Neural Computing and Applications (2022) 34:13339–13353
https://doi.org/10.1007/s00521-022-07361-8

S.I.: DEEP LEARNING FOR TIME SERIES DATA

CGAN-based synthetic multivariate time-series generation: a solution to data scarcity in solar flare forecasting

Yang Chen[1] · Dustin J. Kempton[1] · Azim Ahmadzadeh[1] · Junzhi Wen[1] · Anli Ji[1] · Rafal A. Angryk[1]

Received: 16 July 2021 / Accepted: 27 April 2022 / Published online: 30 May 2022

Abstract
One of the major bottlenecks in refining supervised algorithms is data scarcity. This might be caused by a number of reasons often rooted in extremely expensive and lengthy data collection processes. In natural domains such as Heliophysics, it may take decades for sufficiently large samples for machine learning purposes. Inspired by the massive success of generative adversarial networks (GANs) in generating synthetic images, in this study we employed the conditional GAN (CGAN) on a recently released benchmark dataset tailored for solar flare forecasting. Our goal is to generate synthetic multivariate time-series data that (1) are statistically similar to the real data and (2) improve the performance of flare prediction when used to remedy the scarcity of strong flares. To evaluate the generated samples, first, we used the Kullback–Leibler divergence and adversarial accuracy measures to quantify the similarity between the real and synthetic data in terms of their descriptive statistics. Second, we evaluated the impact of the generated samples by training a predictive model on their descriptive statistics, which resulted in a significant improvement (over 1100% in TSS and 350% in HSS). Third, we used the generated time series to examine their high-dimensional contribution to mitigating the scarcity of the strong flares, which we also observed a significant improvement in terms of TSS (4%, 7%, and 31%) and HSS (75%, 35%, and 72%), compared to oversampling, undersampling, and synthetic oversampling methods, respectively. We believe our findings can open new doors toward more robust and accurate flare forecasting models.

Keywords Multivariate time series · Class imbalance · Generative adversarial network · Flare forecasting

1 Introduction

Living With a Star (LWS) is a NASA scientific program to study the Sun–Earth system and its impacts on human life and society. Dedicated to this program, NASA launched the first mission named Solar Dynamics Observatory (SDO) in February 2010. The SDO mission is an invaluable instrument for researching solar activity, which can produce damaging space weather. This space weather activity can have drastic impacts on space and air travel, power grids, GPS, and communication satellites [1]. For example, in March 1989, geomagnetically induced currents, produced when charged particles from a coronal mass ejection impacted the earth's atmosphere, caused power blackouts and direct costs of tens of millions of dollars to the electric utility Hydro-Quebec in Canada [2]. If a similar event would have happened during the summer months, it is estimated that it would likely have produced

✉ Yang Chen
ychen113@student.gsu.edu

Dustin J. Kempton
dkempton1@cs.gsu.edu

Azim Ahmadzadeh
aahmadzadeh1@cs.gsu.edu

Junzhi Wen
jwen6@student.gsu.edu

Anli Ji
aji1@student.gsu.edu

Rafal A. Angryk
angryk@cs.gsu.edu

[1] Georgia State University, Atlanta, GA 30302, USA

widespread blackouts in the northeastern USA, causing an economic impact to the tune of billions of dollars [2].

A solar flare is an event occurring in the solar corona that is characterized by a sudden orders-of-magnitude brightening in Extreme Ultra-Violet (EUV) and X-ray, and for large events, gamma-ray emissions, from a small area on the Sun, lasting from minutes to a few hours [3, 4]. Since 1974, Geostationary Operational Environmental satellites (GOES) operated by the National Oceanic and Atmospheric Administration (NOAA) are employed to automatically detect and classify X-ray flares into wavelength bands of 1-8 Ångstrom. According to the peak soft X-ray flux in this range, flares are logarithmically categorized as A, B, C, M, and X, from weaker to stronger. In a typical binary classification strategy, the most intense flares, namely the M and X classes, are identified as the positive class. In contrast, no flare occurrence and flares of A, B, and C classes are identified as the negative class.

The class-imbalance issue, an extreme imbalance between positive and negative classes, is intrinsic to many real-world machine learning tasks, including the solar flare forecasting. It is widely known that an improper handling of the class-imbalance issue can result in unrealistic and unreliable analyses, with limited practical value in operational settings [5–7]. The scarcity of the positive class(es) of flare data is an important problem that needs to be appropriately treated in the current research. Inspired by this, we make use of a deep learning-based approach to produce the synthetic multivariate time-series data using generative adversarial networks.

Many remedies have already been put forward to address the class-imbalance issue. The simplest approaches are oversampling, undersampling, and cost-sensitive learning. We recently revisited some of these approaches and their impact on a flare forecasting dataset [8]. In a more complex attempt of mitigating the class-imbalance issue, [9] demonstrates the benefit of using various statistic-based synthetic oversampling techniques compared to naïvely oversampling and undersampling. This motivates us to investigate synthetic sample-based oversampling from a new angle. More specifically, we employ a conditional generative adversarial network (CGAN) to generate realistic time-series data (as opposed to descriptive statistics derived from time series) based on real data and, therefore, to construct a balanced training dataset to facilitate the task of multivariate time-series classification. The rationale behind using CGAN is as follows: (1) we can learn the distribution of existing data and generate informative synthetic samples; (2) the algorithm can explicitly control the category of generated samples, especially to generate minority synthetic samples for mitigating the class-imbalance issue; (3) the CGAN enables us to produce time-series data rather than point-in-time data compared to the statistic-based oversampling method.

This paper extends our previous investigations of using a CGAN to produce time-series data in [10]. This paper provides additional explanations of our input dataset and associated classification problems used to evaluate our synthetic data. Additionally, we have included numerous comparisons with other approaches for remedying the class-imbalance issue. The main contributions of this paper can be categorized as follows:

- We utilize the CGAN to perform synthetic time-series data generation on a recently released flare forecasting benchmark dataset (SWAN-SF). Our experiments indicate that utilizing the synthetic time-series data we have generated may be an effective solution to the data scarcity in the solar flare forecasting problem.
- We perform model selection using two methods: (a) the Kullback–Leibler divergence metric for quantifying the similarity between the distributions of the real and synthetic data, and (b) the adversarial accuracy for monitoring the performance of CGAN directly. We show that these methods unanimously select the same models.
- We conduct three groups of comparative experiments to examine the effectiveness of the generated synthetic samples as a remedy for the class-imbalance issue on the flare forecasting task. We show that the significant improvements are present not only compared to the baseline where no augmentation is utilized, but also compared to other remedies such as under-/over-sampling, and statistical-based synthetic oversampling methods.
- We quantitatively monitor the impact of the number of synthetic samples used for balancing the data on the forecasting performance. Overall, we observe a steady boost in performance in terms of both TSS and HSS2 scores as the imbalance ratio decreases, providing further evidence that the synthetically generated time series are mimicking the real time series.

