Proceedings of the 2025 Winter Simulation Conference
E. Azar, A. Djanatliev, A. Harper, C. Kogler, V. Ramamohan, A. Anagnostou, and S. J. E. Taylor, eds.

GENERATING ARTIFICIAL ELECTRICITY DATA FOR MONITORING ENERGY CONSUMPTION IN SMART CITIES

Sina Pahlavan[1], Wael Shabana[2], and Abdolreza Abhari[3]

[1,2,3]Distributed Systems and Multimedia Processing Lab (DSMP),
Dept. of Computer Science, Toronto Metropolitan University, Toronto, ON, Canada

ABSTRACT

Generation of synthetic data for energy demand allows simulation-based forecasting for infrastructure planning, building optimization, and energy management, key elements of smart cities. This study compares multivariate kernel density estimation (KDE) and time-series generative adversarial networks (TimeGAN) for their ability to generate realistic time series that preserve crucial feature relationships for forecasting. The evaluation is based on both statistical similarity and predictive performance using machine learning models, focusing on seasonal and hourly consumption patterns. The results emphasize the importance of temporal consistency and justify synthetic augmentation when real data is limited, especially for time-aware energy forecasting tasks, and demonstrate how synthesized data can be used when forecasting future energy demand.

1 INTRODUCTION

Accurate predictions of energy consumption in smart cities are necessary for efficient and sustainable resource planning of infrastructure, as highlighted in a study by (Peteleaza 2024). By integrating smart grids with advanced forecasting models, such as a dense encoder with hyperparameter tuning, the study demonstrates how machine learning can enhance urban energy management and sustainability at scale. Accurate forecasting of hourly electricity is vital for reliable grid operations and to respond to daily demand fluctuations.

Synthetic data generation offers a way to produce realistic, privacy-preserving datasets for model development and evaluation. In the absence of high-quality data on real-world electricity consumption data, data generation becomes a useful technique to overcome the lack of data. However, the effectiveness of augmented synthetic data depends on how well it preserves key temporal and structural patterns in energy behavior.

KDE has proven valuable in simulation tasks where real-world data deviate from standard parametric distributions. Giannelos et al. applied KDE to represent the non-Gaussian behavior of building electricity demand and solar generation. Their study shows that KDE-based Monte Carlo simulations capture extreme events more effectively than Gaussian models, leading to more robust energy hub operations. The authors demonstrate that KDE enables accurate predictions, particularly under conditions of high variability and uncertainty (Giannelos, Pudjianto, Zhang, and Strbac 2025).

A 2019 study emphasizes the growing importance of forecasting short-term electrical load, especially in the context of increasing electrification and the integration of low-carbon technologies (Jacob, Neves, and Vukadinović Greetham 2020, pg.15 - 37). Beyond energy modeling, Pozi and Omar employ KDE for privacy-preserving synthetic data generation. Their framework uses KDE to approximate the empirical distribution of dataset features and then generates synthetic shifted datasets that maintain statistical utility

while masking sensitive relationships (Mohd Pozi and Omar 2020). Though designed for data anonymization, their approach highlights KDE's broader applicability for generating realistic, distribution-preserving synthetic data, an insight directly relevant to energy prediction tasks that rely on high-fidelity simulation. These works demonstrate how KDE serves as a robust tool for simulating complex distributions in both operational and privacy-sensitive contexts.

Generative modeling for time-series has progressed from simple statistical tools to deep generative models. Vanilla GANs often struggle with capturing temporal dependencies, which are critical in sequential data. **TimeGAN** addresses this by integrating recurrent networks with adversarial and supervised training to better preserve both temporal and feature dynamics (Yoon, Jarrett, and van der Schaar 2019). Parallel advancements, such as **Wasserstein GANs**, have shown improved training stability and gradient behavior, making them attractive for time-series synthesis tasks where mode collapse and convergence issues are common (Arjovsky, Chintala, and Bottou 2017).

The goal of this study is to evaluate whether synthetic data, generated by KDE and TimeGAN, can be effective in forecasting short-term energy demand at the aggregate level, capturing the combined patterns of residential, commercial, and industrial electricity use.

2 DATA COLLECTION

We have collected and merged multiple datasets collected from different sources and preprocessed them into one dataset, which represents the historical hourly levels of electricity use in Ontario, Canada. The dataset consists of three main sources: energy demand, weather statistics, and HOEP prices, and captures the period between September 2013 and December 2024. Our attempts at synthesizing data will focus on overall electricity levels in the residential, commercial, and manufacturing sectors. The data collected at this stage will be referred to as original, real, or historical data throughout the remainder of the paper.

The energy demand data was sourced from hourly Ontario electricity demand records (provided by the IESO), while the weather data came from Weatherstats (a website containing weather stats for Canada) for multiple cities with temperature and humidity readings. Public holidays were incorporated using the Python Holidays library to mark non-business days. Binary flags were added to capture business and holiday effects. These included a weekend, a statutory holiday flag, and a "business hour" flag.

3 DATA SYNTHESIS

3.1 Kernel Density Estimation (KDE)

KDE is a non-parametric way to estimate the probability density function (PDF) of a dataset. They have been suggested as a good method for synthetic data generation, particularly when the original dataset contains relatively few rows (Plesovskaya and Ivanov 2021). Given a dataset of points $n = \{x_1, x_2, x_3, \ldots, x_n\}$, KDE performs the following calculation to estimate the density at a point x:

$$\hat{f}_x(h) = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right)$$

Where:

- $\hat{f}_x(h)$ is the density estimated at point x.
- h is a smoothing parameter or bandwitdth, that controls the width of the kernel.
- K is the kernel function.
- x_i is an individual data point.

