2024 IEEE 20th International Conference on Intelligent Computer Communication and Processing (ICCP) | 979-8-3315-3997-9/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICCP63557.2024.10792997

Advancing IoT Data Utilization: Generating and Evaluating Synthetic Time Series Data

Raluca-Laura Portase Corina-Mădălina Dragotoniu
Camelia Lemnaru Mihaela Dinsoreanu Rodica Potolea
Computer Science Department of Technical University, Cluj-Napoca

***Abstract*—Synthetic data generation plays a crucial role in various domains where access to real-world datasets is limited or restricted due to legal, ethical, or privacy concerns. Given the growing need for realistic data in analysis and research, we explore three distinct methods to capture real time series data's temporal and distributional characteristics. We provide a comparative study of these methods by using a set of evaluation metrics, highlighting the strengths and benefits of each approach. This analysis offers multiple perspectives on the utility and applicability of the proposed synthetic data generation techniques. Additionally, we investigate the impact of synthetic data when forecasting time series, shedding light on its potential for enhancing predictive models.**

***Index Terms*—Synthetic data, Time series decomposition, Data analysis, Machine learning, Evaluation metrics**

I. Introduction

Due to the advantages the IoT industry brings in many fields, from healthcare to finance, the quantity of collected data has increased significantly. Its applications vary from facilitating research to helping in the decision-making process and enabling the development of new technologies. However, some challenges also need to be handled, such as the accessibility and availability of quality data. The ability to collect, analyze, and interpret data has become a significant step in modern science, leading to faster evolutionary technological progress.

The advantages of a large variety of data are noticeable in many domains. Educational institutions use data analytics to improve students' learning experience, while financial institutions rely on data to optimize investment plans, prevent fraud, and forecast stock market data. Another example is healthcare, where patient information is a good source for developing predictive models to forecast numerous diseases, create personalized treatments for patients, and even improve the overall process [1]. The impact of data in our daily lives is unimaginable, being not only valuable but also indispensable.

The availability of good-quality datasets with meaningful insights is often restricted, mainly because of legal constraints (private companies hold data) and ethical considerations (there are cases when data is challenging to obtain in a real environment). All these limitations challenge researchers and data analysts who need to use data to validate models, train models, and test hypotheses.

Synthetic data is a powerful solution for all these problems. This kind of data is artificially obtained from real datasets, purposely designed to maintain real data's statistical properties and patterns. By creating new datasets that imitate the complexity of real-world data, synthetic data offers an easily accessible alternative for the scientific community to progress and facilitate discoveries.

This paper proposes multiple approaches to generate synthetic data and conducts a comparative analysis of the results from a data quality perspective. The implemented methods retain the key characteristics of real datasets while offering an open-source resource for scientific research and data analysis. These methods are further applied to time series data, with the results validated using a comprehensive set of metrics. The proposed algorithms represent a novel approach to synthetic data generation compared to similar works [2]–[4]. While artificial data generation has gained significant popularity recently, most existing solutions rely heavily on neural networks [5], [6]. In contrast, two of the methods presented in this paper adopt a mathematical approach to modelling time series, while the third utilizes the TimeGAN model [7] to generate data based on previously trained time series.

The remainder of this paper is organized as follows: Section II provides a brief overview of related work and practical applications of synthetic data generation. Section III offers a more detailed description of the methods and a comparative analysis of their performance. Section IV presents concrete examples of synthetic data generated from several real datasets and thoroughly compares the metrics used to evaluate the results. This section concludes with an analysis of the impact of synthetic data on time series forecasting. Finally, Section V presents the conclusions.

II. Related work

A. Synthetic data generation

IoT devices are nowadays equipped with numerous sensors that monitor various properties and functionalities. The data collected by these sensors comes in diverse forms and must be managed and stored efficiently for future purposes.

One of the most commonly used data types is time series, which represents a collection of sequential values recorded over a specific period, offering a dynamic view of a variable's evolution [8]. Time series analysis is a valuable tool for understanding the key features of data and is widely applicable across many domains due to its ability to capture temporal patterns.

[1]979-8-3315-3997-9/24/$31.00 ©2024 IEEE

As described in [9], time series can be decomposed into three main components: trend, seasonality, and noise (irregular components). The trend component reflects the overall direction of the data over a long period, indicating whether it generally increases or decreases, regardless of short-term fluctuations [10]. Conversely, seasonality refers to regular, repeating patterns that occur at fixed intervals, such as daily, weekly, monthly, or yearly cycles [11]. Incorporating trend and seasonality into synthetic data is crucial for achieving data realism and accuracy, especially when representing the cyclical nature of many real-world processes. Synthetic data is often used in forecasting [12] due to the need for large quantities of data to train and test predictive models. Therefore, synthetic datasets must emulate the patterns and dynamics found in real-world data to create a system that accurately forecasts future values based on historical data.

In [2] the authors evaluate three pre-existing dataset generators: Mostly AI, Gretel.ai, and Synthetic Data Vault. These platforms provide convenient online solutions for generating synthetic data. However, they have limitations, such as not being freely accessible and requiring users to provide real datasets, which can raise concerns about data confidentiality.

