IE643: Deep Learning - Theory and Practice **July-Nov 2023**

End-term Project Report: TTS GAN

Team Name: Sigma Enigma *Team Members: Samagra Vijaywargiya (22M0570)*
Saurabh Kumar Khandelwal (22M0574)

Abstract

In this project report, we address the common challenge of limited data in medical machine learning applications, particularly when dealing with time series datasets such as electrocardiogram (ECG) signals. The conventional data augmentation techniques for time-series data are constrained by the necessity to preserve the fundamental characteristics of the signal. To overcome this limitation, we propose TTS-GAN, a novel approach leveraging a transformer-based Generative Adversarial Network (GAN) for the generation of realistic synthetic time-series data sequences. This innovation allows for the generation of synthetic time-series data sequences of arbitrary length, closely resembling real-world ECG signals. We validate the performance of our model through visualizations and dimensionality reduction techniques, demonstrating the striking similarity between the generated and authentic time-series data.

1. Introduction

In the realm of deep learning applications for physiological time-series signals, a significant challenge persists- the scarcity of data. Unlike the abundant datasets readily available for computer vision (CV) and natural language processing (NLP) tasks, physiological signals are acquired through sensor measurements tied to intricate physical or biological processes. Particularly in scenarios involving human subjects, the process of data collection, annotation, and interpretation entails substantial costs. Additionally, the diverse collection configurations further complicate the amalgamation of data from different settings, hindering the creation of extensive datasets crucial for successful deep learning model training. The insufficiency of data in medical machine learning research poses a dilemma, compelling researchers to compromise by training shallower deep learning models. Unfortunately, this compromise limits the models' capacity to capture the full complexity of the physiological problems at hand, leading to overfitting and diminished generalization capabilities.

Amidst these challenges, Generative Adversarial Networks (GANs), introduced in 2014, have emerged as a promising avenue. Widely recognized for their success in generating and manipulating data in CV and NLP domains, GANs have also seen an increasing application in time-series and sequential data generation, including forecasting. This report delves into a comprehensive review of GAN implementations on time-series data, exploring their potential to address data scarcity issues in the context of physiological signals.

A GAN comprises a generator and discriminator, both typically neural network models engaged in a zero-sum game. The generator produces synthetic data akin to real training data, while the discriminator distinguishes between real and generated data. Through alternating back-propagation updates, these components play a crucial role in achieving equilibrium. The transformer architecture, leveraging multiple self-attention layers, has recently gained prominence in deep learning models. Surpassing other neural network architectures in various domains, the transformer exhibits universal computation engine properties. This report introduces a pioneering approach where a pure transformer-based GAN model is constructed, showcasing its performance in multiple image synthesis tasks.
By leveraging the transformer's capabilities within the GAN framework, this project aims to overcome data scarcity challenges in physiological time-series signals. The exploration of this novel approach holds potential not only for enhancing the quality of synthetic data but also for streamlining the training process, presenting a significant advancement in the field of medical machine learning research.

Our project makes significant contributions in the following key aspects:

1. In-Depth Conceptual Explanation: We break down these complex concepts, providing a clear understanding of transformers' self-attention layers and GANs' generator-discriminator dynamics, therefore simplifying the theoretical aspects for readers interested in the fusion of transformers and GANs for time-series generation.

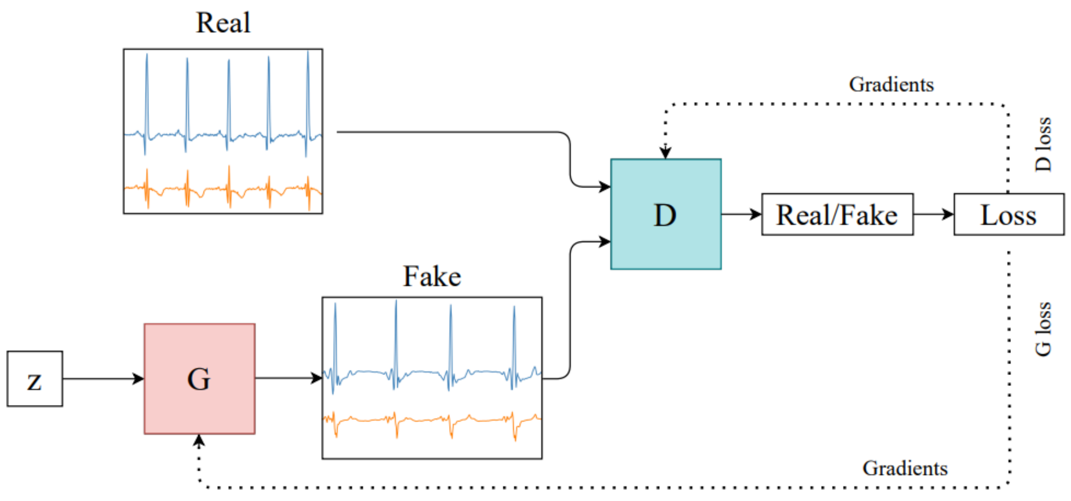
2. Development of a Pure Transformer-Based GAN Model: A pure transformer-based Generative Adversarial Network (GAN) model is designed explicitly for the generation of synthetic time-series data. This departure from conventional approaches signifies a novel application of transformer architectures in addressing the challenges posed by physiological time-series signals.