We used multivariate KDE with a Gaussian kernel to model the joint distribution of several continuous variables simultaneously. The multivariate aspect allows KDE to account for interdependencies among features, enabling it to preserve correlations like higher demand on cold days or varying usage across different hours and days of the week. Now, we must generalize the previous formula to:

$$\hat{f}_H(x) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{\sqrt{(2\pi)^2|H|}} \exp\left(-\frac{1}{2}(x-x_i)^T H^{-1}(x-x_i)\right)$$

This allows KDE to capture correlations between dimensions, which is crucial in the case of synthetic data where temperature, humidity, and demand are interdependent.

Autoencoder For Dimensionality Reduction KDE suffers from the curse of dimensionality. Previous works have recommended autoencoders as a means of battling this curse (Wang, Yao, and Zhao 2016). We have also used dimensionality reduction via an autoencoder, designed to compress input data into a lower-dimensional latent representation (encoding) and then reconstruct it back to its original form (decoding) (Li, Pei, and Li 2023). This structure allows learning a nonlinear representation that retains the most important relationships between features. We trained the autoencoder on the normalized dataset. Then, we passed the original data through the encoder to reduce it to a lower-dimensional latent space.

Sliding Window to Preserve Temporal Context Instead of treating each time step as an independent data point, we grouped sequences of consecutive observations using windows with a duration of four hours (Masood, Abbasi, and Keong 2020). This approach captures short-term temporal dependencies and seasonal fluctuations. The autoencoder was trained to compress and reconstruct these time windows.

Conditional Sampling We implemented a conditional sampling strategy, inspired by conditional GANs. Rather than training a full conditional TimeGAN, we conditioned our KDE-based sampling on a subset of features, such as month, day, hour, temperature, and humidity. This allowed us to model local distributions and generate synthetic energy demand values that reflect realistic temporal contexts.

3.2 Time-Series Generative Adversarial Networks (TimeGAN)

Generative data in the field of energy consumption is required for a variety of reasons, among these are simulating realistic demand profiles under weather and temporal conditions, forecasting market demand, and impute missing data. Time-series data presents some challenges due to temporal dependencies and high dimensionalities. TimeGAN (Time-series Generative Adversarial Network) offers a hybrid architecture, performing autoregressive modeling, unsupervised adversarial learning, and supervised data embedding (Yoon, Jarrett, and van der Schaar 2019).

3.2.1 TimeGAN Architecture

TimeGAN comprises four separate neural networks to model them. These neural networks include an autoencoder, consisting of an embedding network (e) and a recovery network (r), a generator (g), and a discriminator (d), as shown in Figure 1(b). The autoencoder is responsible for learning meaningful feature representations by mapping the input time-series $(X = x_1, x_2, \ldots, x_T)$ to a latent space representation $(H = h_1, h_2, \ldots, h_T)$ and subsequently reconstructing it back into the original input space (Struye, Lemic, and Famaey 2022). This process ensures that the latent representation preserves both the contextual and temporal structure of the original data, serving as a foundation for the generator and discriminator during adversarial training (Yoon, Jarrett, and van der Schaar 2019).

3.2.2 Backpropagation and Training Procedure

TimeGAN's training process as shown by (Yoon, Jarrett, and van der Schaar 2019) comprises the following four sequential stages as shown in Figure 1(a), each guided by specific loss functions.

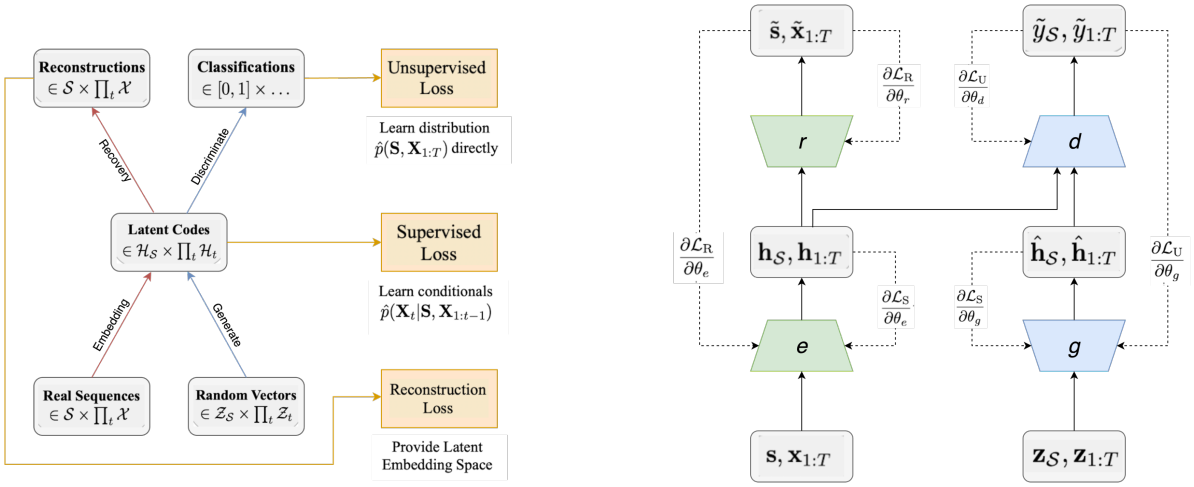
Autoencoder Pretraining: The reconstruction loss $\mathscr{L}_R$ initializes the embedding and recovery networks to reconstruct input sequences.

Supervised Training: The supervised loss $\mathscr{L}_S$ trains the supervisor to predict future latent states, promoting temporal consistency.