2 Related work

2.1 Class-imbalance remedies

One challenge present in many, if not all, natural hazard forecasting problems is the issue of class imbalance, and flare forecasting is no exception to this issue. Class imbalance describes a situation where the events of one or more of the class(es) in the dataset are notably less than that of the other class(es). It is common to apply special remedies to address the class imbalance if present. The

reasoning behind addressing class imbalances is that a significant imbalance in the dataset would affect any classifier by injecting a bias toward the majority classes, and machine learning models generally perform best when classes are roughly equal in size. For example, the authors [11] investigated the impact of class-imbalance issue on a solar flare benchmark dataset, namely SWAN-SF [12]. Their findings show that the classification performance is improved when typical class-imbalance remedies, such as undersampling, oversampling, and incorporating misclassification cost, are applied. Random oversampling balances the dataset by randomly selecting and duplicating samples of the minority class. Conversely, random undersampling randomly removes samples of the majority class to achieve a balance between the two classes. For multi-class problems, either of these remedies can be coupled with preserving the climatology of the subclasses in the original dataset [13]. However, these sampling-based methods can only provide limited improvements in training, with the risk of overfitting, since they do not introduce or utilize any new data. Another approach to addressing class imbalance is to produce synthetic samples based on the existing data. In [14], Synthetic Minority Oversampling Technique, namely SMOTE, was introduced to create new samples between minority instances and their nearest neighbors of the same class. In [9], three naïve random synthetic oversampling methods, namely Random Uniform Synthetic Oversampling (RUSO), Random Normal Synthetic Oversampling (RNSO), and Random NOise Synthetic Oversampling (RNOSO), were employed to generate synthetic samples of the minority classes in the flare forecasting problem. RUSO generates new samples from the uniform distribution between the minimum and maximum values of each feature of minority samples in the training set. In contrast, RNSO uses the normal distribution with the mean and standard deviation values of each feature of minority samples in the training set. RNOSO generates new samples by sampling noise terms from the normal distribution with mean of 0 and standard deviation of 0.1 and adds the noise to the existing minority samples. Although [9] demonstrates the benefit of using various statistic-based synthetic sampling methods over naïvely oversampling and undersampling, it is important to note that: (1) they do not generate time-series data (as shown in Table 1). The development of recurrent neural networks and generative modeling opens the door to generating sequential data, such as videos and music, and time series [15, 16]; (2) they work under the assumption of normality, which is not necessarily valid in many real-world applications. In our study, we know that the peak X-ray fluxes of solar flares follow a power-law distribution, and the distribution of each of the magnetic-field physical parameters of SWAN-SF is impacted by that.

2.2 Generative adversarial network (GAN)

Generative adversarial network is an emerging technique for modeling high-dimensional distributions of real samples implicitly [26]. Initially proposed in [17], the GAN learns to produce realistic data by training two components, the generator and the discriminator, in an adversarial manner. First, the generator is used to capture the data distribution by sampling random vectors from a latent space as inputs and to produce samples similar to the real data. Next, the discriminator receives both generated samples and real samples as inputs, and estimates the probability of the input coming from the real data space. By training the generator and the discriminator simultaneously, a generator is enabled to gradually generate more realistic samples under the supervision of the real samples. This process is repeated until the discriminator cannot distinguish the generated samples from the real ones. Typically, either the generator or the discriminator can be implemented by arbitrary multilayer neural networks consisting of fully connected networks, convolutional neural networks, and recurrent neural networks, depending on the actual data source.

The vanilla GAN has exhibited some limitations on the stability of the model training and the diversity of the generated sample [18]. Therefore, several works have investigated designing new architectures in order to mitigate the training issues and improve the quality of the generated samples. For example, the deep convolutional GAN (DCGAN) utilizes convolutional neural networks as the generator and discriminator and replaces pooling layers with strided convolutions (discriminator) and fractional-strided convolutions (generator) to improve the training stability [27]. The Wasserstein GAN [18] introduces the Earth-Mover distance to improve the learning stability and provide a meaningful learning curve for tuning hyperparameters. The Info GAN [19] incorporates the representation learning by maximizing the mutual information between a fixed subset of the latent variables and the observations. The variational GAN (VAEGAN) [20] combines autoencoder and GAN to encode the real data as inputs of the generator instead of randomly sampling from a latent space, enabling the GAN model to achieve faster and more stable learning. The conditional GAN (CGAN) [21] is another variant dedicated to improving the quality of the generated samples and controlling the classes of the synthetic samples by utilizing conditional information. The most common form of conditional information is the class labels. All the reviewed approaches and their key features are organized in Table 1 for convenience.

Various GAN applications have been proposed to deal with different demands, from art, science, finance, drug

Table 1 Fourteen studies related to this work

No.	Methods	Production types	Features
1	Oversampling/ undersampling	Time series	The simplest one; No new data introduced
2	Advanced oversampling [13]	Time series	Applied on multi-class problems
3	SMOTE [14]	Point-in-time	Provides statistical interpretations
4	RUSO/RNSO /RNOSO [9]	Point-in-time	Provides statistical interpretations
5	Vanilla GAN [17]	Time series	Generating single-class samples
6	WGAN [18]	Time series	Stable and faster training by using a meaningful objective function
7	InfoGAN [19]	Time series	Learns interpretable latent variables in an unsupervised manner
8	VAEGAN [20]	Time series	Stable and faster training
9	CGAN [21]	Time series	Generates multi-class samples by using conditional information; Stable and faster training
10	C-RNN-GAN [16]	Time series	Generates single-category musical data
11	RCGAN [22]	Time series	Generates privacy-free medical data
12	DoppelGANger [23]	Time series	High fidelity; Privacy-free; Deal with mix-type data
13	[24]	Time series	Learns the conditional probability distribution of features by GAN
14	TimeGAN [25]	Time series	Incorporates conditional temporal dynamics into the unsupervised GAN

The over-/under-sampling (Row 1) is the simplest one which is considered as our baseline model. Row 2 corresponds an effective method in multi-class problems that couples oversampling and undersampling by preserving the distribution of the subclasses in the original dataset. Rows 3 and 4 describe statistic-based methods. Rows 5–14 list several GAN-based methods. The type of the generated data for each method is specified in the third column

discovery to video games, and have achieved great success. In the computer vision domain, synthetic image generation has been tested in scenarios, such as cartoon characters [28], face frontal views [29], and new human poses [30]. Image-to-image translation [31, 32] and text-image synthesis [33] applications enable users to transfer objects between different styles or different formats. Moreover, image super-resolution [34, 35] and motion stabilization [36] applications are especially helpful in autonomous driving and navigation tasks since object detection accuracy is improved by utilizing optimized images or videos. The last but not least avenue of applying GANs is data augmentation. Traditional data augmentation techniques usually perform a transformation pipeline on the existing instances of data, and it involves one or more of data manipulations, to name a few, random rotation, translation, reflection, cropping, blurring, sharpening, and hue adjustment. However, these transformations are not applicable to all situations. For example, the chirality of an image of solar filament would be changed if a reflection or affine transformation is performed. GAN provides an alternative way to perform the data augmentation, that is to learn an underlying distribution of real samples and to produce new realistic samples based on the learned distribution.

2.3 Time-series generation using GAN

Various projects in different domains have emerged to shed light on generating time-series data by utilizing the generative adversarial network, as shown in Table 1. In [16], the use of a C-RNN-GAN was proposed as a method to generate musical data. This method applied a unidirectional long short-term memory (LSTM) as the generator and a bidirectional LSTM as the discriminator. In [22], RGAN was developed as a privacy-preserving method for generating synthetic medical data in an effort to mitigate the concern regarding the utilization of the privacy-sensitive patient data to train machine learning models. DoppelGANger [23] is another framework designed for generating synthetic time series data with high fidelity and sharing data with privacy-free properties. Particularly, it deals with mix-type datasets that contain continuous and discrete features. In [37], GAN was utilized as a data augmentation method for generating synthetic biosignal data, including electroencephalographic (EEG) and electrocardiography (ECG). The improved Wasserstein GAN was employed to generate synthetic spiking time series in the banking domain [38]. In [24], the authors used GAN to learn the conditional probability distribution of the key features to generate synthetic time-series data. TimeGAN [25] combined the versatility of the unsupervised GAN

approach with the control over conditional temporal dynamics. This method has two more autoencoding components, including an embedding function and a reconstruction function trained jointly with the generator and the discriminator components. This structure enables the model to iteratively learn to encode features, generate representations, and adjust weighting parameters according to the objective function.

3 SWAN-SF dataset

The data used in this project are a benchmark dataset named Space Weather ANalytics for Solar Flares (SWAN-SF) [12], which is publicly available at the Harvard Dataverse Repository [39]. SWAN-SF is a comprehensive, multivariate time-series dataset extracted from solar photospheric vector magnetograms in HMI Active Region Patch (HARP) data made available as the Space-weather HMI Active Region Patch (SHARP) series [40, 41]. The benchmark dataset has 5 classes, including four flare classes of X, M, C, and B, with an additional class labeled as NF representing the absence of any of the listed flares. Each multivariate time series is labeled by looking at the strongest flare event recorded in the 24-hour prediction window. This interval follows the 12-hour observation window from which the magnetic field parameters are calculated. In this study, we simplify the task to a binary classification by merging the stronger instances (i.e., X- and M-class flares) to form the positive class, and the weaker instances (i.e., of C, B, and NF classes) to represent the negative class. The extreme class imbalance exhibited by SWAN-SF is illustrated in Fig. 1, with each class's sample size annotated. A proper treatment of this imbalance is the objective of this study.

