TSA-GAN: A Robust Generative Adversarial Networks for Time Series Augmentation

1st Zheng Li
School of Cyber Science and Engineering, Wuhan University
Wuhan, China
whulizheng@whu.edu.cn

2nd Chao Ma
School of Cyber Science and Engineering, Wuhan University
Wuhan, China
chaoma@whu.edu.cn

3rd Xiaochuan Shi
School of Cyber Science and Engineering, Wuhan University
Wuhan, China
shixiaochuan@whu.edu.cn

4th Dian Zhang
School of Cyber Science and Engineering, Wuhan University
Wuhan, China
zhangdian@whu.edu.cn

5th Wei Li
School of Artificial Intelligence and Computer Science
Jiangnan University
Wuxi, China
cs_weili@jiangnan.edu.cn

6th Libing Wu
School of Cyber Science and Engineering, Wuhan University
Wuhan, China
wu@whu.edu.cn

2021 International Joint Conference on Neural Networks (IJCNN) | 978-1-6654-3900-8/21/$31.00 ©2021 IEEE | DOI: 10.1109/IJCNN52387.2021.9534001

***Abstract*—Time series classification (TSC) is widely used in various real-world applications such as human activity recognition, smart city governance, etc. Unfortunately, due to different reasons, only part of time series could be collected which may obviously degrade the performance of time series classifiers. To alleviate this problem, time series augmentation aims to generate synthetic time series by learning useful features from collected time series. As the popular generative model, generative adversarial networks (GAN) is regarded as a promising model for time series augmentation. However, applying GAN to the time series data suffers from a challenge in which the generated instances hold low quality but the model has gotten saturation. In this paper, for time series augmentation, we proposed TSA-GAN which is a robust GAN model with a self-adaptive recovering strategy to solve this problem. On 85 datasets of the UCR 2015 archive, our proposed TSA-GAN helps time series classifiers achieve performance improvements ranging from 8.3% to 12.5%, which is far better than the baseline.**

***Index Terms*—time series augmentation, GANs, training saturation**

I. Introduction

A time series is a sequence of numerical values ordered by time, which is usually observed with specific devices at a given sampling rate. Classifying time series into multiple categories (a.k.a Time Series Classification) plays an important role in various fields ranging from finance [1] to signal processing [2] [3], from data governance [4] to biological sciences [5]. Deep learning technique is applied to time series classification due to its superiority in automatically identifying discriminative features of time series. However,a well-fitting time series classifier based on deep learning technique largely relies on the training set with adequate time series samples [6]. If the training set contains time series less than the required quantity, the overfitting might be observed on deep learning based time series classifiers. To prevent the potential overfitting, it is feasible to generate synthetic time series and thus form a new larger training set [7]. This task is named as Time Series Augmentation (TSA).

Improving the model performance by augmenting existing data samples has been applied to multiple fields such as computer vision and achieves satisfactory results [8]. But it is difficult to design time series augmentation method by simply copying existing data augmentation techniques. This is mainly because, for synthetic images generated by specific data augmentation techniques, their category will remain unchanged. For example, an image labeled as 'cat' will not be labeled as 'dog' after a series of transformations such as translation, rotation, scaling, etc. For time series, however, it is a non-trivial task to generate synthetic time series with the guarantee that its class label remains unchanged.

Traditionally, window slicing and window warping [9] are two methods for time series augmentation. For window slicing, the length of generated time series will change while segments containing discriminative features might be removed. For window warping, the temporal correlation among time points of the generated time series might be destroyed due to compression or stretching. Thus, the generated time series may lose temporal information presented in the raw time series. An alternative called Dynamic Time Warping(DTW) Barycentric Averaging (DBA) [10] is proposed to generate synthetic time series by calculating the average distance among existing time

Zheng Li and Chao Ma contribute equally to this work. Wei Li is the corresponding author of this paper.

978-0-7381-3366-9/21/$31.00 ©2021 IEEE
Authorized licensed use limited to: WUHAN UNIVERSITY. Downloaded on November 23,2023 at 06:42:05 UTC from IEEE Xplore. Restrictions apply.

series with DTW. However, DTW is very time-consuming which dramatically reduces the computation efficiency of DBA. Moreover, the averaging strategy adopted by DBA is not able to generate interpolation outside of the given range.

As a popular generative model, Generative Adversarial Networks (GAN) [11] is widely applied based on deep learning technique. The framework of GAN contains two main modules: the generator (G) and the discriminator (D). G and D learn from each other in an adversarial manner to generate synthetic samples. Since GAN was proposed in 2014 [11], this excellent generative model has been applied in many fields, especially in image synthesis [12] [13]. In recent years, GAN has also been used in image augmentation and proved to be effective [14]. In the field of time series augmentation, however, the application of GAN is still immature [15].