Adversarial Training: The generator and discriminator are trained using the unsupervised loss $\mathscr{L}_U$, where the generator learns to synthesize realistic latent sequences, and the discriminator distinguishes them from the real ones.

Joint Training: The full generator loss $\mathscr{L}_G$ combines the three objectives with weights α and β. The discriminator uses $\mathscr{L}_D$. This stage uses *Backpropagation Through Time (BPTT)*, which unrolls recurrent neural networks (RNN) over time, allowing the gradients to flow through all timesteps as explained by (Werbos 1990).

This coordinated process ensures that the generator captures both the temporal and statistical properties of the real data.



(a) TimeGAN Modules (b) TimeGAN Training Sequence

Figure 1: TimeGAN Architecture

From: (Yoon, Jarrett, and van der Schaar 2019)

4 RESULTS AND ANALYSIS OF SYNTHETIC DATA

4.1 KDE Synthesized Data Analysis

To assess synthetic data quality, we used the Kolmogorov-Smirnov test to measure the maximum difference between the datasets' cumulative distribution functions (Berger and Zhou 2014).

Figure 2 illustrates that most of the features in the KDE-generated synthetic dataset exhibit low KS scores compared to the original data, indicating a close match in marginal distributions. Some features show slightly higher KS values, suggesting higher deviations in distributional similarity.

Secondly, we will employ two correlation matrices to compare real and synthetic KDE data, as shown in Figure 3. The synthesized data closely resemble the original data in terms of correlation.

We also used the Pearson correlation score. The high computed value of 0.9926 confirms that the synthetic generation process has effectively preserved the critical multivariate dependencies.

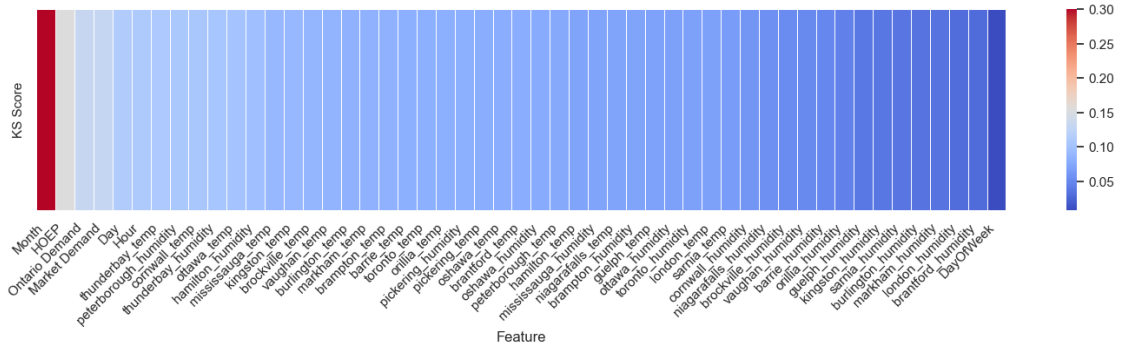


Figure 2: KS score heatmap for different features comparing the similarities between the original and KDE synthesized datasets.

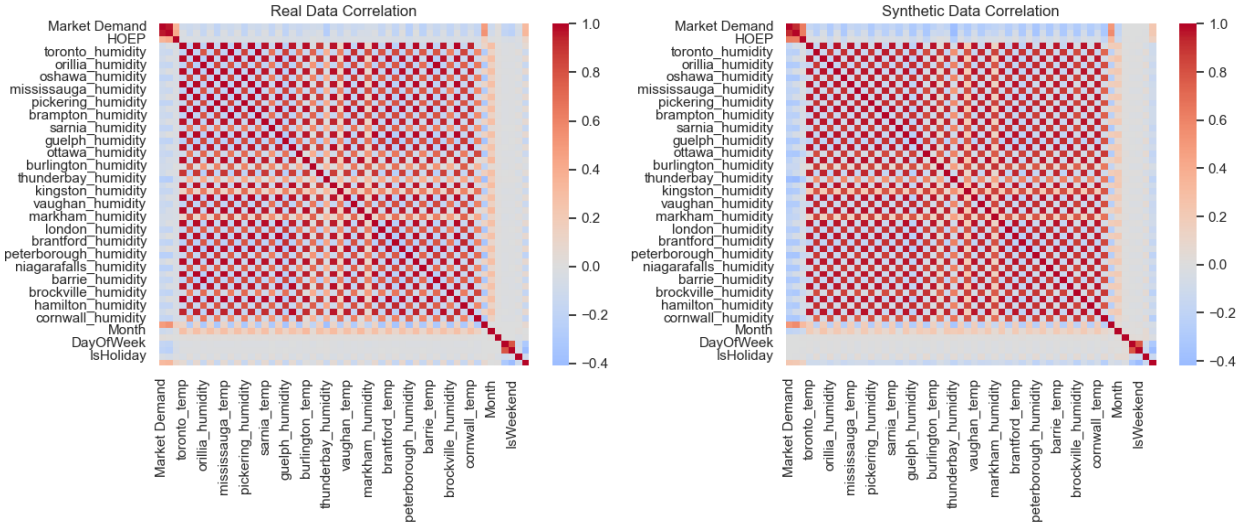


Figure 3: Correlation matrices of the real and KDE synthetic data.