The SWAN-SF is made up of five temporally non-overlapping partitions covering the period from May 2010 through August 2018. Each partition contains approximately an equal number of X- and M-class flares, and there are a total of 6,234 flare records and 324,952 non-flaring records. Each flare record is a multivariate time series (hereafter MVTS) with 60 time steps, each of which has 51 magnetic field parameters. (For the definition of the parameters, see Table 1 in [12].) We limit our investigation to only four of these 51 parameters, abbreviated to TOTUSJH, ABSNJZH, SAVNCPP, and TOTBSQ, which have been listed as the most relevant to the flare forecasting in several studies including [41] and more recently in [42]. Moreover, based on how they are calculated it is easy to see that many of these parameters are highly correlated with each other and a small subset of them suffices our objective in this study.

One major concern for evaluating flare forecasting models is to determine evaluation metrics appropriate for the above-mentioned class imbalance. Many well-known performance metrics are significantly impacted by class imbalance [43], including accuracy, precision, and F1-score, which ignore the number of misclassified instances. From years of exploration, domain experts have agreed on two effective metrics, namely the true skill statistic (TSS) [44] and the updated Heidke skill score (HSS2) [45], as shown in Eqs. 1 and 2, respectively. These are functions of the confusion matrix whose entries are true positive (tp), true negative (tn), false positive (fp), and false negative (fn). We will use both of these metrics to evaluate the performance of our flare forecasting models in Sects. 6.4 and 6.5.

$$\mathrm{TSS} = \frac{\mathrm{tp}}{\mathrm{tp} + \mathrm{fn}} - \frac{\mathrm{fp}}{\mathrm{fp} + \mathrm{tn}} \tag{1}$$

$$\mathrm{HSS2} = \frac{2 \cdot ((\mathrm{tp} \cdot \mathrm{tn}) - (\mathrm{fn} \cdot \mathrm{fp}))}{(\mathrm{tp} + \mathrm{fn}) \cdot (\mathrm{fn} + \mathrm{tn}) + (\mathrm{fp} + \mathrm{tn}) \cdot (\mathrm{tp} + \mathrm{fp})} \tag{2}$$

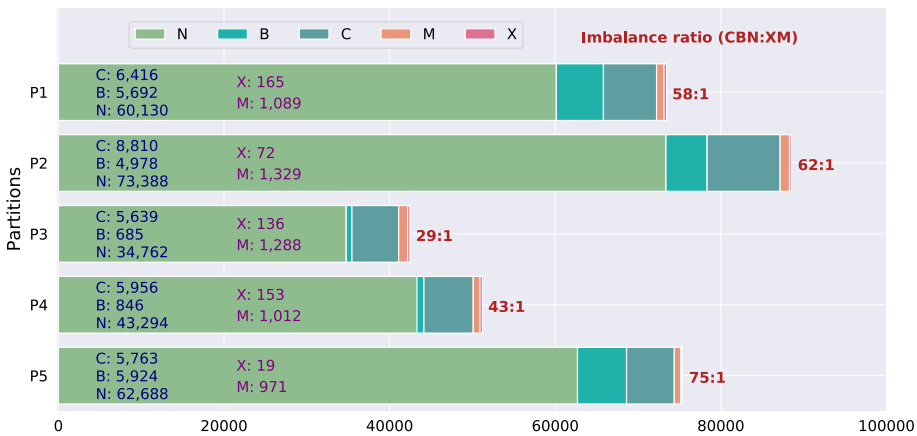

Fig. 1 The plot illustrates the distribution of the 5 flare classes in SWAN-SF dataset. The flare counts and the imbalance ratio (font in red) per partition are annotated. In this study, the flare instances of X and M classes make up the positive class, and the C, B, and N classes account for the negative class

4 Conditional GAN: a RECAP

The algorithm we employ is the conditional generative adversarial network (CGAN) whose architecture is illustrated in Fig. 2. Several reasons make us decide to utilize this algorithm: First, CGAN can control the category of generated samples, allowing us to generate samples of minority classes to mitigate the class-imbalance issue. Second, it can provide stable and faster training compared to the vanilla GAN [46]. Third, the category information of instances in the SWAN-SF dataset is available as conditional information for training CGAN models. We choose LSTM networks as the fundamental components in both the generator and the discriminator since our subject is sequential data.

As mentioned in Sect. 2.2, the ultimate goal of a generator (G) is to generate an output with similar characteristics as the real data. As seen in Fig. 2, the algorithm takes in a random input vector Z_n, which is a tensor with the shape of [*batch_size*, *sequence_length*, *latent_dim*]. In our case, the shape is [32, 60, 3] for 32 multivariate time series in a batch, each of length 60 and the latent dimension of 3. The conditional vector (C_n), as a type of auxiliary information, has the shape of [32, 60, 2] since the binary labels are encoded into a one-hot representation. By concatenating Z_n and C_n, we obtain a tensor of shape [32, 60, 5] as the final input of the generator. Note that the latent space dimension, as a hyperparameter, is determined by the dimension of the parameters and the conditional information. We empirically assume that the total dimension of the latent space and the conditional information should be similar to the dimension of the parameters being produced. Besides, the dimension of the latent space and the conditional information should be balanced, which means neither should dominate the inputs of a generator. The outputs of the generator, regarded as the generated or synthetic samples, are calculated by going through the LSTM and dense layers pipeline. The LSTM layer controls the memorizing process using a gating mechanism. Meanwhile, the dense layers guarantee that the generated samples can maintain the same shape as the real data, i.e., [32, 60, 4] where 4 stands for four magnetic field parameters mentioned in Sect. 3.

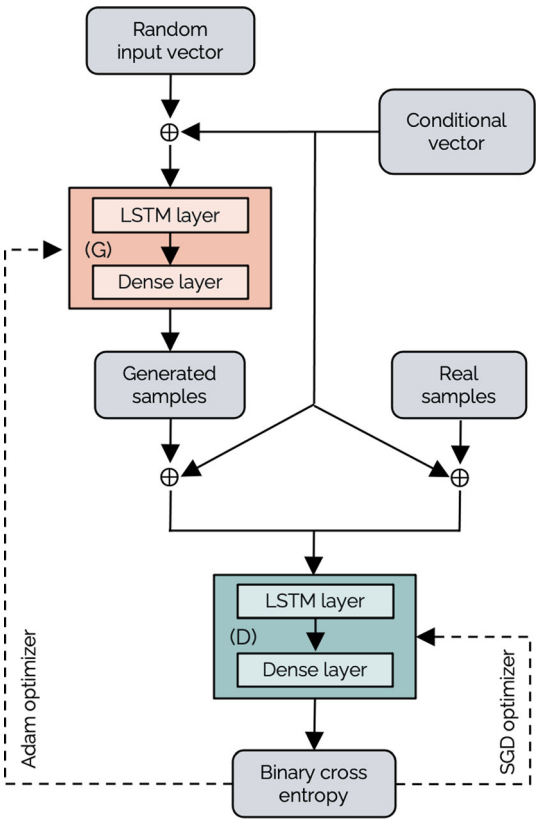

Fig. 2 This is the framework of the CGAN algorithm, including components of the generator (G) and the discriminator (D). Each component is processed by the combination of the LSTM layer and the dense layer. The inputs of the generator are random input vectors concatenated with conditional vectors. The inputs of the discriminator are either generated or real multivariate time series with conditional vectors. The binary cross-entropy is the criterion for optimizing the model

The task of a discriminator (D) is to classify inputs as either being the real or generated samples produced by the generator. As Fig. 2 illustrates, the discriminator takes both the real and the generated multivariate time-series samples as the inputs. To simplify the notation, we use $\widetilde{X_n}$ to denote either real (X_n) or synthetic samples ($G(Z_n|C_n)$) when the difference is clear from the context. By feeding C_n into D, the discriminator produces judgments about whether the sample is generated or real and evaluates if the category of the generated sample corresponds to its conditional information. Finally, the binary cross-entropy loss calculated between the predicted and the ground-truth values is used to update the weighting parameters of the generator and the discriminator using the backpropagation.