As observed in our works, a problem may happen from time to time during the training procedure which severely degrades the quality of synthetic data samples. We term such a scenario as "training saturation" because the training will be saturated and the gradient of the generator will stop updating. Therefore, in this paper, we are interested in applying GAN models to time series augmentation and evaluating their effectiveness. Firstly, by incorporating GAN models, a general framework for time series augmentation is proposed in this paper. Secondly, to avoid a severe training saturation problem, a new GAN model called TSA-GAN with a self-recovering strategy is designed. Specifically, in TSA-GAN, a monitor is implemented to detect potential training saturation during the training process. Once any training saturation is detected, the recovery mode will be activated to restore the generator so that the training process can continue. Finally, to compare the performance of different GAN models for time series augmentation, a set of experiments are conducted on 85 datasets of the UCR 2015 archive [16]. According to the experimental results, our proposed TSA-GAN helps time series classifiers achieve performance improvements ranging from 8.3% to 12.5%, which obviously outperforms the baselines.

In this paper, our contributions could be summarized as follows:

- A framework using GAN models for time series augmentation is proposed in this paper.
- A new method to detect the training saturation and make GAN models recover from that problem is designed.
- Comprehensive experiments are conducted on 85 time series datasets of the UCR 2015 archive while in-depth analysis is provided focusing on the impact of each factor on the time series classification performance.

The rest part of this paper is organized as follows. The time series augmentation problem is formally defined in Section III. A concrete example about time series augmentation and basic ideas of GAN models are provided in Section II. In Section IV, the methodology is illustrated in detail including the time series augmentation method, the overall architecture of GAN, monitor, recover method, and deep learning based classifiers. In Section V, the comprehensive experiments are conducted. Related works about existing TSC augmentation solutions are discussed in Section VI followed with the conclusion made in Section VII.

II. Preliminaries

A. An Example about Time Series Augmentation

To meet the requirements of time series augmentation, we assume that there is a training set S which has n tags and each tag has an indefinite number of time series. Our goal is to generate synthetic time series so that each tag has enough and equal samples in number. 'enough' here means the training set must be big enough to avoid overfitting problems which will be explained in detail in Section VI. And here "equal" makes sure that, especially for the deep learning classification model, each tag gets enough 'attention' [17]. In deep learning models, we iterate and optimize the models by minimizing the loss function. In the multi-classification problem, if a tag takes up a small proportion in the training process, its contribution to the loss function will be reduced. And this will make the model pay less attention to the corresponding category. The goal of our data augmentation method is shown in Figure 1. The black rectangle represents the original training set of time series. The green rectangle represents the set of generated time series. Their length represents the time series number. The objective of time series augmentation technique is to improve the performance of downstream time series analytic tasks (e.g. TSC) by generating the green part and merging both into a new training set with more high-quality time series.

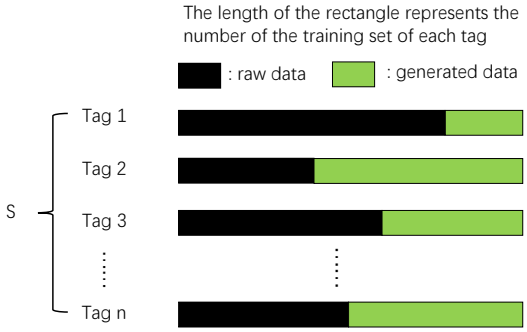


Fig. 1. Schematic Diagram of Time Series Augmentation

B. Basics of Generative Adversarial Networks

Thus, there is a list to tell the generator how much data it should generate for each tag. As for the generator, Generative Adversarial Networks are used for Time Series. Generative Adversarial Networks(GANs) [11] were introduced in 2014 and the model produces good output through the mutual game learning of (at least) two modules in the framework: generative model and discriminative model.

1) Generative model (Generator): that can recover the real data distribution. It receives a random noise Z and generates a picture from this noise, which is called G (z).
2) Discriminative model (Discriminator): that has the task of telling if a sample comes from the training set or the

Authorized licensed use limited to: DALIAN MARITIME UNIVERSITY. Downloaded on November 23,2023 at 06:42:05 UTC from IEEE Xplore. Restrictions apply.

Generator. Its input parameter is x, which may be training data or "false" data generated by the generator. Its output is 1 or 0. 1 represents that the discriminator thinks this is real data, and 0 represents that the discriminator thinks it is false data.

The generative model G has to learn a distribution p_g over data x, by building a mapping function from a prior noise distribution $p_z(z)$ to the data space, $G(z;\theta_g)$, where θ_g are the parameters of the model, e.g. the multilayer perceptrons weights implementing G.