Another approach to generating synthetic data is presented in [3], where the authors describe a method for producing synthetic financial time series using generative models. Several types of generative models are explored, including Generative Adversarial Networks (GANs), Variational Autoencoders (VAEs), and Generative Moment Matching Networks (GMMNs). The process begins by generating a fixed-length, one-dimensional array of samples representing one year of financial data. Additionally, the paper compares different types of GAN architectures, such as fully connected and fully convolutional GANs, to assess their performance.

A similar GAN-based solution is presented in [4], which outlines the architecture of a GAN designed for energy data generation. The model consists of a Generator and a Discriminator, which are trained together through adversarial training, where both networks compete in a dynamic process. The paper further explores improved GAN architectures, including Deep Convolutional GAN (DCGAN), Conditional GAN (CGAN), and Wasserstein GAN (WGAN), highlighting their ability to enhance the data generation process.

The authors in [4] also propose a novel architecture called WDCGAN, which integrates the features of DCGAN, CGAN, and WGAN. Utilizing the Wasserstein loss function, this model improves training stability and addresses common issues such as mode collapse and gradient vanishing, offering a more robust solution for generating synthetic energy data.

B. Metrics for Time Series comparison

Apart from visual representations, several metrics can be employed to evaluate the quality of synthetic data and ensure its similarity to the original dataset. Below is a general description of each metric and the expected values for a well-performing synthetic dataset:

- Dynamic Time Warping (DTW) [13]: DTW measures the similarity between two time series by aligning them. A lower value suggests a good similarity between the original and synthetic data, while a higher value indicates major differences.
- Time Alignment Measurement (TAM) [14]: TAM calculates the synchronization between two time series by analyzing their temporal structure. The result should be between 0 and 1, where 0 means perfect alignment and 1 indicates dissimilarity between time series.
- Normalized Compression Distance (NCD) [15]: NCD assesses the similarity between two time series based on their compressibility. The result is between 0 and 1, where a value close to 0 suggests that the generated data is very similar to the real data.
- Kolmogorov Complexity-Based Distance (KCD) [16]: This metric uses the Kolmogorov complexity (the length of the shortest computer program that can generate the signal) to compare two time series. The result should be a value between 0 and 1, where 0 means perfect similarity.
- Compression-Based Similarity Measures (CBSM): CBSM evaluates the similarity between two time series based on their compressed representation. The result is between 0 and 1, where a value close to 0 suggests that the generated data is very similar to the real data.
- Compression-Distance Framework: This method measures the similarity between two time series by analyzing how one is compressed using the other as a reference. There is not a predefined interval of values, but a lower result is expected for a good synthetic dataset.
- Kullback-Leibler Divergence (KL Divergence) [17]: This metric assesses how one probability distribution diverges from the second probability distribution. A lower KL Divergence value suggests that the data distributions are similar.
- Kumar-Johnson Distance: This distance compares the distribution of two datasets. The range of values is [0, 1], where a smaller value suggests that the synthetic data distribution is closer to the original data distribution.
- Wasserstein Distance [18]: This metric determines the minimum effort required to transform one distribution into another. A value close to 0 indicates a strong relationship between synthetic and real data.
- Maximum Mean Discrepancy (MMD) [19]: MMD measures the difference between the distribution of two datasets. The result should be between 0 and 1, where 0 means a perfect similarity in distribution.

III. Methods

Synthetic data plays a crucial role in data analysis and forecasting algorithms, particularly when access to real-world data is restricted due to ethical or legal constraints. This section presents three distinct methods for generating synthetic time series data and offers a comparative analysis of their strengths, weaknesses, and underlying approaches.

A. DataFusion Method

The first solution is designed to generate synthetic time series based on a real time series with daily values. Initially, we decompose the time series into three key components: trend, seasonal component, and irregular component:

$$y_t = T_t + S_t + R_t$$

where y_t is the time series, T_t is the trend component S_t is the seasonal component, and R_t is noise or the irregular component. This decomposition allows us to leverage the trend and seasonal components from the original dataset to create the synthetic time series, ensuring that key patterns are preserved.

Generating a synthetic time series from a real dataset involves several steps, each designed to preserve the integrity of the original data while analyzing its features. Figure 1 visually represents this method's flow.