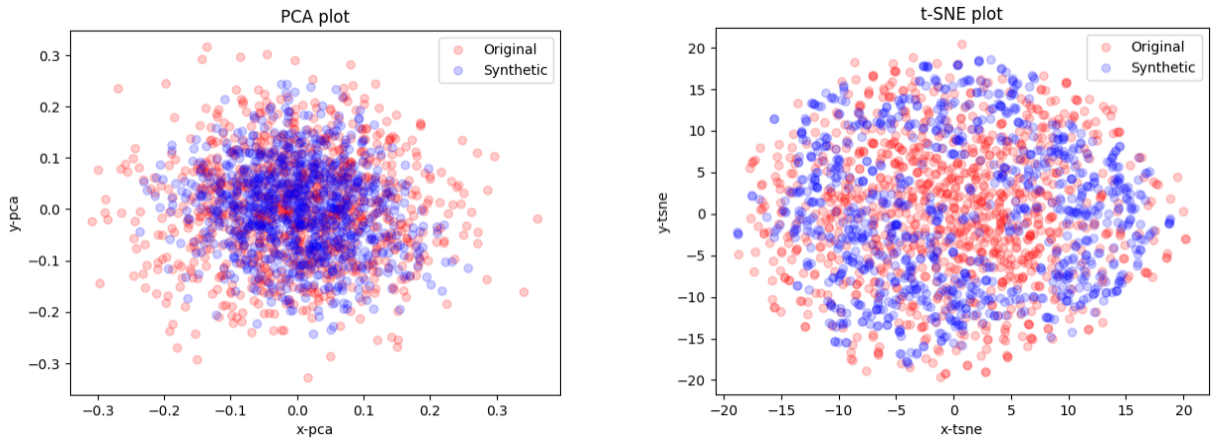
4.2 TimeGAN Synthesized Data Analysis

To assess the quality and fidelity of the synthetic data generated by the model, we conducted both qualitative and quantitative evaluations. Synthetic data was compared with the original dataset using dimensionality reduction techniques (PCA and t-SNE), as well as two key statistical scores: predictive score (*measuring the Mean Absolute Error (MAE) between ground truth and the model's prediction*) and discriminative score (|*Accuracy - 0.5*|) (Yoon, Jarrett, and van der Schaar 2019).

As shown in Figures 4(a) and 4(b), both the PCA and t-SNE plots demonstrate considerable overlap between the original and synthetic samples. In the PCA projection, the synthetic data cluster closely around the original distribution, indicating preservation of the global structure. The t-SNE plot, which emphasizes local and non-linear structures, further reveals that synthetic points intermix well with the original data, suggesting that the model has learned meaningful temporal dependencies.

Furthermore, as shown in Table (1) a low predictive score (close to 0) indicates that synthetic data can be used to train models that perform well on real data. The predictive score of 0.2819 suggests a moderately good generalization from synthetic to real sequences.

The observed discriminative score of 0.4999 confirms that the synthetic data is highly indistinguishable from the real samples. T-SNE diagnostics report a mean σ of 0.158 and a final KL divergence of 2.695



(a) (b)

Figure 4: TimeGAN PCA & t-SNE Charts

after 300 iterations. These values are typical of well-separated yet overlapping clusters, supporting the conclusion that the synthetic data closely match the real data distribution without mode collapse.

Table 1: Quantitative TimeGAN Evaluation

Metric	**Value**
Predictive Score	0.2819
Discriminative Score	0.4999

The combined visual and quantitative evaluations strongly suggest that the generative model has successfully captured the statistical and temporal characteristics of the original dataset and exhibits high fidelity for forecasting or simulation.

The KS heatmap for the TimeGAN synthesized data exhibits lower overall KS scores, suggesting a higher similarity between the original and synthetic data. This is illustrated in Figure 5.

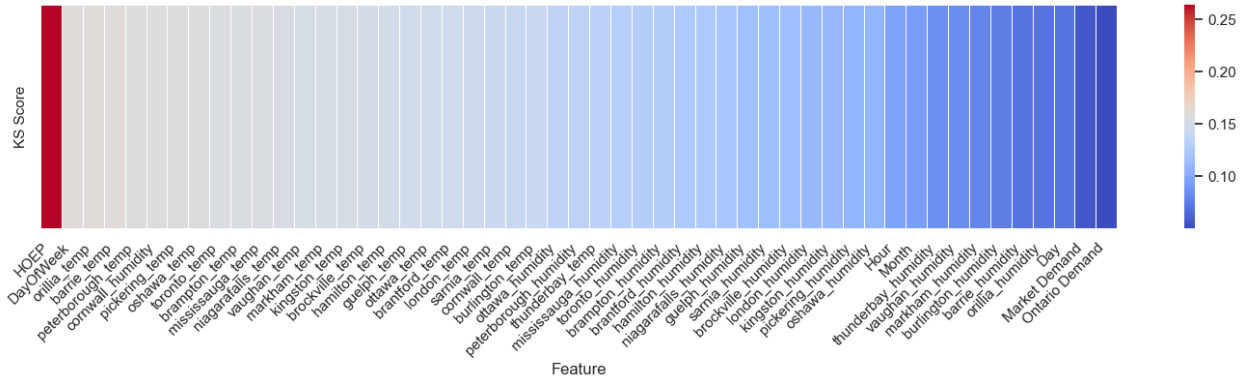


Figure 5: KS score heatmap for different features comparing the similarities between the original and TimeGAN synthesized datasets.

Furthermore, the correlation matrices (Figure 6) of the original and synthetic datasets provided a Pearson correlation of 0.9400, indicating a strong linear relationship and suggesting that the synthetic data effectively preserved the dependencies between those present in the original dataset.

