Transformer-based Conditional Generative Adversarial Network for Multivariate Time Series Generation

Abdellah Madane[1,3,4], Mohamed-djallel Dilmi[3], Florent Forest[2], Hanane Azzag[3], Mustapha Lebbah[1,3], and Jérôme Lacaille[4]

[1] DAVID Lab, University of Versailles, Université Paris-Saclay
mustapha.lebbah@uvsq.fr
[2] IMOS, Ecole polytechnique fédérale de Lausanne
[3] LIPN UMR CNRS 7030, Université Sorbonne Paris Nord
{madane, dilmi, florent, azzag}@lipn.univ-paris13.fr
[4] Safran Aircraft Engines
jerome.lacaille@safrangroup.com

Abstract. Recent works proposed transformer-based time series generative adversarial networks to address the limitations of recurrent networks. However, these models assume a unimodal distribution and try to generate samples around the expectation of the real data distribution. One of the limitations is that they generate random multivariate time series and fail to generate samples in the presence of multiple sub-components within the distribution. In this paper, we propose to overcome these limitations by conditioning the generated samples on a particular encoded context, allowing one model to fit a mixture distribution with multiple sub-components. Technically, it is a Conditional Generative Adversarial Network (CGAN) that models realistic Multivariate Time Series (MTS) under different types of conditions, such as categorical variables or multivariate time series. This paper evaluates our transformer-based CGAN approach (MTS-CGAN) for generating realistic high-dimensional and long sequences under categorical conditions. We evaluate the model on the UniMiB SHAR dataset using qualitative evaluations and quantitative metrics. Additionally, we introduce a specialized version of the Frechet Inception Distance (MTS-FID) to measure its performance and compare statistical similarities between generated and real data distributions.

1 Introduction

Conditional generative adversarial networks have attracted significant interest recently [8,13], whether for data augmentation, scenario simulation, or missing data imputation. The quality of samples generated by such models is improving rapidly. One of their most exciting applications is multivariate time series generation, particularly when considering contextual knowledge to carry out this generation. Most published works address this challenge using recurrent architectures [14], which usually struggle with long time series due to vanishing or exploding gradients. One other way to process sequential data is via Transformer-based architectures. In the span of five years, Transformers have repeatedly advanced the state-of-the-art on many sequence modeling tasks [19,3]. Thus, it was a matter of time before we could see transformer-based solutions for time series [23,18], particularly for multivariate time series generation [11,10]. These studies showed promising results. Consequently, whether Transformer-based techniques are suitable for conditional multivariate time series generation is an interesting problem to investigate. Our work extends the TTS-GAN [11] by conditioning its generated output on a particular encoded context, allowing one model to fit a mixture distribution with multiple sub-components. Our contributions are summarized as follows: (a) We design a transformer-based conditional generative adversarial network architecture for conditional multivariate time series generation, called MTS-CGAN, using different types of conditions. For this paper, we only evaluate the model using the categorical condition; (b) We introduce and study a new parameter α, which controls the trade-off between the desired variability and the importance given to the context in the conditional generation; (c) We rigorously evaluate our approach on the UniMiB SHAR dataset [16] using qualitative and quantitative evaluation methods; (d) We introduce MTS-FID, a new version of FID suitable for evaluating generative models dealing with multivariate time series. We show that MTS-FID exhibits the same behavior as FID and that the results obtained correlate with other evaluations performed; (e) Through a data augmentation study, we demonstrate that our MTS-CGAN can model the mixture data distribution and accurately generate multivariate time series for each of the classes given as conditions.

Our code is available at https://github.com/unsupervise/MTS-CGAN or https://github.com/MadaneA/MTS-CGAN.