The discriminative model $D(x;\theta_d)$ is a second network implementing a binary classifier, which outputs a single scalar representing the probability that x came from training data rather than p_g.

The two models are trained together to play the following two-player min-max game until they reach the Nash Equilibrium:

$$\min_G \max_D V(D,G) = E_{x\sim p_{data}(x)}[logD(x)]+ E_{z\sim p_z(z)}[log(1-D(G(z)))] \quad (1)$$

III. Problem Formulation

The time series augmentation (TSA) problem could be formulated as follows.

Given:

- A train set of time series S = $\{TS_{train}, C\}$; TS_{train} is a set of training time series, and $TS_{train} = \{ts^o_{train}\}(1 \leq o \leq O)$; C is a set of class labels and C = $\{c_k\}(1 \leq k \leq K)$ in which each tag c_k is attached with one time series $ts^o_{train} \in O$.
- n is the number of classes.
- M = $\{m_n\}$, m_n is the number of data of each tag in S.
- M^* = $\{m^*_n\}$, m^*_n is the target number of data of each tag in S.
- min_m is a default value, which is related to the size of the dataset.

Assume:

- The time series ts^o_{train} with the same label c_k belong to the same distribution p_n.

Objective:

- S_{new} = $\{TS_{train} + TS_{new}, C + C_{new}\}$, TS_{new} is manually generated time series and C_{new} is its tags.

According to the problem formulation of time series augmentation, in Section IV, the overall framework and important design issues are introduced in detail.

IV. Methodology

A. Overall Framework of TSA-GAN

As shown in Figure 2, the overall framework of TSA-GAN is composed of four main components: *Generator*, *Discriminator*, *Monitor*, and *Restorer*. *Generator* is responsible for generating synthetic time series while *Discriminator* judges the authenticity of the input time series and conducts adversarial training with *Generator*. *Monitor* keeps monitoring the loss value of *Generator* during the training process and activates *Restorer* if any training saturation is detected. *Restorer* is specially designed to make a try to drag *Generator* out of the training saturation situation and make it continue the training until the Nash Equilibrium is reached.

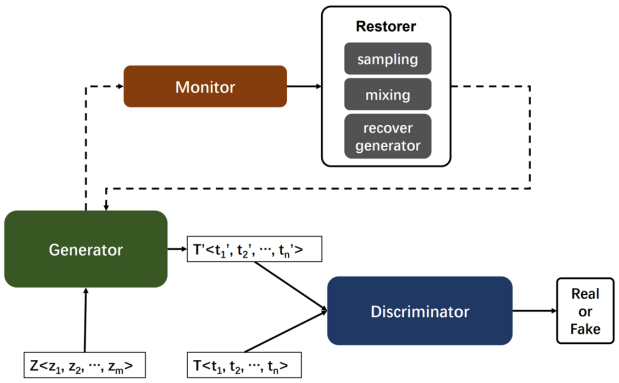


Fig. 2. Overall Framework of TSA-GAN

B. Augmentation Planner

To complete the steps of Figure 1, it is required to calculate M^*. As defined in the Section III, M^* is the size of the target dataset which can be computed with Algorithm 1.

Through the size and distribution of the original training set, the *Augmentation Planner* can calculate the reasonable size of the target data set, and take into account the data balance problem and the performance of the classifier's training at the same time.

It should be noted that many parameters that can be adjusted in the Algorithm 1. In different training sets, these parameters can be also adjusted manually to get better results.

C. Monitor

Monitor is responsible for monitoring the loss value of *Generator* during the whole training process. And if any training saturation sign is detected, *Monitor* must immediately pause the current training process and activates *Restorer* for making *Generator* get rid of the training saturation situation. To accurately detect the potential training saturation, a threshold is employed. If the loss value of *Generator* becomes higher than the predefined threshold, it will be considered that a training saturation appears, and then the *Restorer* will be activated for restoring. More details about *Monitor* could be found in Algorithm 2.

D. Restorer

As *Restorer* starts to work, it will continue the training of *Generator* for several epochs and extract the weights of *Generator* from different epochs. And then, by merging the weights of *Generator* from different epochs, the merged weights of *Generator* would obtain the diversity of the output time series distribution. Therefore, it is expected

Authorized licensed use limited to: DALIAN MARITIME UNIVERSITY. Downloaded on November 23,2023 at 06:42:05 UTC from IEEE Xplore. Restrictions apply.

