Machine Learning Based Time Series Generation for the Nuclear Industry

Hungary

Tamás HOLCZER

Budapest University of Technology and Economics

Laboratory of Cryptography and System Security (CrySyS)

holczer@crysys.hu

Keywords: machine learning, time series, radiation detection system

Abstract

We need a lot of data for various purposes. We want to test new algorithms or make a cyber exercise, but sometimes we do not have enough original publicly available data. In this case we must generate synthetic data. A special case of data generation is where we need a time series. This paper discovers different methods of time series generation and test a method called TimeGAN for generating synthetic radiation detection system data. Similar approach can be used for temperature, pressure, or other synthetic time series relevant for the nuclear industry.

1. Introduction

Time series are timestamped data sequences of measured values. In industrial environments, time series are stored by historians for incident response, reporting or for later analysis. In the nuclear field, sources of time series can be e.g., radiation detection, temperature, or pressure sensors.

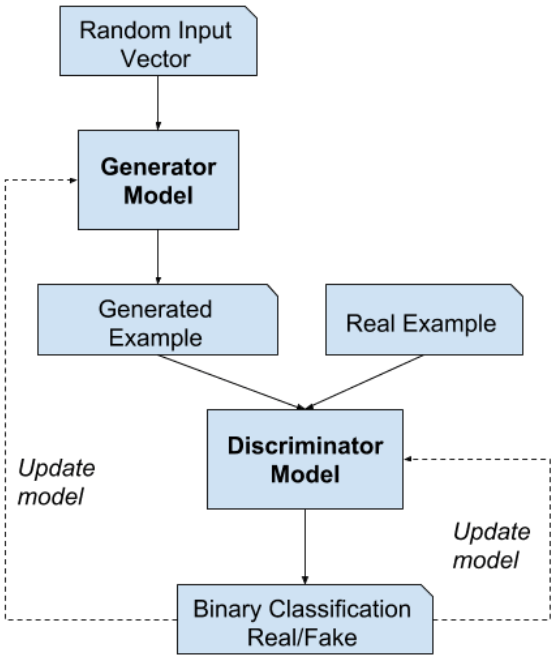
In some cases, we do not have enough real-life time series data, or it cannot be published. In this case, we must generate synthetic time series. These synthetic time series can be used for cyber exercises by simulating a real-world sensor, or for training and testing an anomaly detection algorithm before deployment. Machine learning based anomaly detection algorithms need a lot of data to learn the characteristics of normal and anomalous behavior. Generally, we have a lot of normal data but in many cases, we do not have enough attacks. Synthetic data generation can help us getting malicious events which can be learnt by different anomaly detectors making them more accurate. In this way synthetic data generation makes the anomaly detectors more reliable and the system using the anomaly detectors a more secure system.

As said, synthetic data generation is a hot topic in machine learning as it can be used to enrich small datasets. There are different methods for generating synthetic data, some are better in preserving the distribution of the original data, some are better at preserving the short or long trends. In this paper we start with an overview of some lately published methods for time series generation in Section 2. After that we introduce our main contribution, the implementation for generating synthetic radiation detection sensor data in Section 3. We shortly evaluate our results in Section 4, while we conclude our paper in Section 5.

2. Time-series generation

Time-series generation is a task, where a data sequence must be generated with some given properties. If the statistical distribution of the sequence is given and the values are independent, then a pseudo-random generator can generate the time-series from uniform distribution based on the inversion method [1]. Unfortunately, in most of the cases the exact distribution is unknown and the values following each other are dependent. This makes the realistic synthetic data generation a challenge. Most of the proposed sophisticated methods are using machine learning, where the properties of the time-series are learnt by a system and then reproduced on demand.

Generative Adversarial Networks (GAN) [2], [3] are based on deep learning methods. A generative model can explore the properties of an original dataset without supervision and generate new samples like the original data. The problem is reshaped as a supervised learning problem with two models. One model is the generator which tries to generate new examples of the original dataset and the other is the discriminator, who tries to classify the samples as original or generated. The two models are competing against each other and trained at the same time. The generator tries to make samples like the original, and the discriminator tries to decide if they are generated or original. The generator is successful if the discriminator has only 50% chance of correct decision. That means that the generator creates plausible examples.



