and EV datasets, respectively. The AUC of 0.626 for Pecan-A and AUC of 0.651 for Pecan-B show that the attacker has a slightly higher chance to identify a data record to be a member of target model rather than non-member data among all threshold settings. The EV dataset has an AUC of 0.539 which shows that the chances of risk in membership inference attack is similar to random guessing.

1) Pecan-A: Figure 9 shows the ROC and AUC for self-leakage check using different synthetic dataset. It is seen that the attack AUC of 3DAE GAN synthetic data is 0.5, AUC of WGAN-GP synthetic data is 0.548 and the AUC of VAE GAN synthetic data is 0.486. All of them are around 0.5 and lower than that of the original self-leakage check. It means that the attacker does not have high probability to identify a data point to be a member of target model if any parts of the whole data population are known to public. It shows that by having the knowledge of some synthetic data, there is not much privacy leakage to the other data points in the same synthetic data population. The ROC plots of the cross-leakage check using synthetic dataset to make inference conclusions on original data are plotted in Figure 10. This evaluation helps to examine the extent of privacy leakage to the original data while having knowledge of the synthetic data that are generated from the original data. Low AUC in this cross-leakage shows that the attacker cannot directly get the same inference conclusion on original dataset when the attacker has the conclusion on synthetic dataset. Thus, there is little or no privacy leakage in the 3DAE GAN and WGAN-GP generated synthetic data. However, the higher AUC from the VAE GAN synthetic data suggests that this synthetic dataset can lead the attacker to find more member data in original data (the true positives) with higher

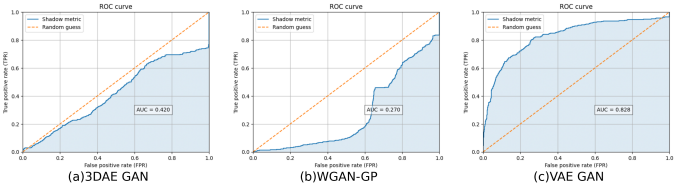


Fig. 14. EV: membership inference attack ROC curves of cross-leakage using synthetic data from different generating method: (a) 3DAE GAN, (b) WGAN-GP, (c) VAE GAN.

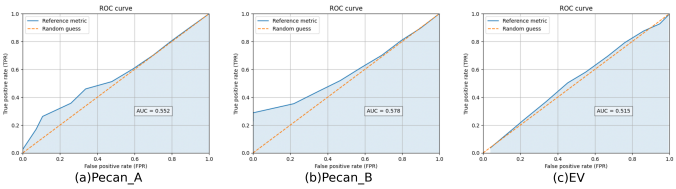


Fig. 15. Population attack ROC curves of original datasets: (a) Pecan-A, (b) Pecan-B, (c) EV.

confidence. This result can be explained because the VAE GAN method forces the original data to form a normal distribution and generates synthetic data from this distribution. This process compresses the information into a smaller boundary and thus makes both original and synthetic data to be more compactly represented within a reduced dimensional space. However, this compression also brings privacy leakage more easily.

2) Pecan-B: The self-leakage check of MIA for Pecan-B dataset is shown in Figure 11. The attack AUC of 3DAE GAN synthetic data is 0.458, AUC of WGAN-GP synthetic data is 0.511 and the AUC of VAE GAN synthetic data is 0.481. All of them are also smaller than that of the self-leakage check using original data. This shows that by having the knowledge of some synthetic data, there is not much privacy leakage to the other data points in the same synthetic data population. Figure 12 shows the cross-leakage check for MIA ROC. The AUC when using 3DAE GAN generated synthetic data is 0.096 and the AUC is 0.117 when using WGAN-GP generated synthetic data. These low AUC values suggest that there is little or no privacy leakage in the 3DAE GAN and WGAN-GP generated synthetic data. However, the higher AUC of 0.812 from the VAE GAN synthetic data shows there is higher risk to reveal the original data information from the synthetic data.

3) EV: The self-leakage check of MIA for EV dataset is shown in Figure 13. Similar to the two Pecan Street datasets, the scores of AUC for synthetic data self-leakage are also around 0.5. However, the AUC of 0.566 from WGAN-GP synthetic data is slightly larger the value of 0.539 of the original dataset. This shows that it has a little higher chance to reveal its self information. For both our 3DAE GAN and VAE GAN, the AUC is smaller as 0.508 and 0.515, respectively. Nevertheless, all three types of data provide chances like random guessing to MIA attackers. The cross-leakage check of MIA for EV dataset is shown in Figure 14. The plots show that 3DAE GAN and WGAN-GP have smaller chance to reveal to the information of original EV data, because their AUC values are much smaller than the VAE GAN method.

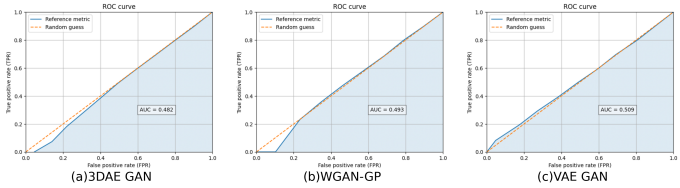


Fig. 16. Pecan-A: population attack ROC curves of cross-leakage using synthetic data from different generating method: (a) 3DAE GAN, (b) WGAN-GP, (c) VAE GAN.