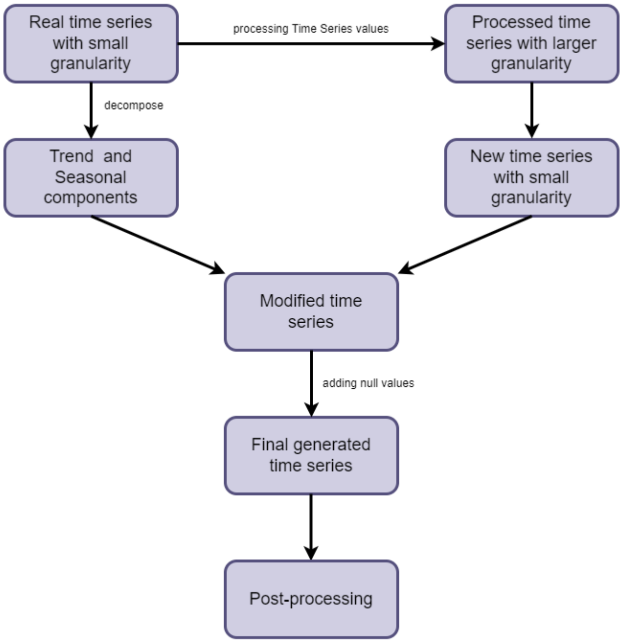

Real time series with small granularity
processing Time Series values
Processed time series with larger granularity
decompose
Trend and Seasonal components
New time series with small granularity
Modified time series
adding null values
Final generated time series
Post-processing

Fig. 1: Flow diagram of the DataFusion method

The first step of the algorithm consists on the preparation of the original time series, ensuring that the data is correctly formatted with dates as the primary index. Missing values are handled to ensure accurate analysis.

Next, the time series data is aggregated to a higher level of granularity. For example, if the original sampling rate is a day, the next logical granularity level for data is a week. This step involves calculating the sum of the small granularity data to compute the higher granularity one (for example, the daily values over seven days to generate a weekly time series), which serves as the foundation for creating the synthetic time series.

After the aggregation step, a new small granularity time series is generated using the values of the higher granularity one as reference points. This can be achieved, for example, by applying weighted averages of weekly values to each day, with the weights based on predefined seasonal factors and day-of-week variations. These weights ensure the synthetic daily time series captures realistic fluctuations.

The next step involves performing a seasonal decomposition of the original time series into its main components: trend, seasonal, and residual (irregular). These components are critical for preserving the structure of the data.

Once the trend and seasonal components are extracted, they replace the corresponding elements in the generated time series, ensuring that the synthetic data captures the key features of the original time series.

Finally, additional adjustments are made to refine the synthetic data further. These adjustments may include introducing null values at random positions or applying minimum thresholds to simulate missing or zero-value data points, thereby better reflecting real-world conditions.

B. ResidualReshape method

ResidualReshape method also considers the key features of a time series: trend and seasonality. Unlike the DataFusion method, it does not create a new fine-granularity time series or replace the trend and seasonality components. Instead, it generates a synthetic time series by introducing a new residual component into the original data, thereby creating a completely new artificial dataset while preserving the most important characteristics of the original series.

A complete pipeline of this method is represented in Figure 2.

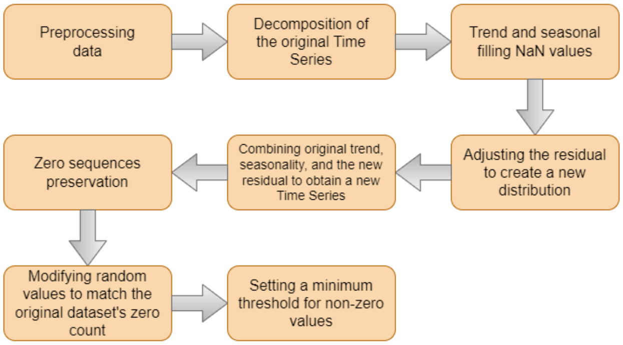

Preprocessing data
Decomposition of the original Time Series
Trend and seasonal filling NaN values
Adjusting the residual to create a new distribution
Combining original trend, seasonality, and the new residual to obtain a new Time Series
Zero sequences preservation
Modifying random values to match the original dataset's zero count
Setting a minimum threshold for non-zero values

Fig. 2: Pipeline for ResidualReshape method

The process begins by decomposing the original time series into its fundamental components: trend, seasonal, and residual. An additive decomposition technique can achieve this, helping to better understand the series' temporal dynamics.

Once the components are separated and stored, the algorithm processes them by addressing any invalid data points (such as NaN values). A specialized function replaces these invalid values with estimates based on neighbouring points, ensuring the time series remains intact.

The next step is to generate the synthetic time series. The residual component is modified in this step to introduce new values, adding variability and fluctuations that simulate real-world noise. Random values within predefined ranges are infused into the residual component, mimicking the unpredictable oscillations observed in the original time series. Additionally, sequences of null values from the original dataset are preserved, maintaining the temporal structure. Further adjustments, such as applying minimum thresholds, are made to ensure consistency and usability for subsequent analysis.

C. DataSynthetizedGAN method

For our third approach, we used TimeGAN, a generative model specifically designed for time series data for the last method. TimeGAN extends the main principles of a Generative Adversarial Network for sequential data, becoming a good option for generating synthetic time series that preserve the complex temporal features and relations present in the original data. From an architectural point of view, a TimeGAN consists of two major components: a Generator and a Discriminator [20]. These components are specially designed to work with sequential data, with a recurrent neural network architecture at their core.