2 Related works

Conditional Generative Adversarial Networks (CGANs) [17] are an extension of the popular vanilla GANs [5]. It allows a certain degree of control over the generated samples by setting a condition that the generation must meet. This condition is generally incorporated in the form of a vector $\mathbf{y}$ concatenated with the noise vector $\mathbf{z} \sim p_{\mathbf{z}}$ at the input of the generator G and provided as an input to the discriminator D. Thus, this corresponds to conditioning the distributions of D and G. Therefore, the standard CGAN objective is defined as

$$\min_G \max_D \mathbb{E}_{\mathbf{x}\sim p_{\text{data}}} [\log D(\mathbf{x}, \mathbf{y})] + \mathbb{E}_{\mathbf{z}\sim p_{\mathbf{z}}} [\log(1 - D(G(\mathbf{z}, \mathbf{y}), \mathbf{y}))] . \qquad (1)$$

Transformer-based GANs. Transformers [21] are a family of architectures relying entirely on the attention mechanism to learn global dependencies between input and output, without any notion of recurrence. It proved to be sufficient to understand and extract features from the input given to the model. TransGAN [9] introduces a GAN built using solely transformers to generate synthetic images. The generator takes a one-dimensional noise vector as an input and gradually increases the resolution of the feature map computed using transformer encoder blocks until a synthetic image with the required resolution is generated. For the discriminator, the authors adopted the architecture of Vision Transformer (ViT) [4], an image classifier based on transformers. Similarly, the authors of TTS-GAN [11] propose an adaptation of TransGAN to multidimensional time series generation, for data augmentation purposes. The generator and discriminator adopt the Trans-encoder architecture, and a multivariate time series is considered as an image of height one, length equal to the number of time steps and the number of variables as the number of channels.

CGANs for Time Series. The TTS-GAN was recently extended in [12] by proposing a conditional version, TTS-CGAN, conditioned on a categorical class label. They notably added a classification head to the discriminator, predicting the input signal's class. Unlike the unconditional version TTS-GAN, the study of the conditional version TTS-CGAN has not received sufficient attention yet. It still lacks a proper and exhaustive evaluation of results for some evaluation metrics.

3 MTS-CGAN: Multivariate Time Series Conditional Generative Adversarial Network

We propose a conditional GAN where the generator and the discriminator are purely transformer-based networks. The MTS-CGAN architecture consists of a generator G and a discriminator D, as shown in Figure 1. This model generates multivariate time series conditioned on a context that can be either categorical variables or another multivariate time series.

The conditional generator (G) takes two inputs: a noise vector $\mathbf{z}$ of dimension d_z and the encoded context to condition the generation. The latent dimension is a hyper-parameter depending on the data. The context is encoded into a latent space of dimension d_z to facilitate its concatenation with $\mathbf{z}$. We then apply linear transformations to the concatenated vectors to obtain a vector of size equal to the target sequence length and with d_c channels, where d_c needs to be tuned. Finally, we encode the position of each element using a positional embedding vector learned jointly with the model during training. The resulting vector passes through the consecutive layers of a Transformer encoder. Each has a multi-head self-attention layer that extracts the contextual inter-dependencies between the generated signal and the provided context. The final output passes through a $(1, 1)$-convolution layer with a number of output channels equal to the target dimension.

Trade-off between variability and condition: Every modeling deals with the object of interest – in our case, a multivariate time series – as a composition of two components: a regular and an irregular one. Contextual knowledge and prior information encode evidence in favor of some propositions and help to characterize the regular component of the time series, while a random variable models the irregular one. Thus, it seems natural to associate a weight α to