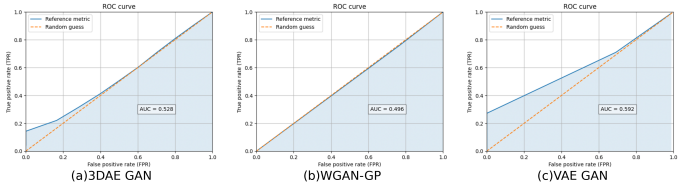


Fig. 17. Pecan-B: population attack ROC curves of cross-leakage using synthetic data from different generating method: (a) 3DAE GAN, (b) WGAN-GP, (c) VAE GAN.

F. Population Attack

Population attack decides whether a given data point belongs to the same population as the target data. This is a different type of privacy attack compared to the membership inference attack, but it also examines the information correlation between different data points. The results of population attack ROC for each original dataset is shown in Figure 15. The AUC of Pecan-A under population attack is 0.552, the AUC of Pecan-B is 0.578 and the AUC of EV dataset is 0.515. All of them are slightly larger than 0.5, showing slightly better performance than random guessing for attackers.

1) Pecan-A: The results of population attack on synthetic Pecan-A data are shown in Figure 16. For synthetic data from 3DAE GAN method, the AUC is 0.482. The AUC is 0.493 for WGAN-GP synthetic data and 0.509 for VAE GAN synthetic data. It is noted those results are also smaller than the AUC when attacking original Pecan-A dataset as shown in Figure 15. Thus, synthetic data increases the privacy protection. Moreover, they are a bit larger than the AUC when performing self-leakage MIA attacks. This is expected because to conclude whether a data point belongs to a certain population is easier than to decide if this data point is linked with another particular data point.

2) Pecan-B: The population attack AUC for Pecan-B synthetic data is shown in Figure 17. The AUC scores are 0.528, 0.496 and 0.592 for 3DAE GAN, WGAN-GP and VAE GAN, respectively. Similarly to Pecan-A dataset, both our 3DAE GAN and WGAN-GP methods have smaller AUC than that of the original dataset. However, the VAE GAN synthetic data has a larger value than original AUC. It follows the findings in previous sections that VAE GAN method contains more correlation in a compact hidden space to the original data, making the synthetic data have higher privacy risk.

3) EV: The results of AUC for population attack on EV synthetic data are shown in Figure 18. In this case, the 3DAE

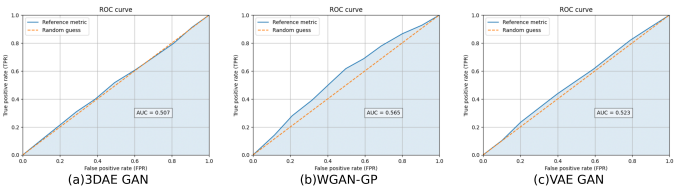


Fig. 18. EV: population attack ROC curves of cross-leakage using synthetic data from different generating method: (a) 3DAE GAN, (b) WGAN-GP, (c) VAE GAN.

GAN synthetic data is the only one to have lower AUC than the original EV dataset. The original AUC as in Figure 15 is 0.515 and our 3DAE GAN has AUC of 0.507. For WGAN-GP and VAE GAN synthetic data, the AUC scores are 0.565 and 0.523,respectively. These results also comply with previous findings that both WGAN-GP and VAE GAN methods have potentially higher risk of disclosing original data information. Nevertheless, all methods' AUC are close to 0.5 that is marginally equal to random guessing.

In summary, from the evaluations among different types of dataset, 3DAE GAN method has better performance than the existing WGAN-GP and VAE GAN in data similarity, utility, and privacy leakage protection in most cases.

V. Conclusion

This paper introduced the 3DAE GAN model as a novel approach for generating synthetic data for power systems. The primary objective is to not only uphold security and privacy standards but also to preserve the inherent insights and functionalities of the original dataset. Our proposed method integrates an autoencoder with a conventional GAN to handle multivariate time-series data in a 3D process, effectively uncovering hidden interconnections between various features and their temporal patterns. Statistical results indicate that our approach successfully generates power consumption data with maintained statistical coherence compared to the original datasets, outperforming benchmarks such as WGAN-GP and VAE GAN. Furthermore, the synthetic data exhibits similar functionalities in tasks like predicting energy consumption using LSTM model. In addition, the absence of privacy leakage check was demonstrated by conducting membership inference attacks and population attacks. The robustness of the prediction models trained on synthetic data is also evaluated against the typical FGSM adversarial attack. All these experiments contribute to a comprehensive understanding of the proposed method.

VI. Acknowledgement

This work was supported in part by the Asian Institute of Digital Finance (AIDF) under grant A-0003504-09-00.

References

[1] E. Brophy, Z. Wang, Q. She, and T. Ward, "Generative adversarial networks in time series: A systematic literature review," *ACM Comput. Surv.*, vol. 55, no. 10, feb 2023.