3. Qualitative and Quantitative Evaluation: We conduct a comprehensive evaluation of the generated sequences produced by our pure transformer-based GAN model. The assessment encompasses both qualitative and quantitative analyses, comparing the quality of the synthetic time-series data against real-world sequences.

The rest of the paper is organized as follows. Section 2 discusses the background and most popular applications of GANs and transformer models. In Section 3, we provide the details of our TTS-GAN model architecture and how we process time-series data to feed this model. After describing the dataset in section 4, we give details on experiments in Section 5, followed by the results presented in section 6. Description of future work is given in Section 7. Section 8 summarizes our work and concludes this paper.

2. Literature Survey

In recent times, there has been a notable trend in leveraging Generative Adversarial Networks (GANs) for the generation of time-series and sequential data. A comprehensive review paper [2] provides a detailed overview of GAN implementations in this domain, emphasizing the advantages of GANs as tools for time-series data augmentation. Notably, GANs address challenges related to data scarcity by augmenting smaller datasets and generating novel, hitherto unseen data. They prove effective in recovering missing or corrupted data while mitigating data noise. Furthermore, GANs contribute to data privacy by generating differentially private datasets devoid of sensitive information from the source datasets.



$$\underset{G}{min}\,\underset{D}{max}\,V(G,D) = \mathbb{E}_{x\sim p_{data}(x)}[logD(\mathbf{x})] + \mathbb{E}_{z\sim p_{\mathbf{z}}(z)}[log(1 - D(G(\mathbf{z})))]$$

Figure 1: Generative Adversarial Network (Wang et al.)

The review paper highlights various state-of-the-art GAN models and algorithms tailored for time-series data generation, including C-RNN-GAN, RCGAN, TimeGAN, and SigCWGAN. It's noteworthy that these models predominantly rely on recurrent neural networks (RNNs) as the foundational architecture for their GAN models. However, a key limitation of RNN-based GANs emerges in their struggle to generate long synthetic sequences with sufficient realism for practical use. This challenge stems from the sequential processing of time-steps, wherein more recent time-steps exert a disproportionate influence on the generation of subsequent time-steps. In essence, RNNs face difficulties in establishing relationships between distant time-steps within lengthy sequences.

The transformer architecture, first proposed by Vaswani et al. (2017) [3] for machine translation, relies on multiple self-attention layer, has recently become a prevalent deep learning model architecture. It has been shown to surpass many other popular neural network architectures, such as CNNs over images and RNNs over sequential data in classification tasks. In [4], the author draws inspiration from the successful scaling of Transformers in Natural Language Processing (NLP) and extends their application to image data with minimal modifications. The approach involves treating image patches as analogous to tokens in NLP, presenting them as sequences of linear embeddings to a standard Transformer. This innovative adaptation is employed for image classification in a supervised learning setting.
The discriminator architecture, resembling the Vision Transformer (ViT) model, demonstrates a binary classification task. Similar to the ViT model, the discriminator assesses whether the input sequence corresponds to a genuine or synthetic signal. This approach is reminiscent of the ViT model's division of images into uniformly sized patches for analysis.

