Artificial Intelligence Research and Development
I. Sanz et al. (Eds.)

doi:10.3233/FAIA230656

Applying Generative Models and Transfer Learning to Physiological Data Classification

José Fernando Núñez [a,1] Jamie Arjona [a] Adrián Tormos [b] Darío García [b] Javier Béjar [a]
[a] *Computer Science Department, Universitat Politècnica de Catalunya (Barcelona Tech), Barcelona, Spain*
[b] *Barcelona Supercomputing Center (BSC), Spain*

Abstract. The scarcity and imbalance of datasets for training deep learning models in a specific task is a common problem. This is especially true in the physiological domain where many applications use complex data collection processes and protocols, and it is difficult to gather a significant number of subjects.

In this paper, we evaluate generative deep learning algorithms by training them to create data based on open physiological datasets and conduct a study on their potential for transfer learning. We measure the performance change of classifiers when the training data is augmented with the synthetic samples and also perform experiments in which we fine-tune classification models trained with the generated data adding increasing amounts of the real data to investigate the transfer learning capabilities of synthetic datasets.

Finally, we advise and provide the best option for researchers interested in augmenting ECG datasets using these algorithms and the best fine-tuning strategies that would generalize correctly when tested on new data from the same domain but for a different classification task.

Keywords. Synthetic Data, Transfer Learning, Time Series, Physiological Signals, ECG

1. Motivation

Given the limited availability of open physiological datasets due to regulations and data privacy, researchers face the problem of finding enough data to train data-hungry deep learning models. In addition, in many applications, there are severe class imbalances due to the scarcity of examples which are very often the class of interest for the diseases studied. One way to amend these problems is the use of synthetic data as a data augmentation technique, as opposed to the more *classical* techniques.

Many of these techniques for the time series [1] domain are inspired by augmentations used in image processing (scaling, transposition, cut-outs, etc.) or are domain-specific transformations (noise, jittering, warping, etc.), but it is difficult for this augmented data to both capture the attributes of the problem and to be more than simply

[1]Corresponding Author: José Fernando Núñez, Universitat Politècnica de Catalunya, E-mail: jose.fernando.nunez@upc.edu.

small alterations of the original data. Also, in the time series domain, applying these techniques results in data that does not capture the underlying dataset distribution.

Generative models provide us with a more interesting way to augment all types of data. From text-to-image generation to NLP tasks, generative data has been a hot topic for quite some time. However, training these models on time series proves harder than other types of data like images and text since there is a very limited literature on generative models in the domain of physiological time series (EEG, ECG, EMG, etc.).

The aim of this paper is to present a short study in this domain, focusing on the selection of a generative model and how to evaluate the performance of the synthetic data, both graphically and when used to augment existing physiological datasets by pretraining a classification model and thus perform transfer learning.

This research is being developed as part of the Artificial Intelligence for Healthy Aging (AI4HA) project. The goal of the project is to develop diagnostic models for aging-associated diseases, including Parkinson's disease, cognitive decline, heart failure, sarcopenia, hearing loss, etc. Most of the data used in the project come from a wide range of physiological time series such as electroencephalography (EEG), electrocardiography (ECG), electromyography (EMG), and inertial sensors (IMU).

2. Modeling data for synthetic generation

Modeling data in machine learning can be approached from two perspectives. The discriminative approach assumes that it is sufficient to train a model to associate the input data to a class label or a continuous value without capturing the inner process behind the data. The generative approach assumes that modeling a problem also involves capturing the process that defines the data, usually represented explicitly or implicitly as a probability distribution. This is a harder problem, but allows us to do sampling on the trained model to obtain new data samples.

Currently, there are many deep learning approaches for modeling data generatively as probability distributions. Some models define proper probability distributions directly from the attributes of the data (autoregressive models [2], normalizing flows [2]) or by learning latent variables models (VAEs [2], Denoising Diffusion [3]); others capture implicitly the data distribution with models that can be sampled from (Generative Adversarial Networks [4]). These models can be used to generate data and have been successfully applied in other domains, especially in natural language processing and for many computer vision tasks.

One advantage of these models is that they can generate data without conditioning, thus capturing a joint probability distribution, or conditionally given additional information that allows partitioning the data distribution accordingly. This allows, for instance, the usage of class labels, but the conditioning factor can be any supplementary information available for the data. In this work, we condition the generative models on the class labels, as we would like to be able to generate data resembling specific classes from the original datasets.

