International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9
***Article Received:** 25 July 2023 **Revised:** 12 September 2023 **Accepted:** 30 September 2023*

Implementation of Synthesize GAN Model to Detect Outlier in National Stock Exchange Time Series Multivariate Data

Swati Jain[1], Dr. Naveen Choudhary[2], Dr. Kalpana Jain[3]
[1]Research Scholar : Department of Computer Science and Engineering
[2]Head of Department : Department of Computer Science and Engineering
[3]Assistant professor: Department of Computer Science and Engineering
College of Technology and Engineering , MPUAT University,Udaipur, Rajasthan, India
swati.subhi.9@gmail.com

Abstract— This research work explores a novel approach for identifying outliers in stock related time series multivariate datasets, using Generative Adversarial Networks (GANs). The proposed framework harnesses the power of GANs to create synthetic data points that replicate the statistical characteristics of genuine stock related time series. The use of Generative Adversarial Networks to generate tabular data has become more important in a number of industries, including banking, healthcare, and data privacy. The process of synthesizing tabular data with GANs is also provided in this paper. It involves several critical steps, including data collection, preprocessing, and exploration, as well as the design and training using Generator and Discriminator networks. While the discriminator separates genuine samples from synthetic ones, the generator is in charge of producing synthetic data. Generating high quality tabular data with GANs is a complex task, but it has the potential to facilitate data generation in various domains while preserving data privacy and integrity. The results from the experiments confirm that the GAN framework is useful for detecting outliers. The model demonstrates its proficiency in identifying outliers within stock-related time series data. In comparison, our proposed work also examines the statistics and machine learning models in related application fields.

Keywords- GAN, Outlier Detection, Multivariate Data, Synthetic Data Generation, Dimensionality Reduction, Deep Learning.

I. INTRODUCTION

Data has always been a valuable resource, and this is even more true nowadays with the relentless progress of artificial intelligence. As any resource, data needs to be stored in an organized way. A very direct and efficient way to proceed is with tables, which are the well-known data structures composed of rows and columns, such as Excel spreadsheets for instance. Data stored under this format is called tabular data and is probably the most common data modality. Never the less, qualitative tabular data is not easy to come by besides, even if qualitative data is available, quantity can also be an issue. Indeed, for ML algorithms to function as intended, a lot of data is usually required and there are many domains where only few data are available. One possible way to address these accessibility and quantity issues is to leverage synthetic data, i.e., artificially created data that presents the same properties as some given existing data. However, the production of qualitative synthetic tabular data is not an easy task. This is mainly due to the difficulty of modelling the complex relationships between the columns of a table, but also to the wide variety of data types that these different columns may contain. Categorical data is hard to deal with due to its very nature, and numerical data can be hard to process when it follows complex non-Gaussian like distributions. As a result, there is still no unanimously accepted generative model for tabular data synthesis. Generative adversarial networks have literally taken over the field of image generation. Their incredible success is due to their ability of accurately modelling the underlying distribution of a given set of real images, and then sampling from this distribution to produce realistic looking images that could convincingly belong to the same set.[1] A question that begs to be asked is then: since images are nothing more to a computer than big tables of numbers corresponding to pixel values, couldn't we apply GANs to tabular data? Yes, we could. It is not without any difficulties, but we can definitely use GANs to synthesize tabular data, and this is exactly what we did in our research work. Incomplete data is a well-known problem with large databases, which raises challenges for many data mining applications.[2] The main focus will be on developing scalable and adaptable anomaly detection techniques that can spot unusual trade patterns in vast amount of stock related data. In the proposed method, a GAN based unsupervised outlier detection for multivariate time series data has been trained by deep learning networks. Further, we also estimated value of outlier which is treated as missing value in dataset using our proposed algorithm.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9
***Article Received:** 25 July 2023 **Revised:** 12 September 2023 **Accepted:** 30 September 2023*

