ANALYSIS OF TIME SERIES GENERATIVE ADVERSARIAL NETWORKS, IMPROVEMENTS AND IDEAS

FINAL REPORT

Harshavardhan. B*
Dept. of Computer Science and Engineering
University of Minnesota
battu018@umn.edu

Praveen. M
Dept. of Chemical Engineering and Materials Science
University of Minnesota
mural101@umn.edu

Youchi. Z
Dept. of Chemical Engineering and Materials Science
University of Minnesota
zhan8020@umn.edu

Jonathan. Z
Dept. of Chemistry
University of Minnesota
zajac028@umn.edu

1 Introduction

Generative adversarial networks (GAN) have rapidly gained interest since their inception by Goodfellow et al. [2014]. In this framework, the core idea is that a generator generates synthetic data that is then fed through a discriminator to evaluate how realistic the data are. In this framework, the training objective of the generative network is to increase the error rate of the discriminative network, while the training objective of the discriminative network is to distinguish false data produced by the generator from the true data distribution. Since its development, the applications of the GAN framework has accelerated for various purposes and in various domains. Some of the state of the art applications using GAN architecture as a backbone include StyleGAN by Karras et al. [2018] for style transfer by disentangling the latent representation, U-Net for image segmentation by Ronneberger et al. [2015], face aging by Antipov et al. [2017], Astrophysical Image Processing (AIP) by Schawinski et al. [2017], and spatial mapping for geographical data translation by Wu and Biljecki [2022].

Although much of the work on GANs is done on images, GANs for time series generation is explored recently. GAN for time series plays an important role in preserving the privacy of the sensitive time data, and can also help in augmenting datasets which can be to train a classifier for downstream tasks like classification, forecasting and clustering. In this report, we go through how to formulate the problem for time series generation using GANs, followed by recent works in this area. We compared the datasets generated by training the discriminator with Wasserstein Loss - Gradient Penalty(WGAN-GP) introduced by Gulrajani et al. [2017] on different GAN architectures for time series. In the conclusion, we shared some of the ideas and areas that we are planning to study further.

2 Related Work

Among novel mechanisms derived from the original GAN framework include Recurrent Conditional GAN (RCGAN) by Esteban et al. [2017], In RCGAN, the Generator and Discriminator are realized by recurrent neural networks (RNN). More specifically, they use LSTM (long short-term memory) cells as the structure for both generator and discriminator to enhance the long-term memorization. LSTM has more operations inside a single iteration to decide how much information from previous iterations will be forgotten and how much will be carried on to future. This enabled the model to convey memories from a flexible distance compared with the original recurrent neural network, which can only remember the nearest information.

*This work is submitted as the final report for the project component of CSCI 5525.

Time-series GAN (TimeGAN) by Yoon et al. [2019], include a dimensionality reduction step at first and then train generator and discriminator networks on latent space. The latent space is a transformation of the feature matrix with much less number of features. It is generated by an embedding process in which we pass all the input data through a network with a bottleneck layer. The loss function for getting a good dimensionality reduction is the difference between the input data and the output data. This strategy will force the input and output to be the same and thus make sure enough information has been embedded into bottleneck layer and thus can be learned on a lower dimension. This embedding strategy is achieved using an autoencoder like architecture. After building the dimensionality reduction part, two networks, namely generator and discriminator will be trained on both low dimensional original data (supervised learning) and also generated data (unsupervised learning).

TimeGAN and RCGAN pretty much covers the architectures and ideas of the most of the methods. Some similar methods were proposed by Mogren [2016], where the architecture C-RNN-GAN is used for music generation. C-RNN-GAN uses the output of $t-1$ timestep for t timestep in RNN. Where as RCGAN uses ground-truth just like teacher forcing. DoppleGANger by Lin et al. [2020] focuses on time series generation with meta data, an interesting approach where each observation has time dependent records along with the meta data and meta data generation is also involved in the problem. Smith and Smith [2020] proposed architecture identical to TimeGAN where they projected the timeseries into spectrograms and retrieved it using a decoding network. By contrast, TimeGAN uses embedding space instead.

3 Experimental Setup

3.1 Preprocessing

Time series generation problem respects the temporal relationships accross time (T), and inter dimensional relationships across the features (D). So, we divide the training data, ($X \in R^{T \times D}$) into sequences of length (W), corresponding to time series with W time steps. We do this by using a sliding window of length (W) and stride of 1. The final dimensions of the dataset will be (N, W, D). We normalize the data using a MinMax scalar and shuffle the data across (N) at random.