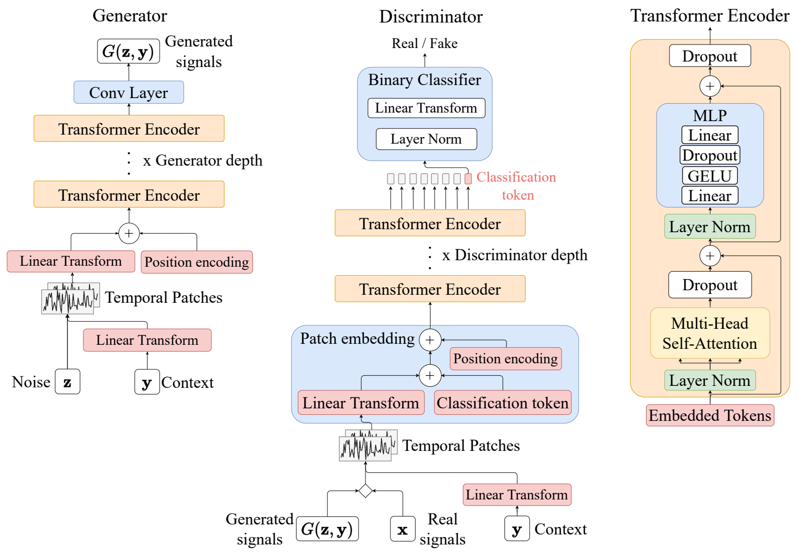


Fig. 1. The proposed MTS-CGAN architecture.

the random variable that models the lack of knowledge. This hyper-parameter $0 < \alpha < 1$ defines the weighting between the desired noise/variability and the relevance given to the context, by complementary. Therefore, we scale the noise $\mathbf{z}$ by multiplying it by α and the context vector $\mathbf{y}$ by $1 - \alpha$.

The conditional discriminator (D) aims to classify the time series as real or synthetic. Its input is either a real or a generated time series, with its corresponding condition. First, it concatenates the input vectors and applies a linear transformation. Then, the resulting embedding is broken down into multiple patches associated with their respective temporal positions and a classification token. The whole set is then fed to the consecutive layers of the Transformer's encoder. Finally, a binary classifier uses the information embedded into the classification token to distinguish between real and fake signals.

Loss function. We compare three different training objectives:

– Standard CGAN objective [5] (see Eq. 1)
– Least Squares GAN (LSGAN) [15], that replaces the cross-entropy loss used in the original GAN with the least squares loss:

$$\min_D \frac{1}{2}\mathbb{E}_{\mathbf{x}\sim p_{\text{data}}}\left[(D(\mathbf{x},\mathbf{y})-1)^2\right] + \frac{1}{2}\mathbb{E}_{\mathbf{z}\sim p_{\mathbf{z}}}\left[(D(G(\mathbf{z},\mathbf{y}),\mathbf{y}))^2\right]$$
$$\min_G \frac{1}{2}\mathbb{E}_{\mathbf{z}\sim p_{\mathbf{z}}}\left[(D(G(\mathbf{z},\mathbf{y}),\mathbf{y})-1)^2\right]$$

– WGAN-GP [6]: an improved version of the WGAN [1]. Instead of the weight clipping method introduced in the WGAN to enforce the Lipschitz constraint, it computes a gradient penalty, which is added to the loss of the discriminator:

$$\min_D \mathcal{L}_{\text{D}}^{\text{WGAN}} + \lambda \mathbb{E}_{\mathbf{z}\sim p_{\mathbf{z}}} \left[(\|\nabla_{\mathbf{z}} D(G(\mathbf{z},\mathbf{y}),\mathbf{y})\|_2 - 1)^2 \right]$$
$$\mathcal{L}_{\text{D}}^{\text{WGAN}} = -\mathbb{E}_{x\sim p_{real}}[D(\mathbf{x},\mathbf{y})] + \mathbb{E}_{z\sim p_z}[D(G(\mathbf{z},\mathbf{y}),\mathbf{y})]$$
$$\min_G -\mathbb{E}_{\mathbf{z}\sim p_{\mathbf{z}}}[D(G(\mathbf{z},\mathbf{y}))]$$