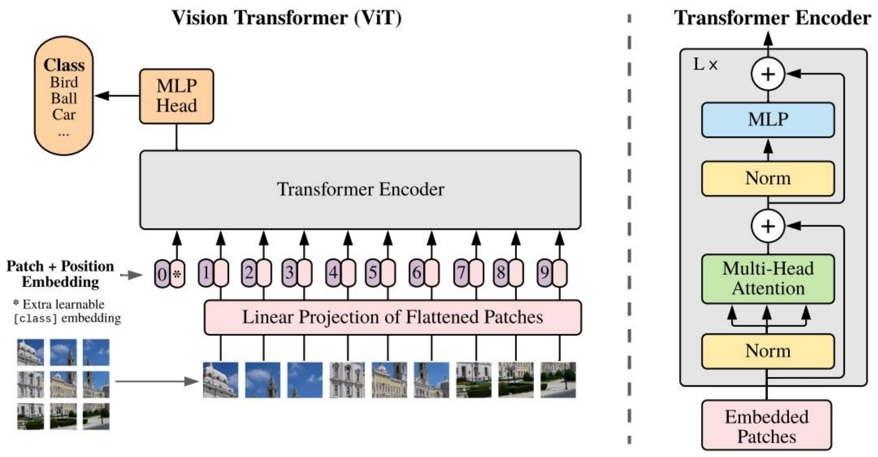


Figure 2: Model overview.

We split an image into fixed-size patches, linearly embed each of them, add position embeddings, and feed the resulting sequence of vectors to a standard Transformer encoder. In order to perform classification, we use the standard approach of adding an extra learnable “Classification token” to the sequence. The illustration of the Transformer encoder was inspired by Vaswani et al. (2017).

Authors in [5] construct a GAN exclusively based on transformer architectures, diverging from convolutional methods. Motivated by the recent success of transformers in computer vision, the authors introduce TransGAN, a novel architecture designed to balance memory efficiency, global feature statistics, and local fine details through a memory-friendly generator, a multi-scale discriminator, and a unique grid self-attention mechanism. Techniques such as data augmentation, modified layer normalization, and the adoption of relative position encoding enhance the optimization and generalization of TransGAN.

3. Methods and Approaches

3.1 Work done before mid-term project review:

Fig. 3 illustrates the architecture of the TTS-GAN model, consisting of two main components: a generator and a discriminator. Both components are constructed based on the transformer encoder architecture, comprising two compound blocks. The first block utilizes a multi-head self-attention module, followed by a second block consisting of a feed forward MLP with a GELU activation function. Normalization layers precede both blocks and dropout layers follow each block, with residual connections employed in both.