1. Figure Structure of the competing models [2]

Several GANs are proposed in the literature, which can be used to generate synthetic data e.g., pictures. They can be used to generate time series as well, but most of them are not optimized to preserve the temporal dynamics and connections of the original data. This weakness is overcome in [4] where a Wasserstein GAN is used for this purpose. An enhanced version of this approach is introduced in [5], where a method called TimeGAN is developed. The result of the generator is a realistic time-series data with good similarity properties. The authors also published an open-source implementation of their work (https://github.com/ydataai/ydata-synthetic and https://github.com/jsyoon0823/TimeGAN), which was used also in this paper. The usage of the library is well described in [6], where energy consumption related time-series are generated.

There are several other approaches for time series generation. One lightweight and efficient method called ExtraMAE involves autoencoders [7]. The method is self-supervised which means it masks some parts of the time-series and tries to predict it. This method helps understand and capture the temporal dynamics of the original time series. ExtraMAE outperforms TimeGAN on some datasets. In the future we want to compare their performance considering the scenario used in this paper.

Markov models and chains are also used for synthetic time-series generation in the financial sector or in weather prediction [8], [9]. A Markov model consist of states and state transition probabilities. A stochastic system possesses the Markov property if the next state of the system only depends on the current state of the system (the system has no memory). This is a very simple but powerful model for different application areas. More complex systems can be modeled with a multi-step model (e.g., first the direction of the change is modeled and after that the size of the change). In the future we want to analyze and compare the results obtained in this paper to a Markov-model based solution as well.

The strength of Markov models (explicit transitions) and advantages of GANs (multi-step distributions) are mixed in [10]. It has similar results of other GANs on traditional benchmark time-series but has a huge potential in the future.

Some time series are incomplete or inaccurate. Traditional GAN based solutions cannot tackle with this problem. The authors of [11] can handle this problem with their RTGAN solution. Fortunately, this complex solution is not required in our case, as we have access for high quality complete and accurate time-series in most of the time.

3. Generator implementation

Before starting the implementation, first we must define the exact use-case. For this purpose, a radiation detection system is envisioned. A radiation detection system (RDS) can measure radiation data periodically and send the measured values to a database for storage and later analysis. The background radiation is measured in nSv/h (nanoSievert/hour); thus, a measurement record consists of three fields:

Timestamp	Location	Measured value

, where the timestamp defines the time, the record was created; the location is the place where the measurement was done; and the measured value stores the actual measurement in nSv/h.

Any kind of machine learning based approach requires a large learning set to be trained on. We used data from the openly available Hungarian academic radiation detection network. The network consists of twelve higher educational institutes with thirteen sites. Each site measures the background radiation with a 5–10-minute interval. Some sites use radiation sources around the detectors, so higher than natural radiation readings are also possible. The measured data can be accessed for several years back in time on the public webpage (http://omosjer.reak.bme.hu/). Unfortunately, the mass download of the time series is not supported by the webpage (however it is not prohibited as well). So, after finding a good use-case and a source of training set, we must solve the automatic download of the records.

This task was carried out by a Python script using the requests module. The script made several get requests to query the webpage for different time intervals and stations. The resulting html page was parsed, and the measurement values were stored in comma separated value (csv) files for later use. Collecting the data for thirteen stations for the last 6 years took a few minutes and resulted in almost one

thousand files (separate files were generated for each station and each month) occupying approximately one hundred megabytes disk space. This amount of training set is considered enough for any machine learning algorithm.

After downloading the learning set, we can start to work with the machine learning models. As stated in the previous section, we used TimeGAN [5] as our model.

Two slightly different models were created to get an insight about the learning times of the TimeGAN algorithm. The first model was taught on a one-month long sample. The data was sampled with a sliding window of size 1 day (144 samples in one window assuming that we get a reading in every 10 minutes). This leads to 4174 different partially overlapping samples for the algorithm. The algorithm was taught in 100 iterations. It took around a half day on a laptop with a recent i7 CPU but without any GPU support. Presumably this time can be largely reduced using a decent GPU.