Algorithm 1 Augmentation Planner

Input: S in
Output: M^* out

$M = [m_1, m_2, m_3, ..., m_n]$ get the num of time series of each tag.
M^* = []
$a\_m = average(M)$
if $a\_m \leq min_m$ **then**
  for each $m_i \in m$ do **do**
    if $m_i < min_m$then **then**
      append min_m to M^*
    else
      append m_i to M^*
    end if
  end for
else
  for each $m_i \in m$do **do**
    if $mi < a\_m * 90\%$ **then**
      append a_m to M^*
    else
      append m_i to M^*
    end if
  end for
end if
return M^*

Algorithm 2 Monitor

Input: $Threshold, G\_loss$ in

$G\_loss.update()$ update G loss
while 1 **do**
  if $G\_loss <= Threshold$ **then**
    $continuetraining$
  else
    $suspendtraining$
    $new_Generator = Recover()$
    $set_Generator(new_Generator)$
  end if
  $G\_loss.update()$ update G loss
end while

that $Generator$ with the newly merged weights would get recovered from the training saturation. More details about $Restorer$ could be found in Algorithm 3.

When the data augmentation part is ready, the next step is to use the data to train a classifier. Fully Convolutional Network (FCN) [18] classifier is selected, we test two classifiers, one is trained with data augmentation and the other is trained without it, by their performances on the test set.

V. Evaluation

A. Experimental Settings

As the DE facto standard, the UCR archive has been chosen by many researchers for evaluating time series analytic models in various kinds of tasks. In this paper, the vanilla GAN and our proposed TSA-GAN are validated on the 85

Algorithm 3 Restorer

Input: $Generator, epoch, Recover\_times, Frequency$ in
Output: $new\_Generator$ out

$counter = 0$
$Weights = []$
$new\_weight = []$
while $s < S_num$ **do**
  $Generator.train()$ Train Generator
  if $epoch\%Frequency == 0$ **then**
    $Weights.append(get_weight())$Extract the weights of each layers from Generator
    $s = s + 1$
  end if
end while
for $weight \in Weights$ **do**
  $new\_weight = mix(weight, new\_weight)$ mix weight in same position
end for
return new_weight

datasets of the UCR 2015 archive. For the sake of fairness, both the vanilla GAN and TSA-GAN adopt exactly the same neural network structure (i.e. multi-layer perceptron denoted as MLP for short) and hyper-parameters setting. In detail, min_m and threshold are set to 32 and 1.2, respectively. And Recover_times and Frequency are set to 6. Since it is difficult to directly evaluate the quality of the generated time series, the time series classification task is utilized to help distinguish the effectiveness of vanilla GAN and TSA-GAN in the task of time series augmentation. Thus, the fully convolutional network (FCN) is employed to play the role of a time series classifier. The detailed structures of MLP and FCN are shown in Figure 3.

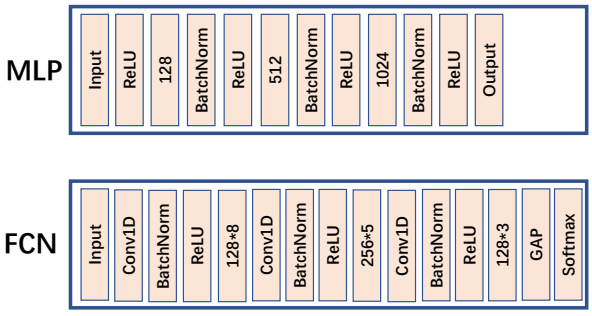


Fig. 3. Structures of MLP & FCN

B. Experimental Results

GAN and TSA-GAN are applied to time series augmentation on the training set of each dataset of the UCR 2015 archive, respectively. And as stated in Section V-A, it is feasible to evaluate the effectiveness of GAN and TSA-GAN by comparing the classification accuracy of FCN on the test set with or without time series augmentation. If the classification

Authorized licensed use limited to: DALIAN MARITIME UNIVERSITY. Downloaded on November 23,2023 at 06:42:05 UTC from IEEE Xplore. Restrictions apply.