In our implementation of TimeGAN, we carefully configured the hyperparameters to optimize performance and obtain good characteristics in the synthetic dataset: the sequence length used is 365, which reflects the daily nature of the time series grouped in years; the number of sequences is 100 to produce a variety of synthetic samples that capture the major features of the original data, and the model has 256 hidden dimensions, being able to capture complex temporal patterns and relations. The noise dimension was set to 512 to increase the generated time series' variability and realism, and the model dimension was decided to be 256 in order to control the complexity of the model architecture and balance model capacity and its efficiency. We set the batch size to 128, which brings accessible optimization and parameter updates; the learning rate of 1e-4 permits a good regulation of the parameter updates, and an interval of 100 steps in monitoring the training process helps to track the model's performance. Finally, the training has a total of 5000 epochs to ensure good learning and adaptation of the model to the real data distribution.

By implementing a TimeGAN with these parameter settings to generate synthetic data, we want to provide an alternative method for generating artificial time series compared to the methods described in this section.

IV. Application of methods on real data

The first dataset used in this paper, the household devices running time set, includes daily usage data of smart home appliances over the course of one year, measured in seconds. Each data point represents the total operational time of an IoT device on a specific day. This dataset provides valuable insights into usage patterns and dynamics over an extended period, making it an ideal foundation for synthetic data generation.

We selected three devices from each usage category to evaluate the methods: low-used, medium-used, and highly used devices [21]. This selection ensures a representative sample, allowing us to assess the methods' effectiveness across various usage patterns and behaviours.

To further demonstrate the general applicability of our synthetic data generation methods, we applied them to additional datasets that capture diverse temporal patterns and behaviours. These datasets include:

- The Daily Minimum Temperatures dataset [22], which contains daily temperature recordings from Melbourne and is well-suited for weather-related time series analysis.
- The Daily Total Female Births dataset [23], which tracks daily birth counts and provides insights into demographic trends.
- The Daily Delhi Climate dataset [24], which includes four attributes—temperature, humidity, wind speed, and mean pressure—offering a multi-dimensional view of Delhi's climate.

This broad range of datasets, encompassing weather data, climate profiles, and demographic trends, demonstrates that each method is capable of generating realistic and useful synthetic time series across different domains. This validation highlights the robustness and versatility of the methods in various contexts."

A. Evaluation by visual comparison

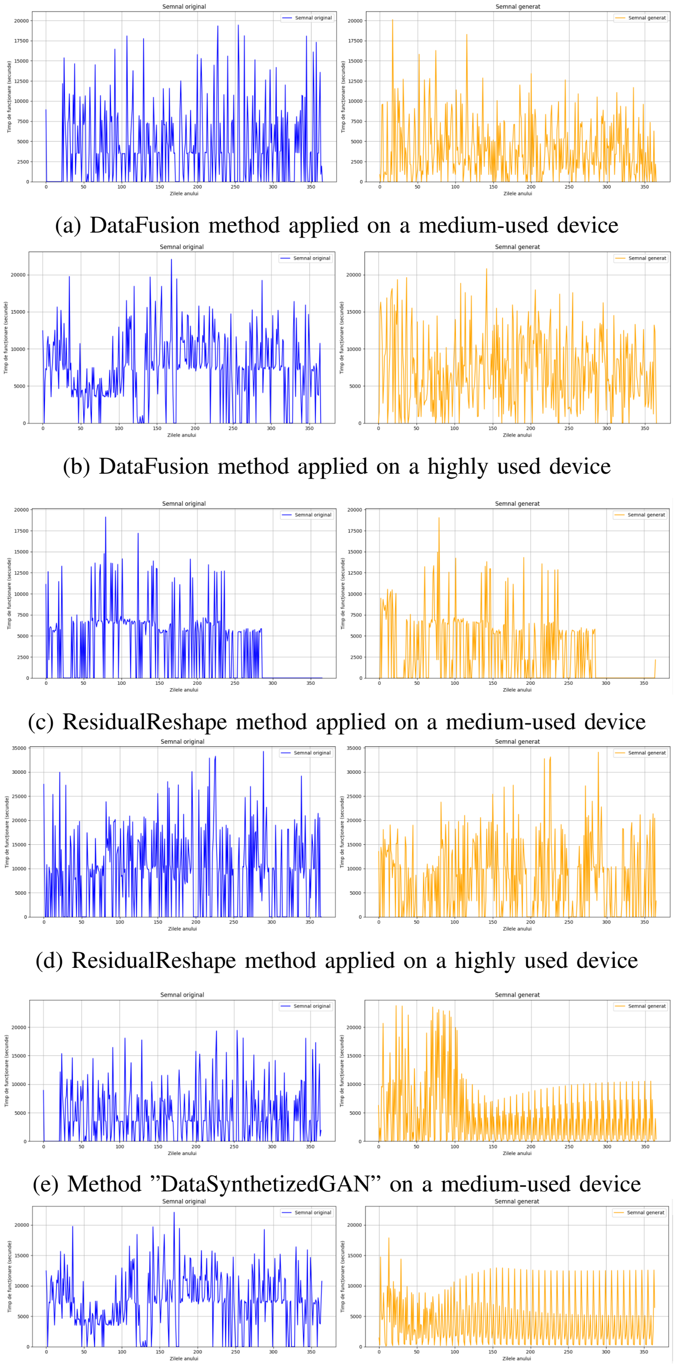
Visual comparison is a fundamental and cost-effective method for assessing the quality of synthetic data generated by the proposed techniques. By plotting both real and synthetic time series, we can identify similarities and differences, thereby evaluating how well each method captures and simulates the main features of the time series, regardless of its complexity.