So far, we have described the structures and functionalities of the generator and the discriminator. Next, we define the objective function used for optimizing the algorithm. In our framework, the objective function is divided into two parts: the generator loss ($Loss_G$) and the discriminator loss ($Loss_D$). The discriminator loss is obtained by calculating the cross-entropy between the ground truth and the outputs of the discriminator, as shown in Eq. 3,

$$Loss_D(\widetilde{X_n}|C_n, y_n) = -\mathrm{CE}\Big(D(\widetilde{X_n}|C_n), y_n\Big) \tag{3}$$

where $\widetilde{X_n}$ is the set of inputs of the discriminator, and C_n is the conditional vector. $D(\widetilde{X_n}|C_n)$ returns the likelihood of $\widetilde{X_n}$ being a real or a generated sample, and CE stands for

the cross-entropy loss function. Note that $\widetilde{X_n}$ is composed of two different types of data sources, as formulated in Eq. 4.

$$\widetilde{X_n} = \begin{cases} X_n & \textbf{if inputs are real samples} \\ \mathrm{G}(Z_n|C_n) & \textbf{if inputs are generated samples} \end{cases} \tag{4}$$

Correspondingly, y_n takes two different values depending on the source of the sample in $\widetilde{X_n}$.

$$y_n = \begin{cases} \boldsymbol{1} & \textbf{if inputs are real samples} \\ \boldsymbol{0} & \textbf{if inputs are generated samples} \end{cases} \tag{5}$$

The generator loss (Loss_G) is also formulated in Eq. 6, where the input $G(Z_n|C_n)$ is the generated samples, and its corresponding predictions are $D\Big(G(Z_n|C_n)|C_n\Big)$. To optimize the generator, we need to guide the discriminator to classify the generated samples as real. To do so, we initialize the ground-truth labels with $\mathbf{1}s$ (same as the real samples). By minimizing $Loss_G$, the predictions of the discriminator approach $\mathbf{1}s$ gradually indicating the generated samples are realistic enough that the discriminator cannot distinguish them from the real samples.

$$\mathrm{Loss}_G(Z_n|C_n) = -\mathrm{CE}\Big(D\Big(G(Z_n|C_n)|C_n\Big), \mathbf{1}\Big) \tag{6}$$

5 Methodology

The main objective of this study is to examine the effectiveness of CGAN as a possible remedy to the class-imbalance issue on SWAN-SF. In this section, we focus on proper assessments of the contribution of CGAN-generated synthetic multivariate time series of SWAN-SF and answer whether the generated time series are reliable for machine learning use.

There are two main concerns in evaluation of GAN models and their synthetically generated data: (1) to determine the learning progress and (2) to examine the effectiveness of synthetic data for the original problem. Regarding the former, in most image-based GAN projects, researchers can determine the training progress by visually examining the synthetic images. However, the visual inspection of synthetic time series does not give us much evidence as to whether the synthetic samples are realistic or not. To address this concern, we present two statistical-based approaches to handle the model selection issue. Regarding the latter, we design multiple experiments by applying different class-imbalance remedies to tackle the flare forecasting problem. We elaborate on our methodologies in the following text.

5.1 Model selection using distributions of statistical features

To provide a statistical evaluation for our model, we compare a few descriptive statistics extracted from the real and synthetic time-series data. This establishes a high-level similarity criterion that must be satisfied if the distributions of the real and generated time series are indeed similar.

Suppose we have sets of real (T) and synthetic (S) samples, with equal number of multivariate time series. For each instance, we extract its mean, median, and standard deviation. We then construct the corresponding probability distributions P_T and P_S, with setting the bin size to M. To quantitatively measure the similarity, we calculate the Kullback–Leibler (KL) divergence [47] between distributions of P_T and P_S using Eq. 7. The KL divergence is a nonnegative measure, which means $D_{KL}(P_T||P_S) \geq 0$. The smaller value indicates the higher similarity between P_T and P_S.

$$D_{KL}(P_T||P_S) = \sum_{m \in M} P_T(m) \cdot log\Big(\frac{P_T(m)}{P_S(m)}\Big) \tag{7}$$

5.2 Model selection using adversarial accuracy

The adversarial accuracy, as formulated in Eq. 8, is put forward by Yale et al. [48], which is used for measuring the similarity of two sets of data samples through their nearest neighbors.

$$\begin{aligned} AA_{TS} = \frac{1}{2}\Big(&\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{1}(d_{TS}(i) > d_{TT}(i)) \\ &+ \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{1}(d_{ST}(i) > d_{SS}(i))\Big) \end{aligned} \tag{8}$$

$$\begin{cases} d_{TS}(i) = min_j ||X_T^i - X_S^j||_2 \\ d_{TT}(i) = min_{j,j\neq i} ||X_T^i - X_T^j||_2 \end{cases} \tag{9}$$

In Eq. 8, the subscripts T and S refer to the sets of real and synthetic samples, respectively. The distance function d is defined in Eq. 9 as the minimum (Euclidean) distance between each real sample X_T^i and all synthetic samples X_S^j (i.e., $d_{TS}(i)$), and all other real samples X_T^j (i.e., $d_{TT}(i)$). The shortest distance generally means the highest similarity between two samples. If $d_{TS}(i) > d_{TT}(i)$, it means no synthetic sample is found in S that is more similar to X_T^i than any other real samples in T. Otherwise, a synthetic sample, which is more similar to the X_T^i, can be found. A realistic sample is generated when $d_{TS}(i) < d_{TT}(i)$. The range of

adversarial accuracy is [0, 1]. The outcome 1 indicates that there is no resemblance between the set of real samples and the set of synthetic samples. The outcome 0 indicates that the two sets are exactly the same, yielding no new information. The desirable outcome of adversarial accuracy is close to 0.5, implying that the real and synthetic samples generated by the generators are indistinguishable [48].

5.3 Synthetic data v.s. over-/under-sampling

To assess the effectiveness of the synthetic data, we design several experiments where we compare the impact of different balancing remedies on the classification of flaring and non-flaring instances of SWAN-SF, with that of balancing using our synthetic data. As shown in Table 2, we set up three groups of experiments, namely A, B, and C, and each comprises four experiments. For A and B, the primary difference between them is that in the former we utilize the last-value statistic of MVTS samples, whereas in the latter, median and standard deviation of time series are used. The last-value is literally the last value of each time series. This makes our results comparable with those in [9] where point-in-time data were used. The mean statistic is sensitive to outliers, which makes us eliminate it for flare forecasting. For C, we aim to examine the effectiveness of synthetic samples in their original high-dimensional format, i.e., time series. The question is whether the unwanted noise of the synthetic MVTS samples was obscured by the summary descriptive statistics. Therefore, we conduct the experiments in C to verify the hypothesis.

For each group, we train four classifiers with the same parameter setting, but with different training datasets. The models in A1, B1, and C1 are trained on the highly imbalanced, real dataset without any changes. The models in A2, B2, and C2 are trained on the dataset that is made balanced by adding synthetic minority (flaring) samples. The models in A3, B3, and C3 are trained on the dataset that is made balanced by random oversampling of (i.e., duplicating) the minority instances. Lastly, the models in A4, B4, and C4 are trained on the dataset that is made balanced by random undersampling of the majority (non-flaring) instances. The models in A1, B1, and C1 are considered as the baseline.