Time series appear in many applications and are a challenge for generative models, since many techniques applied in other areas are difficult to adapt or do not make sense. Literature on applying these models for time series is limited compared to other domains and is usually focused on specific applications and datasets. As a contribution to this

topic, in the next sections we will experiment with different generative models trained with physiological time series, propose methods for assessing the quality of the synthetic data using dimensionality reduction methods, use the trained models to generate data for solving classification tasks, and use the generated data to pretrain models to demonstrate the potential of transfer learning.

3. Models

For data generation, we will select models from the families of Generative Adversarial Networks (GANs) [4] and Denoising Diffusion [3]. The former models the problem of data generation as a game between a network that can obtain samples from a latent Gaussian vector (generator) and a network whose task is to distinguish these generated samples from the real ones (discriminator). The latter model assumes that a network can be trained to progressively add noise to the data samples, and the model then learns how to reverse this process so it can start from Gaussian noise and iteratively remove the noise until a sample with maximium likelihood is obtained.

We train these models conditionally based on the classification task that will be solved. In other words, we take into account the class labels of the dataset when fitting the generative models. For the GANs, we mix the class label with the latent noise that the model learns to convert to data, and we also add classification as an additional output for the discriminator to determine the class of the real and generated samples. This procedure biases the model so that the generator is able to transform the Gaussian latent on samples of the selected class. For the diffusion model, the conditioning appears as an embedded component added to the original noise prediction network. A cross-attention layer is used to bias the model so the path followed by the denoising process ends on sample of the desired class.

We chose to train two GANs, one based on CNNs with 1D convolutions [5] whose discriminator output is an array of numbers corresponding to different parts of the sequence instead of a single number measuring the realness of the data [6], and the other based on transformers [7]. We also chose a diffusion model based on DiffWave [8], a popular model used in audio synthesis. The selected models have been adapted to work with physiological time series and class conditioning, by modifying the architecture to work with single-channel time series and adding a conditional embedding representing the labels attributed to the corresponding datasets.

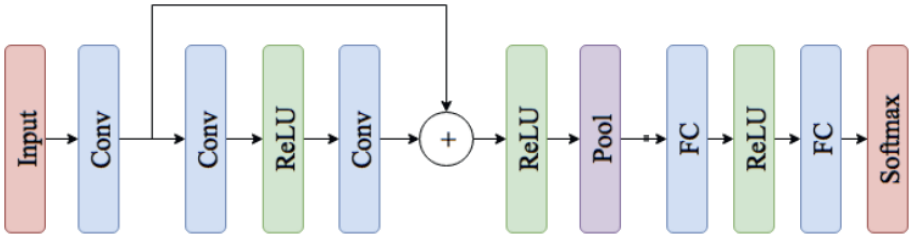

Figure 1. Deep residual convolutional classifier architecture.

For the classification tasks models, to study the capabilities of synthetic data, we have considered training a deep residual convolutional neural network model (Figure 1) under different settings.

4. Datasets

For creating the synthetic data, we used open source datasets from PhysioNet. These consist of ECG signals from various patients with different anomalies. Specifically, the MIT-BIH dataset [9] offers data related to cardiac arrhythmias (five classes based on the categories proposed by the AAMI EC57 standard, subdivided into healthy with 90,589 observations, atrial premature with 2,779 observations, ventricular malfunction with 7,236 observations, fusion of ventricular and normal with 803 observations and a last class of 8,039 unclassified observations), while the PTB dataset [10,11] contains ECG signals corresponding to patients diagnosed with myocardial infarction and healthy patients used as control data (two classes, the abnormal one with 10,505 observations, and the normal or healthy one with 4,045). For both datasets, we used the same preprocessing technique as presented in the work [12], where the ECG signals are divided into QRS segments of length 187.

5. Methodology and Evaluation

When it comes to methods for evaluating the generated data, there is no agreement on the correct way. The most common ones in the literature are the ones used for synthetic images, namely the Fréchet Inception Distance [4] and other similar metrics. In the case of images, this is measured based on the features obtained from a pre-trained network on a massive dataset of natural images. There is, however, no such concept of *natural* time series, nor pretrained networks that can be used for feature extraction, so this is not a viable approach. The evaluation of conditional generation is also an open problem given the difficulty in assessing how well the distribution of the generated data has captured the underlying information present in the real dataset.