We applied the proposed methods to the real household devices' running time set, focusing on a representative subset of medium-used and highly-used appliances. The results are illustrated in Figure 3, providing a visual comparison of the real and synthetic data to demonstrate the effectiveness of each method.

B. Evaluation using time series comparison metrices

We apply the evaluation metrics from Section 2B to assess the performance of each method in generating synthetic time series data for various household devices. The tables I, II, and III provide a detailed breakdown of the results of all these metrics applied to the three methods in generating synthetic time series for each household device.

The first three household devices (D1, D2, D3) represent low-usage appliances, which typically exhibit sparse or irregular activity patterns. These devices may operate infrequently; thus, the challenge lies in generating synthetic data that accurately reflects these intermittent usage patterns. The next set of devices (D4, D5, D6) corresponds to medium-usage appliances. The methods evaluated must balance reflecting these recurring patterns while accommodating occasional fluctuations. The final group of devices (D7, D8, D9) consists of high-usage appliances. These devices are characterized by frequent and intensive operation and often show well-defined



(a) DataFusion method applied on a medium-used device

(b) DataFusion method applied on a highly used device

(c) ResidualReshape method applied on a medium-used device

(d) ResidualReshape method applied on a highly used device

(e) Method ”DataSynthetizedGAN” on a medium-used device

(f) DataSynthetizedGAN method applied on a highly used device

Fig. 3: Visual comparison of the proposed methods applied on the medium-used and highly-used devices

daily or weekly usage cycles. The synthetic time series for these devices must not only replicate the high intensity of use but also preserve the temporal consistency and correlations present in real-world data.

Overall, all three methods obtained promising results in generating synthetic time series that imitate reality, each proving unique advantages and areas for future improvement.

The first method, DataFusion, demonstrates a solid foundation by capturing the trends and patterns present in the original data. Even though it is a simplistic approach, it performs well across different metrics, showing a valuable understanding of the real data's temporal dynamics and structural features.

The ResidualReshape method offers significant improvements over the DataFusion in terms of accuracy and fidelity. It reproduces the time series's temporal dependencies and distributional properties well, resulting in synthetic data that mirrors real-world information.

The DataSynthetizedGAN method showcases the immense potential of generating synthetic data by implementing TimeGAN. The results show high consistency and complexity in the data, which is a valuable foundation for future analysis in this domain of generating synthetic data using neural networks.

Compared to the first method, ResidualReshape proves superior performance across most of the metrics, obtaining approximately half the values of the first solution in terms of DTW distance, TAM values, or Compression-Distance Framework (CDF) scores. This indicates its appropriateness in expressing the temporal dynamics, distributional features, and overall similarity between real and synthetic time series. Other metrics, such as Kullback-Leibler divergence and Kumar-Johnson distance, show good scores for all three methods. While DataSynthetizedGAN performed well compared to ResidualReshape, its results are similar to those obtained for the first method. The high TAM scores and low DTW distances suggest good alignment and similarity to real data.

Overall, all three methods obtained promising results, being a reliable and effective approach for generating synthetic time series. The ResidualReshape method stands out as the most satisfactory solution. It consistently produces synthetic data that imitates real datasets, demonstrating superior complexity and distributional similarity. DataFusion method offers a good alternative, while DataSynthetizedGAN is very promising in some aspects, being a good starting point for future refinement.

We applied the same methods to three additional datasets: Daily Minimum Temperatures (Temp), Daily Total Female Births (Births), and Daily Delhi Climate (Climate 1, Climate 2, Climate 3, Climate 4). A comparative analysis of the results is presented in Table IV. The findings indicate that the performance of each metric and method is consistent with the results obtained from the household devices running time set. This suggests that the methods for generating synthetic time series are effective across diverse datasets. The consistency observed across different types of data (ranging from temperature and climate variables to demographic information) demonstrates the robustness and general applicability of the synthetic data generation methods in producing realistic and comparable synthetic datasets.