Table 2 All experiments carried out to examine various class-imbalance remedies

Group	No.	Method	Description	Statistic
A	A1	Baseline (BL)	No data augmentation applied on P1	Last value
	A2	Synthetic Oversampling using CGAN (CGAN)	Adding synthetic flaring samples to the minority class of P1	
	A3	Random oversampling (RO)	Randomly oversampling samples of the minority class on P1	
	A4	Random undersampling (RU)	Randomly undersampling samples of the majority class on P1	
B	B1	Baseline (BL)	No data augmentation applied on P1.	Median & standard deviation
	B2	Synthetic oversampling using CGAN (CGAN)	Adding synthetic flaring samples to the minority class of P1	
	B3	Random oversampling (RO)	Randomly oversampling samples of the minority class on P1	
	B4	Random undersampling (RU)	Randomly undersampling samples of the majority class on P1	
C	C1	Baseline (BL)	No data augmentation applied on P1	Time series
	C2	Synthetic oversampling using CGAN (CGAN)	Adding synthetic flaring samples to the minority class of P1	
	C3	Random oversampling (RO)	Randomly oversampling samples of the minority class on P1	
	C4	Random undersampling (RU)	Randomly undersampling samples of the majority class on P1	

Groups of A and B have experimented on the extracted descriptive statistics of MVTS data. Group A utilizes the last value statistic of MVTS samples as inputs, whereas in group B, median and standard deviation of samples are used. All experiments in A and B utilize Partition 1 (P1) as the training set and Partitions 2, 3, and 5 as the test sets. Partition 4 is not involved in this experiment. The experiments in C are conducted to examine various class-imbalance remedies by taking time series as inputs. Similarly, Partition 1 is utilized as the training set and Partitions 2, 3, and 5 as the test sets. Partition 4 is reserved for validation of the hyperparameters

6 Experiments and results

In this section, we conduct experiments to evaluate the effectiveness of the proposed assessment methods. First, we show the results of two model selection methods based on the distributions of statistical features and the adversarial accuracy. Then, we present multiple experiments by applying different class-imbalance remedies to tackle the flare forecasting problem. Furthermore, we exhibit a quantitative analysis of the quality and usefulness of the synthetic flaring time series generated by the CGAN model to balance the training dataset.

6.1 Experimental settings

After exploring various settings based on the defined objective function, we found that using the Adam optimizer for the generator and the gradient descent optimizer for the discriminator produced optimal results. We tune the performance of CGAN model by setting different hyperparameters, i.e., latent space dimensions: $\{2, 3, 4, 5\}$, learning rates: $\{0.5, 0.1, 0.01, 0.001, 0.0001\}$, batch sizes: $\{16, 32, 64\}$. Empirically, we concluded our optimal hyperparameter setting with the latent space dimension of 3, the conditional information dimension of 2 (since we have two classes), the learning rate of 0.1, the batch size of 32, and the LSTM hidden size of 100. The model was trained with 300 epochs, and intermediate models were saved at every five epochs. We have implemented CGAN using the TensorFlow 2.1 library [49].

For preprocessing of SWAN-SF, we linearly transformed all five partitions to the range $[-1, 1]$ for training the CGAN model and evaluations. We train the generator on partition 1 of SWAN-SF, with the four magnetic field parameters mentioned in Sect. 3.

We employ the support vector machine (SVM) as the standard classifier for experiment groups A and B in Sect. 6.4. The models are trained on Partition 1 and evaluated on Partitions 2, 3, and 5. Partition 4 is not involved in this experiment. For the experiments in A, we use the same hyperparameters as were used in [9], i.e., kernel, C and gamma set to 'rbf', 0.5 and 8, respectively. Since the input of the experiments in group B has double dimensions compared to A (from 4 to 8), we adjust the hyperparameters accordingly following the instructions in [50–52], and set the kernel, C and gamma to 'rbf', 0.25 and 0.25, respectively. We conduct the time-series-based classification experiments (group C in Sect. 6.4) using the time-series-specific support vector classifier (T-SVC). Similarly, Partition 1 is for training and Partitions 2, 3, and 5 are for evaluation. Partition 4 is reserved for validation of the hyperparameters. We performed a grid search on C and gamma to find the optimal setting, i.e., C: {0.001, 0.01, 0.1, 1, 10, 100}, gamma: {0.001, 0.01, 0.1, 1, 10, 100}, using the 'rbf' kernel. We conclude the optimal setting with C and gamma to 0.01 and 0.01.

6.2 Evaluation using distributions of statistical features

We have conducted this analysis on all of the four selected physical parameters, but for brevity, we present only the results of TOTUSJH. In Fig.3, we compare the results by monitoring the improvement of the models at every 5 epochs, and the quality of the samples they generate. Specifically, we utilize 1254 real flare samples in Partition 1 of SWAN-SF and 1254 synthetic samples generated by the CGAN model in the evaluation. The columns A and B in Fig. 3 compare the distributions of the three descriptive statistics of the real and synthetic time series based on two intermediate models saved in the training process. Column A corresponds to a model trained after 50 epochs, whereas B shows the results after 250 epochs. Comparing A with B, it is evident that, at least in terms of the three descriptive statistics, the generator gradually learns to generate synthetic time series, which are more and more similar to the real flaring time series. To draw a more comprehensive picture, we calculate the Kullback–Leibler (KL) divergence between the distributions of three descriptive statistics of the real and the synthetic time series every 50 epochs. We observe, as shown in the column C of Fig. 3, that the KL divergence decreases as training progresses. We found that on average, the models between the epochs 201–250 achieve the best performance, with lower KL divergence for the mean, median, and standard deviation distributions. We also see that the variance between the results produced by intermediate models trends downward until we surpass the 250 epoch mark. We further need to examine the overfitting issue. That is, the KL divergence can be low if the CGAN model just memorizes the training set, resulting in no or limited new information produced. We assess this question in the next section.

6.3 Evaluation using adversarial accuracy

The evaluation of our intermediate models using adversarial accuracy is illustrated in Fig. 4, for the physical parameter TOTUSJH, as an example. We again utilize 1254 real flare samples in Partition 1 of SWAN-SF and 1254 synthetic samples generated by the CGAN model. As the box plots suggest, the models between 201 to 250 epochs achieve the adversarial accuracy of 0.55, 0.60, and 0.68, in terms of mean, median, and standard deviation of