In this work, we chose to perform two simple types of evaluation. The first is a 2D plot of the transformation of the data sets using the dimensionality reduction algorithms t-Stochastic Neighbor Embeding (t-SNE, [13]) and Locally Linear Embeding (LLE, [14]), which give visual insight into the distribution of the synthetic data in spaces of reduced dimensionality. Properly generated data would have an *overlapped* distribution with the original dataset, when fitting t-SNE and LLE to both the original and generated data.

The second type of evaluation is the change in the classification scores (*accuracy*, *precision*, and *recall*) when using the aforementioned classifier. We perform several kinds of evaluations on the synthetic quality of generated data: training on synthetic data and testing on real data (TSTR), training on real data and testing on synthetic data (TRTS), training and testing on real data (TRTR), training and testing on synthetic data (TSTS), training on a mix of real and synthetic data and testing on real data (TRSTR), and training on synthetic and adding increasing proportions of real data. These tests were carried out to see how different training affects the classification task, with the aim of improving the performance of the classifiers using only real data.

6. Generative and Classification models experiments

By training all the generative models with the original datasets, we have found the TTS-CGAN and Diffwave models to be the best choices, as they were both successful in different ways. The Diffwave model seems to be better at capturing the underlying probabilistic distribution and also generating an evenly distributed array of synthetic data, while the TTS-CGAN based model generates the best pretraining data for the MIT-BIH dataset. The GAN with 1D convolutions, arguably the simplest model, does not do a good job in capturing the probabilistic distribution of the real dataset with the generated samples, and seems to cluster the data in dense patches corresponding to the class labels.

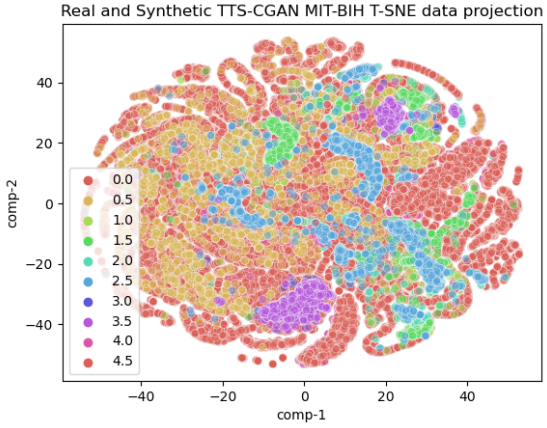

Figure 2. t-SNE fitted with real and synthetic data generated with TTS-CGAN model trained on the MIT-BIH dataset. Real classes are decimals ending in .0 and synthetic ones in .5 (e.g. healthy data: real=0.0, generated=0.5)

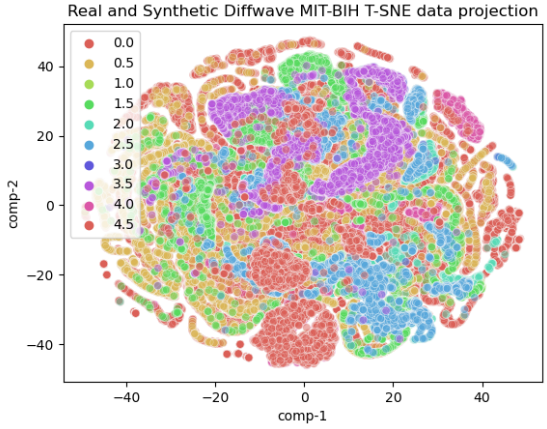

Figure 3. t-SNE fitted with real and synthetic data generated with Diffwave model trained on the MIT-BIH dataset. Real classes are decimals 0 and synthetic ones in .5 (e.g. healthy data: real=0.0, generated=0.5)

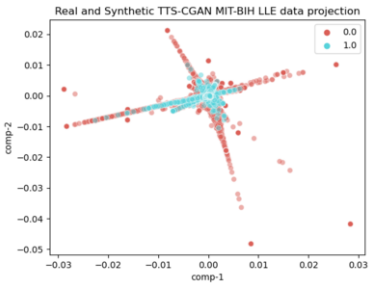

Figure 4. LLE fitted with the MIT-BIH dataset and TTS-CGAN generated data

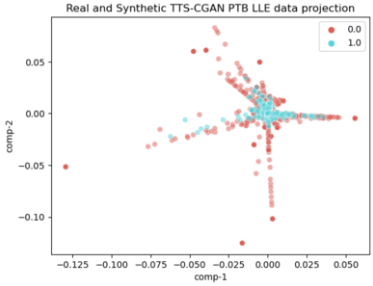

Figure 5. LLE fitted with the PTB dataset and TTS-CGAN generated data

As we can observe in figure 3, the data distribution generated by the Diffwave model trained on the MIT-BIH dataset almost completely overlaps the original data's distribution, informing us that the model learned to capture the underlying attributes of the data and is thus considered of good probabilistic quality. Furthermore, we can appreciate that the TTS-CGAN model struggled to do the same, with many gaps in the synthetic data's overlapped regions. However, we will show how this model could be of interest for augmenting datasets and pretraining classifiers. Furthermore, the LLE proved to be better at showing outliers, as the data was segmented into narrow shapes resembling lines in the reduced space (see figs 4, 5) and most of the data is found near the *center* of the shapes.