C. Impact of synthetic data on time series forecasting

One of the primary purposes of synthetic data is its utility in forecasting models. To assess this, we use the generated synthetic data in forecasting tasks, employing the Random

TABLE I: Results obtained when using DataFusion method

Device	DTW	TAM	NCD	Kolmogorov d.	CBSM	CDF	Kullback-Leibler div.	Kumar-Johnson d.	Wasserstein d.	MMD
D1	**496272**	0.721	0.810	0.507	0.193	1421	-0.0002	0.0003	**551**	0.262
D2	636985	0.795	0.797	0.450	0.190	1275	-0.0002	0.0002	727	0.300
D3	723296	0.710	0.825	0.525	0.159	1240	3.436e-5	0.0002	1241	0.353
D4	694360	0.851	**0.705**	0.138	0.184	**540**	0.001	0.0005	1688	0.496
D5	1104505	0.788	0.787	**0.093**	0.208	1445	6.506e-5	5.136e-5	993	0.205
D6	946788	**0.625**	0.784	0.384	0.188	1162	0.0002	0.0003	1347	0.326
D7	1802945	0.800	0.793	0.214	0.214	1377	**-8.072e-5**	3.830e-5	2018	0.233
D8	2036895	0.688	0.762	0.261	0.225	1240	1.094e-5	0.0001	2006	0.230
D9	1356181	0.708	0.782	0.246	**0.240**	1657	4.058e-5	**8.895e-5**	794	**0.104**

TABLE II: Results obtained when using ResidualReshape method

Device	DTW	TAM	NCD	Kolmogorov d.	CBSM	CDF	Kullback-Leibler div.	Kumar-Johnson d.	Wasserstein d.	MMD
D1	**216938**	0.270	0.581	0.203	0.361	88	4.252e-5	1.820e-5	**407**	0.153
D2	234455	0.255	0.558	0.241	0.352	40	2.840e-5	2.261e-5	518	0.151
D3	225429	0.242	0.563	0.153	0.335	33	3.878e-5	3.149e-5	461	0.104
D4	496126	0.445	**0.536**	0.056	0.327	**8**	1.142e-5	1.821e-5	464	0.191
D5	426672	0.260	0.553	0.239	0.410	12	2.087e-5	2.100e-5	888	0.165
D6	408246	0.234	0.569	0.218	0.374	79	8.347e-5	8.471e-5	526	0.119
D7	833151	0.266	0.570	0.265	0.368	60	**8.725e-6**	9.014e-6	1441	0.160
D8	969095	0.297	0.584	0.138	0.384	126	7.127e-6	1.487e-5	1132	0.116
D9	429195	**0.168**	0.601	**0.030**	**0.411**	226	1.044e-5	**9.140e-6**	644	**0.101**

TABLE III: Results obtained when using DataSynthetizedGAN method

Device	DTW	TAM	NCD	Kolmogorov d.	CBSM	CDF	Kullback-Leibler div.	Kumar-Johnson d.	Wasserstein d.	MMD
D1	730294	1	0.777	0.336	0.129	757	0.0002	0.0001	1431	0.468
D2	**729644**	0.963	0.780	0.158	0.110	783	**-7.728e-5**	0.0002	1010	0.498
D3	1065257	1	0.821	**0.043**	0.030	934	0.001	0.0002	1413	0.493
D4	738182	0.947	0.786	0.440	0.073	823	0.0001	**8.405e-5**	1292	0.479
D5	1299379	1	0.739	0.286	0.173	625	0.0002	7.569e-5	**928**	0.362
D6	756462	1	0.780	0.223	0.065	824	-1.334e-5	0.0005	2421	0.373
D7	1877616	**0.936**	0.744	0.184	0.158	611	-0.0002	0.0003	4180	0.428
D8	2162752	1	0.718	0.338	0.187	546	1.808e-5	0.0003	6266	0.344
D9	1521725	1	**0.692**	0.110	**0.231**	**449**	2.817e-5	0.0001	3191	**0.164**

Forest model to evaluate its impact on predicting real datasets. This evaluation is essential for validating the quality and applicability of the synthetic data produced by all three methods.

In this approach, we first trained a Random Forest model on a dataset comprising 5,000 real time series from the household devices running cycles dataset, each covering 11 months of data. The goal is to forecast the functionality time of a real device for the final 30 days of the year. A second model was trained on a combined dataset of 5,000 real time series and 1,000 synthetic time series. This model also uses data from the first 11 months for training and focuses on predicting the last month of the year. By comparing the forecasts from this combined dataset with those from the purely real dataset, we can assess the impact of synthetic data on the model's performance.

We evaluate the performance of the Random Forest model using three key metrics: Mean Absolute Error (MAE), Symmetric Mean Absolute Percentage Error (SMAPE), and R-squared (R²). MAE and SMAPE measure the average magnitude of errors in the predictions, while R² indicates the proportion of variance in the dependent variable that the independent variables can explain. R² values range from 0 to 1, where a value close to 1 signifies a good model fit.

Table V presents the results for each metric, showcasing the performance of the Random Forest model on every dataset for each of the three methods. E1, E2, E3, and E4 represent highly used household devices with less than 30% null values.

For the first method, DataFusion, the results show that the Mean Absolute Error (MAE) and R-squared (R²) values remain consistent between real data and mixed datasets. For instance, the MAE slightly decreases from 2560.39 to 2532.90 for device D1. However, the Symmetric Mean Absolute Percentage Error (SMAPE) value increases for mixed data, particularly for devices D2 and D3. This suggests that while the synthetic data does not negatively impact the forecasting model's performance, an increase in SMAPE values is observed, which is expected.