1 month dataset, sliding window size: 144 sample (one day), 4174 sample, 100 iteration learning	
Phase	**Time**
Embedding network training	00:07:01
Supervised network training	00:03:55
Joint networks training	12:17:23
Synthetic data generation	00:00:10

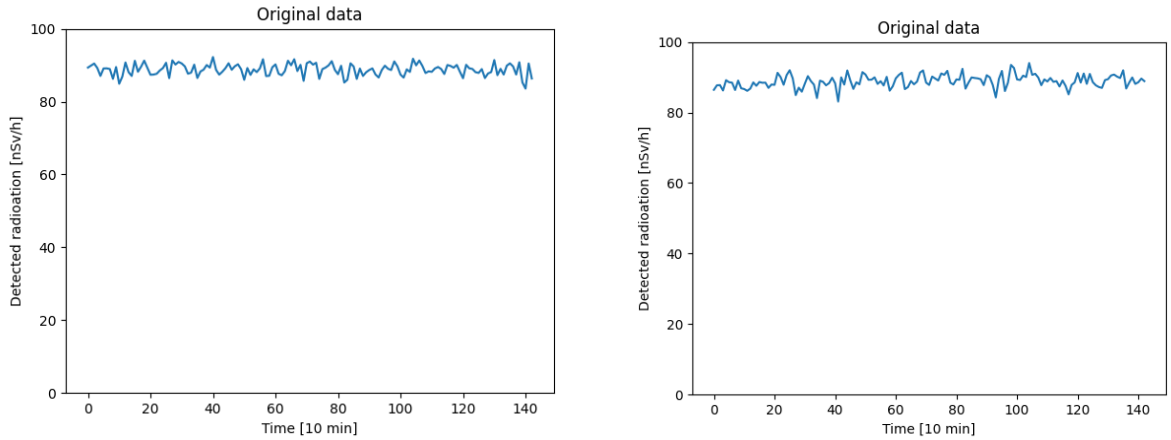
The second model was taught on a 2 week long dataset but with 500 iterations of learning. This task took more than two days.

2 weeks dataset, sliding window size: 144 sample (one day), 2016 sample, 500 iteration learning	
Phase	**Time**
Embedding network training	1:02:11
Supervised network training	00:22:47
Joint networks training	50:05:26
Synthetic data generation	00:00:11

Most of the machine learning models use real numbers in the interval of zero to one to represent every feature. If the original feature is from a different interval, then all the values are transformed. A widely used scaler is the MinMax scaler. It is very important to make an inverse transformation before using the synthetic data. This inverse transformation was properly done in the next section where the evaluation of the results. is discussed

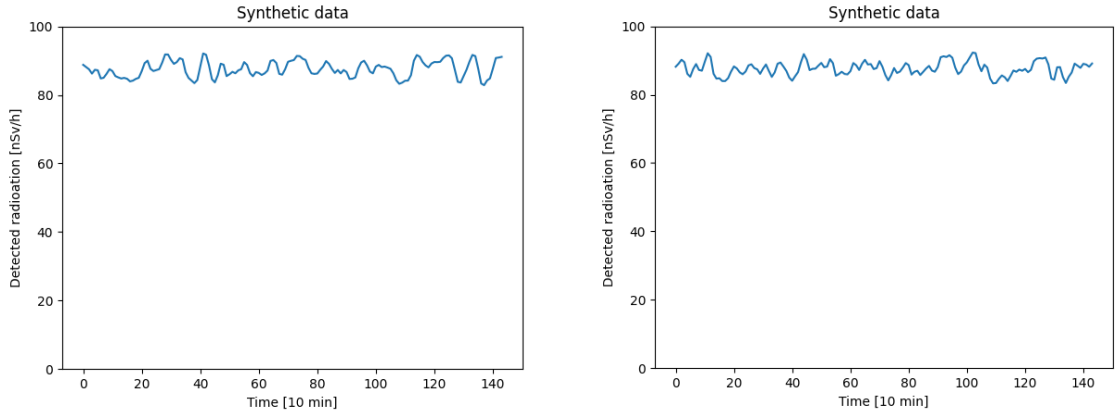
4. Evaluation of the generator

Before evaluating the results, we should take a look at the original data. It is visualized on Figure 2. The values are varied between 80 and 100 nSv/h. The measurements were made with 10 minutes intervals, but the consecutive measurements are displayed next to each other.



