

Proceedings of Statistics Canada Symposium 2024: The Future of Official Statistics

Exploration of Deep Learning Synthetic Data Generation for Sensitive Utility Data Sharing

by Benjamin Santos, Rafik Chemli, and Julian Templeton

Release date: September 8, 2025



Statistics
Canada

Statistique
Canada

Canada

Exploration of Deep Learning Synthetic Data Generation for Sensitive Utility Data Sharing

Benjamin Santos, Rafik Chemli, and Julian Templeton¹

Abstract

Utilities hold crucial information about energy usage and building characteristics which can be utilized by government agencies to improve their corresponding analytics. However, this data is associated with private customer records and thus the building data and energy usage may be too sensitive to share. Often, high-level aggregated versions of this data are shared through robust contracts, limiting the statistics that can be derived. With the advancement of generative machine learning techniques, Statistics Canada and Natural Resources Canada have explored the feasibility of using these models to produce synthetic versions of utility data which may be shared in full to requesting organizations. These synthetic datasets can be created by a utility company through a locally run program and the outputs can be approved before being sent. This work has identified that certain generative models can feasibly be used by utilities to generate new versions of a dataset and has identified the issues which must be addressed prior to implementing this in practice. Both tabular and time-series models have been tested for different data sharing scenarios, where the TimeGAN model successfully captured the general energy peaks and valleys over a given day with reasonable computational requirements. Although this process takes days for annual energy amounts over thousands of customer records, this can enable new data sharing initiatives between utilities and National Statistical Offices while managing privacy risks. As work progresses in future phases with real utility partners, trust can be built for these approaches, and they can begin being tested on real data by actual data holders.

Key Words: Synthetic data; Time series; Generative models; Data sharing; Confidentiality; Utility data.

1. Introduction

Utility data, crucial for building analytics, is sensitive and often restricted due to privacy concerns. Natural Resources Canada (NRCAN) aims to collect and utilize utility data from varying sources to enhance the research and analysis they are doing. However, the customers' privacy, whose records are within the utility datasets, is one of several concerns for the data holders. Thus, there are significant hurdles to accessing this data which can lead to long delays in performing any analysis which would need to use this data. To overcome these issues, we propose the use of modern deep learning synthetic data (SD) generation models. This approach offers the potential to balance data utility with privacy, enabling enhanced analytics without compromising customer confidentiality. In this article, we will give an overview of the work conducted within the project and provide clear discussion points from the tests performed. This will start with a section clearly outlining the project's scope and limitations. Next, we outline all background information surrounding the data and models used. Then we report the tests conducted. The article concludes by summarizing the work and outlining future steps.

1.1 Scope

This research aims to answer three questions regarding the use of deep learning SD generation techniques when applied to sensitive utility data: 1) What are some of the available methods to be used for generating quality SD?, 2) Based on testing a subset of these models, are these computationally feasible to run and are they able to capture the general distributions and trends (i.e. energy peaks and valleys) of the original data?, 3) From the testing outputs and discussions held, what are the key factors which must be considered to transition this research, if successful, into practice?

¹ Statistics Canada, 150 Tunney's Pasture, Canada, K1A 0T6 (benjamin.santos@statcan.gc.ca)

Some limitations of this work are: 1) we focus only on understanding the runtime feasibility of the methods, whether the SD captures the general distribution of the original data, and whether the energy peaks and valleys could be captured for each day; 2) not all techniques researched could be tested; and 3) based on the computing power available, only basic optimization is performed.

2. Background

2.1 Synthetic Data and Privacy Protections

The SD research field is vast, with many techniques and evaluation criteria defined. Here, we define a synthetic dataset to be one that maintains the same statistical properties and behaviors as the original dataset but is an entirely unique set of data (see fully synthetic data in UNECE, 2022). Each synthetic record represents a fictitious customer record. Ideally, the overall synthetic dataset and trends found within it will resemble that of the original despite being new datapoints. Generative deep learning models aim to generate a new dataset based on the statistical properties learned from the original data. These can be applied to time-series data and can be appropriately tuned for specific tasks, such as maintaining the peaks and valleys of energy data. This new dataset will inherently be different from the original data, adding privacy to the process. These methods do come at a cost despite their potential, typically, they require a Graphics Processing Units (GPU) or accelerators and enough Random Access Memory (RAM), moreover, the time it takes to train a model over hourly measurements of a particular day can vary from 20 minutes to several hours depending on the configuration and type of model used. Furthermore, the evaluation of the SD is difficult to measure and requires a strong understanding of how it will be used (for more details see UNECE, 2022). Thus, there are many tradeoffs to analyze when deciding on the type of generative model to utilize. Despite the model complexity and outputs being a new dataset, using these models can open opportunities in an *all-or-nothing scenario*. That is, when NRCAN cannot retrieve the original data as-is and does not want only basic analytics from a utility's data. While SD is different from the original data, it could be useful for in-depth analysis, it can be useful for exploratory data analysis, and for testing and learning purposes. This can be a big undertaking but can also open the door to new initiatives and insights that could not otherwise be derived.

