Synthetic Time-series Data Generation with 3D Convolution for EV Systems

Xudong Hu*
Electrical and Computer Engineering
National University of Singapore
Singapore
e0459193@u.nus.edu

Guihai Zhang*
Electrical and Computer Engineering
National University of Singapore
Singapore
e0534010@u.nus.edu

Biplab Sikdar
Electrical and Computer Engineering
National University of Singapore
Singapore
elebisik@nus.edu.sg

***Abstract*—As electric vehicles (EVs) gain widespread acceptance for sustainable transportation, robust testing and validation for related technologies are becoming difficult due to challenges in acquiring real-world data due to limited availability, high costs, and privacy concerns. To address this issue, this paper introduces the 3D-time-series Generative Adversarial Network (3DTS GAN) to generate high-resolution, multivariate synthetic driving data for EV systems. Integrating Auto-encoder and GAN structures, the proposed method addresses the shortcomings of existing data generation methods, offering a more comprehensive representation of driving data. Evaluation results show that this method is able to generate synthetic data that is similar to original driving data with higher similarity scores than those attained using existing methods. Moreover, a functional check is done to demonstrate that there is no significant difference between using the original driving data and the synthetic data to perform further tasks such as energy consumption prediction.**

***Index Terms*—Electrical Vehicle, 3D Convolution, Synthetic Data, Time series data, Energy Consumption prediction**

I. Introduction

With the wide and rapid adoption of electric vehicles (EVs) towards sustainable transportation, the need for robust testing and validation processes is becoming increasingly paramount. However, acquiring real-world data for electric vehicles poses significant challenges due to factors such as limited availability, high cost, and privacy concerns. In response to these problems, the utilization of synthetic data has emerged as a promising solution to augment and expedite the development, testing, and validation of EV systems. Synthetic data is the artificial data generated from real-world data and shares the same characteristics and structure of the original data. Thus, a given analysis method performed on synthetic data and original data would provide similar results [1]. Synthetic data can replicate real-world scenarios, providing a versatile and scalable alternative to traditional data collection methods. By publishing synthetic data, there is no privacy concern as the original confidential data is under protection. Even when there is data breach of the synthetic data, driver's private information is not leaked.

However, existing data generation methods have less considerations on the time-series characteristics of the data. For example, there are more data corresponding to the whole trip and the data are spread over a longer period of time. Common methods only concentrate on short periods of time, while the information and inter-connection between data that are distributed far from each other is not well captured and thus, the generated synthetic data does not fully represent the characteristics of original data. Moreover, the generated synthetic data are usually univariate that only refers to one type of data. A more utilizable group of synthetic driving data should contain more types of information like environmental and traffic conditions, road type, vehicle motor performance, temperature, driving distance, speed, acceleration, battery recharging ability etc.

To address these problems, the 3D-time-series Generative Adversarial Network (3DTS GAN) is proposed to generate synthetic data in time-series format. This method can handle multivariate data at high resolution (i.e., per-second data). The synthetic data can be further used for EV system testing and validation like energy consumption prediction task. The proposed method combines the structure of Auto-encoder and typical GAN. The 3D convolution process helps to examine and store time-series features by considering time as the third axis. By feeding input of random Gaussian noise, the model produces synthetic data which contains a total of 35 features like the speed, acceleration, temperature, power consumption etc. Evaluation results show that our proposed method can generate driving data that are similar to original data with tiny differences while maintaining the utility of the data for further prediction and testing tasks. The main contributions of our work are as follows:

- We propose the 3DTS GAN to generate multivariate synthetic data in time-series format to overcome the scarcity and privacy issues of driving data.
- We compare the proposed technique with existing methods and results show that our model performs better at generating data that is similar to the original data.
- We examine the generated synthetic data in terms of its ability to perform energy consumption prediction and the accuracy is comparable to the results when using original data.

The rest of the paper is organized as follows. Section II is the literature review of related works. Section III describes the

*Xudong Hu and Guihai Zhang are co-first authors.

details of proposed model to generate synthetic data. Section IV discusses the evaluation results. Section V presents the concluding remarks of our work.