II. LITRETURE REVIEW

A variety of anomaly detection techniques have been developed over the past few years in response to the wide range of anomalous kinds, data types, and application circumstances. These techniques are intuitive, but they are rigid and unable to identify irregularities in the context.[12] Prior until recently, the majority of research on deep generative models concentrated on models that offered a probability distribution function's parametric definition. The log likelihood can then be maximized to train the model. The deep Boltzmann machine is arguably the most effective model in this family. These models typically feature likelihood functions that are intractable, necessitating many approximations of the likelihood gradient. These challenges prompted the creation of models for generative machines, which may produce samples from the intended distribution even in the absence of an explicit likelihood representation.[14] A generative machine that can be trained using precise backpropagation instead of the several approximations needed for Boltzmann machines is a generative stochastic network.[16] By doing away with the Markov chains present in generative stochastic networks, this work expands on the concept of a generative machine. Nevertheless, their method is not intended for anomaly detection, but rather for time series representation learning.[21] To the best of our knowledge, we are the first to present cycle consistent GAN designs for time series data, allowing time series reconstructions to be performed directly using Generators. Furthermore, we thoroughly explore the use of Critic and Generator outputs in the estimation of anomaly scores.[27] To be used with GANs, a comprehensive framework for time series anomaly detection is presented.

III. MATERIALS AND METHODS

Tabular Data and Data Synthesis

A. *Tabular Data*

Data arranged in rows and columns is referred to as tabular data. Each row is a representation of something with tidbits of information. These bits of information are captured in some defined attributes and recorded in cells, each of which contains the value of one attribute.

1) *Table wise properties*

There are no restrictions on the number of rows or columns in a table, thus it can take on any shape. There are two possible scenarios: either there are significantly more rows than columns, or there are much less rows than columns. Besides, a table can be incomplete and contain empty cells and replace missing values using any stablished methods.

2) *Row Wise Properties*

This is the case when working with time-series data such as market stock prices. When put in tabular form, each row contains data related to a given point in time. The values of the attributes in the row corresponding to time may then be dependent on the values of the same attributes in some other rows corresponding to time points.

3) *Column Wise Properties*

Columns can be all independent of one another but that would make data far less interesting. However, this knowledge is not easy to extract because columns can interact in infinitely many ways, and most of them are nontrivial.

4) *Data Synthesis*

Data synthesis is the action of creating data. This new data can either be created without any connection to some already existing data or, on the contrary, in close relation to some given real-world data. In the first case, one can easily generate

somewhat arbitrary data using the adequate statistical tools. The second case is trickier and perhaps of greater interest. The goal is to generate realistic synthetic data based on some given example data. With tabular data, this means combining attributes of some rows to produce new synthetic rows. This approach has the advantage to work both with numerical and categorical data, and to provide realistic synthetic samples. Some common examples are variational auto encoders, deep belief networks and generative adversarial networks.

Generative Adversiable Network

GAN has two main components, namely Generator and Discriminator.

Generator: This method of learning is done without supervision. It will produce phony data that is derived from actual data. In addition, it is a neural network with activation, loss, and hidden layers.

Discriminator: This supervised method uses a basic classifier to determine if data is legitimate or fraudulent. It gives generator feedback after being trained on actual data.

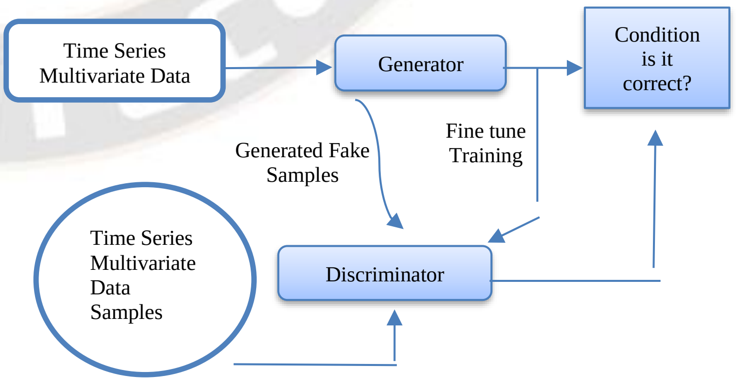

Figure 1: GAN Model.

IV. PROPOSED MODEL

Generating Adversarial Networks can be adapted for tabular data synthesis, such as generating rows of data in a structured format like a spreadsheet or a database. Here's a general approach for generating tabular data with GANs.