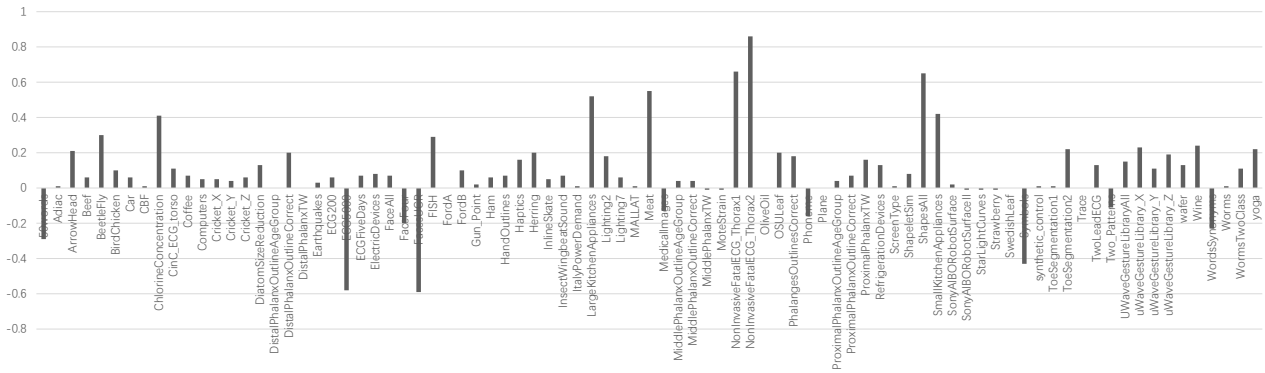


Fig. 4. Classification Accuracy Changes using GAN for Time Series Augmentation on 85 Datasets of UCR 2015 Archive

accuracy is improved by at least 0.01 by using specific time series augmentation technique, it will be considered as a win. If the classification accuracy is decreased for at least 0.01 lower due to the usage of specific time series augmentation technique, it will be considered as a loss. Otherwise, it will be regarded as a tie. According to the experimental results shown in Table I, TSA-GAN won on 64 datasets while W_DBA and GAN models only won 56 datasets. And TSA-GAN gets failed to obviously improve the classification accuracy on only 4 datasets while the other two baselines failed on 9 and 22 datasets.

TABLE I
COMPARISON OF MODEL PERFORMANCE IN TERM OF NO. OF WIN/LOSS/TIE DATASETS

Models	*Win*	*Loss*	*Tie*
W_DBA	56	22	7
GAN	56	9	20
TSA-GAN	64	4	17

Win: if the accuracy is at least 0.01 higher.
Loss: if the accuracy is at least 0.01 lower.
Tie: otherwise.

Moreover, as shown in Table II, TSA-GAN's final average accuracy is the best, and its negative impact on individual data is also the smallest, which shows that TSA-GAN is a robust model.

TABLE II
COMPARISON OF MODEL PERFORMANCE IN TERM OF ACCURACY OF AVERAGE INCREASE/AVERAGE DECREASE/INTEGRATED AVERAGE

Models	*a_increase*	*a_decrease*	*average*
W_DBA	3%	-1%	1.80%
GAN(MLP)	17%	-30%	8.30%
TSA-GAN(MLP)	16%	-3%	12.50%

The overall performance of GAN relative to that of W_DBA already has a good performance, but in the process of experiment, we found that for some data sets, GAN appeared a kind of training saturation phenomenon, leading to the training can not converge, so that the quality of the generated results is very poor, not only can not be used for data augmentation, on the contrary, it will make the performance of the training set worse.

The performance of GAN(MLP) on the data set is shown in Figure 4. The horizontal axis is the name of the data set, and the vertical axis is the difference in the accuracy of the classifier trained before and after data augmentation on the test set.

We checked the training process and found that this poor performance training appeared the training saturation phenomenon, the loss of its generator will rise rapidly, and the training will not converge, as shown in Figure 6.

To solve this problem, we designed the TSA-GAN model. When using TSA-GAN to train the above data sets, we observed the process of TSA-GAN gradually recovering training, stopping the training saturation, and finally converging training, as shown in Figure 7.

The performance of TSA-GAN on the data set is shown in Figure 5. The horizontal axis is the name of the data set, and the vertical axis is the difference in the accuracy of the classifier trained before and after data augmentation on the test set.

C. Analysis and Discussion

The problem of Figure 6 will appear when GAN is used in time series augmentation training. The loss of the generator becomes larger and remains unchanged, resulting in the final failure to learn the distribution of the target data set.

The reason is that in the process of training, the training saturates, which makes the judgment given by the discriminator unable to update the gradient of the generator. The deep reason is that the training degree of the generator is difficult to synchronize with the discriminator. There are many reasons for this kind of problem [19]. If we solve these problems separately, it will be too complicated.

The examples of outputs are shown in Figure 8. Subgraph "a" is the result of GAN training on the dataset "Adiac" and training saturation happened. Subgraph "b" is the result of TSA-GAN after training saturation and then recovery on the same dataset "Adiac". Subgraph "c" is an example of the "Adiac" training set. It can be seen from those graphs that recovery can effectively recover the training process from training saturation and generate time series close to the target.

Authorized licensed use limited to: DALIAN MARITIME UNIVERSITY. Downloaded on November 23,2023 at 06:42:05 UTC from IEEE Xplore. Restrictions apply.