II. Related works

The field of synthetic data generation has witnessed significant advances in recent years, characterized by a diverse array of methodologies spanning statistical and neural network-based approaches. These methods have been applied across various domains, each presenting unique strengths and challenges.

Statistical methods traditionally rely on analyzing linear correlations within data, employing mathematical equations to model these interconnections. For instance, the work presented in [2] focuses on generating data for EV charging sessions, while another study employs Markov chain approaches for evaluating battery pack state of health [3]. While those models offer straightforward extrapolation within linear space, they often fall short in capturing the complexity and diversity of data types. In [4], statistical characteristics were analyzed to simulate EV load profiles, yet, these models often lack transferability and scalability to different evaluation scenarios. Bayesian Networks (BNs), as probabilistic graphical models, leverage prior probabilities for data generation. The work in [5] demonstrates their effectiveness in scenarios where intrinsic correlations between features are pivotal. However, the reliance on prior probabilities can sometimes constrain their applicability to a narrower range of use cases.

Neural network methods, particularly various forms of GANs, have emerged as potent tools for capturing non-linear correlations within data. Studies by [6], [7] utilize conventional GAN architectures to enhance prediction accuracy in EV demands and charging load curves. These models adeptly uncover hidden patterns without necessitating an in-depth exploration of time-series characteristics inherent in the data. Furthermore, a specialized time-series GAN [8] incorporates self-attention mechanisms, enabling it to learn internal time-series properties, albeit with the limitation of handling only single-variant data. The CycleGAN framework has emerged as a pivotal technique in synthetic data generation, particularly noted for its proficiency in handling unpaired data. This attribute is especially advantageous in scenarios where one-to-one data correspondences are absent [9]. Nonetheless, CycleGAN's reliance on an ancillary data set potentially curtails the quantity of synthetic samples produced, especially when alterations to this secondary dataset are constrained. In parallel, Wasserstein Generative Adversarial Network (WGAN) [10] have gained prominence, primarily owing to their incorporation of the Wasserstein distance, which has been instrumental in enhancing training convergence. Despite its strengths, the WGAN architecture demands significant computational resources. Additionally, its inclination towards generating a broader range of diverse samples, as a strategy to mitigate mode collapse, may inadvertently compromise the fidelity of the generated data in terms of its resemblance to the original dataset. This trade-off between diversity and similarity is a critical aspect that needs careful consideration in the application of WGANs for synthetic data generation.

In summary, the landscape of synthetic data generation is rich and evolving, with each method offering specific advantages and limitations. Statistical methods excel in scenarios with linear correlations and simpler data structures, while neural network approaches, particularly GANs, are more adept at handling complex, non-linear relationships and diverse data types. The choice of method ultimately hinges on the specific requirements of the application domain and the nature of the data at hand.

III. Methodology

A. Dataset

In our endeavor to replicate the diverse driving behaviors prevalent in real-life scenarios, we utilized an open-source tool (emobpy) for electric vehicle (EV) driving simulations, as detailed in [11]. This tool is particularly proficient in generating granular energy consumption data at a resolution of one second, allowing for an intricate depiction of driving dynamics. We tailored the simulation parameters, encompassing a range of driver categories and travel habits, to realistically model the driving behaviors for each journey. Table I presents the specific configuration settings employed for this purpose.

The simulation resulted in a comprehensive dataset, amounting to 5,000 minutes of driving activity. This equates to an extensive collection of 330,000 individual data records. Of these, 300,000 records were allocated for the creation of synthetic data, while the remaining 30,000 were reserved for functional testing, as elaborated in Section III-D. Each data entry encompasses 35 features, each intricately linked to specific driving conditions, such as immediate power usage, road type, vehicle speed, and passenger count. This compilation of data culminates in a dataset structured as $300,000 \times 35$.