A. DATA PREPROCESSING

Start with a dataset of real tabular data that you want to mimic. Normalize and preprocess the data, handling missing values, categorical variables, and any other necessary data transformations.

B. GENERATOR NETWORKS

Design the generator network to produce synthetic rows of tabular data. The input to the generator might be random noise or some structured input. The output layer of the generator should produce data points in the same format as your real data.

C. DISCRIMINATOR NETWORKS

Create a discriminator network that can identify between rows of actual and fake data. A binary output from the discriminator should indicate whether a certain data row is synthetic or real.

D. TRAINING

To train the GAN, feed the discriminator rows of both fake and genuine data. While the generator tries to provide data that can "fool" the discriminator, the discriminator aims to correctly discriminate between real and synthetic data.

E. LOSS FUNCTIONS

To direct the training process, use suitable loss functions, such as binary cross-entropy loss for the generator and discriminator.

F. HYPERPARAMETER TUNING

Experiment with hyperparameters, including the architecture of the generator and discriminator, learning rates, and batch sizes, to achieve better results.

G. EVALUATION

The quality of the data generated can be assessed using evaluation metrics, which can include mean squared error, mean absolute error, or other metrics specific to your task and dataset.

H. DATA POST PROCESSING

You may need to use post-processing procedures, depending on your data's properties, to make sure the created data follows the same restrictions as the original data.

I. DATA GENERATION

You can utilize the generator to create fresh synthetic data samples after the GAN has been trained.

Generating tabular data ensures that the generated data is not only realistic but also maintains the integrity of relationships within the data is crucial. The entire sequential process flowchart for obtaining the desired output from the supplied input data is displayed as follows:

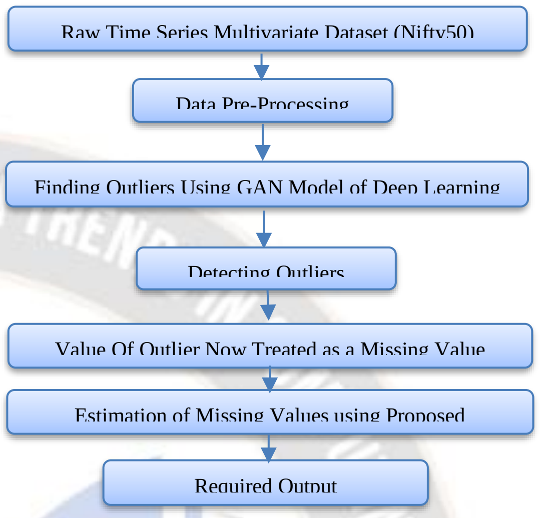
Raw Time Series Multivariate Dataset (Nifty50)
Data Pre-Processing
Finding Outliers Using GAN Model of Deep Learning
Detecting Outliers
Value Of Outlier Now Treated as a Missing Value
Estimation of Missing Values using Proposed
Required Output

Figure 2: Flow Chart of Proposed Model.