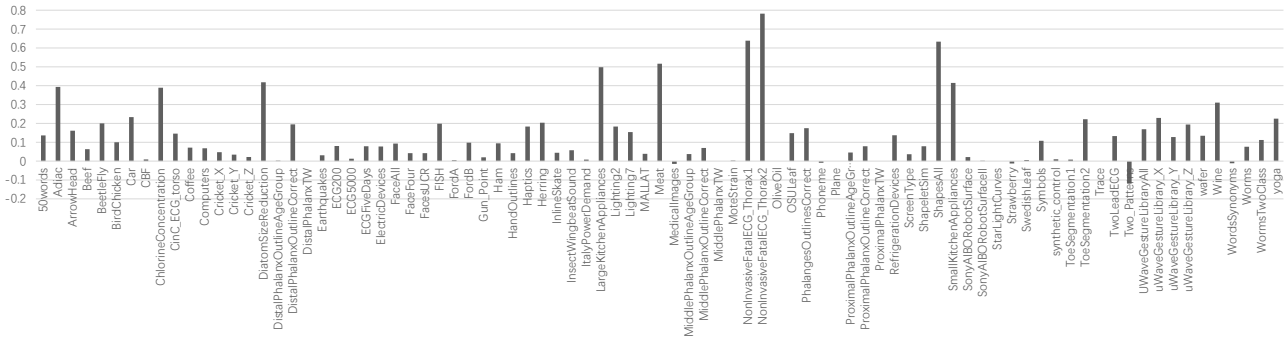


Fig. 5. Classification Accuracy Changes using TSA-GAN for Time Series Augmentation on 85 Datasets of UCR 2015 Archive

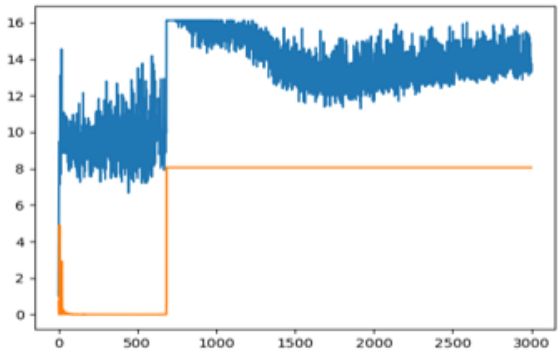


Fig. 6. Loss Values While Training GAN. The horizontal axis of the chart represents the epoch of training, the vertical axis represents the loss value, the blue represents the loss value of the discriminator, and the orange represents the loss value of the generator.

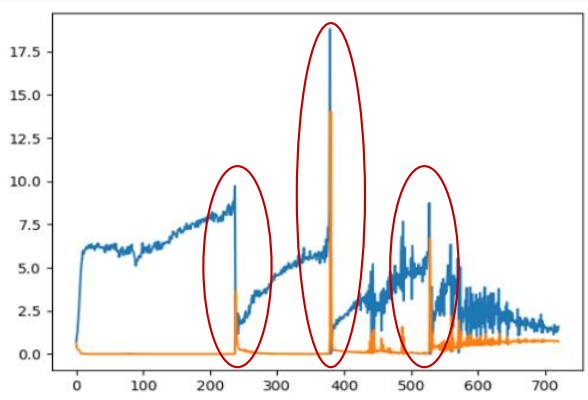
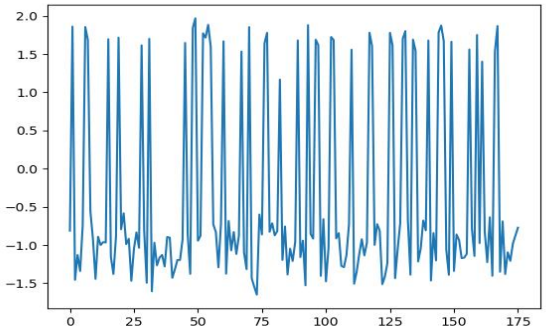
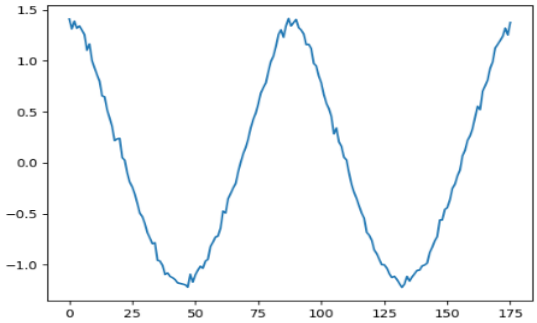