3.2 Time Series Generation

A standard Generative Adversarial Network (GAN) for time series will have two components, the generator (G), and a discriminator (D). The objective of the generator is to generate a window of synthetic samples, $\tilde{x} \in \mathbb{R}^{d \times w}$ using a random latent vector $z \in \mathbb{R}^z$ as an input. The objective of the discriminator is distinguish the synthetic series, $\tilde{x} = G_\theta(z) \sim p_G(z)$ from real series, $x \sim p_{data}(x)$. The generator, G_θ is then trained along with the discriminator (D) until they reach an equilibrium. The parameters, (θ) of the generator are updated using the gradient of the output of the discriminator. A GAN with BCE loss is prone problems like vanshing gradient, and mode collapse due to the stability of discriminator during the training process. To address this problem, Wasserstein GANs were introduced by Arjovsky et al. [2017]. WGANs replace standard binary cross entropy objective function with earth mover's distance.

We used the improved version of Wasserstein loss or Earth movers distance proposed by Gulrajani et al. [2017] for the experiments. We noticeably got good results with RCGAN when trained with W-Loss and GP. We also compared the synthetic data generated by modified RCGAN, (WRGAN) with the current state of the art methods.

3.3 Evaluation Metrics

As this is a generation problem, different from prediction or forecasting. We cannot use the elements of the confusion matrix to measure the performance of the model as there are no ground truths. In case of Images, synthetic images are evaluated using metrics like fidelity and diversity. We used the same metrics to evaluate the synthetic time series data.

3.3.1 Diversity — Synthetic samples should be distributed to cover the real data.

t-SNE and PCA can be used to analyse the original and synthetic datasets by flattening the time dimension. By using them we can visualize how closely the distribution of generated samples resembles that of the original in 2-dimensional space, giving a qualitative assessment of Diversity.

3.3.2 Fidelity — Synthetic samples should be indistinguishable from the real data.

For a quantitative measure of similarity, we train a post-hoc time-series classification model using a 2-layered LSTM network to classify the real sequences from synthetic sequences. First, each original sequence is labeled real, and each

generated sequence is labeled not real. Then, an off-the-shelf (RNN) classifier is trained to distinguish between the two classes as a standard supervised task. We then report the classification error on the held-out test set, which gives a quantitative assessment of Fidelity.

3.3.3 Usefulness—Synthetic samples should be just as useful as the real data when used for the same predictive purposes (i.e. train-on-synthetic, test-on-real)/ TSTR setting.

In order to be useful, the sampled data should inherit the predictive characteristics of the original. In particular, we expect TimeGAN to excel in capturing conditional distributions over time. Therefore, using the synthetic dataset, we train a post-hoc sequence-prediction model (by optimizing a 2-layer LSTM) to predict next-step temporal vectors over each input sequence. Then, we evaluate the trained model on the original dataset. Performance is measured in terms of the mean absolute error (MAE); for event-based data, the MAE is computed as $|1 - \text{estimated probability that the event occurred}|$. This gives a quantitative assessment of Usefulness.

4 Observations

We tested the performance of the RCGAN, WRGAN, TimeGAN and TimeGAN with W-loss using all the evaluation metrics mentioned above using the **Stocks** dataset. The dataset consists multiple dimensions which is very good for Multi-variate time series generation setting. The historical Google stocks data from 2004 to 2019, includes features such as volume, high, low, opening, closing, and adjusted closing price for the stock.

We used Adam optimizer with default values of learning rate ($\alpha = 0.001$), (β_1, β_2) = (0.9, 0.999). In order to reduce gains through network changes, we used GRU cells in all experiements and trained the models for same steps.

4.1 Checking Diversity, Fidelity and Usefulness

As mentioned in section 3, we used PCA and t-SNE plots as shown in Fig 1 for visually inspecting the synthetic data and real data. These plots give a better understanding of the distribution by converting the data to lower dimensions. As the overlap of the plots increases means the diversity is good, and the generator has learnt all types of sequences. By looking at the t-SNE plots in Fig 1, we can notice that TimeGAN and TimeGAN-Wloss generated datasets with high diversity when compared with RGAN and WRGAN-GP.

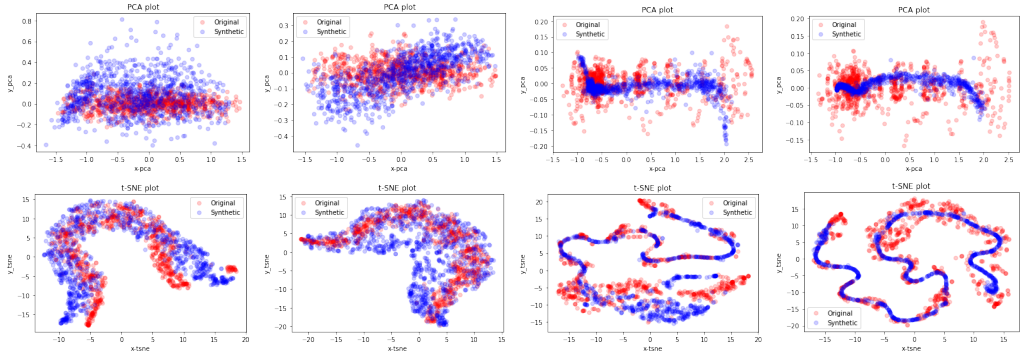

Figure 1: From left to right, we have the PCA plots of the real and synthetic datasets using RGAN, WRGAN, TimeGAN, and TimeGAN with W-loss on top. t-SNE plots of the real and synthetic datasets of the same models in the bottom.