Since privacy concerns are the forefront issue guiding this research, it is critical to understand the privacy protections provided by the solutions explored. As previously discussed, the SD generated by the models within this work are different from the original datapoints and the quality of the generated data will differ based on the selected model and the hyperparameters it uses. Additionally, some generative models apply differential privacy (DP) in the training process which offers strong mathematical guarantees on the amount of privacy, represented as noise, being added to the data. Models using DP, such as DPGAN (Xie, 2018), will thus provide further privacy guarantees to better protect customer records at a corresponding cost to their usefulness. Outside of the utility, NRCAN would not be able to say with certainty whether it is or is not different from the original, and by how much. Balancing privacy and usefulness of the generated SD is difficult but helps provide more confidence to the utility that their customer's data is protected.

2.2 Tabular and Time-Series Data

Different generative models are designed to work with different types of data, making it imperative to understand what will be used as input and what the outputs should capture. Within this project we are using simulated time-series data relative to customer records of energy consumption for different buildings. Thus, if the goal is to capture the full time-series, the model used must be designed to accept time-series energy consumption as input and to generate data within a time-series format. Here, the data is within hours of a day for every day within an arbitrary year. Therefore, the model will learn from the input time-series and generate a new time-series which sequentially outputs samples to match the total amount used as input. For example, if a model learns a single day's worth of customer records, it will be able to output exactly one day's worth of records for the same number of records used for inputs. Within the captured time-period, the information from prior times is used as context when predicting the results at future times. This is more complex than working with tabular data which does not require the time to be captured within the inputs themselves. Table 2.2.1-1 provides an example of how the data may look within a time-series.

Table 2.2.1-1
Example of time-series datapoints (extract)

datapoint_id	:dcv_type	:ext_wall_cond	hour	energy_elec	energy_gas
a	Occupancy_based	0.278	0	0.252	0.001
a	Occupancy_based	0.278	1	0.211	0.532
a	Occupancy_based	0.278	2	0.157	0.741
b	No_DCV	0.183	0	0.211	0.532
b	No_DCV	0.183	1	0.157	0.741

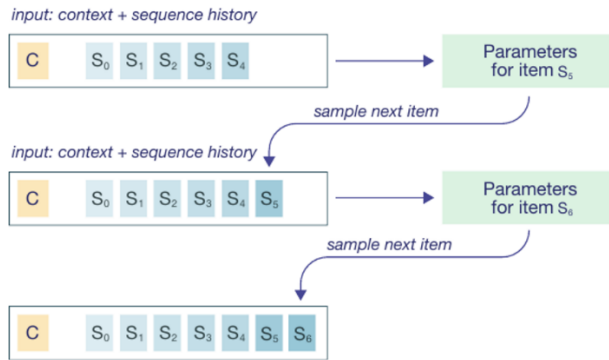
From Table 2.2.1-1, we see that this data is a mixture of static features (building characteristics) and time series data (energy). Even if the data is a time-series, it can also be used as tabular data by ignoring the time aspect. Generative models using tabular model learn how to output individual customer records without providing context on the time. Using these models, we will not capture time-series information, which may be fine if the input data does not contain trends reliant on the time. It may still be useful to work within a tabular setting to capture information that may be challenging within a time-series model. Within the scope of this research, we can use a tabular model to predict a single output per customer record where this record contains auxiliary information from the summarized time-period. For instance, if we know that days will have a single peak and valley in the energy usage for electricity and for gas, we can have the model learn to output SD versions of records for each day, including peak/valley times and values.

2.3 Conditional Probabilistic Auto-Regressive Generative Model

This model called the Conditional Probabilistic Auto-Regressive (CPAR) model (Zhang, 2022) aims to generate SD from real data comprised of a context or static features and a time series. These inputs may contain a mix of data types: numeric, categorical, and time; from Table 2.2.1-1: “:dcv_type” is a categorical feature (it contains a small set of possible values), “energy_elec” can be considered as numeric (continuous), and “hour” is the time. Also note that each time series or sequence (blue or red) has a different set of static features (context).