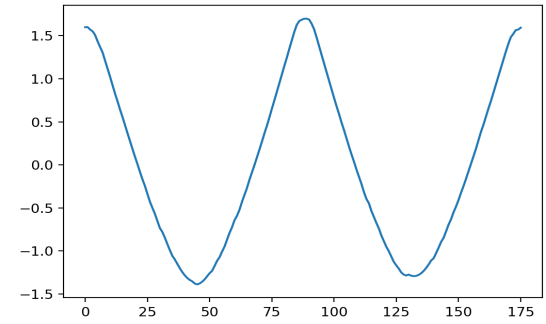
Fig. 7. Loss Values While Training TSA-GAN. The horizontal axis of the chart represents the epoch of training, the vertical axis represents the loss value, the blue represents the loss value of the discriminator, and the orange represents the loss value of the generator, The red round frame indicates that TSA-GAN detects a training saturation and the recovery generator allows training to continue.



(a) Synthetic sample generated by GAN when training saturation is observed. (Epoch: 3000; Dataset: “Adiac”.)



(b) Synthetic sample generated by TSA-GAN. (Epoch: 3000; Dataset: “Adiac”.)



(c) Original sample from the training set. (Dataset: “Adiac”.)

Fig. 8. Synthetic samples generated by GAN and TSA-GAN compared with the original sample from the training set

Authorized licensed use limited to: DALIAN MARITIME UNIVERSITY. Downloaded on November 23,2023 at 06:42:05 UTC from IEEE Xplore. Restrictions apply.

By experiences, it can effectively deal with this situation and restore the training of the generator. By mixing weights, the gradient update of the generator can continue normally instead of falling into the dilemma of training saturation.

VI. Related Work

GANs have many improvements in the computational vision(CV) space to solve their problems [20] and improve their generating quality [21]. Unlike data augmentation for CV [22], data augmentation for time series has not yet been systematically reviewed to the best of our knowledge. The most used data augmentation method for time series classification is Window Slicing and Window Warping [9]. This method is inspired by the image cropping technique for data augmentation in computer vision tasks. It is a subsampling method to randomly extract continuous slices from the original time series. The problem is that, for CV, we can easily tell which parts are important for this picture when cutting, but not for time series. Moreover, in Window Warping, if only for TSC, it works and improves the Support Vector Machines accuracy for classifying electroencephalographic time series [9], but for the time property which is also very important for time series would be changed after Window Warping.

Another interesting method is introduced by [Fawaz et al., 2018]. This method generates new time series with DTW [23] and ensemble them by a weighted version of the DBA [24] algorithm. It performs well theoretically but when we When try to apply it in practice, the DTW algorithm is very time-consuming, especially for long time series.

Recently, T-CGAN [25] is proposed to perform data augmentation on time series, using one CGAN(conditional GAN) [26] model to generate all time series of different tags from one data set. We question its flexibility and training difficulty because it is very difficult to fit all kinds of time series with only one cGAN model, especially the mode collapse problem of GAN has not been solved perfectly.

However, most of the above-mentioned works are not specifically designed for time series augmentation which motivates our work in this paper.

VII. Conclusion

In this paper, we propose a framework named TSA-GAN by incorporating GAN models for time series augmentation. Besides the $Generator$ and $Discriminator$ adopted in vanilla GAN models, $Monitor$ and $Restorer$ are designed and integrated with $Generator$ and $Discriminator$ to prevent GAN models from training saturation during the training process. By conducting the time series classification task on 85 datasets of the UCR 2015 archive, our proposed TSA-GAN helps the time series classifier (i.e. FCN) achieve performance improvements ranging from 8.3% to 12.5%, which is obviously better than the baselines.

Although TSA-GAN has shown superiority over selected baselines, well training GAN models for time series augmentation is still full of challenges. For instance, besides the training saturation problem, mode collapse is another potential risk that may degrade the quality of synthetic time series. The challenging issue is that so far there is no effective way to detect and fix the mode collapse during the training process of GAN models. Therefore, in our future works, we will focus on the study of mode collapse detection and fixing strategies from which the time series augmentation task could benefit a lot.

Acknowledgment

This work is supported by Wuhan University's teaching research project "The innovation of contents construction and organization format of AI laboratory courses" under the category YB-10.

References

[1] Q. Yang and X. Wu, "10 challenging problems in data mining research," *International Journal of Information Technology Decision Making (IJITDM)*, vol. 05, pp. 597–604, 12 2006.

[2] C. Ma, X. Shi, W. Li, and W. Zhu, "Edge4tsc: Binary distribution tree-enabled time series classification in edge environment," *Sensors*, vol. 20, no. 7, 2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/7/1908

[3] C. Ma, X. Shi, W. Zhu, W. Li, X. Cui, and H. Gui, "An approach to time series classification using binary distribution tree," 12 2019, pp. 399–404.