Similarly, for the DataSynthetizedGAN method, the differences between real and mixed data results are minimal. For example, for device D2, the SMAPE rises from 28.36% to 33.70%, while the MAE decreases from 3136.46 to 2991.40, and the R² value improves from 0.48 to 0.50. These results indicate that the forecasting model's performance is comparable for both real and mixed datasets.

The ResidualReshape method demonstrates a significant improvement in results for mixed datasets. In the case of device D1, the MAE decreases substantially from 2604.04 to 647.80, and the SMAPE drops from 36.57% to 20.09%, which shows

TABLE IV: Data generation metrics results for other datasets

Dataset	Method	DTW	NCD	Kolmogorov d.	CBSM	CDF	Kullback-L. div.	Kumar-J. d.	Wasserstein d.	MMD
Temp	DataFusion	1253	0.822	0.077	0.033	2059	0.075	0.148	2	0.144
	ResidualReshape	221116	0.548	0.074	0.296	408	1	0.356	61	0.589
	DataSynthetizedGAN	744358	0.766	0.105	0.253	579	0.301	0.076	726	0.486
Births	DataFusion	2542	0.821	0.079	0.100	1588	0.287	0.165	5	0.265
	ResidualReshape	22467	0.683	0.020	0.213	390	0.511	0.167	52	0.575
	DataSynthetizedGAN	692919	0.639	0.076	0.303	926	0.853	0.008	505	0.387
Climate 1	DataFusion	1071	0.686	0.143	0.322	303	0.351	0.115	4	0.256
	ResidualReshape	6251	0.497	0.526	0.299	144	1	0.339	45	0.618
	DataSynthetizedGAN	133953	0.841	0.238	0.288	1226	0.299	0.024	940	0.503
Climate 2	DataFusion	3784	0.682	0.290	0.342	264	0.229	0.078	29	0.245
	ResidualReshape	8745	0.464	0.470	0.331	163	0.253	0.092	54	0.520
	DataSynthetizedGAN	430911	0.779	0.227	0.179	779	1	0.103	313	0.388
Climate 3	DataFusion	403	0.682	0.044	0.410	257	0.166	0.042	2	0.186
	ResidualReshape	8065	0.462	0.176	0.362	170	1	0.458	66	0.595
	DataSynthetizedGAN	268445	0.901	0.099	0.337	838	0.476	0.069	1028	0.681
Climate 4	DataFusion	6522	0.713	1	0.321	442	0.043	0.012	555	0.364
	ResidualReshape	15107	0.550	1	0.430	173	0.012	0.026	119	0.371
	DataSynthetizedGAN	853728	0.647	0.173	0.284	641	0.002	0.047	764	0492

TABLE V: Forecasting results obtained for real data set compared to real and augmented data set

Device	Data	”DataFusion”			”ResidualReshape”			”DataSynthetizedGAN”		
		MAE	SMAPE	R^2	MAE	SMAPE	R^2	MAE	SMAPE	R^2
E1	Real	2560.39	36.07%	0.32	2604.04	36.53%	0.31	2560.39	36.07%	0.32
	Mixed	**2532.90**	48.97%	0.31	647.80	20.29%	0.78	2675.58	37.37%	0.28
E2	Real	3136.46	28.36%	0.48	3079.17	27.92%	0.49	3136.46	28.36%	0.48
	Mixed	3121.85	41.62%	**0.48**	530.80	12.88%	**0.97**	**2991.40**	33.70%	**0.50**
E3	Real	5023.34	33.46%	0.39	5108.16	34.06%	0.37	5023.34	33.46%	0.39
	Mixed	5090.09	47.01%	0.36	1048.10	**8.13%**	0.91	5140.27	47.28%	0.34
E4	Real	1379.53	31.40%	0.25	1409.32	31.84%	0.24	1379.53	31.40%	0.25
	Mixed	1379.37	**31.38%**	0.25	**348.49**	21.94%	0.94	1443.28	**32.44%**	0.22

enhanced forecasting accuracy and consistency. Additionally, the R^2 values for all forecasted devices increase considerably to around 0.8 and 0.9, indicating a better explanation of the variability in the data with mixed datasets. These improvements highlight that the ResidualReshape method provides the highest quality synthetic data among the methods tested.

V. Conclusions

In conclusion, this paper thoroughly examines three distinct methods for generating synthetic time series data, each contributing uniquely to the scientific field. Through extensive experimentation and evaluation, we have demonstrated the effectiveness of these methods in capturing both the temporal and distributional characteristics of real-world data.

This paper's primary contribution is its comparative analysis of these methods, showcasing their strengths and advantages in the context of synthetic data generation. The resulting data sets are made publicly available [25]. This research highlights the methods' capabilities and lays the groundwork for future studies in artificial data generation. It is a step towards advancements in various domains, such as anomaly detection, forecasting, and predictive modelling. With the foundation established by this study, researchers can continue innovating in synthetic data generation, thereby creating new opportunities for discovery and advancement in data-driven research.