For a more practical evaluation, we have trained a deep residual convolutional neural network similar to the one presented in [9] but with only one residual block. The model has been trained and tested with the MIT-BIH and PTB datasets and has been evaluated through its performance on different classification scores, as presented in Table 1.

Next, with the goal of assessing how synthetic data can help in the training a Deep Learning model when data are scarce or highly unbalanced, we focus on the MIT-BIH dataset's minority classes (related to arrythmias), as there is an overwhelming prevalence of the healthy class compared to the others, making the dataset highly unbalanced. Once we have a trained model using a synthetic dataset consisting of 550k samples (110k from each class), we fine-tune it using different sizes of real data, starting with 20% of the real data until we use the whole dataset, with increments of 20%. Table 2 shows the evolution of the scores as more real data is used.

Another experiment performed to evaluate the model trained on synthetic data has been to fine-tune the model with the real data from the PTB dataset changing the task to myocardial infarction classification (binary classification). Similarly as before, we start with a pre-trained model trained with synthetic PTB data (112k observations for each class) then fine-tune the model adding increments of 20% of the real data in order to study the evolution of the classification scores. Results are shown in Table 3.

7. Results

On the basis of the previous results, we can conclude that the data generated by the TTS-CGAN model proved to be more useful and that the t-SNE plots were the most helpful measure of synthetic quality. For the experiments, we first evaluate the synthetic data and compare them with the real data, the results are shown in Table 1. These results

Table 1. Results from testing the models under different train and test combinations.

Dataset	Setting	Accuracy	Precision	Recall
MIT	**TSTR**	0.551	0.430	0.753
	TSTS	0.999	0.999	0.999
	TRTR	0.980	0.961	0.840
	TRSTR	0.971	0.933	0.783
PTB	**TSTR**	0.660	0.660	0.699
	TSTS	1.000	1.000	1.000
	TRTR	0.914	0.908	0.874

Table 2. Comparison table between trained model on synthetic data and fine-tuned on real data vs model trained with real data using different percentages of real data. The data used corresponds to the MIT dataset.

Percentage	Accuracy		Precision		Recall		Execution	
	Synthetic	*Real*	*Synthetic*	*Real*	*Synthetic*	*Real*	*Synthetic*	*Real*
0.2	0.9760	0.9664	0.9098	0.9153	0.8651	0.7700	33.5108	32.1254
0.4	0.9803	0.9741	0.9329	0.9476	0.8703	0.7854	50.4500	51.5347
0.6	0.9814	0.9763	0.9551	0.9586	0.8502	0.8074	65.3325	65.7416
0.8	0.9804	0.9783	0.9591	0.9599	0.8311	0.8162	81.9592	82.6904
1.0	0.9799	0.9751	0.9633	0.9612	0.8168	0.7920	100.0152	110.3127

Table 3. Comparison table between trained model on synthetic data and fine-tuned on real data vs model trained with real data using different percentages of real data. The data used corresponds to the PTB dataset.

Percentage	Accuracy		Precision		Recall		Execution	
	Synthetic	*Real*	*Synthetic*	*Real*	*Synthetic*	*Real*	*Synthetic*	*Real*
0.2	0.6627	0.7785	0.6810	0.7901	0.6684	0.7700	1.7065	7.3009
0.4	0.7888	0.8092	0.8166	0.8196	0.8052	0.7989	12.8148	16.2860
0.6	0.7961	0.8669	0.8131	0.8641	0.8234	0.8562	20.4001	26.8113
0.8	0.8248	0.8621	0.8187	0.8564	0.8504	0.8394	25.0002	34.6997
1.0	0.8221	0.8902	0.7885	0.8660	0.8352	0.8647	27.1476	42.3554

highlight that the synthetic data can be more easily classified (observations of each class are very similar, but very different from observations of other classes). Indeed, a classifier trained and tested with synthetic data has perfect performance, while models trained on synthetic data and tested on real data have very low performance. We added scores from training and testing on real data to give insight into the difficulty of the tasks. An insight to be drawn from this is that the generative models studied produce data maximizing the likelihood that it belongs to a specific class, as evidenced in figure 2 where the mass of the generated data sits well within the boundaries of the real data distribution. This could mean that the generated data is in a sense a *simplification* of the real data, but more work is to be done to find the actual explanation of this behavior.