The final preparatory phase involves normalizing the dataset using a min-max scaling approach. This process was crucial to standardize the range of feature values and ensure a balanced weighting across different features. The scaling formula employed can be articulated as follows:

$$\widetilde{f}_i = \frac{(f_i - max(F)) + (f_i - min(F))}{max(F) - min(F)}. \tag{1}$$

This approach to data simulation and preparation was instrumental in forging a dataset that not only reflects the complexity of real-world driving scenarios but also provides a robust foundation for subsequent analyses.

B. Time-series Data

Inspired by the approach established in [12], this study employs Gramian Angular Fields (GAF) as the preferred method for data encoding, aiming to augment the inter-feature correlations. Following the normalization of the features, a vector is composed using features from identical timestamps. This vector is then transformed into a polar coordinate system, defined as follows:

TABLE I: Driving behavior setting for each trip

Condition	Full-time commuters		Part-time commuters		Non-commuters	
	Weekday	Weekend	Weekday	Weekend	Weekday	Weekend
Category probability	0.4		0.3		0.3	
Trip to work	At least 1	Based on need	At least 1	Based on need	N.A.	N.A.
Minimum time at workplace	7	3	3.5	3	N.A.	N.A.
Maximum Time at workplace	8	4	4	4	N.A.	N.A.
Minimum state duration at workplace	3.5	3	3.5	3	N.A.	N.A.
Minimum state duration except for workplace	0.25	0.25	0.25	0.25	0.25	0.25

$$\begin{cases} \phi = arccos(\widetilde{f}_i), -1 \leq \widetilde{f}_i \leq 1, \widetilde{f}_i \in \widetilde{F}_i \\ r = \frac{s_i}{D}, s_i \in D \end{cases}. \tag{2}$$

In this context, s_i denotes the positional aspect of each feature, while D signifies the radial distance within the polar coordinate framework. The formulation of the GAF matrix is achieved through the computation of the inner product of the features, which is detailed as follows:

$$\widetilde{F}' \cdot \widetilde{F} - \sqrt{I - \widetilde{F}^2}' \cdot \sqrt{I - \widetilde{F}^2}. \tag{3}$$

Here, I represents the unit vector. Consequently, this approach yields an $N \times N$ square matrix, where N denotes the total number of features. This matrix effectively encapsulates the temporal interrelations among the selected features, providing a comprehensive representation of their dynamics.

To form the time-series data representation that captures the inter-correlation between time-stamps, we construct a 3D matrix where the third dimension is filled with timestamp expansion of 30 seconds. With the help of 3D convolution operation, the network can learn from the correlation between features and time together. Thus, the $300,000 \times 35$ data has been divided and encoded into 10,000 individual 3D matrix of shape of $30 \times 35 \times 35$ for network training. Each 3D matrix represents a time-series data block spanning 30 seconds, encompassing 35 encoded features.

C. Proposed Method

In this Section I, we introduce a novel 3D-time-series Generative Adversarial Network framework, tailored for the generation of synthetic data. This model is architecturally composed of an encoder, a decoder, a generator, and a discriminator, as illustrated in Figure 1. The training process is demarcated by a blue arrow, where the training data X, characterized by the dimensions of $30 \times 35 \times 35$, is fed into the encoder to yield the latent representations H_X. The decoder, upon receiving H_X, reconstructs data X' to approximate the original input X. The discrepancy between X and X' gives rise to the recovery loss (L_r), quantified as:

$$L_r = \mathbb{E}_{X',X}[\|X - X'\|_2]. \tag{4}$$

Simultaneously, a randomly generated noise vector V, of length 100, is introduced to the generator to synthesize analogous latent representations H_V. The divergence between H_X and H_V contributes to the similarity loss (L_h):