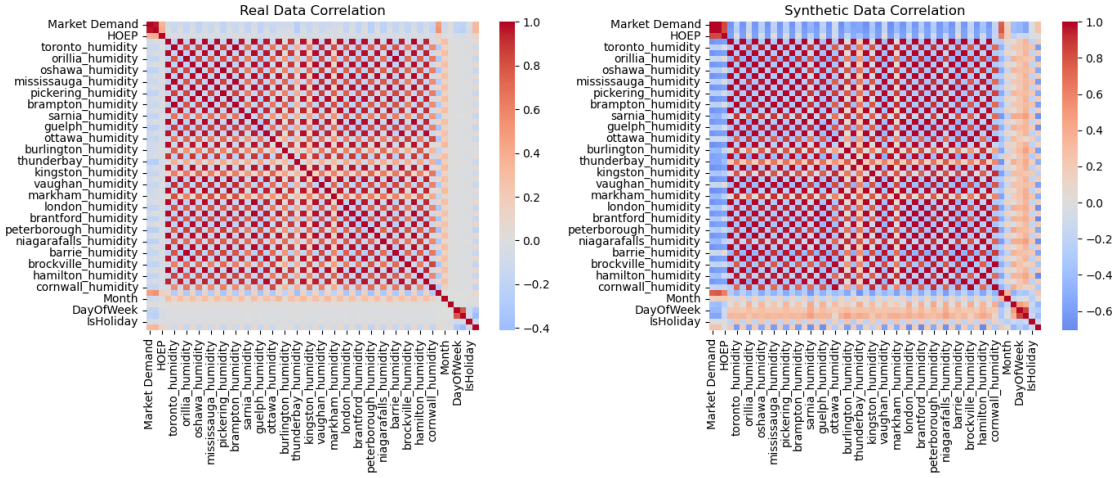


Figure 6: Correlation matrices of the real and TimeGAN synthetic data.

4.3 Analyzing Real and Synthetic Data Patterns

Figure 7 compares the average hourly demand in the real, KDE, and TimeGAN datasets. The real data display a characteristic daily load curve, with low demand in the early morning, a morning ramp-up, and a peak in the late afternoon to early evening. KDE replicates the general shape of the daily demand curve, but exhibits notable step-like trends, particularly between hours 4–5 and 12–13, which break the smoothness seen in the original data. TimeGAN, on the other hand, produces a smoother curve overall but overestimates peak evening demand (hours 18–20) and underestimates early morning demand (hours 0–6), indicating a tendency to amplify or shift the peaks. Although both synthetic sources approximate the overall trend, neither fully captures the nuances in intraday dynamics found in the real data.

Figure 8 shows the average demand across seasons. Real data exhibit distinct seasonal patterns, with summer and winter showing higher average loads than spring and fall. KDE follows these seasonal contours reasonably well, though with a slight smoothing effect that reduces the contrast between peak

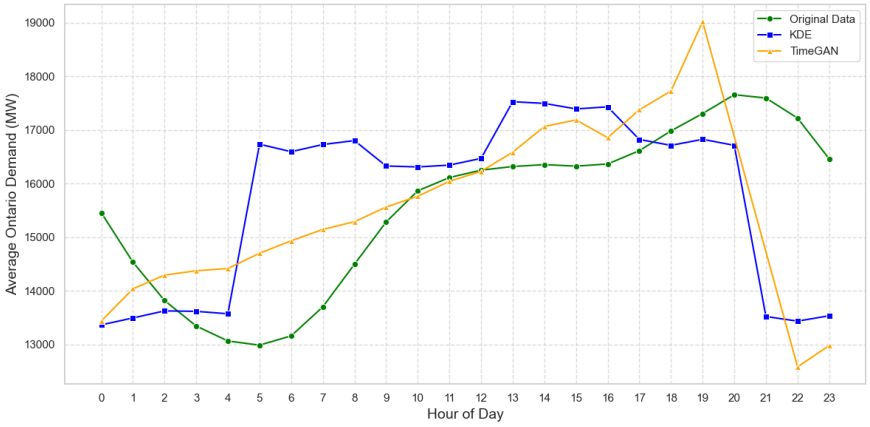


Figure 7: Average hourly demand: real, KDE, and TimeGAN data.

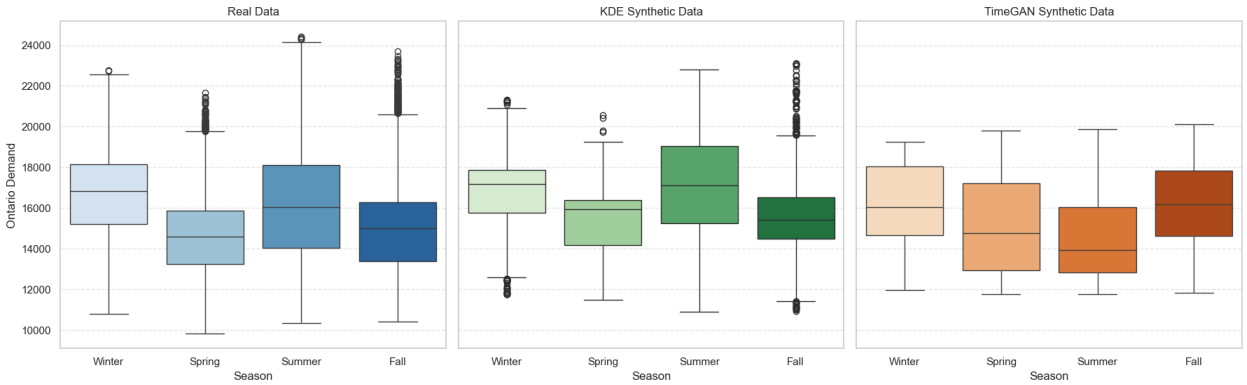


Figure 8: Comparison of average demand by season across real, KDE, and TimeGAN data.

and shoulder seasons. TimeGAN data also reproduce the broad seasonal structure but show elevated demand levels in Spring and Fall. While KDE shows a conservative fit with consistent seasonal transitions, TimeGAN introduces higher variance and potential overfitting to extreme values, highlighting differences in generalization behavior between the two synthetic approaches.