2. Figure Two samples of the original data

Two synthetic dataset is displayed on Figure 3. where the first model was used. They look quite like the original, but obviously they are different. This is exactly what expect from a synthetic dataset.



3. Figure Two samples of synthetic data generated by TimeGAN

A simple comparison of the mean of the datasets shows, that they are quite similar. The two original series have mean values of 88.79 and 88.87 while the synthetic series has means of 88.83 and 87.83. Further comparisons could be made by comparing the standard deviation of the series. More adequate statistical test could also be used like the Principal Component Analysis (PCA) or t-Distributed Stochastic Neighbor Embedding (t-SNE), but they remained for future work.

5. Conclusion

This paper presented the problem of time series generation in the nuclear industry with some motivation. We made a short overview of possible methods and tested a promising method called TimeGAN for generating synthetic time series radiation detection sensor data. The method worked well; however, the evaluation of the method was shallow. In the future we want to make a more thorough validation which can help uncovering subtle problems or help defining parameters of the underlying neural networks. Other methods like ExtraMAE or Markov based approaches are also planned to be tested.

Acknowledgement

The research reported in this paper was partially supported by the International Atomic Energy Agency (IAEA) within the CRP-J02017 coordinated research project, the Ministry of Innovation and Technology NRDI Office of Hungary within the framework of the Artificial Intelligence National Laboratory Program, and is part of project no. BME-NVA-02, implemented with the support provided by the Ministry of Innovation and Technology of Hungary from the National Research, Development and Innovation Fund, financed under the TKP2021 funding scheme.

References

[1] A. Rényi, *Probability theory*. Dover Publications, 2007.

[2] J. Brownlee, "A Gentle Introduction to Generative Adversarial Networks (GANs)," *https://machinelearningmastery.com/what-are-generative-adversarial-networks-gans/*. 2019.

[3] I. Goodfellow, "NIPS 2016 Tutorial: Generative Adversarial Networks," Dec. 2016.

[4] K. E. Smith and A. O. Smith, "Time Series Generation using a One Dimensional Wasserstein GAN," in *Proceedings of International Conference on Time Series and Forecasting*, 2019, pp. 771–781.

[5] J. Yoon, D. Jarett, and M. van der Schaar, "Time-series Generative Adversarial Networks," in *Advances in Neural Information Processing Systems 32 (NeurIPS 2019)* , 2019.

[6] A. Yadav, "Modeling and Generating Time-Series Data using TimeGAN," *https://towardsdatascience.com/modeling-and-generating-time-series-data-using-timegan-29c00804f54d*. 2021.

[7] M. Zha, S. Wong, M. Liu, T. Zhang, and K. Chen, "Time Series Generation with Masked Autoencoder," Jan. 2022.

[8] A. Shamsad, M. Bawadi, W. Wanhussin, T. Majid, and S. Sanusi, "First and second order Markov chain models for synthetic generation of wind speed time series," *Energy*, vol. 30, no. 5, pp. 693–708, Apr. 2005, doi: 10.1016/j.energy.2004.05.026.

[9] C. AIasseur, L. Husson, and F. Perez-Fontan, "Simulation of rain events time series with Markov model," in *2004 IEEE 15th International Symposium on Personal, Indoor and Mobile Radio Communications (IEEE Cat. No.04TH8754)*, 2004, pp. 2801–2805. doi: 10.1109/PIMRC.2004.1368831.

[10] D. Jarrett, I. Bica, and M. van der Schaar, "Time-series Generation by Contrastive Imitation," in *Proceedings of Neural Information Processing Systems*, 2021.

[11] H. Pei, K. Ren, Y. Yang, C. Liu, T. Qin, and D. Li, "Towards Generating Real-World Time Series Data," Nov. 2021.