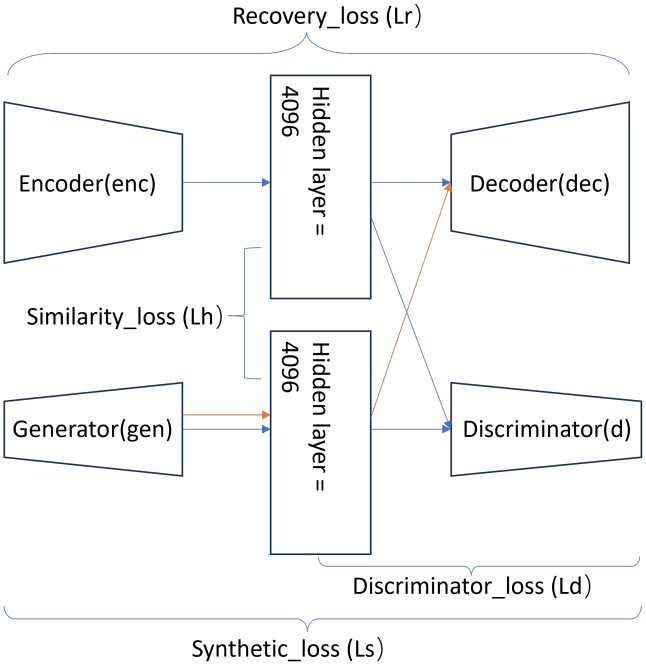

Fig. 1: 3DTS GAN Architecture. Training process is marked with blue arrow; Synthetic data generation process is marked with red arrow.

$$L_h = \mathbb{E}_{H_V,H_X}[\|H_V - H_X\|_2]. \tag{5}$$

Subsequently, both latent representations are classified by the discriminator to ascertain their authenticity, leading to the discriminator loss (L_d):

$$L_d = log(Y_X) + log(1 - Y_V). \tag{6}$$

Here, Y_X denotes the discriminator's output for the real latent representations, while Y_V signifies the discriminator's output for the synthetic counterparts. As the objective is to align H_V closely with H_X, a synthetic loss (L_s) is introduced to gauge their disparity:

$$L_s = log(1 - Y_V). \tag{7}$$

The backpropagation phase is unique in that the encoder and generator also incorporate the loss from the latent representa-

tion disparities. Thus, the aggregate loss for each component is summarized as:

$$\begin{aligned} Decoder\ loss &= L_r \\ Discriminator\ loss &= L_d \\ Encoder\ loss &= \lambda_e \cdot L_h + L_r \\ Generator\ loss &= \lambda_g \cdot L_h + L_s, \end{aligned} \tag{8}$$

where λ_g and λ_e represent the impact factors for the generator and encoder, respectively, emphasizing the importance of aligning the latent representations.

The synthetic data generation process, indicated by a red arrow, simply involves passing the noise vector V through the generator and decoder to yield the synthetic data X_V. This process even accommodates oversampling requirements.

D. Evaluation

To verify if the generated synthetic data is close to original data and can provide learning insights for a neural network in training session, we conduct both statistical measurements and functional checks. Statistical assessments include cosine similarity, Euclidean distance, and Jensen–Shannon distance. Given that the synthetic data generation does not adhere to a one-to-one mapping, it is imperative for the generated points to feasibly reside within the original dataset's domain. Cosine similarity assesses the alignment between points by measuring the angle between them, effectively determining if they are oriented in the same direction. In contrast, Euclidean distance quantifies the shortest path between two points, thus evaluating if they are close to each other. Conversely, the Jensen–Shannon distance treats each point as a distinct distribution, enabling an evaluation of their resemblance. Employing these three metrics together facilitates the identification of the nearest corresponding point within the original dataset.

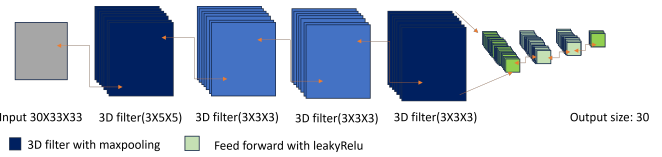

Fig. 2: 3D convolutional predictive model for time-series energy consumption prediction.