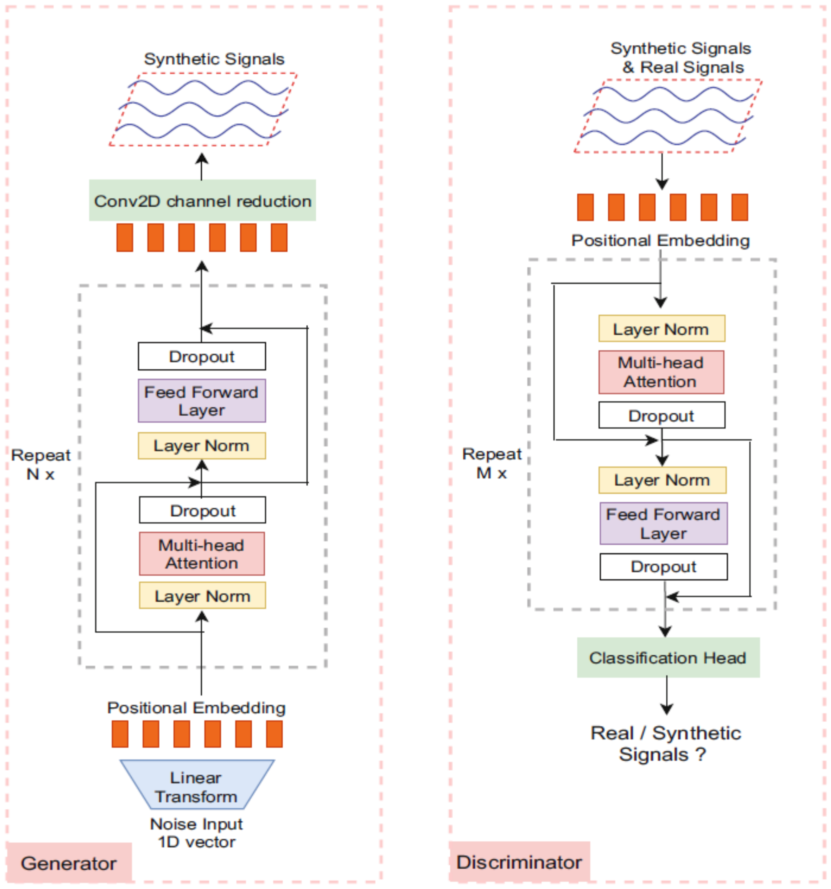


Figure 3: TTS-GAN Architecture

The generator initially receives a 1D vector containing N uniformly distributed random values within the range (0,1), denoted as $N_i \sim U(0, 1)$. Here, N represents the latent dimension of synthetic signals, a tunable hyperparameter. The vector is then transformed into a sequence with the same length as real signals and M embedding dimensions, with M being another hyperparameter. This sequence is divided into patches, and positional encoding values are added to each patch. The patches are subsequently input into the transformer encoder blocks. The encoder block outputs then pass through a Conv2D layer to reduce the synthetic data dimensions. The Conv2D layer has a kernel size of (1, 1) and a filter size matching the dimension of real data sequences, ensuring the width and height of synthetic data remain unchanged. Consequently, a random noise vector is transformed into a sequence with the same shape as real signals.

The discriminator's architecture works as a binary classifier to distinguish between real and synthetic signals. In TTS-GAN, input sequences are treated as images where the timesteps of the inputs represent the image widths. This approach allows for the addition of positional encoding on time series inputs.

Time-series data sequences are viewed as images, where the number of timesteps corresponds to the image width (W). Sequences may have single or multiple channels, akin to the number of channels (RGB) in an image (C). Input sequences are represented as matrices of size (Batch_Size, C, W). A patch size (N) is chosen to divide a sequence into W/N patches, each ending with a soft positional encoding value learned during model training. Consequently, inputs to the discriminator encoder blocks have a data shape of (Batch_Size, C, (W/N) + 1).

Both the generator and discriminator transformer blocks employ Mean Squared Error (MSE) loss for parameter updates. Let z denote input vectors to the generator, G(z) represent synthetic data generated by the generator, and D(x) denote the classification output of the discriminator for real or synthetic signals (x). Real_label is set to 1, and fake_label is set to 0. The discriminator loss is the sum of losses for real and fake data, while the generator loss is represented as MSE loss between the discriminator's output for generated data and real_label. To enhance GAN model training stability, heuristics such as soft labels or flipping label values can be employed, though their effectiveness has been tested on a case-by-case basis. The discriminator loss (d_loss) is given by the sum of d_real_loss (MSELoss(D(real), real_label)) and d_fake_loss (MSELoss(D(G(z)), fake_label)), while the generator loss (g_loss) is expressed as MSELoss(D(G(z)), real_label).

3.2 Work done after mid-term project review:

We conducted a thorough assessment of TTS-GAN using qualitative visualizations and quantitative metrics. To further illustrate the resemblance between real and synthetic data, we employed t-SNE visualizations in Figure 4. The plots depict data point distributions mapped to two dimensions, with red dots representing original data and blue dots denoting synthetic data from TTS-GAN. The distribution patterns between the two types of data remain notably similar.

For a quantitative analysis of similarity, we introduced two metrics: average cosine similarity (avg_cos_sim) and average Jensen-Shannon distance (avg_jen_dis). These metrics were applied to signal features from each signal channel, forming a feature vector for each sequence.

Average Cosine Similarity: When calculating the average cosine similarity between two feature vectors f_a and f_b, each of size m, the cosine similarity is computed using the following formula:

$$cos\_sim_{ab} = \frac{f_a \cdot f_b}{\|f_a\| \, \|f_b\|} = \frac{\sum_{i=1}^{m} f_{ai} f_{bi}}{\sqrt{\sum_{i=1}^{m} f_{ai}^2}\sqrt{\sum_{i=1}^{m} f_{bi}^2}}$$

The average cosine similarity score is then obtained by averaging these cosine similarity values across all pairs of feature vectors corresponding to real and synthetic signals of the same class. The formula for calculating the average cosine similarity is as follows, where n represents the total number of signals:

$$avg\_cos\_sim = \frac{1}{n}\sum_{i=1}^{n} cos\_sim_i$$

This metric provides a measure of the average similarity between feature vectors of real and synthetic signals within the same class.