4 Experiments and Evaluations

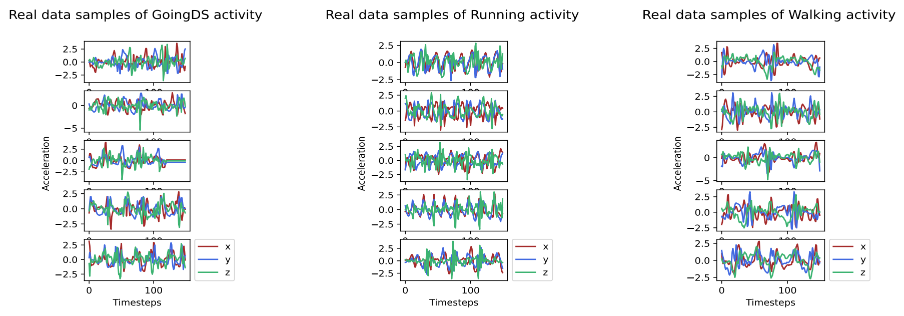


Fig. 2. Five multivariate acceleration signals from the UniMiB dataset. From left to right, samples of Going downstairs, Running and Walking activity classes.

Datasets. In this experiments we only use UniMiB SHAR [16], which is a human activity recognition and fall detection dataset. It consists of acceleration signals on the (X, Y, Z) axes, collected for 9 types of Activities of Daily Living (ADL) and 8 types of falls. In our study, we will focus on generating samples of ADL (see Figure 2). Due to class imbalance, we include three ADL classes (`walking`, `running`, and `going downstairs`), which make up 4034 out of 11771 total samples. Each class of acceleration signals has specific characteristics.

ADL as a categorical condition. We consider activities of daily living class labels as the condition and generate multivariate time series (X, Y, Z) for a given activity using MTS-CGAN. We encode each label as a one-hot numerical vector, apply linear transformations, and concatenate the output to the noise vector $\mathbf{z}$.

The hyperparameter α represents the lack of knowledge. We trained MTS-CGAN using different values of α to evaluate its relevance and impact. We have evaluated the values $\{0.1, 0.25, 0.7, 0.9\}$. One can expect low α values to produce higher inter-class separation and low intra-class variability, while high α values tend to reduce inter-class separation and increase intra-class variability.

4.1 Discussion on α

Intra-class inertia represents the variability within classes. It may increase (resp. decrease) with high (resp. small) α values. Also, inter-class inertia is correlated to prior knowledge. The more importance is given to noise, the more diverse the data generated within the classes are; therefore, the greater the interclass inertia and the smaller the intraclass inertia is. Therefore small α values can represent highly separated components. In this section, we present the results of MTS-CGAN with categorical conditions. Figure 3 shows the t-SNE visualization of real and 3-class conditioned generated Multivariate time series for the four MTS-CGAN models trained with different values of α. The global analysis of the four scatterplots corroborates our expectations. Indeed, for low values $(0.1, 0.25)$ of α, inter-class distances for the generated time series (red, green, and Blue) are much more important in comparison with those obtained for high values $(0.75, 0.9)$. Also, since the hyperparameter α weights the noise and the irregular component of the time series, it's natural that low values $(0.1, 0.25)$ of α produce compact components while high values $(0.75, 0.9)$ produce more spread ones.

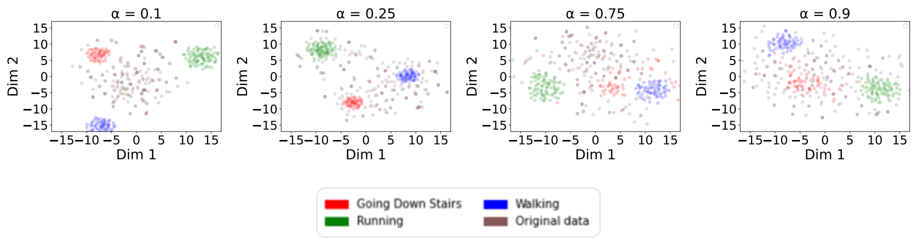
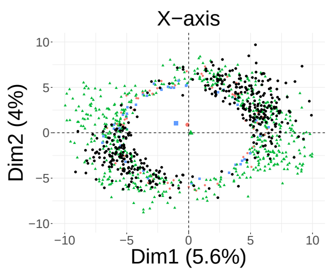


Fig. 3. t-SNE embeddings of real and generated time series for different α values. Real samples from UniMiB are represented in brown. Generated time series for the three ADL classes are represented in blue (walking), red (going downstairs) and green (running).