For the second experiment, we assess the effect of the synthetic data when training a Deep Learning classification model when data are scarce or highly unbalanced. The results (see table 2) show that starting with a model trained with synthetic data provides better classification scores. This is of interest in particular for the case of *precision*

Table 4. Comparison table between trained model on synthetic MIT data and fine-tuned on real PTB data vs model trained with real PTB data using different percentages of real data.

Percentage	Accuracy		Precision		Recall		Execution	
	Synthetic	*Real*	*Synthetic*	*Real*	*Synthetic*	*Real*	*Synthetic*	*Real*
0.2	0.8362	0.7785	0.8551	0.7901	0.8441	0.7700	11.6971	7.3009
0.4	0.8926	0.8092	0.8999	0.8196	0.9036	0.7989	19.8847	16.2860
0.6	0.9452	0.8669	0.9396	0.8641	0.9507	0.8562	33.1194	26.8113
0.8	0.9466	0.8621	0.9359	0.8564	0.9542	0.8394	39.0200	34.6997
1.0	0.9641	0.8902	0.9490	0.8660	0.9655	0.8647	46.4486	42.3554

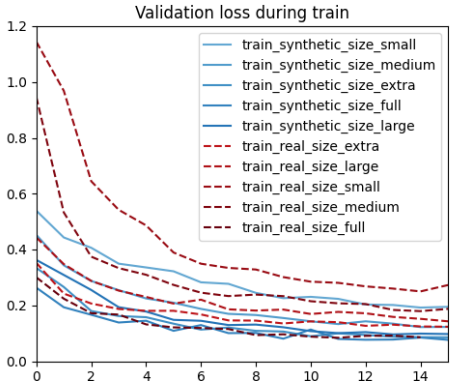

Figure 6. Loss curves on validation set of the fine tuned synthetic model (blue solid lines) and the real data model (red dashed lines) trained under different sizes of the MIT-BIH dataset. Lines become darker as more data has been used

and *recall*, which provide information about the biases in the classifier resulting from class imbalance. While *precision* is nearly the same for the fine-tuned model and the one trained with real data, it is clear that *precision* is better in the fine-tuned model, which improves the final accuracy. Furthermore, although the loss curves (Figure 6) show that models trained with synthetic data start with less loss, we can observe in the second-to-last column (time needed to train the model in seconds) that it is unclear whether the training time is influenced by fine-tuning the model. It is clear that as more data becomes available, less time is needed to train a fine-tuned model. Finally, observe that the scores in both models become more similar as the percentage of real data used increases, probably as an effect of the model becoming accustomed to the real data and *forgetting* patterns from the synthetic data.

The same experiment has been performed with the PTB dataset, where we pre-train the same classification model on PTB synthetic data and then fine-tune it with different sizes of real data from the same dataset. The results (see table 3), show that the synthetic data does not reflect the patterns of the real data with either model. This could be caused by the small amount of data available in the PTB dataset (roughly 1/10th the size of the MIT-BIH dataset), and the difficulty to train a generative model on such a scarce dataset.

Moreover, the third and last experiments consisted in using the classification model pre-trained with synthetic data for the arrythmia detection task (MIT-BIH dataset) and then fine-tuning it in order to solve the myocardial infarction detection task (PTB dataset). Results are shown in table 4, and it can be seen that the fine-tuned model has better overall scores with a considerable margin compared to the classifier trained only on real data. This indicates two things: arrhythmia detection and the myocardial infarction task have similar patterns. Additionally, the synthetic data generated using the MIT-BIH dataset are good enough to capture the patterns that are useful for both tasks. Therefore, in this case of data scarcity, having more data through synthetic generation has significantly improved the baseline.

8. Conclusions and future work

Our results show that synthetic data generation can be useful when classifying a scarce dataset or one with an overwhelming minority class, especially when the task highly depends on the data quantity and quality. This work also shows the need for a better synthetic time series evaluation method, as the t-SNE and LLE plots do not give all the probabilistic insight needed and the classification experiment does not tell us if we are truly capturing the patterns in the distribution of physiological time series.