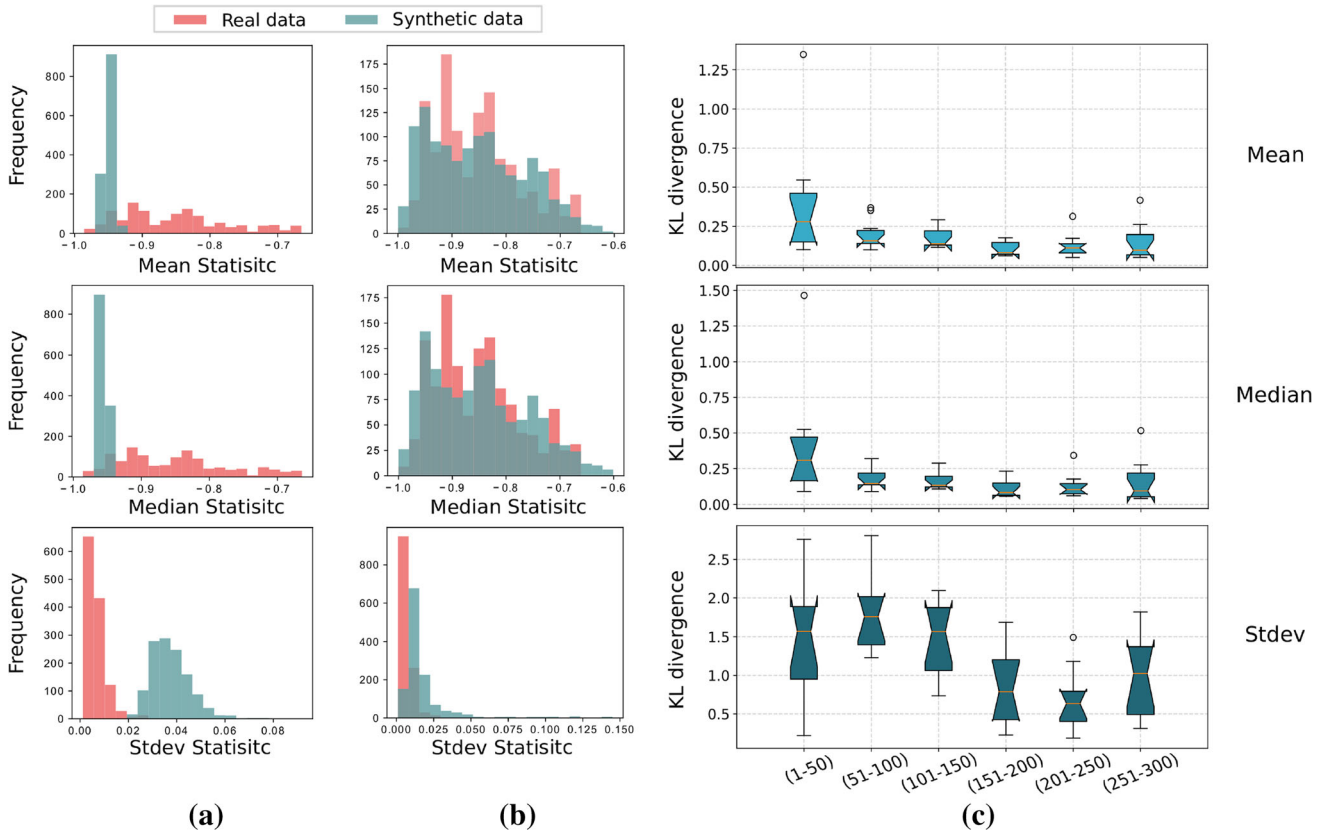

Fig. 3 The plots show the distributions of mean, median, and standard deviation of the physical parameter TOTUSJH and its synthetic counterpart using 20 equal-width bins. Columns A and B show the distributions of the descriptive statistics at two intermediate epochs, 50th and 250th, respectively. Column C shows the distributions of KL divergence scores calculated by comparing distributions of synthetic samples and real samples across all intermediate models divided into six groups

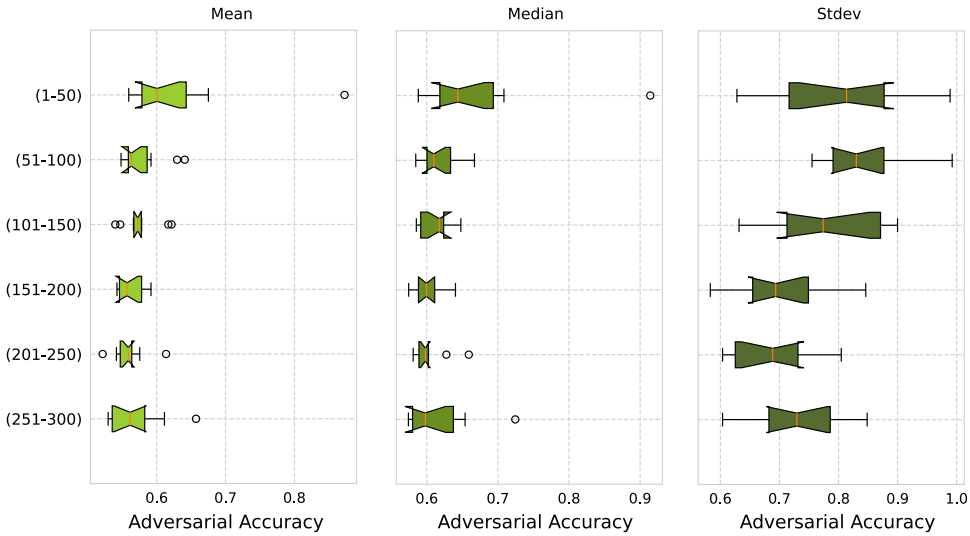

Fig. 4 The box plots show the distributions of adversarial accuracy of the three descriptive statistics of TOTUSJH, namely mean, median, and standard deviation, evaluated with all intermediate models divided into six groups

the generated time series, respectively. This shows that the CGAN model can generate realistic synthetic samples by maintaining a good balance between underfitting and overfitting. Moreover, the adversarial accuracy results are consistent with our evaluation using KL divergence.

6.4 Examining descriptive statistics of synthetic time series

We conducted two groups of flare forecasting-based experiments (A&B) to examine the effectiveness of the

synthetic data using descriptive statistics. Four classifiers are trained for each group, with the same parameter setting but different training datasets. In A2 and B2, we generate 70, 984 synthetic flare samples to balance the training set. For A3 and B3, the training dataset is made balanced by random oversampling 70, 984 duplicates of the minority instances. For A4 and B4, the training dataset is made balanced by random undersampling 1254 of the majority instances. Of course, data manipulation is only served for the purpose of training, and test sets are made entirely of real data.

The results of the group A experiments are illustrated in Fig. 5. Comparing A1 and A2, it is evident that the performance of SVM trained on the synthetically balanced data is significantly higher than that of the baseline classifier, by both metrics, TSS and HSS2. This observation confirms that the model generally performs best when classes in the training dataset are roughly equal in size. Specifically, the CGAN classifier results in a fivefold improvement compared to the baseline experiment in terms of TSS (an increase from 0.11 to 0.76). The HSS2 shows an over onefold improvement (from 0.18 to 0.39). The HSS2 improvement in A2 compared to A3 and A4 is also significant; from 0.19 to 0.39. TSS, however, remains roughly stagnant in these cases, which is simply due to the difference in what the two metrics measure. It is crucial to note that while balancing the data seems to be the main reason for the significant improvement in performance from A1 to A2, it would not have happened by balancing the dataset with unrealistic flaring instances. This is the main takeaway from our synthetically generated samples that we are evaluating through A2 experiment.

Furthermore, compared to the statistic-based oversampling methods purposed in [9], the CGAN-based method achieves a significant improvement in terms of TSS while maintaining HSS2 at its highest value, i.e., 0.39. Overall, the experiment results show that our method can produce a better flare forecasting performance than the random sampling-based methods or the statistic-based oversampling methods.

Next, we examine the forecasting performance of the group B experiments, as shown in Fig. 6. In these experiments, we observe that B2 achieves the highest TSS and HSS2. The result shows that the CGAN model can successfully learn the median and standard deviation of real multivariate time-series samples.

Putting together the results shown in Figs. 5 and 6, we demonstrated that our method has multiple advantages compared to other remedies. First, comparing to the random oversampling method (A3 and B3), the CGAN-based method can bring new information through generating realistic synthetic samples instead of duplicating existing samples. Second, comparing to the random undersampling strategy (A4 and B4), the CGAN-based approach can produce unlimited synthetic samples. Thus, more data provide a path toward training more powerful machine learning models. This significantly benefits flare forecasting models based on deep neural networks. Third, comparing to the statistic-based oversampling methods (RUSO, RNSO, and RNOSO), the CGAN-based method can learn the descriptive statistics of the real MVTS samples and, therefore, generate realistic samples. All in all, we can so far conclude that CGAN algorithm can be used to remedy the imbalance issue of MVTS flare datasets. What we have not yet examined, however, is the temporal characteristics of the synthetic time series, and whether they are realistic beyond their median and standard deviation summaries. Next, we put this question to the test.