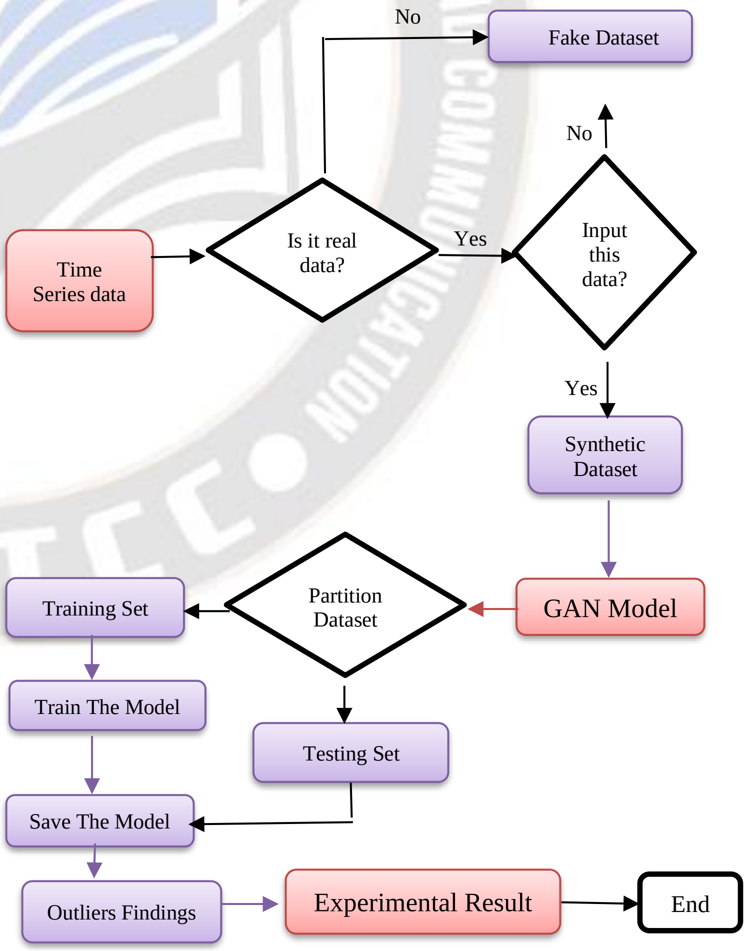

Figure 3: Proposed Method for Outlier Detection

V. RESULTS

We had to compile the inputs from the entire dataset, wherein 20% of the real data was used as the test set and 80% of the data for both synthetic and real datasets was utilized as train data. We used the validation set loss as the early termination criteria for both the models trained on synthetic and real data. The Sequential class from TensorFlow is used to specify the code for implementing a GAN. It comprises of two dense layers with ReLU activation for the generator and discriminator, and one dense layer with sigmoid activation for the final layer. Use Pandas to load the actual data into a.csv file and then load it using Python libraries and convert it to a Numpy array. Now train the model using a for loop that trains the discriminator and the generator alternately to detect outliers. Here is the Coal India NSE stock price Nifty dataset with their graphs of dataset and outliers.

The experiment yielded highly fascinating and promising results regarding the production of synthetic sequential data. A few warnings should be noted: the data under investigation was modest and had a daily time frequency, so it may be regarded quite straightforward. There were also no unexpected missing values, and the dimensionality was low.

Following is the graph of Coal India NSE stock price data from the year 2010-2021.

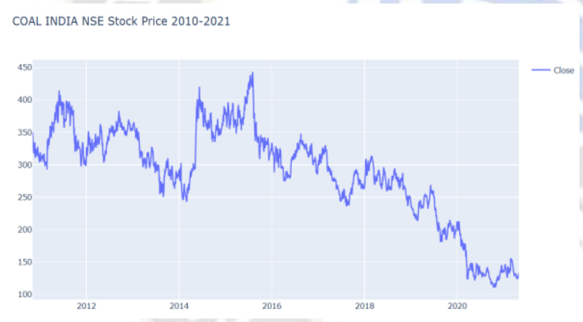

Figure 4: Graphical representation of Coal India stock price.

The experiment yielded highly fascinating and promising results regarding the production of synthetic sequential data.

TABLE I: The summarized results obtained for the test set.

	R2	MAE	MRLE
Real	0.905402	0.012814	0.000629
Synthetic	0.912356	0.019088	0.000715

Now, plot metrics and evaluate model loss as shown in the graph below:

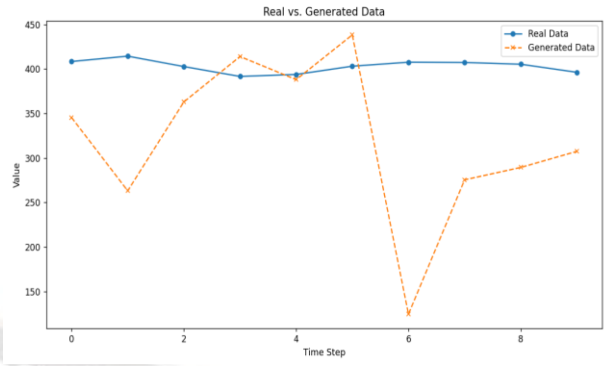

Figure 5: Graphical representation of model loss.