As future work, it would be of interest to perform new experiments on the recently released PTB-XL data set [15], one of the largest clinical data sets focused on ECG signals, as well as to work with other physiological time series such as EEGs and EMGs. Also of great interest to us is to compare the presented generative models with newer ones coming from the diffusion family or those which incorporate attributes from different families of models. Lastly, we want to find other methods to evaluate the generative quality of these models (e.g. UMAP looks promising) [16].

Finally, beyond physiological time-series data, new generative models for images based on diffusion techniques can also be used to enhance classifiers related to aging-related illnesses, for instance tasks like colon polyp detection on colonoscopies, detection of prostate cancer using MRI scans, histopathology analysis, etc. which are tasks related to our current project as well.

9. Acknowledgments

This research has been financed by the Artificial Intelligence for Healthy Aging (AI4HA, MIA.2021.M02.0007) project from the Programa Misiones de I+D en Inteligencia Artificial 2021.

References

[1] Talavera E, Iglesias G, González-Prieto Á, Mozo A, Gómez-Canaval S. Data Augmentation techniques in time series domain: A survey and taxonomy. arXiv preprint arXiv:220613508. 2022.

[2] Bond-Taylor S, Leach A, Long Y, Willcocks CG. Deep generative modelling: A comparative review of vaes, gans, normalizing flows, energy-based and autoregressive models. IEEE transactions on pattern analysis and machine intelligence. 2021.

[3] Yang L, Zhang Z, Hong S. Diffusion Models: A Comprehensive Survey of Methods and Applications. arXiv preprint arXiv:220900796. 2022.
[4] Gui J, Sun Z, Wen Y, Tao D, Ye J. A Review on Generative Adversarial Networks: Algorithms, Theory, and Applications. IEEE Transactions on Knowledge and Data Engineering. 2023;35(4):3313-32.
[5] Li Z, Ma C, Shi X, Zhang D, Li W, Wu L. TSA-GAN: A Robust Generative Adversarial Networks for Time Series Augmentation. In: IJCNN. IEEE; 2021. p. 1-8.
[6] Isola P, Zhu J, Zhou T, Efros AA. Image-to-Image Translation with Conditional Adversarial Networks. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Los Alamitos, CA, USA: IEEE Computer Society; 2017. p. 5967-76. Available from: https://doi.ieeecomputersociety.org/10.1109/CVPR.2017.632.
[7] Li X, Ngu AHH, Metsis V. TTS-CGAN: A Transformer Time-Series Conditional GAN for Biosignal Data Augmentation. arXiv preprint arXiv:220613676. 2022.
[8] Kong Z, Ping W, Huang J, Zhao K, Catanzaro B. Diffwave: A versatile diffusion model for audio synthesis. arXiv preprint arXiv:200909761. 2020.
[9] Moody GB, Mark RG. The impact of the MIT-BIH Arrhythmia Database. IEEE Engineering in Medicine and Biology Magazine. 2001;20(3):45-50.
[10] Bousseljot R, Kreiseler D, Schnabel A. Nutzung der EKG-Signaldatenbank CARDIODAT der PTB über das Internet. Biomedizinische Technik/Biomedical Engineering. 1995.
[11] Oeff M, Koch H, Bousseljot R, Kreiseler D. The PTB diagnostic ECG database. National Metrology Institute of Germany. 2012.
[12] Kachuee M, Fazeli S, Sarrafzadeh M. ECG Heartbeat Classification: A Deep Transferable Representation. In: 2018 IEEE International Conference on Healthcare Informatics (ICHI); 2018. p. 443-4.
[13] van der Maaten L, Hinton G. Visualizing Data using t-SNE. Journal of Machine Learning Research. 2008;9:2579-605. Available from: http://www.jmlr.org/papers/v9/vandermaaten08a.html.
[14] Roweis ST, Saul LK. Nonlinear dimensionality reduction by locally linear embedding. science. 2000;290(5500):2323-6.
[15] Wagner P, Strodthoff N, Bousseljot RD, Kreiseler D, Lunze FI, Samek W, et al. PTB-XL, a large publicly available electrocardiography dataset. Scientific Data. 2020 May;7(1):154. Available from: https://doi.org/10.1038/s41597-020-0495-6.
[16] McInnes L, Healy J, Melville J. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. The open journal. 2018. Cite arxiv:1802.03426Comment: Reference implementation available at http://github.com/lmcinnes/umap. Available from: http://arxiv.org/abs/1802.03426.