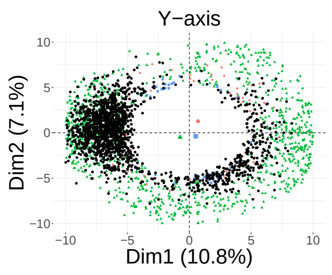
4.2 Evaluation using Principal Component Analysis

Using a properly set-up PCA on the real data, we project generated samples on the two-dimensional principal component plane and compare it to the real data distribution.

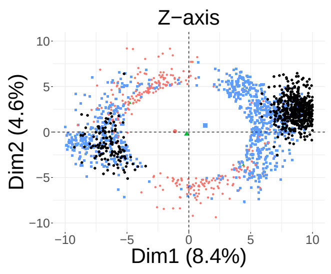
ADL as a categorical condition. Figure 4 shows the projection of the generated data onto the factor space corresponding to the PCA of each axis (X,Y,Z). The data quantity unbalance noticed between the generated and real distributions is due to us keeping only projections with a good quality of representation. Figures (a-c) show that the distribution of the generated data (points in black) overlaps with the real data according to the three axes. It proves that the MTS-CGAN generates synthetic data that looks very similar to the real dataset.



(a) X-axis factor space



(b) Y-axis factor space



(c) Z-axis factor space

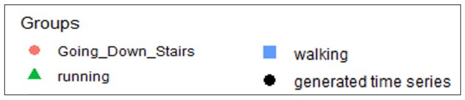
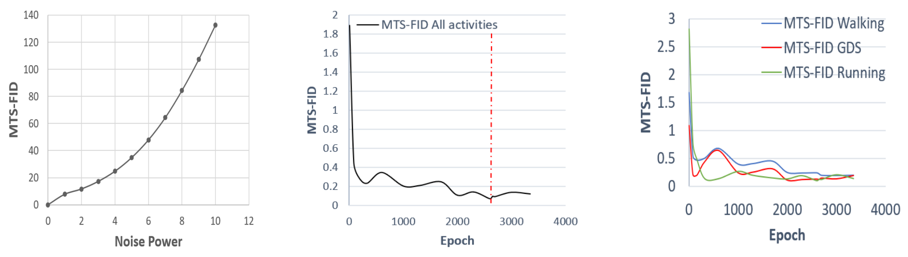


Fig. 4. Projection of real data and data generated by MTS-CGAN on the real data PCA plane, for each of the (X,Y,Z) acceleration axes. Each color indicates one of the three classes in the real data (red: Going downstairs, green: running, blue: walking) and the generated distribution is represented in black.

4.3 FID for Multivariate Time Series

The objective of a generative model is to produce outputs similar to the real data. Therefore, the distance between the generated and real data distribution can be used to measure the model's performance. A well-performing metric to evaluate the quality of GANs is the Frechet Inception Distance (FID) [7]. It calculates the distance between the real and the generated distribution based on statistics computed from a set of samples of each distribution. Considering (μ_r, Σ_r) and (μ_g, Σ_g) are respectively the mean and covariance of the sample



(a) Behavior of MTS-FID when adding white gaussian noise

(b) MTS-FID on all activities during MTS-CGAN training

(c) MTS-FID by activity during MTS-CGAN training

Fig. 5. MTS-FID during MTS-CGAN training (categorical condition).

embeddings from the real data distribution and the generated data distribution, the FID is formulated as follows :

$$FID = \|\mu_r - \mu_g\|^2 + T_r\left(\Sigma_r + \Sigma_g - 2\left(\Sigma_r\Sigma_g\right)^{1/2}\right) \tag{2}$$