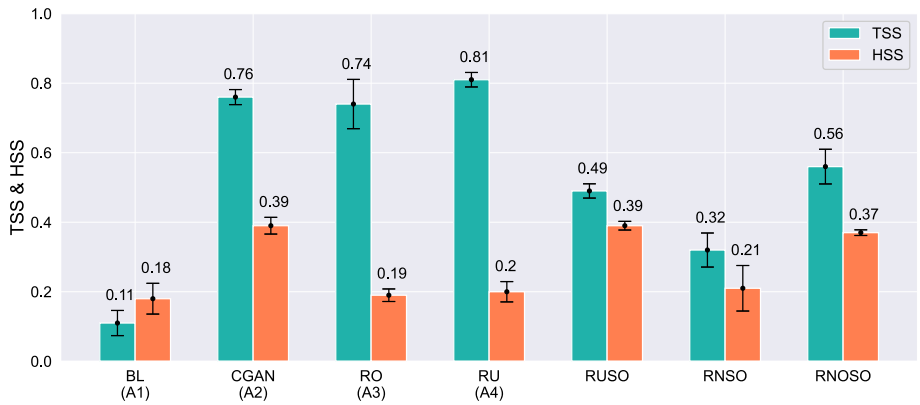

Fig. 5 The bar plot compares CGAN's synthetically generated data (A2) with the other group A experiments listed in Table 2. The choice of the last-value statistic in A makes our results comparable with the naïve random synthetic oversampling methods of RUSO, RNSO, and RNOSO purposed in [9]. The reported TSS and HSS2 values are averaged over three separate evaluation trials on Partitions 2, 3, and 5 of SWAN-SF. Partition 4 is not involved in this experiment. Error bars show the standard deviation of the obtained TSS/HSS2 values

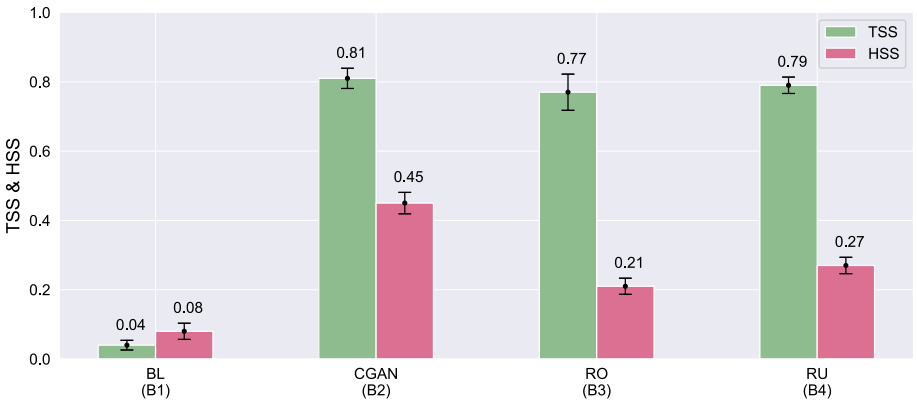

Fig. 6 The bar plot compares CGAN's synthetically generated data (B2) with the other group A experiments listed in Table 2. The reported TSS and HSS2 values are averaged over three separate evaluation trials on Partitions 2, 3, and 5 of SWAN-SF. Partition 4 is not involved in this experiment. Error bars show the standard deviation of the obtained TSS/HSS2 values

6.5 Examining synthetic time series v.s. over-/under-sampling

In this section, we examine the effectiveness of synthetic samples in time-series format. For the experiments in group C, we use the same setting of training datasets with groups A and B mentioned in Sect. 6.4. The forecasting results of experiments in group C are reported in Fig. 7. We observe that the model in C2 trained on the dataset balanced with the synthetic samples beats the models trained in C1 and C3, in terms of both TSS and HSS2 scores. The experiment C2 shows a 31% improvement in terms of TSS comparing to the model trained in C4. Although the model in experiment C2 does not obtain the highest HSS2 score, it still gives a comparable performance. The experimental result validates our assumption that adding informative synthetic samples to balance the training dataset can result in a more robust forecasting model.

6.6 Examining incremental incorporation of synthetic time series

To further demonstrate the effectiveness of the synthetic multivariate time series, we conduct another experiment to show how varying the number of incorporated synthetic samples affects the forecasting performance. More specifically, we fix the number of real flaring and non-flaring samples in the training dataset, and gradually add synthetic flaring samples while monitoring the model's performance on the test set. As illustrated in Fig. 8, we conduct ten experiments by varying the imbalance ratios of the training dataset from 1:58 to 1:1. The ratio of 1:58 is the original imbalance ratio of Partition 1, including 1254 real flares and 72,238 non-flaring samples.

Through observing the result, we can see that the performance generally increases as we reduce the imbalance ratio using our synthetic MVTS data. While the strict increase of TSS values indicates that the incorporated synthetic time series are of high quality (when compared with the real time series), we notice that the HSS2 values slightly decline at the very end. Familiar with the different behavior of these two metrics, we believe this is caused due to the lack of a per-experiment hyperparameter tuning. In other words, the added synthetic time series eventually made the default hyperparameters ineffective and consequently the model suboptimal. This change seems to have been overlooked by TSS, but not by HSS2, which is the main reason for using them as a couple. Overall, the results show that the trained CGAN model can indeed generate realistic multivariate time-series samples.

We would like to recapitulate that our main objective is to show the effectiveness of CGAN as a possible remedy to

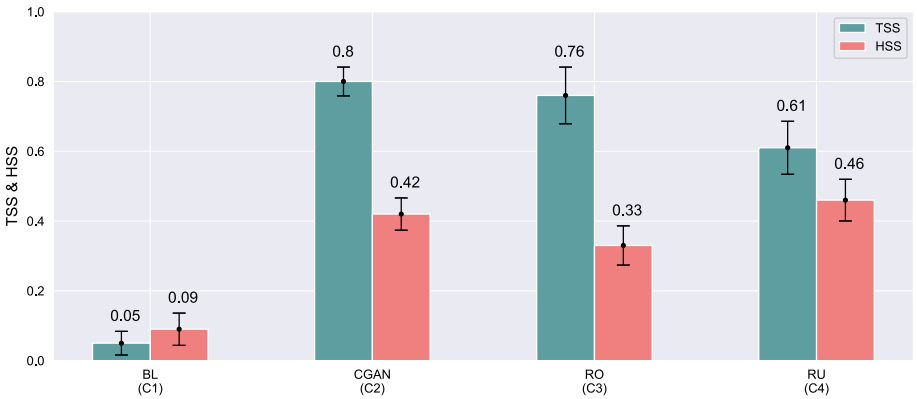

Fig. 7 The bar plot compares CGAN's synthetically generated data (C2) with the other group C experiments listed in Table 2. The reported TSS and HSS2 values are averaged over three separate evaluation trials on Partitions 2, 3, and 5 of SWAN-SF. Partition 4 is reserved for validation of the hyperparameters. Error bars show the standard deviation of the obtained TSS/HSS2 values

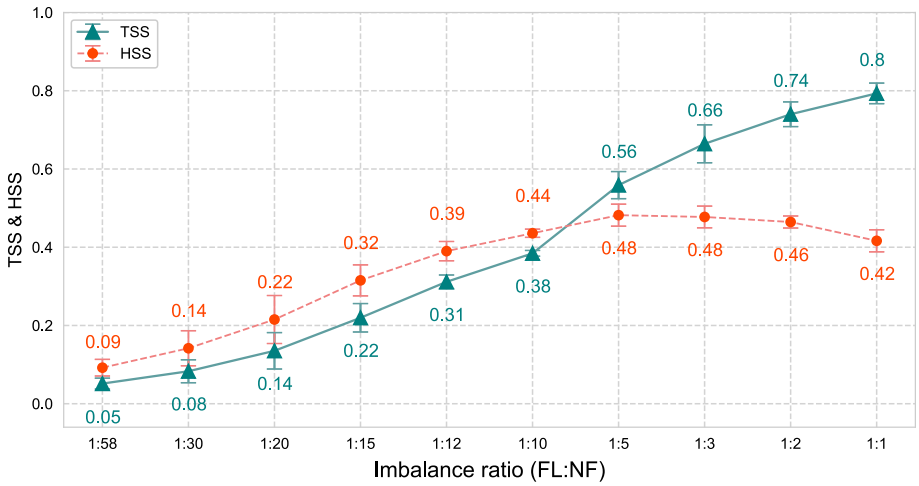

Fig. 8 The plot illustrates the gradual impact of reducing the imbalance ratio of the training set on performance, by incrementally adding synthetic flaring samples. The reported TSS and HSS2 values are averaged over three separate evaluation trials on Partitions 2, 3, and 5 of SWAN-SF. Partition 4 is reserved for validation of the hyperparameters. Error bars show the standard deviation of the obtained TSS/HSS2 values

the class-imbalance issue on SWAN-SF. Therefore, we do not claim the superiority of this approach over any other existing methods, nor do we infer that our findings can be extended to any other multivariate time-series datasets. To this end, we did not include multiple datasets, and we did not compare the performance of CGAN with other GAN-based algorithms. Instead, we kept our focus on evaluating the contribution of CGAN-generated synthetic MVTS of SWAN-SF, and the reliableness of the generated time series for machine learning use.