After finding loss, our system generated number of Outliers and their position using proposed model as given below:

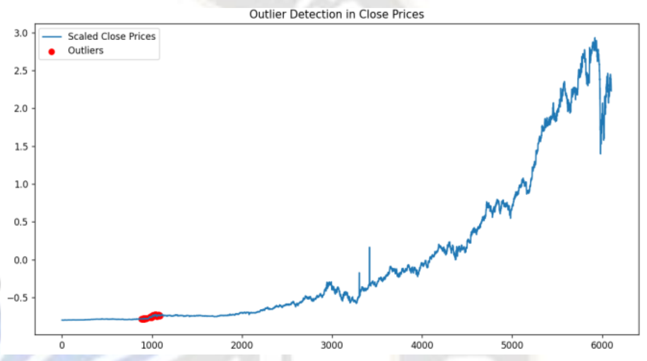

Figure 6: Graph showing outliers in given dataset.

Further, each outlier was treated as missing value which in turn was calculated and estimated using our model. Their locations and estimated missing values are given below:

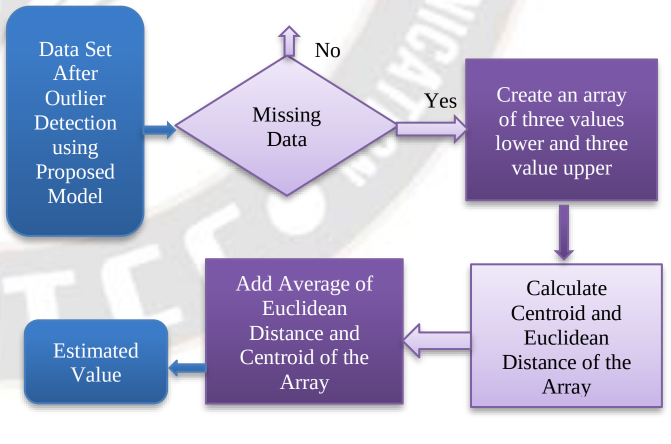
Data Set After Outlier Detection using Proposed Model
Missing Data
No
Yes
Create an array of three values lower and three value upper
Calculate Centroid and Euclidean Distance of the Array
Add Average of Euclidean Distance and Centroid of the Array
Estimated Value

Figure 7: Proposed model for estimation of missing value.

VI. COMPARATIVE ANALYSIS

We examined more than 20 datasets and based on the total number of outliers detected in these datasets, we found our work showed far better results compared to other methods. We compared our proposed method with other machine learning and statistical techniques such as LSTM, SVM and IQR concluded that our method is more selective and conservative in identifying outliers compared to others.

TABLE II: Comparitive analysis of proposed method with existing methods.

Dataset tested with their rows and columns		Proposed Model (GAN)	LSTM	ML (one class SVM)
HDFC Bank Dataset (6229 ,7)	Total no of outliers	12	15	559
Coal India Dataset (2598,15)	Total no of outliers	34	37	245
Maruti Dataset (4093, 15)	Total no of outliers	21	25	369
Sun-Pharma Dataset (5059, 15)	Total no of outliers	30	34	448

Comparison with LSTM and GAN. All methods are benchmarked on the same dataset. The inference time is the average of several single inference steps. The effects outliers can have in squared errors such as MAE, MSE or RMSE. The goal here we evaluate MAE, MSE and RMSE for each set of observations. As the sample size increases, RMSE typically exceeds MAE. Large errors are not always penalized by MAE. Large mistakes are penalized by MSE. Big errors are penalized by RMSE. MAE is a preferable alternative if you want to ignore data outliers; if you want to include them in the loss function, MSE/RMSE is the superior choice.

TABLE II: Comparitive analysis of proposed method with existing methods.