Average Jensen-Shannon distance: It is determined by calculating the Jensen-Shannon distance for each feature pair from real signals f_{i_real} and synthetic signals f_{i_syn}. The Jensen-Shannon distance for a pair of feature vectors is computed as follows:

$$jen\_sim_i = \sqrt{\frac{D(f_{i\_real}||m) + D(f_{i\_syn}||m)}{2}}$$

Here, m represents the pointwise mean of signals f_{i_real} and f_{i_syn}, and D denotes the Kullback-Leibler divergence. The average Jensen-Shannon distanc is then calculated as the average of these Jensen-Shannon distance values across all feature pairs. The formula is as follows:

$$avg\_jen\_dis = \sum_{i=1}^{m} jen\_sim_i$$

In essence, this metric quantifies the average dissimilarity between each feature from real signals and synthetic signals, considering the Jensen-Shannon distance for each pair.

4. Data set Details

The PTB-XL electrocardiography dataset comprises 21837 clinical 12-lead ECG records of 10 seconds length from 18885 patients, where 52% are male and 48% are female with ages covering the whole range from 0 to 95 years.

5. Experiments

5.1 Training procedure and algorithm

- Real ECG data from the PTB Diagnostic ECG Database is loaded from files using the load_challenge_data function, which reads both signal values and header information.
- The downsample_ecg function is used to downsample the ECG signals from 500Hz to 100Hz and stored.
- Several components for a transformer-based model, including multi-head attention, layer normalization, position-wise feedforward, encoder layers, encoder, and positional encoding are defined.
- A GAN architecture consisting of a Generator and a Discriminator based on a transformer-based architecture (`Encoder`) with positional encoding is used for synthetic ECG data generation.
- The generator takes random noise (z) as input and generates synthetic ECG signals.
- The discriminator is designed to classify whether the input is real or synthetic.
- The training loop iterates over epochs and batches, training the generator and discriminator alternately.
- The generator aims to generate synthetic ECGs that are indistinguishable from real ones, while the discriminator aims to correctly classify real and synthetic ECGs.
- Binary cross-entropy loss is used for both the generator and discriminator.
- Adam optimizer is used for both the generator and discriminator.
- The Generator and Discriminator losses, as well as the cosine similarity and Jensen-Shannon distance between real and synthetic ECG data are displayed.
- Every 5 epochs, a synthetic ECG signal is generated and visualized using Matplotlib.
- t-Distributed Stochastic Neighbor Embedding (t-SNE) is applied to visualize the distribution of real and synthetic ECG data in a 2D space.
- The script saves the generator and discriminator models' state dictionaries every 5 epochs.

5.2 Settings

- Algorithm Settings:

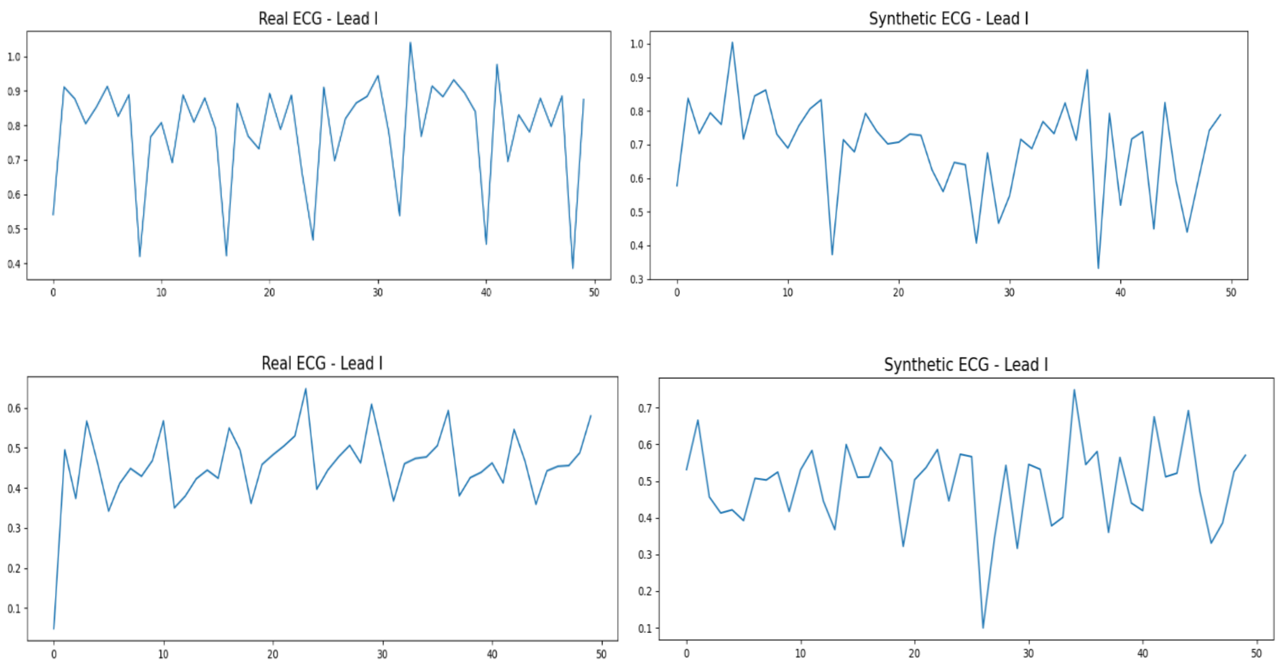
Generator Settings:
- `d_model`: 12
- `num_heads`: 4
- `drop_prob`: 0.1
- `ffn_hidden`: 2048
- `num_layers`: 5