[2] D. Zhang, M. Ma, and L. Xia, "A comprehensive review on gans for time-series signals," *Neural Computing and Applications*, vol. 34, no. 5, pp. 3551–3571, 2022.

[3] X. Li, V. Metsis, H. Wang, and A. H. H. Ngu, "Tts-gan: A transformer-based time-series generative adversarial network," in *International Conference on Artificial Intelligence in Medicine*. Springer, 2022, pp. 133–143.

[4] L. Diao, Y. Sun, Z. Chen, and J. Chen, "Modeling energy consumption in residential buildings: A bottom-up analysis based on occupant behavior pattern clustering and stochastic simulation," *Energy and Buildings*, vol. 147, pp. 47–66, 2017.

[5] S. Ge, J. Li, H. Liu, Y. Wang, H. Zhou *et al.*, "Domestic energy consumption modeling per physical characteristics and behavioral factors," *Energy procedia*, vol. 158, pp. 2512–2517, 2019.

[6] M. Pyne, B. J. Yurkovich, and S. Yurkovich, "Generation of synthetic battery data with capacity variation," in *2019 IEEE Conference on Control Technology and Applications (CCTA)*. IEEE, 2019, pp. 476–480.

[7] M. Lahariya, D. F. Benoit, and C. Develder, "Synthetic data generator for electric vehicle charging sessions: modeling and evaluation using real-world data," *Energies*, vol. 13, no. 16, p. 4211, 2020.

[8] Y. Mao, Q. Zhai, Y. Zhou, J. Zhao, Z. Shao, Y. Yang, and H. Hou, "Data generation method for power system operation considering geographical correlations and actual operation characteristics," *Energy Reports*, vol. 9, pp. 1480–1489, 2023.

[9] S. El Kababji and P. Srikantha, "A data-driven approach for generating synthetic load patterns and usage habits," *IEEE Transactions on Smart Grid*, vol. 11, no. 6, pp. 4984–4995, 2020.

[10] M. N. Fekri, A. M. Ghosh, and K. Grolinger, "Generating energy data for machine learning with recurrent generative adversarial networks," *Energies*, vol. 13, no. 1, p. 130, 2019.

[11] W. Li, J. Chen, J. Cao, C. Ma, J. Wang, X. Cui, and P. Chen, "Eid-gan: Generative adversarial nets for extremely imbalanced data augmentation," *IEEE Transactions on Industrial Informatics*, vol. 19, no. 3, pp. 3208–3218, 2022.

[12] J. Liu, F. Qu, X. Hong, and H. Zhang, "A small-sample wind turbine fault detection method with synthetic fault data using generative adversarial nets," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 7, pp. 3877–3888, 2019.

[13] Y. Chen, Y. Wang, D. Kirschen, and B. Zhang, "Model-free renewable scenario generation using generative adversarial networks," *IEEE Transactions on Power Systems*, vol. 33, no. 3, pp. 3265–3275, 2018.

[14] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *International conference on machine learning*. PMLR, 2017, pp. 214–223.

[15] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," *Advances in neural information processing systems*, vol. 30, 2017.

[16] A. Srivastava, D. Sinha, and V. Kumar, "Wcgan-gp based synthetic attack data generation with ga based feature selection for ids," *Computers & Security*, vol. 134, p. 103432, 2023.

[17] M. Razghandi, H. Zhou, M. Erol-Kantarci, and D. Turgut, "Variational autoencoder generative adversarial network for synthetic data generation in smart home," in *ICC 2022-IEEE International Conference on Communications*. IEEE, 2022, pp. 4781–4786.

[18] P. S. INC, "Dataport. pecan street inc. dataport," [Online]. Available: https://www.pecanstreet.org/dataport/, 2023.

[19] Z. Wang and T. Oates, "Imaging time-series to improve classification and imputation," *arXiv preprint arXiv:1506.00327*, 2015.

[20] D. G. da Silva, M. T. B. Geller, M. S. dos Santos Moura, and A. A. de Moura Meneses, "Performance evaluation of lstm neural networks for consumption prediction," *e-Prime-Advances in Electrical Engineering, Electronics and Energy*, vol. 2, p. 100030, 2022.

[21] J. Ye, A. Maddi, S. K. Murakonda, V. Bindschaedler, and R. Shokri, "Enhanced membership inference attacks against machine learning models," in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, 2022, pp. 3093–3106.

[22] M. Nasr, R. Shokri, and A. Houmansadr, "Comprehensive privacy analysis of deep learning," in *Proceedings of the 2019 IEEE Symposium on Security and Privacy (SP)*, 2018, pp. 1–15.

[23] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *2017 IEEE symposium on security and privacy (SP)*. IEEE, 2017, pp. 3–18.

[24] S. Kumar and R. Shokri, "Ml privacy meter: Aiding regulatory compliance by quantifying the privacy risks of machine learning," in *Workshop on Hot Topics in Privacy Enhancing Technologies (HotPETs)*, 2020.

[25] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.