First, the model splits the context from the sequences. Then, it applies a Gaussian Copula model to the context to understand the correlations between the static features (the building parameters). Specifically, the marginal distributions for each column are extracted and transformed into uniform distributions, then the joint distribution is computed as a multivariate normal distribution which will capture the relationship between the static features. From the Gaussian Copula model, a new context is generated. Next, the model aims to capture the dependencies on the time series using the history of the sequence (S) and the context (C). By doing so, it can generate synthetic time series data. This idea is illustrated in Figure 2.3-1(Zhang, 2022). Thus, a neural network is trained using the sequences as input. Specifically, Gated Recurrent Units (GRU) are used here. Along with Long Short-Term Memory units (LSTM), these are specific implementations over a broader category of networks called Recurrent Neural Networks (RNN). RNNs allow previous outputs to be processed as inputs.

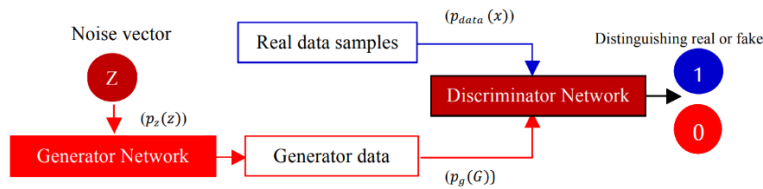
Figure 2.3-1
Example of how the sequence and context are used within the CPAR model.



2.4 Tabular Generative Model

Generative Adversarial Networks (GANs) are generative models consisting of two parts. These two parts are neural networks. The first is a discriminator (D) which estimates the probability of a given sample to be from the real dataset. The second a generator (G) which generates synthetic samples starting from the uniform distribution $p_z(z)$, see figure 2.4-1 (Jabbar, 2021). In other words, D acts as a verifier and aims to not be cheated by G, and G is a forger which tries to create realistic samples to fool D. To improve the performance of D and G, the networks compete against each other during the training process. Thus, they participate in a zero-sum game, where when one of them wins the other loses. Because the goal of G is to learn the real data distribution $p_{data(x)}$, and the goal of D is to discern which are real samples (taken from the real and synthetic samples $p_{g(G)}$), when training, the loss function must consider both G and D losses to improve their tasks.

Figure 2.4-1
The general architecture of a GAN.



2.4.1 Conditional Tabular GAN (CTGAN)

The first results using GANs were the generation of new images. Several variations of GAN models surfaced for tabular data generation, one of them is CTGAN (Xu, 2019) which introduced improvements to handle the type mixing found in tabular data, the imbalance on categorical features, the learning from one-hot encoded vectors, and the conditional generation of samples. Since vanilla GANs cannot deal with these issues, we added CTGAN to this research's experiments.

2.4.2 Anonymization through Data Synthesis using GANs (ADSGAN) and DPGAN

ADSGAN builds on top of the ideas of conditional GANs, and claims to outperform DPGAN, which is a GAN model trained with DP guarantees. DP is defined as follows: for all datasets D_1 and D_2 that differ on a single datapoint, and for all subsets S for a randomized algorithm A (or an ML model or a SD generator) (Yoon, 2020):

$$P(A(D_1) \in S) \leq e^\epsilon P(A(D_2) \in S)$$

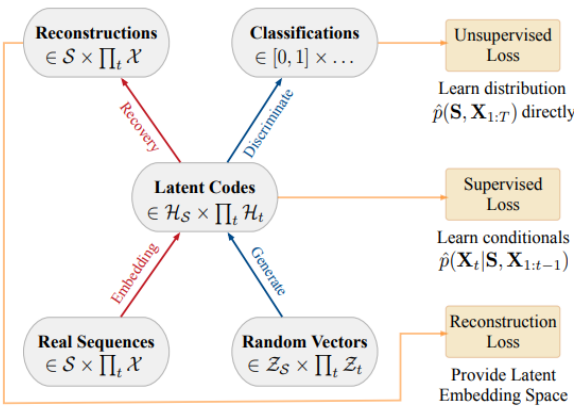
The algorithm or generator A provides ϵ -DP, if the difference in the probability distribution of the generator trained with dataset D_1 and the probability distribution of the generator trained with same dataset excluding one datapoint is

less than e^ϵ . Therefore, DPGAN is said to generate differentially private SD (DPSD). The authors of ADSGAN (Yoon, 2020) argue that DPSD could be of poor quality with respect to having a similar joint distribution to the original dataset (see Gaussian Copulas above), and that DP cannot be directly computed over finite datasets. To overcome these limitations, the authors have developed a different concept called ϵ -identifiability which offers less protection than DP but satisfies their use case on synthesizing health records.