Optimization Settings:
- Learning rate: 0.0002
- Betas: (0.5, 0.999)

Training Settings:
- Batch size: 50
- Maximum sequence length: 50
- Number of training epochs: 100

6. Results

In Fig. 3b, synthetic data samples generated by TTS-GAN are displayed alongside real data in Fig. 3a, showcasing visually similar signal patterns.



(a) Real signals (b) TTS-GAN Synthetic signals

Fig. 3: A visual comparison of real data and their corresponding synthetic data generated by TTS-GAN.

To further illustrate the resemblance between real and synthetic data, we employed t-SNE visualizations in Figure 4.

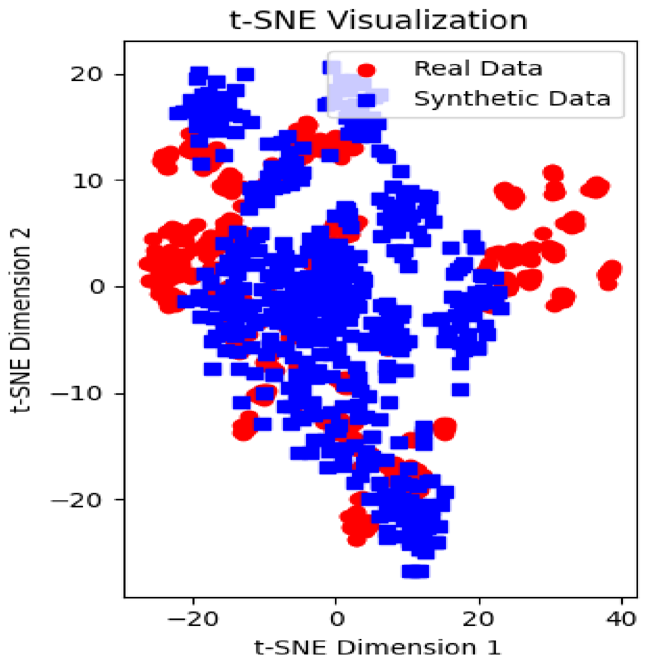


Fig. 4: The t-SNE test for real and synthetic data generated by TTSGAN.

The experimental results, detailed in Table 1, demonstrate average cosine similarity and low Jensen-Shannon distance for synthetic samples across various signal classes.

As suggested in End-term review, we have modified the methodology which involves computing the cosine similarity between flattened representations of real and synthetic tensors using PyTorch's `F.cosine_similarity` function. The tensors are reshaped to facilitate the comparison. The maximum cosine similarity is then selected for each real sample comparing with all synthetic samples, and the average of these maximum values is calculated to provide an overall measure of similarity between the two sets of tensors. This process captures the most significant cosine similarity for each real sample when compared to all synthetic samples.

Score	Value
Average cosine similarity	0.669
Average Jensen-Shannon distance	0.058

Table 1: The metric scores between real data and synthetic data

For the generator and discriminator, losses are plotted for every epoch.