Fidelity is measured quantitatively using the Discriminative score, Table 1 shows the Discriminative and Predictive Scores of the model different models with **stocks** dataset. The lower the discriminative score, the better. If the classifier cannot distinguish the synthetic data from the real data, it means the quality of generated data is good and has high fidelity.

From Fig 1, TimeGAN with W-loss performed the best out of all four models and this is corroborated by the t-SNE plots as well. Surprisingly RGAN performed well compared to WRGAN. One possible explaination to this is, as we constrained all models to be trained for same steps, this caused instability in calculating the gradient penalty using the interpolated series as described in the paper Gulrajani et al. [2017], W-loss approximation with gradient penalty would need more time for convergence and as well as computation. In the case of usefulness, the predictive score of the models with Wasserstein loss was relatively better but not by a large margin.

Table 1: Results of different GAN architectures on **stocks** dataset

Scores		
Metric	Method	Stocks
Discriminative Score (Lower the Better)	RGAN	0.196 ± 0.21
	WRGAN-GP	$\mathbf{0.202 \pm 0.01}$
	TimeGAN	0.102 ± 0.021
	TimeGAN(W-Loss)	$\mathbf{0.006 \pm 0.01}$
Predictive Score (Lower the Better)	RGAN	0.056 ± 0.013
	WRGAN-GP	0.028 ± 0.005
	TimeGAN	0.038 ± 0.001
	TimeGAN(W-Loss)	$\mathbf{0.039 \pm 0.004}$

5 Conclusion

In this work, we have analyzed different time series generation algorithms, and also the synthetic data from these algorithms both qualitatively and quantitatively. We replaced BCE loss with W-Loss in the current state of the art model for time series generation (**TimeGAN**) and noticed modest gains in performance. Synthetically generated data can be some times very valuable when the available data is scarce or sensitive. The synthetically generated data from these models can be used for training the base models and the real dataset can be used for fine tuning. We performed the experiment in **HW 4** of the course and achieved significant gains over the base model which is trained on real dataset alone. In the future, we are planning to analyze how one attribute is influencing the other in the generation problem using graph structures.

References

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014. URL https://arxiv.org/abs/1406.2661.

Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks, 2018. URL https://arxiv.org/abs/1812.04948.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015. URL https://arxiv.org/abs/1505.04597.

Grigory Antipov, Moez Baccouche, and Jean-Luc Dugelay. Face aging with conditional generative adversarial networks, 2017. URL https://arxiv.org/abs/1702.01983.

Kevin Schawinski, Ce Zhang, Hantian Zhang, Lucas Fowler, and Gokula Krishnan Santhanam. Generative adversarial networks recover features in astrophysical images of galaxies beyond the deconvolution limit. *Monthly Notices of the Royal Astronomical Society: Letters*, page slx008, jan 2017. doi:10.1093/mnrasl/slx008. URL https://doi.org/10.1093%2Fmnrasl%2Fslx008.

Abraham Noah Wu and Filip Biljecki. GANmapper: geographical data translation. *International Journal of Geographical Information Science*, pages 1–29, mar 2022. doi:10.1080/13658816.2022.2041643. URL https://doi.org/10.1080%2F13658816.2022.2041643.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of wasserstein gans, 2017. URL https://arxiv.org/abs/1704.00028.

Cristóbal Esteban, Stephanie L. Hyland, and Gunnar Rätsch. Real-valued (medical) time series generation with recurrent conditional gans, 2017. URL https://arxiv.org/abs/1706.02633.

Jinsung Yoon, Daniel Jarrett, and Mihaela van der Schaar. Time-series generative adversarial networks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/c9efe5f26cd17ba6216bbe2a7d26d490-Paper.pdf.

Olof Mogren. C-rnn-gan: Continuous recurrent neural networks with adversarial training, 2016. URL https://arxiv.org/abs/1611.09904.

Zinan Lin, Alankar Jain, Chen Wang, Giulia Fanti, and Vyas Sekar. Using GANs for sharing networked time series data. In *Proceedings of the ACM Internet Measurement Conference*. ACM, oct 2020. doi:10.1145/3419394.3423643. URL https://doi.org/10.1145%2F3419394.3423643.

Kaleb E Smith and Anthony O Smith. Conditional gan for timeseries generation, 2020. URL https://arxiv.org/abs/2006.16477.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan, 2017. URL https://arxiv.org/abs/1701.07875.