2.5 TimeGAN Model

TimeGAN is a generalization of generators for tabular data with time series (Yoon, 2019). The structure of the data is the same as explained above for the CPAR model. For a given dataset $D = \{(s_n, x_{n,1:T_n})\}_{n=1}^N$ of N datapoints, there is a static set of features \mathbf{S} and sequences \mathbf{X} . TimeGAN tackles two goals. First, to learn a density $\hat{p}(S, X_{1:T})$ which better approximates the underlying joint distribution $p(S, X_{1:T})$ (this is considering all the static features and the sequences). Second, to learn the conditionals $p(X_t; X_{1:T-1})$. The model has four total components: the embedding and recovery functions, the sequence generator, and the sequence discriminator. The first two components are trained jointly with the generator and discriminator (Figure 2.5-1). A prominent aspect of this model is that the “Embedding” and “Recovery” components can be anything with the constraint they must be autoregressive and obey causal ordering. This is why we can use RNNs, LSTMs, GRUs and other autoregressive models for learning the conditionals.

Figure 2.5-1
TimeGAN model architecture.



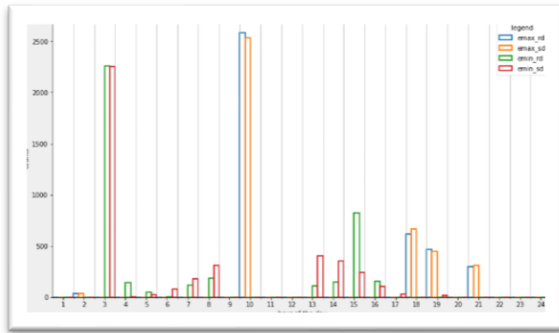
3. Results and conclusions

3.1 Tabular generative model experiments

The idea is to test if the generators can capture the relationship, if any, between the building characteristics (4000 different buildings) and those maximum and minimum of consumption values with the time they occur (peaks and valleys). We focused on a 24-hour window where (t_{max}, e_{max}) and (t_{min}, e_{min}) , are the time of the occurrence of the maximum and its value, and the time of the occurrence of the minimum and its value, respectively, for a specific utility. We tested several generative models from the *synthcity* package (Qian, 2023): **CTGAN**, **ADSGAN**, **DPGAN**, and **marginal distributions**. The overall metrics can be found in the extended report (Templeton, 2024). From those metrics, we conclude that in general **ADSGAN** offers the better utility and privacy trade-off, **DPGAN** degrades the utility and **marginal distributions** the utility and privacy. This is expected due to DP noise being added on the training phase in the case of **DPGAN**, while **marginal distributions** just construct a datapoint by sampling it from the marginal distributions. **CTGAN** performs well and the privacy metrics are not too off from the other methods. In terms of runtime performance, all generators except the marginal distributions take around the same amount of time to be trained: 10 minutes on an AMD 16 core processor and needs at least a GPU with 2Gb of RAM. The marginal

distributions generator is used as a baseline and it trains incomparably fast. Surprisingly, both ADSCAN and CTGAN were able to capture the peaks and valleys (fig. 3.1-1). These observations are also confirmed by the classification metrics (fig. 3.1-1), where we can see that peaks on electricity are very well represented in the SD, on the contrary, the valleys for both utilities have been missed. These metrics are averaged over all the classes (macro average).

Figure 3.1-1
(left) Comparison of counts of real (rd) and synthetic (sd) peak hour times for ADSCAN, (right) Peaks/valleys classification metrics.