For a functional evaluation, we compare the utility of both the original and synthetic datasets in consumption prediction tasks, focusing on their performance and robustness. With a dataset dimensioned at $300,000 \times 35$ data, we designate instant and average power consumption as prediction targets, utilizing the remaining 33 features as inputs. The architecture of the consumption prediction model is depicted in Figure 2. We employ the Mean Absolute Percentage Error (MAPE) as the accuracy metric, calculated as:

$$MAPE\ Loss = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{Y_i - \hat{Y}_i}{Y_i}\right|. \tag{9}$$

Here, $\hat{Y}_i$ and Y_i represent the predictive output and actual consumption for the i-th second, respectively. This metric offers a relative and balanced evaluation of prediction accuracy.

To assess model robustness, we implement a white-box Fast Gradient Sign Method (FGSM) as an adversarial challenge. FGSM alters inputs to deviate the model's predictions. Finally, we extend this evaluative approach to include comparisons with WGAN-GP and CycleGAN, popular alternatives in synthetic data generation, to provide a comprehensive perspective.

IV. Results

The proposed 3DTS GAN is trained as explained in the previous section. Figure 3 and Figure 4 show the training losses of all components. Figure 3(a) is the loss for the encoder and Figure 3(b) is the loss for the decoder. The figures show that the encoder and decoder have converged to learn the structure and patterns of input and the recovered data is closer to the input data. As for the generator and the discriminator, their training losses are plotted in Figure 4. The blue line is the loss of each component and the orange line is the changing trend. We can see there are oscillations of the loss for generator around 5 and loss for discriminator around 1. This is because the overall training process is like a battle between the generator and the discriminator where the generator tries to generate data to fool the discriminator and the discriminator tries to differentiate the real and fake data. Nevertheless, from the trend line of the changes in Figure 4(a) and Figure 4(b), both generator and discriminator have converged with small error.

A. Synthetic Time-series Data

Table II summarises the comparison of similarity scores of generated synthetic data from our proposed method and existing methods. The best performance metrics are made bold for each column. It is obvious to see that our method has generated the synthetic data that are most similar to original data for all the metrics. For cosine similarity, larger value

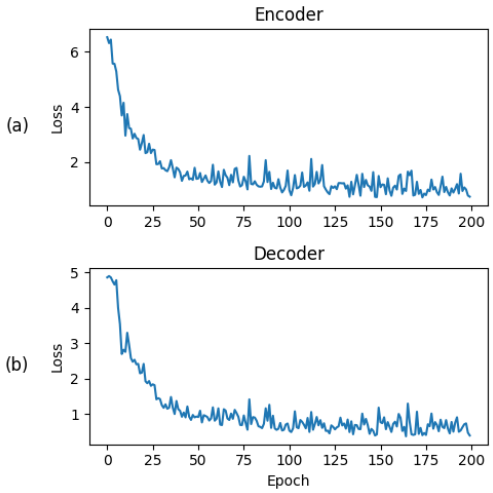

Fig. 3: Training performance for encoder and decoder.

TABLE II: Similarity scores between real and synthetic data using different methods

Method	Cosine Similarity			Euclidean Distance			JS Distance		
	Mean	Min	Max	Mean	Min	Max	Mean	Min	Max
Ours	**0.979**	**0.977**	**0.981**	**2.705**	**2.613**	**2.812**	**0.335**	**0.193**	**0.520**
WGAN-GP	0.944	0.898	0.969	4.681	3.537	6.770	0.354	0.211	0.576
CycleGan	0.802	0.796	0.805	7.555	6.950	9.619	0.449	0.342	0.572

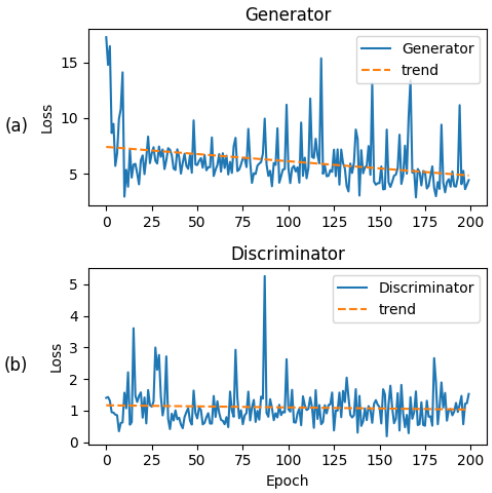

Fig. 4: Training performance for generator and discriminator.