However, in the case of time series, Vanilla FID does not fit our case since it uses InceptionV3 which is adapted to the images. Instead, we propose to use a feature extractor adapted to multivariate time series. Some previous works [20] have proceeded likewise to adapt FID to their data type. Therefore, we train a CNN-based classifier inspired by [22] on UNiMiB data to classify (X, Y, Z) acceleration time series according to their activity class (Walking, Running, Jumping, Going downstairs, Sitting down, Going upstairs, Standing up from sitting, Lying down from standing, Standing up from laying), then we eliminate the classification layer keeping only the feature extractor. We test our new FID version (MTS-FID) by computing the score between a collection of real samples and itself, which we expect to be 0. Then, we gradually add white Gaussian noise and compute MTS-FID between the two sets. We expect a higher score as we increase the noise power. Figure 5 (a) shows that similar to the vanilla FID, lower MTS-FID scores correlate with better-quality multivariate times series.

We use MTS-FID to monitor the training of our models. It helps detect model convergence to avoid overfitting. Figure 5 (b) and Figure 5 (c) show the MTS-FID evolution during MTS-CGAN training with categorical conditions. It starts with a high score since, in the beginning, the generator produces noise, then it gradually decreases as the generator becomes more capable of generating realistic data. The red dotted line shows the exact epoch when the model reached the smallest value of MTS-FID during its training and thus its convergence. Using the previous evaluation methods, we find that stopping the training at this stage also gives the best results. This proves the correlation between the different evaluation methods. We also use MTS-FID to compare models trained using different hyperparameters, alpha values, and loss functions, as shown in tables 1 and 2.

Table 1. MTS-FID scores of MTS-CGAN trained using different loss functions.

Loss function	All activities	Walking	GDS	Running
Standard	0.23	0.34	0.14	0.42
LSGAN	**0.07**	**0.23**	**0.12**	**0.10**
WGAN-GP	2.81	2.58	1.82	3.89

Experiments conducted on the generation task with categorical conditions and presented in Table 1 were achieved with $\alpha = 0.9$, and results in Table 2 were obtained using the LSGAN loss. Results show that MTS-CGAN generates better quality data when trained on LSGAN combined with $\alpha = 0.9$. Also, it shows that it did learn to model the activities of Walking and Going downstairs better than the activity of Running.

Table 2. MTS-FID values of MTS-CGAN trained using different α values.

α	All activities	Walking	GDS	Running
0.9	**0.07**	0.23	**0.12**	**0.10**
0.75	0.26	**0.19**	**0.12**	0.79
0.25	0.43	0.50	0.53	0.99
0.1	0.44	0.56	0.74	0.28

Table 3. Classification performance before and after data augmentation

	Before Data Augmentation					After Data Augmentation				
	Train size	Precision	Recall	F1-score	Acc.	Train size	Precision	Recall	F1-score	Acc.
Going Down Stairs	782	0.77	0.77	0.77	88%	1164	0.91	0.80	0.92	**93%**
Walking	1132	0.85	0.96	0.90		1132	0.87	0.99	0.92	
Running	1164	1.00	0.86	0.94		1164	1.00	0.96	0.98	

4.4 Data augmentation using MTS-CGAN

We built a multivariate time series classifier that classifies each MTS (X, Y, Z) according to its respective ADL (walking, running, going downstairs). We split the data into training, validation, and test sets. First, we train and test the classifier on the original train/val/test splits. Second, we balance the number of samples in the training set for each class by performing data augmentation using MTS-CGAN (with categorical conditions), train the classifier on this augmented training set, and test it on the original test set of 1013 samples, all classes included. Results show that enhancing the data with our model's synthetic samples improves the classifier's performance, especially on the minority class: F1 score increased from 77% to 92%, as shown in Table 3. Note that this method has already been used to evaluate a conditional GAN on multivariate time series in [2].