Performance Metrics		Proposed Model (GAN)	LSTM	One class SVM (ML)	IQR(Inter -Quartile Range)
HDFC Bank Dataset	MAE	0.62323	0.82323	1.01332	5.363
	RMSE	0.87902	0.90732	1.11543	2.316
Coal India Dataset	MSE	0.59083	0.91620	1.021555	5.441
	RMSE	0.90876	0.95718	1.14681	2.332
Maruti Dataset	MSE	0.09287	0.18011	0.61588	5.230
	RMSE	0.38765	0.42439	0.92601	2.286
Sun Pharma Dataset	MSE	0.10004	0.10215	1.01304	5.451
	RMSE	0.24567	0.31961	1.11421	2.334

VII. CONCLUSION

This research uses the design of a Generative Adversarial Network to provide a unique method for identifying outliers in multivariate time series data. In this approach, time sequences are projected onto a latent space using inverse mapping after a GAN based generator is trained to learn the general data distribution of the training set. The reconstruction loss of the sequences as they are recreated by the generator is used to estimate anomaly scores. The generator and discriminator of the GAN increases the accuracy of anomaly detection for datasets. GAN works better for anomaly identification in time series than other existing statistical, machine learning methods, according to experimental results on a variety of real world time series National Stock Exchange datasets. We intend to evaluate GAN on more sizable multivariate datasets and contrast it with more baseline models in the future in order to obtain a deeper understanding of these problems and improve GAN's performance.

REFERENCES

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza,B. Xu, D. Warde-Farley ,S. Ozair, Y. Bengio. Generative adversarial nets. In Advances in neural information processing systems ,2014.

[2] Radford, A., Metz, L., &amp; Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434, 2015.

[3] Hochreiter, S., &amp; Schmidhuber, J. . Long short-term memory. Neural computation, 9(8), 1997, 1735-1780.

[4] Graves, A., Mohamed, A. R., &amp; Hinton, G. Speech recognition with deep recurrent neural networks. In Acoustics, speech and signal processing (ICASSP), 2013 IEEE international conference on,2013, (pp. 6645-6649). IEEE.

[5] Hinton, G. E., &amp; Salakhutdinov, R. R. Reducing the dimensionality of data with neural networks. Science, 313(5786),2006 504-507.

[6] Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., &amp; Manzagol, P. A. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. Journal of Machine Learning Research, 11(Dec), 2010, 3371-3408.

[7] Rousseeuw, P. J., &amp; Leroy, A. M. Robust regression and outlier detection (Vol. 589). Wiley, 1987.

[8] Filzmoser, P., Maronna, R., &amp; Werner, M. Outlier identification in high dimensions. Computational Statistics &amp; Data Analysis, 2008, 52(3), 1694-1711.

[9] Chandola, V., Banerjee, A., &amp; Kumar, V. Anomaly detection: A survey. ACM computing surveys (CSUR), 2009, 41(3), 1-58.

[10] Yize Chen, Yishen Wang, Daniel Kirschen, and Baosen Zhang. Model-free renewable scenario generation using generative

adversarial networks. IEEE Transactions on Power Systems, 33(3):3265–3275, 2018.

[11] Andrew M Dai and Quoc V Le. Semi-supervised sequence learning. In Advances in neural information processing systems, pages 3079–3087, 2015.

[12] Xinrui Lyu, Matthias Hueser, Stephanie L Hyland, George Zerveas, and Gunnar Raetsch. Improving clinical predictions through unsupervised time series representation learning. arXiv preprint arXiv:1812.00490, 2018.

[13] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhudinov. Unsupervised learning of video representations using lstms. In International conference on machine learning, pages 843–852, 2015.

[14] Otto Fabius and Joost R van Amersfoort. Variational recurrent auto-encoders. arXiv preprint arXiv:1412.6581, 2014.

[15] Yingzhen Li and Stephan Mandt. Disentangled sequential autoencoder. arXiv preprint arXiv:1803.02991, 2018.10

[16] Alex Graves. Generating sequences with recurrent neural networks. arXiv preprint arXiv:1308.0850, 2013.

[17] Ilya Sutskever, James Martens, and Geoffrey E Hinton. Generating text with recurrent neural networks. In Proceedings of the 28th International Conference on Machine Learning (ICML-11), pages 1017–1024, 2011.

[18] Chris Donahue, Julian McAuley, and Miller Puckette. Adversarial audio synthesis. arXiv preprint arXiv:1802.04208, 2018.