means better. Our method obtains a mean value of 0.979, the minimum value of 0.977 and the maximum value of 0.981. The method using WGAN-GP has slightly smaller values than our method but the method using CycleGAN has much worse results. For the Euclidean distance (the smaller the better), our method also obtains the smallest values: 2.705 for mean value, 2.812 for minimum value and 2.613 for maximum value. Both WGAN-GP and CycleGAN generate data that have larger Euclidean distance than our method. Similarly, our method achieves the smallest values for the JS distance (the smaller the better). The mean value is 0.335, the minimum value is 0.193 and the maximum value is 0.52. It can also been seen that the WGAN-GP method has very similar results to ours. From this comparison, we can conclude that the proposed method generates data that is closest to the original data, in terms of both of the magnitudes and the distribution characteristics. The WGAN-GP and CycleGAN methods have advantage in handling the spatial relationships of adjacent pixels. The models are inclined to overlook the nuances and correlations inherent in distant data points. Consequently, their effectiveness may diminish when applied to complex time-series datasets.

TABLE III: Prediction MAPE using different data

Data	No attack	Under FGSM attack
Original	6.62	67.64
Ours	7.69	36.41
WGAN-GP	10.69	16.53
CycleGAN	27.94	31.52

B. Functionality Evaluation

As stated previously, the synthetic data generated from our method and other GAN methods are used to perform the same EV energy consumption prediction task. This is to confirm that the synthetic data can produce similar energy consumption prediction results as to make prediction using original data. Moreover, a typical FGSM attack is conducted against the prediction task to examine the performance under adversarial attack. Table III is the result of prediction MAPE using different sets of data. The benchmark is the MAPE score when using the original dataset to train prediction model. The prediction MAPE is 6.62% when the model is clean and this values increases to 67.64% when FGSM attack is carried out on the model. This shows that the original driving data can train a prediction model with small difference but the model is vulnerable to adversarial attacks and the prediction error increases significantly in the presence of an attack. When training the prediction model using the synthetic data generated from our method, the normal MAPE is 7.69% which is only slightly larger than that of the original data. This shows our method can generate synthetic data that has similar functionality as original data. Under the FGSM attack setting, the prediction MAPE of the model increases to 36.41%. This value is much smaller than that of the original data. This can be explained because when using our method to generate the synthetic data, we are not making identical copies of original data and the synthetic data can be thought of as the original data with noise. This introduction of noise to the original data leads the trained model to be more robust to changes in the input data. Although the attack increases the

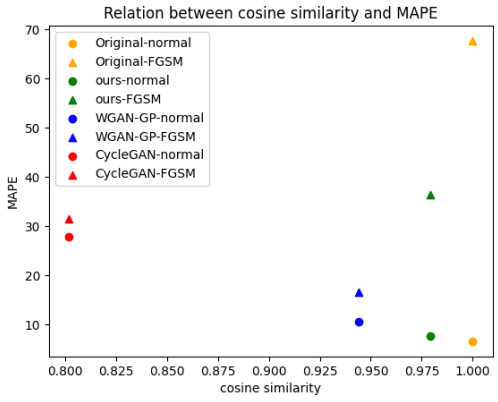

Fig. 5: Relationship between the cosine similarity score and MAPE.

prediction error, this error is much smaller than that of the original trained model. As a comparison, the prediction MAPE of model trained using WGAN-GP generated synthetic data is 10.69% under normal case and is 16.54% when there is FGSM attack. The normal MAPE is larger than using data from our synthetic data, but the error is smaller than ours when there is FGSM attack. This is as expected because from the previous similarity score we can observer that the synthetic data generated using WGAN-GP method is not that close to original data. Therefore, when predicting the power consumption, the MAPE is larger. However, since the difference between the synthetic data and original data is large, it is more robust to the FGSM attack and the error just increases by about 6%. The model trained with CycleGAN generated data has normal prediction MAPE of 27.94% and a value of 31.52% when under FGSM attack. This model has slightly smaller MAPE than ours under FGSM attack, but much larger MAPE than ours under normal scenarios. However, the model trained using our 3DTS GAN generated data can achieve much higher cosine similarity score. This result also follows the rules we have discussed above.