5 Conclusion

In this paper, we presented a novel approach to model multivariate time-series data for conditional generation tasks, solely based on Transformer architecture, thus dispensing recurrence entirely. The training process is efficient, and the model simultaneously learned the observed multivariate time series data distribution for each condition. Self-attention mechanisms proved their ability to learn the conditional aspect of the generation while learning the complex dependencies between the multivariate time series. Moreover, we conducted multiple rigorous evaluations in order to validate MTS-CGAN. The qualitative and quantitative evaluation results showed the high quality of the synthetic data and the effectiveness of MTS-CGAN in generating realistic multivariate time series using categorical conditions. This emphasizes the success of attention in conditional modeling of multivariate time series and its potential to model more complex conditions, a future direction we plan to explore.

References

1. Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein generative adversarial networks. In: International conference on machine learning. pp. 214–223. PMLR (2017)
2. Chen, Y., Kempton, D.J., Ahmadzadeh, A., Angryk, R.A.: Towards synthetic multivariate time series generation for flare forecasting. In: International Conference on Artificial Intelligence and Soft Computing. pp. 296–307. Springer (2021)
3. Conneau, A., Lample, G.: Cross-lingual language model pretraining. Advances in neural information processing systems **32** (2019)
4. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)
5. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. Advances in neural information processing systems **27** (2014)
6. Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., Courville, A.C.: Improved training of wasserstein gans. Advances in neural information processing systems **30** (2017)
7. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. Advances in neural information processing systems **30** (2017)

8. Hu, T., Long, C., Xiao, C.: A novel visual representation on text using diverse conditional gan for visual recognition. IEEE Transactions on Image Processing **30**, 3499–3512 (2021)
9. Jiang, Y., Chang, S., Wang, Z.: Transgan: Two transformers can make one strong gan. arXiv preprint arXiv:2102.07074 **1**(3) (2021)
10. Leznik, M., Michalsky, P., Willis, P., Schanzel, B., Östberg, P.O., Domaschka, J.: Multivariate time series synthesis using generative adversarial networks. In: Proceedings of the ACM/SPEC International Conference on Performance Engineering. pp. 43–50 (2021)
11. Li, X., Metsis, V., Wang, H., Ngu, A.H.H.: Tts-gan: A transformer-based time-series generative adversarial network. arXiv preprint arXiv:2202.02691 (2022)
12. Li, X., Ngu, A.H.H., Metsis, V.: Tts-cgan: A transformer time-series conditional gan for biosignal data augmentation. arXiv preprint arXiv:2206.13676 (2022)
13. Liu, Z., Yin, X.: Lstm-cgan: Towards generating low-rate ddos adversarial samples for blockchain-based wireless network detection models. IEEE Access **9**, 22616–22625 (2021)
14. Lu, C., Reddy, C.K., Wang, P., Nie, D., Ning, Y.: Multi-label clinical time-series generation via conditional gan. arXiv preprint arXiv:2204.04797 (2022)
15. Mao, X., Li, Q., Xie, H., Lau, R.Y., Wang, Z., Paul Smolley, S.: Least squares generative adversarial networks. In: Proceedings of the IEEE international conference on computer vision. pp. 2794–2802 (2017)
16. Micucci, D., Mobilio, M., Napoletano, P.: Unimib shar: A dataset for human activity recognition using acceleration data from smartphones. Applied Sciences **7**(10), 1101 (2017)
17. Mirza, M., Osindero, S.: Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784 (2014)
18. Mohammadi Farsani, R., Pazouki, E.: A transformer self-attention model for time series forecasting. Journal of Electrical and Computer Engineering Innovations (JECEI) **9**(1), 1–10 (2020)
19. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al.: Language models are unsupervised multitask learners. OpenAI blog **1**(8), 9 (2019)
20. Smith, K.E., Smith, A.O.: Conditional gan for timeseries generation. arXiv preprint arXiv:2006.16477 (2020)
21. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017)
22. Wang, Z., Yan, W., Oates, T.: Time series classification from scratch with deep neural networks: A strong baseline. In: 2017 International joint conference on neural networks (IJCNN). pp. 1578–1585. IEEE (2017)
23. Wu, N., Green, B., Ben, X., O'Banion, S.: Deep transformer models for time series forecasting: The influenza prevalence case. arXiv preprint arXiv:2001.08317 (2020)