5 SIMULATION AND FORECASTING USING REAL AND SYNTHETIC DATA

To evaluate how synthetic data can be used, we conduct simulation of different scenarios and test the prediction capabilities of machine learning models when using them for simulation and forecasting. A diverse set of machine learning models are used in the experiment that are explained next. Our ANN consists of an input layer, three dense hidden layers with ReLU activation, and an output layer. It is trained over 100 epochs with a batch size of 32 using the Mean Squared Error (MSE) loss function, which is well suited for continuous prediction tasks (Bramer 2007). ANNs are effective in capturing complex relationships by adjusting internal weights through backpropagation as demonstrated in the examples in the book Topics In Data Science (Abhari 2018).

Support Vector Regression (SVR) is a type of supervised learning model that seeks to find a function that approximates the target values within a certain error margin, as shown by (Zhang and O'Donnell 2020). We use SVR with both a linear kernel and a Radial Basis Function (RBF) kernel, which enables the model to capture non-linear patterns (Han, Qubo, and Meng 2012). Our Regression Tree model splits the data based on feature thresholds to minimize prediction error, with a maximum tree depth of 15 to balance model complexity and overfitting. Each leaf node provides a prediction based on the average target value of the samples it contains (Tso and Yau 2007).

5.1 Short and Long-Term Forecasting with Synthetic Data

To assess the robustness and generalizability of synthetic data across different forecasting horizons, we designed two complementary experiments: a short-term and a long-term evaluation. In both cases, models were trained exclusively on real data and then tested separately on real, KDE-generated, and TimeGAN-generated datasets. The short-term experiment focused on within-year generalization, where models were trained on data from the first ten months of a year and tested on synthetic and real data from the final two months. This setup evaluates how well synthetic data can support near-future forecasting when only limited historical context is available.

The long-term experiment extended the training period to span four full years of real historical data. This broader context allows the models to learn deeper seasonal and temporal dependency patterns before being tested on synthetic data representing a future period. As in the short-term experiment, evaluation was carried out using real, KDE, and TimeGAN test sets. This long-term setting helps reveal whether

synthetic data can adequately reflect long-range dynamics, such as interannual variability and complex seasonal effects. Taken together, these experiments provide a robust framework to examine the utility of synthetic data across both immediate and extended forecasting contexts.

5.2 Results of Short-Term and Long-Term Forecasting

Table 2 compares the model performance with long-term and short-term forecasts using real, KDE, and TimeGAN-generated test data, revealing several key patterns. Across all models, TimeGAN data consistently outperform KDE in both MAE and R^2, highlighting a stronger ability to capture temporal dynamics and preserve structure. In particular, ANN achieves the best overall performance, especially in the short term with TimeGAN (MAE = 355.13, $R^2 = 0.9249$), even surpassing the results on real data.

Table 2: Model performance on Training Real, and Testing Real, KDE, and TimeGAN-generated datasets (Short-Term vs Long-Term).

Model	**Long-Term (4 yrs training + 1 yr test = 5 yrs)**						**Short-Term (10 mo training + 2 mo test) = 1 yr)**					
	Real		**KDE**		**TimeGAN**		**Real**		**KDE**		**TimeGAN**	
	MAE	R^2	MAE	R^2	MAE	R^2	MAE	R^2	MAE	R^2	MAE	R^2
Regression Tree	731.91	0.8074	1103.36	0.5312	826.66	0.7629	884.05	0.6783	953.07	0.3829	732.22	0.6838
SVM RBF	1042.04	0.6079	1187.44	0.4240	1088.70	0.5691	1004.07	0.5957	915.20	0.4705	805.57	0.6450
SVM Linear	1505.83	0.1357	1569.48	0.1038	1493.17	0.4807	976.40	0.5354	941.13	0.4199	810.63	0.6948
ANN	675.78	0.8553	829.39	0.7325	503.94	0.9118	785.17	0.7452	850.17	0.5341	355.13	0.9249

5.3 Simulation Using Synthetic Data

Table 3 shows the short-term prediction results when training and testing on synthetic datasets. Further, it reinforces that synthetic TimeGAN data modeling using ANN outperforms ANN modeling of KDE data, indicating that TimeGAN retains its internal structure even when used for both training and testing. The Regression Tree performs better when trained and tested using TimeGAN data. SVM models degrade more under KDE, which lacks a reliable structure for generalization. In general, TimeGAN proves to be more effective than KDE in generating synthetic energy demand sequences, with SVM emerging as the most robust predictive model when used in simulation.

Table 3: Model performance when training and testing on the same synthetic dataset.