[19] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. Journal of machine learning research, 9(Nov):2579–2605, 2008.

[20] Fred B Bryant and Paul R Yarnold. Principal-components analysis and exploratory and confirmatory factor analysis. 1995.

[21] Jinsung Yoon, James Jordon, and Mihaela van der Schaar. PATE-GAN: Generating synthetic data with differential privacy guarantees. In International Conference on Learning Representations, 2019.

[22] Ronald J Williams and David Zipser. A learning algorithm for continually running fully recurrent neural networks. Neural computation, 1(2):270–280, 1989.

[23] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In Proceedings of the 26th annual international conference on machine learning, pages 41–48. ACM, 2009.

[24] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. The Journal of Machine Learning Research, 17(1):2096–2030, 2016.

[25] Vijay R Konda and John N Tsitsiklis. Actor-critic algorithms. In Advances in neural information processing systems, pages 1008–1014, 2000.

[26] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784,2014.

[27] Yizhe Zhang, Zhe Gan, and Lawrence Carin. Generating text via adversarial training. In NIPS workshop on Adversarial Training, volume 21, 2016.

[28] Luca Simonetto. Generating spiking time series with generative adversarial networks: an application on banking transactions. 2018.

[29] prediction with recurrent neural networks. In Advances in Neural Information Processing Systems, pages 1171–1179, 2015.

[30] Alex M Lamb, Anirudh Goyal Alias Parth Goyal, Ying Zhang, Saizheng Zhang, Aaron C Courville, and Yoshua Bengio. Professor forcing: A new algorithm for training recurrent networks. In Advances In Neural Information Processing Systems, pages 4601–4609, 2016.

[31] Dzmitry Bahdanau, Philemon Brakel, Kelvin Xu, Anirudh Goyal, Ryan Lowe, Joelle Pineau, Aaron Courville, and Yoshua Bengio. An actor-critic algorithm for sequence prediction. arXiv preprint arXiv:1607.07086, 2016.

[32] Olof Mogren. C-rnn-gan: Continuous recurrent neural networks with adversarial training. arXiv preprint arXiv:1611.09904, 2016.

[33] Cristóbal Esteban, Stephanie L Hyland, and Gunnar Rätsch. Real-valued (medical) time series generation with recurrent conditional gans. arXiv preprint arXiv:1706.02633, 2017.

[34] Giorgia Ramponi, Pavlos Protopapas, Marco Brambilla, and Ryan Janssen. T-cgan: Conditional generative adversarial network for data augmentation in noisy time series with irregular sampling. arXiv preprint arXiv:1811.08295, 2018.

[35] Shota Haradal, Hideaki Hayashi, and Seiichi Uchida. Biosignal data augmentation based on generative adversarial networks. In 2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), pages 368–371. IEEE, 2018.

[36] Moustafa Alzantot, Supriyo Chakraborty, and Mani Srivastava. Sensegen: A deep learning architecture for synthetic sensor data generation. In 2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), pages 188–193. IEEE, 2017.

[37] Chi Zhang, Sanmukh R Kuppannagari, Rajgopal Kannan, and Viktor 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), pages 1–6. IEEE, 2018.

[38] Wei-Ning Hsu, Yu Zhang, and James Glass. Unsupervised learning of disentangled and interpretable representations from sequential data. In Advances in neural information processing systems, pages 1878–1889, 2017.

[39] Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. Autoencoding beyond pixels using a learned similarity metric. arXiv preprint arXiv:1512.09300, 2015.

[40] Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Olivier Mastropietro, Alex Lamb, Martin Arjovsky, and Aaron Courville. Adversarially learned inference. arXiv preprint arXiv:1606.00704, 2016.

[41] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. Adversarial autoencoders. arXiv preprint arXiv:1511.05644, 2015.
[42] Yoon Kim, Kelly Zhang, Alexander M Rush, Yann LeCun, et al. Adversarially regularized autoencoders. arXiv preprint arXiv:1706.04223, 2017.
[43] Aäron Van Den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew W Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. SSW, 125, 2016.
[44] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473, 2014.