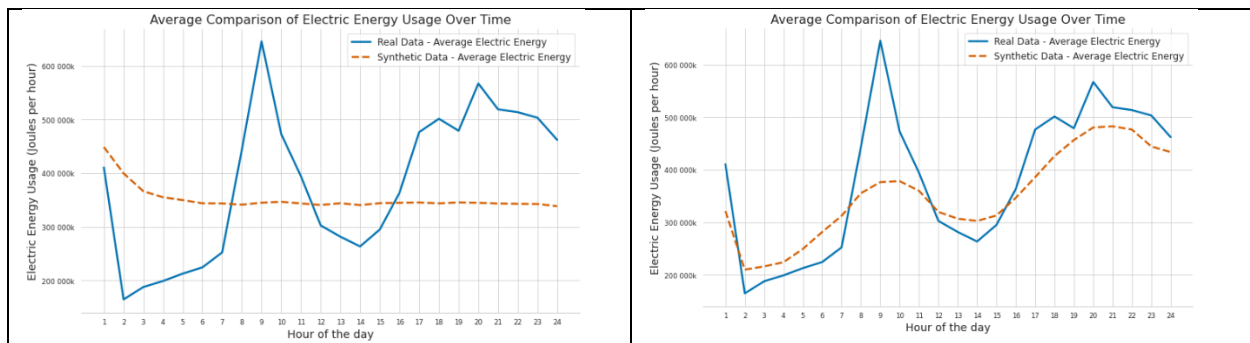


Extremum	Accuracy	Precision	Recall	F1-score
Max electricity	0.98	0.95	0.99	0.97
Min electricity	0.72	0.35	0.41	0.32
Max gas	0.97	0.25	0.25	0.25
Min gas	0.72	0.25	0.27	0.20

3.2 Time-series generative model experiments

In Figure 3.2-2, a comparison between the real and SD of the hourly electricity average of the 24-hour time-series for January 1 is shown for the electricity energy. SD from CPAR (left) seems to average out, meaning that the conditional probability on the sequence was not captured by the model. Note that a hyperparameter exploration is needed to fully assess this model, despite not expecting the model to be great for privacy protection. On the right, we see TimeGAN results using RNN (default) from *synthcity* package, which reflect that the trends are better preserved. In terms of the extrema in energy, we found that there is a higher error regarding the counts of when the hours occur within the synthetic data when compared to the real data. These are worse than what has been observed from the tabular model tests.

Figure 3.2-2
CPAR vs TimeGAN results on hourly averaged electricity for all buildings.



To conclude, we performed an exploratory analysis into the feasibility of using generative deep learning models for the generation of synthetic data of sensitive utility data. We started with an exploration into different methods which can be used to accomplish this task and into the behaviors of the input data being used for testing. With a better understanding of different approaches and the data, we performed tests with CPAR, TimeGAN, and Generative Tabular models. While we did initially observe clear patterns between the distributions of the datasets and the distribution of the original data, additional testing has been conducted to analyze whether the models can maintain the peaks/valleys of energy observed from the original data. This is a challenging task where, within the project's

timeframe, we observed TimeGAN and the Generative Tabular model ending up maintaining the peaks/valleys after a substantial number of experiments. Thus, these models have proven to be feasible potential options to be further explored in future phases. Additional considerations are still required, for instance in the case of sparse gas utility values (not all buildings use gas) could be handled by a separate generative model. Overall, the testing has proven the feasibility of using generative deep learning models for synthetic data generation, but this is only a preliminary look into a large, yet important, undertaking.

References

- Jabbar, A., Li, X., and Omar, B. (2021), "A survey on generative adversarial networks: Variants, applications, and training", *ACM Computing Surveys (CSUR)*, 54(8), pp. 1-49.
- Qian, Z., Cebere, B. C., and Van der Schaar, M. (2023), "Synthcity: facilitating innovative use cases of SD in different data modalities", *arXiv preprint*, arXiv:2301.07573.
- Templeton, J., Santos, B. and Chemli, R. (2024), "Exploration of Deep Learning Synthetic Data Approaches to Help Retrieve Sensitive Utility Data", Statistics Canada, internal report.
- UNECE. (2022), "Synthetic Data for Official Statistics. A Starter Guide", *United Nations*, [ECECESSTAT20226.pdf](#)
- Xie, L., Lin, K., Wang, S., Wang, F., and Zhou, J. (2018), "Differentially Private Generative Adversarial Network", *arXiv preprint*, arXiv:1802.06739.
- Xu, L., Skoularidou, M., Cuesta-Infante, A., and Veeramachaneni, K. (2019), "Modeling tabular data using conditional gan", *Advances in neural information processing systems*, 32, pp. 7303-7313.
- Yoon, J., Jarrett, D., and Van der Schaar, M. (2019), "Time-series generative adversarial networks", *Advances in neural information processing systems*, 32, pp. 5485-5495.
- Yoon, J., Drumright, L. N., and Van Der Schaar, M. (2020), "Anonymization through data synthesis using generative adversarial networks (ads-gan)", *IEEE journal of biomedical and health informatics*, 24(8), pp. 2378-2388.
- Zhang, K., Patki, N., and Veeramachaneni, K. (2022), "Sequential models in the SD vault", *arXiv preprint*, arXiv:2207.14406.