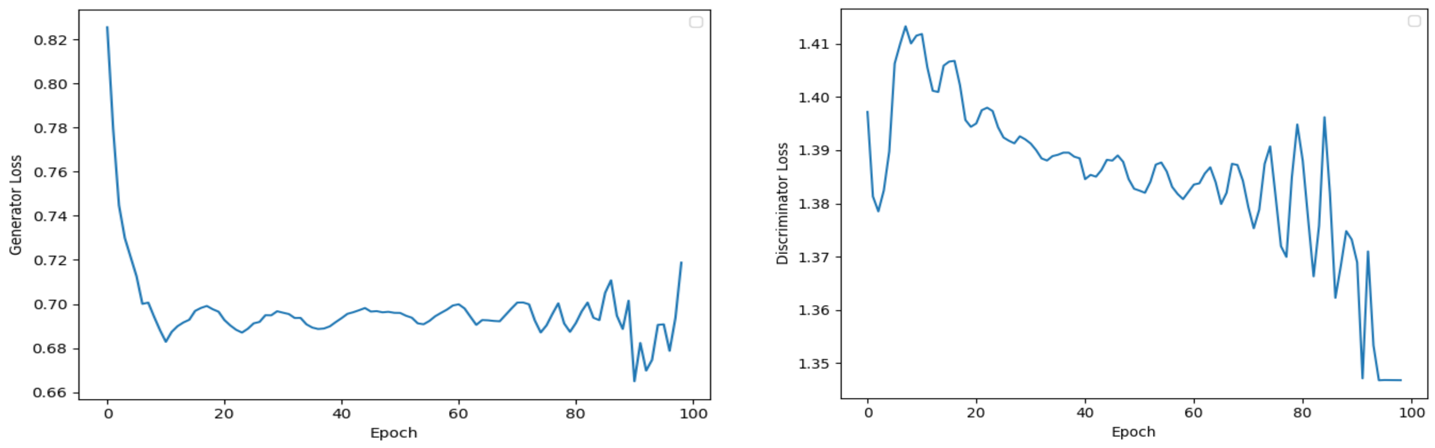


Fig. 5: GAN Losses for real and synthetic data generated by TTSGAN.

7. Scope for Future Work

- Further exploration and validation of the TTS-GAN model should include testing on a more extensive array of datasets exhibiting diverse characteristics. This will enable a comprehensive assessment of the model's robustness and generalization capabilities across various medical contexts.
- Investigating methods to enhance the interpretability and explainability of the TTS-GAN model's synthetic outputs. This could involve attention visualization techniques or developing interpretability tools to provide insights into the model's decision-making process.
- In addressing the current limitation of TTS-GAN, which assumes a unimodal data distribution and generates samples based on this expected pattern, promising future research could involves exploring the integration of a conditional generative adversarial network (cGAN) framework.

8. Conclusion

In this study, we have developed a transformer-based Generative Adversarial Network model, denoted as TTS-GAN, with the capability to generate multi-dimensional time-series data of diverse lengths. Visual comparisons of raw signal patterns and two-dimensional mappings of data point distributions reveal a remarkable similarity between the original and synthetic data. Furthermore, two distinct similarity scores are employed to quantitatively validate the fidelity of the synthetic data. The experimental findings collectively affirm the effectiveness of TTS-GAN as a robust generator of realistic time-series data, particularly when trained on authentic samples.

9. References

[1] Li, X., Metsis, V., Wang, H., Ngu, A.H.H. TTS-GAN: A Transformer-Based Time-Series Generative Adversarial Network. 20th International Conference on Artificial Intelligence in Medicine, AIME 2022, Halifax, Canada. vol 13263. Springer, Cham. (2022) https://arxiv.org/abs/2202.02691 ,

[2] E. Brophy, Z. Wang, Q. She, and T. Ward, “Generative adversarial networks in time series: A survey and taxonomy,” https://arxiv.org/abs/2107.11098 (2021)

[3] Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, pp. 5998–6008. https://arxiv.org/abs/1706.03762 (2017)

[4] Dosovitskiy, A., et al.: An image is worth 16 × 16 words: transformers for image recognition at scale. . https://arxiv.org/abs/2010.11929 (2020)

[5] Jiang, Y., Chang, S., Wang, Z.: TransGAN: two pure transformers can make one strong GAN, and that can scale up. In: Thirty-Fifth Conference on Neural Information Processing Systems https://arxiv.org/abs/2102.07074 (2021)