[4] G. Huang, C. Luo, K. Wu, Y. Ma, Y. Zhang, and X. Liu, "Software-defined infrastructure for decentralized data lifecycle governance: Principled design and open challenges," 07 2019, pp. 1674–1683.

[5] D. S. Stoffer and H. Ombao, "Editorial: Special issue on time series analysis in the biological sciences," *Journal of Time Series Analysis*, vol. 33, no. 5, pp. 701–703, 2012. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-9892.2012.00805.x

[6] Y. Zheng, Q. Liu, E. Chen, Y. Ge, and J. L. Zhao, "Time series classification using multi-channels deep convolutional neural networks," in *International Conference on Web-Age Information Management*, 2014.

[7] D. Brain and G. Webb, "On the effect of data set size on bias and variance in classification learning," *Proceedings of the Fourth Australian Knowledge Acquisition Workshop*, 06 2000.

[8] L. Perez and J. Wang, "The effectiveness of data augmentation in image classification using deep learning," 2017.

[9] A. L. Guennec, S. Malinowski, and R. Tavenard, "Data augmentation for time series classification using convolutional neural networks," 2016.

[10] H. Ismail Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, "Data augmentation using synthetic data for time series classification with deep residual networks," 08 2018.

[11] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Advances in Neural Information Processing Systems*, vol. 3, pp. 2672–2680, 2014.

[12] W. Li, L. Xu, Z. Liang, S. Wang, J. Cao, C. Ma, and X. Cui, "Sketch-then-edit generative adversarial network," *Knowledge-Based Systems*, vol. 203, p. 106102, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0950705120303701

[13] W. Li, W. Ding, R. Sadasivam, X. Cui, and P. Chen, "His-gan: A histogram-based gan model to improve data generation quality," *Neural Networks*, vol. 119, pp. 31–45, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0893608019301893

[14] S.-W. Huang, A. Lin, S.-P. Chen, Y.-Y. Wu, P.-H. Hsu, and S.-H. Lai, "Auggan: Cross domain adaptation with gan-based data augmentation," 08 2018.

[15] Q. Wen, L. Sun, X. Song, J. Gao, X. Wang, and H. Xu, "Time series data augmentation for deep learning: A survey," 2020.

[16] H. A. Dau, A. Bagnall, K. Kamgar, C.-C. M. Yeh, and E. Keogh, "The ucr time series archive," 2018.

[17] J. M. Johnson and T. M. Khoshgoftaar, "Survey on deep learning with class imbalance," *Journal of Big Data*, 2019.

[18] E. Shelhamer, J. Long, and T. Darrell, "Fully Convolutional Networks for Semantic Segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, pp. 640–651, 2017.

Authorized licensed use limited to: DALIAN MARITIME UNIVERSITY. Downloaded on November 23,2023 at 06:42:05 UTC from IEEE Xplore. Restrictions apply.

[19] J. Gui, Z. Sun, Y. Wen, D. Tao, and J. Ye, "A review on generative adversarial networks: Algorithms, theory, and applications," 01 2020.
[20] W. Li, L. Fan, Z. Wang, C. Ma, and X. Cui, "Tackling mode collapse in multi-generator gans with orthogonal vectors," *Pattern Recognition*, vol. 110, p. 107646, 2021.
[21] W. Li, Z. Liang, P. Ma, R. Wang, X. Cui, and P. Chen, "Hausdorff gan: Improving gan generation quality with hausdorff metric," *IEEE Transactions on Cybernetics*, 2021.
[22] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, no. 1, 2019.
[23] T. Rakthanmanon, B. Campana, A. Mueen, G. Batista, M. B. Westover, Q. Zhu, J. Zakaria, and E. Keogh, "Searching and mining trillions of time series subsequences under dynamic time warping," vol. 2012, 08 2012.
[24] G. Forestier, F. Petitjean, H. A. Dau, G. I. Webb, and E. Keogh, "Generating synthetic time series to augment sparse datasets," in *2017 IEEE International Conference on Data Mining (ICDM)*, 2017, pp. 865–870.
[25] G. Ramponi, P. Protopapas, M. Brambilla, and R. Janssen, "T-cgan: Conditional generative adversarial network for data augmentation in noisy time series with irregular sampling," 11 2018.
[26] M. Mirza and S. Osindero, "Conditional generative adversarial nets," 11 2014.

Authorized licensed use limited to: DALIAN MARITIME UNIVERSITY. Downloaded on November 23,2023 at 06:42:05 UTC from IEEE Xplore. Restrictions apply.