References

[1] K. El Emam, L. Mosquera, and R. Hoptroff, *Practical synthetic data generation: balancing privacy and the broad availability of data*. O'Reilly Media, 2020.

[2] D. Pavlov, “Comparison of synthetic data generation tools using internet of things data,” *VU Bachelor Thesis*, 2022.

[3] M. DOGARIU, L. Stefan, B. A. Boteanu, C. Lamba, B. Kim, and B. Ionescu, “Generation of realistic synthetic financial time-series,” *ACM Trans. Multimedia Comput. Commun. Appl*, vol. 37, no. 4, 2018.

[4] J. Li, Z. Chen, L. Cheng, and X. Liu, “Energy data generation with wasserstein deep convolutional generative adversarial networks,” *Energy*, vol. 257, p. 124694, 2022.

[5] 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.

[6] C. Zhang, S. R. Kuppannagari, R. Kannan, and V. K. Prasanna, “Generative adversarial network for synthetic time series data generation in smart grids,” in *2018 IEEE international conference on communications, control, and computing technologies for smart grids (SmartGridComm)*. IEEE, 2018, pp. 1–6.

[7] S. Nord, “Multivariate time series data generation using generative adversarial networks: Generating realistic sensor time series data of vehicles with an abnormal behaviour using timegan,” 2021.

[8] P. Esling and C. Agon, “Time-series data mining,” *ACM Computing Surveys (CSUR)*, vol. 45, no. 1, pp. 1–34, 2012.

[9] L. Franzini and A. Harvey, “Testing for deterministic trend and seasonal components in time series models,” *Biometrika*, vol. 70, no. 3, pp. 673–682, 1983.

[10] J. Verbesselt, R. Hyndman, G. Newnham, and D. Culvenor, “Detecting trend and seasonal changes in satellite image time series,” *Remote sensing of Environment*, vol. 114, no. 1, pp. 106–115, 2010.

[11] P. G. Gould, A. B. Koehler, J. K. Ord, R. D. Snyder, R. J. Hyndman, and F. Vahid-Araghi, “Forecasting time series with multiple seasonal patterns,” *European Journal of Operational Research*, vol. 191, no. 1, pp. 207–222, 2008.

[12] G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[13] M. Müller, "Dynamic time warping," *Information retrieval for music and motion*, pp. 69–84, 2007.

[14] D. Folgado, M. Barandas, R. Matias, R. Martins, M. Carvalho, and H. Gamboa, "Time alignment measurement for time series," *Pattern Recognition*, vol. 81, pp. 268–279, 2018.

[15] P. M. Vitányi, F. J. Balbach, R. L. Cilibrasi, and M. Li, "Normalized information distance," *Information theory and statistical learning*, pp. 45–82, 2009.

[16] D. T. Mihailović, G. Mimić, E. Nikolić-Djorić, and I. Arsenić, "Novel measures based on the kolmogorov complexity for use in complex system behavior studies and time series analysis," *Open Physics*, vol. 13, no. 1, 2015.

[17] M. Tumminello, F. Lillo, and R. N. Mantegna, "Kullback-leibler distance as a measure of the information filtered from multivariate data," *Physical Review E—Statistical, Nonlinear, and Soft Matter Physics*, vol. 76, no. 3, p. 031123, 2007.

[18] E. Bernton, P. E. Jacob, M. Gerber, and C. P. Robert, "Inference in generative models using the wasserstein distance," *arXiv preprint arXiv:1701.05146*, vol. 1, no. 8, p. 9, 2017.

[19] G. K. Dziugaite, D. M. Roy, and Z. Ghahramani, "Training generative neural networks via maximum mean discrepancy optimization," *arXiv preprint arXiv:1505.03906*, 2015.

[20] T. Juneja, S. B. Bajaj, and N. Sethi, "Synthetic time series data generation using time gan with synthetic and real-time data analysis," in *The International Conference on Recent Innovations in Computing*. Springer, 2022, pp. 657–667.

[21] R. L. Portase, R. Tolas, and R. Potolea, "From sensors to insights: An original method for consumer behavior identification in appliance usage," *Electronics*, vol. 13, no. 7, p. 1364, 2024.

[22] S. Faraday, " Daily Minimum Temperatures in Melbourne," https://www.kaggle.com/datasets/samfaraday/daily-minimum-temperatures-in-me, accessed: 16.09.2024.

[23] T. Tran, " Daily Total Female Births," https://www.kaggle.com/datasets/tientd95/dailytotalfemalebirths, accessed: 16.09.2024.

[24] Sumanthvrao, " Daily Climate time series data," https://www.kaggle.com/datasets/sumanthvrao/daily-climate-time-series-data, accessed: 16.09.2024.

[25] Dragotoniu, " Synthetic Time Series IoT Devices," https://www.kaggle.com/datasets/corinadragotoniu/synthetic-time-series-iot-devices, accessed: 23.09.2024.