Short-Term (10 mo training + 2 mo test) = 1 yr				
Model	**KDE → KDE**		**TimeGAN → TimeGAN**	
	MAE	R^2	MAE	R^2
Regression Tree	694.66	0.6792	460.16	0.8369
SVM RBF	857.55	0.5376	576.44	0.7545
SVM Linear	1059.52	0.2038	375.15	0.9068
ANN	1037.46	0.3374	446.20	0.8292

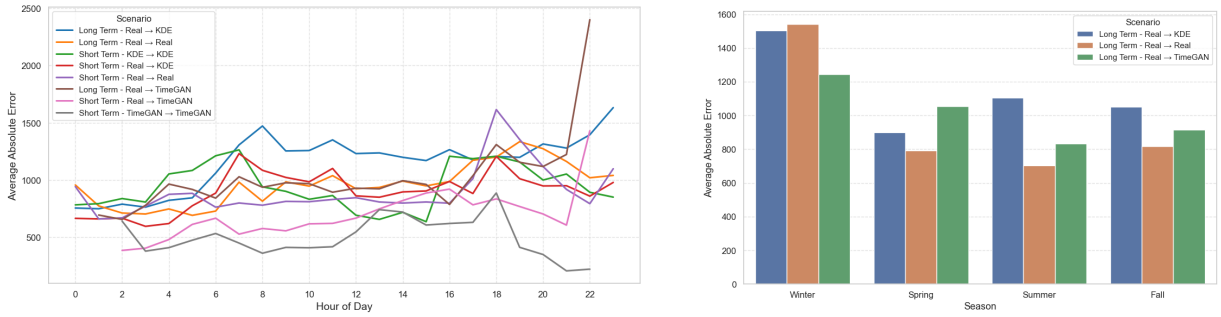
5.4 Seasonal and hourly analysis of error occurence

To better understand the behavior of forecasting errors, we analyze performance trends by season and by hour of day. Real-world data serves as a benchmark for comparison against predictions using data generated by KDE and TimeGAN. This allows us to assess how the quality and consistency of synthetic data affect model generalization and accuracy across time dimensions.

Figure 9(a) presents the average absolute error by hour across both short-term and long-term experiments. In the long-term setting, where models are trained on real data and tested on either real or synthetic data, predictions on TimeGAN test data consistently yield the lowest error throughout the day. Forecasts tested on real data follow closely behind and remain relatively stable, while those tested on KDE data show substantially more fluctuation and consistently higher errors, particularly during mid-morning (8 AM) and late evening hours. This suggests that TimeGAN captures long-term temporal features more effectively than KDE, which struggles to generalize over a broader range of time and consumption behaviors.

In the short-term experiments, a wider range of training-testing combinations were evaluated. Notably, the setting where we trained and tested TimeGAN data produced the lowest and most stable errors across all hours, underscoring the internal coherence of the data it generates. This stability contrasts with the KDE-KDE configuration, which exhibits the highest volatility and error spikes, particularly during peak energy usage hours such as 8AM and 6PM. *Real→KDE* predictions similarly demonstrate increased error and instability, while *Real→TimeGAN* forecasts perform much better, often nearly matching the *Real→ Real* baseline. These findings reinforce the advantage of using TimeGAN over KDE for short-term synthetic data generation in forecasting scenarios.

Figure 9(b) shows seasonal error trends using only long-term testing scenarios. Here, real data consistently produces the lowest forecast error across most seasons, with the exception of winter, where TimeGAN yields slightly better results. KDE incurs the highest errors in summer and fall, suggesting it has trouble replicating the subtleties of energy consumption in these transitional seasons. Interestingly, TimeGAN maintains competitive performance in fall and summer but demonstrates higher errors in spring, possibly due to difficulties in modeling peak cooling loads that occur sporadically in that season. Overall, seasonal patterns support the conclusion that TimeGAN offers a closer approximation to real data performance than KDE, particularly in long-term contexts.



(a) Hourly Mean absolute error for each data type.

(b) Seasonal Mean absolute of predictions errors.

Figure 9: Comparing real, KDE, and TimeGAN data in terms of hourly error across the experiments.

6 DISCUSSION

6.1 Implications

The findings highlight the different strengths and weaknesses of the multivariate KDE and TimeGANs for generating synthetic data in the context of forecasting energy demand. Although KDE achieved higher statistical fidelity with the original dataset, demonstrated by elevated Kolmogorov-Smirnov (KS), scores, and Pearson's correlation coefficients (0.9926 vs. 0.94 for TimeGANs), this similarity did not translate into better model performance since the ML models trained on KDE data showed lower predictive accuracy.

In contrast, TimeGANs, despite producing synthetic data that appeared statistically less similar to the original in marginal distributions, enabled greater forecast accuracy. This was especially apparent in both experiments, where training with real data and testing on TimeGAN data produced the lowest results.

The adversarial training process of TimeGANs appears to encode richer structural and temporal dynamics. Thus, TimeGANs are more effective for downstream predictive modeling and simulation tasks where there is a lack of real data.

6.2 Future Work

Beyond standard metrics such as Pearson's correlation and KS scores, future evaluations should incorporate temporal indicators such as autocorrelation, lag analysis, and cross-correlation to assess how well synthetic data reflect time-dependent behaviors. Importantly, the integration of agent-based simulations can model how behavioral dynamics in net-zero communities affect both real energy consumption and synthetic data quality. Incorporating such behavior-driven scenarios could improve both the realism and predictive utility of synthetic datasets. We also consider incorporating time series plots of forecast outputs to visually assess how well models capture temporal dynamics and respond to short-term fluctuations. This would help identify systematic errors over time under different synthetic data conditions.

6.3 Limitations

KDE preserves marginal distributions well, but struggles with complex multivariate dependencies, limiting its forecasting utility. TimeGAN captures temporal and structural patterns more effectively, but introduces higher variance, which can be problematic in precision-critical applications.

A key limitation of this study is the disparity in dataset size. Although the real dataset had 70,000 rows, only 7,000 synthetic samples could be generated due to the high computational cost of KDE and TimeGAN. This may have affected model performance, particularly for ANN, which benefits from larger datasets. However, our findings show that synthetic data can be a valuable tool for predicting energy demand, provided there are resources that enable the generation of sufficient data.