Figure 5 illustrates the relationship between the cosine similarity score and the MAPE of the prediction model.The yellow markers are the results using original data. The green markers are the results obtained from our 3DTS GAN, the blue markers are from the WGAN-GP, and the red markers are from CycleGAN. At the same time, the circle marker represents the normal case and triangle marker represents the FGSM case. We can see there is a trade-off between data similarity and the resilience against FGSM attack. The original data has the largest changes in MAPE when encountering FGSM attack. The CycleGAN generated data is the most unlike to original data but it has the smallest changes in MAPE under FGSM attack.

V. Conclusion

This paper discusses an exploration of synthetic data generation for electric vehicle applications, leveraging the emobpy simulator to procure high-resolution driving data. The primary objective is to generate a synthetic counterpart of the driving data that not only upholds the security and privacy standards, but also preserves the intrinsic insights and functionalities of the original dataset.

We introduced an innovative 3DTS GAN architecture, adept at handling multi-variant time-series data. This model excels in capturing the intricate interplay between various features and their temporal dynamics. Our findings indicate that the synthetically generated driving data maintain statistical coherence with the original dataset and exhibit superior performance when bench-marked against implementations like CycleGAN and WGAN-GP. Notably, the synthetic data also demonstrates commendable efficacy in predicting energy consumption, as gauged by the MAPE, thereby reinforcing its functional relevance.

Moreover, the robustness analysis results shows that predictive models trained with this synthetic data showcase enhanced resilience to adversarial attacks, thereby enhancing model security. This aspect is particularly significant, highlighting an additional layer of protection.

References

[1] J. Jordon, L. Szpruch, F. Houssiau, M. Bottarelli, G. Cherubin, C. Maple, S. N. Cohen, and A. Weller, "Synthetic data – what, why and how?" 2022.

[2] 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.

[3] 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)*, 2019, pp. 476–480.

[4] J. Schäuble, T. Kaschub, A. Ensslen, P. Jochem, and W. Fichtner, "Generating electric vehicle load profiles from empirical data of three ev fleets in southwest germany," *Journal of Cleaner Production*, vol. 150, pp. 253–266, 2017.

[5] H. M. Combrink, V. Marivate, and B. Rosman, "Comparing synthetic tabular data generation between a probabilistic model and a deep learning model for education use cases," 2022.

[6] S. Chatterjee and Y.-C. Byun, "A synthetic data generation technique for enhancement of prediction accuracy of electric vehicles demand," *Sensors*, vol. 23, no. 2, p. 594, 2023.

[7] R. Buechler, E. Balogun, A. Majumdar, and R. Rajagopal, "Evgen: Adversarial networks for learning electric vehicle charging loads and hidden representations," *arXiv preprint arXiv:2108.03762*, 2021.

[8] J. Yoon, D. Jarrett, and M. Van der Schaar, "Time-series generative adversarial networks," *Advances in neural information processing systems*, vol. 32, 2019.

[9] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2242–2251.

[10] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Proceedings of the 34th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, D. Precup and Y. W. Teh, Eds., vol. 70. PMLR, 06–11 Aug 2017, pp. 214–223. [Online]. Available: https://proceedings.mlr.press/v70/arjovsky17a.html

[11] C. Gaete-Morales, H. Kramer, W.-P. Schill, and A. Zerrahn, "An open tool for creating battery-electric vehicle time series from empirical data, emobpy," *Scientific Data*, vol. 8, no. 1, p. 152, 2021.

[12] Z. Wang, T. Oates *et al.*, "Encoding time series as images for visual inspection and classification using tiled convolutional neural networks," in *Workshops at the twenty-ninth AAAI conference on artificial intelligence*, vol. 1. AAAI Menlo Park, CA, USA, 2015.