7 CONCLUSION

This study compared synthetic data generation using KDE and TimeGANs to simulate and forecast electricity demand in smart city applications. Although KDE achieved strong distributional similarity to the original data, it consistently underperformed in predictive modeling. TimeGAN, in contrast, produced data with slightly lower statistical similarity but much stronger simulation and forecast accuracy, thanks to its ability to capture temporal dependencies and structural patterns in energy usage. This highlights a key trade-off: Generating data using KDE excels in replicating distributions, while TimeGAN better supports modeling tasks that require realistic temporal behavior. For applications like energy forecasting—where capturing evolving dynamics matters more than strict distributional fidelity—TimeGANs offer a more effective solution. In addition, this work demonstrated that by adjusting TimeGAN's parameters, synthetic scenarios such as energy demand surges due to, for instance, increase in electric vehicle driving in smart cities can be generated. These realistic simulations can help train forecasting and monitoring models used in digital twins or grid simulations, enhancing their robustness under evolving future conditions.

REFERENCES

Abhari, A. 2018. *Topics in Data Science with Practical Examples*. CreateSpace. pp. 75–88.

Arjovsky, M., S. Chintala, and L. Bottou. 2017. "Wasserstein GAN". *arXiv preprint arXiv:1701.07875*.

Berger, V. W., and Y. Zhou. 2014. "Kolmogorov–smirnov test: Overview". *Wiley statsref: Statistics reference online*.

Bramer, M. 2007. "Avoiding overfitting of decision trees". *Principles of data mining*:pp. 119–134.

Giannelos, S., D. Pudjianto, T. Zhang, and G. Strbac. 2025. "Energy Hub Operation Under Uncertainty: Monte Carlo Risk Assessment Using Gaussian and KDE-Based Data". *Energies* 18(7) https://doi.org/10.3390/en18071712.

Han, S., C. Qubo, and H. Meng. 2012. "Parameter selection in SVM with RBF kernel function". In *World Automation Congress 2012*, pp. 1–4. IEEE.

Jacob, M., C. Neves, and D. Vukadinović Greetham. 2020. *Short Term Load Forecasting*, pp. 15–37. Cham: Springer International Publishing https://doi.org/10.1007/978-3-030-28669-9_2.

Li, P., Y. Pei, and J. Li. 2023. "A comprehensive survey on design and application of autoencoder in deep learning". *Applied Soft Computing* 138:110176.

Masood, M. A., R. A. Abbasi, and N. W. Keong. 2020. "Context-aware sliding window for sentiment classification". *IEEE Access* 8:pp. 4870–4884.

Mohd Pozi, M. S., and M. H. Omar. 2020, 11. "A Kernel Density Estimation Method to Generate Synthetic Shifted Datasets in Privacy-Preserving Task" https://doi.org/10.22667/JISIS.2020.11.30.070.

Peteleaza, D., e. a. 2024. "Electricity consumption forecasting for sustainable smart cities using machine learning methods". *Internet of Things* 27:101322 https://doi.org/https://doi.org/10.1016/j.iot.2024.101322.

Plesovskaya, E., and S. Ivanov. 2021. "An empirical analysis of KDE-based generative models on small datasets". *Procedia Computer Science* 193:pp. 442–452.

Struye, J., F. Lemic, and J. Famaey. 2022, October. "Generating Realistic Synthetic Head Rotation Data for Extended Reality using Deep Learning". In *Proceedings of the 1st Workshop on Interactive eXtended Reality (IXR '22)*, pp. 1–10. Lisboa, Portugal: ACM https://doi.org/10.1145/3552483.3556458.

Tso, G. K., and K. K. Yau. 2007. "Predicting electricity energy consumption: A comparison of regression analysis, decision tree and neural networks". *Energy* 32(9):pp. 1761–1768.

Wang, Y., H. Yao, and S. Zhao. 2016. "Auto-encoder based dimensionality reduction". *Neurocomputing* 184:pp. 232–242 https://doi.org/https://doi.org/10.1016/j.neucom.2015.08.104. RoLoD: Robust Local Descriptors for Computer Vision 2014.

Werbos, P. J. 1990. "Backpropagation through time: what it does and how to do it". *Proceedings of the IEEE* 78(10):pp. 1550–1560.

Yoon, J., D. Jarrett, and M. van der Schaar. 2019. "Time-series Generative Adversarial Networks". In *Advances in Neural Information Processing Systems (NeurIPS)*. Vancouver, Canada. NeurIPS 2019.

Zhang, F., and L. J. O'Donnell. 2020. "Support Vector Regression". In *Machine Learning*. Elsevier. pp. 123–140.

AUTHOR BIOGRAPHIES

SINA PAHLAVAN is a Master's student in the Department of Computer Science at Toronto Metropolitan University, Toronto, ON, CANADA. His research interests include data science and computer vision for sports analysis. His email address is spahlavan@torontomu.ca.

WAEL SHABANA is a Ph.D. student in the Department of Computer Science at Toronto Metropolitan University, Toronto, ON, CANADA. His research interests include generative deep learning. His email address is wael.shabana@torontomu.ca.

ABDOLREZA ABHARI is a professor in the Department of Computer Science at Toronto Metropolitan University and the director of the DSMP lab (http://dsmp.ryerson.ca). He holds a Ph.D. in Computer Science from Carleton University. His research interests include web data modeling and simulation, Data Science and social networks, AI and Agent systems, Network Simulation, and distributed systems. His email address is aabhari@torontomu.ca and his home page is https://cs.torontomu.ca/~aabhari/.