7 Conclusion & future work

In this project, we utilized the conditional generative adversarial network (CGAN) to perform data-informed augmentation of multivariate time series (MVTS) on a recently released flare forecasting benchmark dataset (SWAN-SF). We tailor several verification methods to show that the generated MVTS samples indeed preserve the distribution of the real physical parameters: (1) we utilize Kullback–Leibler divergence metric to quantify the similarity between the distributions of the real and synthetic data; (2) we use adversarial accuracy to monitor the performance of CGAN directly; (3) we use the synthetic MVTS samples to balance our dataset and compare the classification performance with that trained on the original data and that on the dataset that was balanced by other oversampling, undersampling, and statistic-based synthetic oversampling methods such as RUSO, RNSO, and RNOSO. The results showed that the CGAN-based approach can remarkably boost flare forecasting performance in terms of TSS and HSS2. Therefore, we consider that the CGAN method is an effective remedy for mitigating the class-imbalance issue in flare forecasting.

The CGAN-based approach provides a preliminary attempt to generate meaningful synthetic physical features. There are still many model-related aspects that can be improved further such as incorporating an advanced loss function of Wasserstein GAN, or exploring more complex structures or more layers of the generator and the discriminator. Next, we plan to try deep learning models with flare forecasting, such as convolutional neural networks or recurrent neural networks. In the future, we also wish to investigate how to explore and interpret the meaning of synthetic samples from the astrophysics point of view by collaborating with domain experts.

Funding This project has been supported in part by funding from the Division of Advanced Cyberinfrastructure within the Directorate for Computer and Information Science and Engineering, the Division of Atmospheric & Geospace Sciences within the Directorate for Geosciences, under NSF awards #193155 and # 1936361.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Council NR (2008) Severe space weather events-understanding societal and economic impacts: a workshop report. The National Academies Press, Washington, DC
2. Boteler DH (2003) Geomagnetic hazards to conducting networks. Natl Hazards 28(2):537–561
3. Benz AO (2008) Flare observations. Living Rev Sol Phys https://doi.org/10.12942/lrsp-2008-1
4. Martens PC, Angryk RA (2017) Data handling and assimilation for solar event prediction. In: Proceedings of the international astronomical union, 13(S335), pp 344–347. https://doi.org/10.1017/S1743921318000510
5. Kubat M, Matwin S et al (1997) Addressing the curse of imbalanced training sets: one-sided selection. In: Icml, vol 97. ICML, pp 179–186

6. Mazurowski MA, Habas PA, Zurada JM, Lo JY, Baker JA, Tourassi GD (2008) Training neural network classifiers for medical decision making: the effects of imbalanced datasets on classification performance. Neural Netw 21(2–3):427–436
7. Krawczyk B (2016) Learning from imbalanced data: open challenges and future directions. Progress Artif Intell 5(4):221–232
8. Ahmadzadeh A, Aydin B, Kempton DJ, Hostetter M, Angryk RA, Georgoulis MK, Mahajan SS (2019) Rare-event time series prediction: a case study of solar flare forecasting. In: 2019 18th IEEE international conference on machine learning and applications (ICMLA), pp 1814–1820. https://doi.org/10.1109/ICMLA.2019.00293
9. Hostetter M, Angryk RA (2020) First steps toward synthetic sample generation for machine learning based flare forecasting. In: Wu X, Jermaine C, Xiong L, Hu X, Kotevska O, Lu S, Xu W, Aluru S, Zhai C, Al-Masri E, Chen Z, Saltz J (eds) IEEE international conference on big data, big data 2020, Atlanta, GA, USA, December 10–13, 2020, IEEE, pp. 4208–4217. https://doi.org/10.1109/BigData50022.2020.9377986
10. Chen Y, Kempton DJ, Ahmadzadeh A, Angryk RA (2021) Towards synthetic multivariate time series generation for flare forecasting. Cham, pp 296–307. https://doi.org/10.1007/978-3-030-87986-0_26
11. Ahmadzadeh A, Aydin B, Georgoulis MK, Kempton DJ, Mahajan SS, Angryk RA (2021) How to train your flare prediction model: revisiting robust sampling of rare events. Astrophys J Suppl Ser, 254(2), p 23. https://doi.org/10.3847/1538-4365/abec88
12. Angryk RA, Martens PC, Aydin B, Kempton D, Mahajan SS, Basodi S, Ahmadzadeh A, Cai X, Boubrahimi SF, Hamdi SM, Schuh MA, Georgoulis MK (2020) Multivariate time series dataset for space weather data analytics. Sci Data, https://doi.org/10.1038/s41597-020-0548-x
13. Ahmadzadeh A, Hostetter M, Aydin B, Georgoulis MK, Kempton DJ, Mahajan SS, Angryk R (2019) Challenges with extreme class-imbalance and temporal coherence: A study on solar flare data. In: 2019 IEEE international conference on big data (Big Data), pp 1423–1431. https://doi.org/10.1109/BigData47090.2019.9006505
14. Chawla N, Bowyer K, Hall L, Kegelmeyer WP (2002) Smote: synthetic minority over-sampling technique. J Artif Intell Res 16:321–357. https://doi.org/10.1613/jair.953
15. Chan C, Ginosar S, Zhou T, Efros A (2019) Everybody dance now. In: 2019 IEEE/CVF international conference on computer vision (ICCV). IEEE. https://doi.org/10.1109/iccv.2019.00603
16. Mogren O (2016) C-rnn-gan: a continuous recurrent neural network with adversarial training. In: Constructive machine learning workshop (CML) at NIPS 2016, p 1
17. Goodfellow IJ, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y (2014) Generative adversarial nets. In: Proceedings of the 27th international conference on neural information processing systems - Volume 2, ser. NIPS'14. Cambridge, MA, USA: MIT Press, pp 2672-2680. https://doi.org/10.5555/2969033.2969125
18. Arjovsky M, Chintala S, Bottou L (2017) Wasserstein generative adversarial networks. In: Proceedings of the 34th international conference on machine learning - Volume 70. JMLR.org, pp 214–223. https://dl.acm.org/doi/10.5555/3305381.3305404
19. Chen X, Duan Y, Houthooft R, Schulman J, Sutskever I, Abbeel P (2016) Infogan: interpretable representation learning by information maximizing generative adversarial nets. In: Proceedings of the 30th international conference on neural information processing systems, ser. NIPS'16. Red Hook, NY, USA: Curran Associates Inc., p. 2180–2188. https://doi.org/10.5555/3157096.3157340
20. Larsen ABL, Sønderby SK, Larochelle H, Winther O (2016) Autoencoding beyond pixels using a learned similarity metric. In: Proceedings of the 33rd international conference on international conference on machine learning - Volume 48, ser. ICML'16. JMLR.org, pp 1558–1566. https://doi.org/10.5555/3045390.3045555
21. Mirza M, Osindero S (2014) Conditional generative adversarial nets. http://arxiv.org/abs/1411.1784
22. Esteban C, Hyland SL, Rätsch G (2017) Real-valued (medical) time series generation with recurrent conditional gans. arXiv:1706.02633
23. Lin Z, Jain A, Wang C, Fanti G, Sekar V (2020) Using gans for sharing networked time series data: challenges, initial promise, and open questions. In: Proceedings of the ACM internet measurement conference, ser. IMC '20. New York, NY, USA: Association for Computing Machinery, pp 464-483. https://doi.org/10.1145/3419394.3423643
24. Zhang C, Kuppannagari SR, Kannan R, Prasanna VK (2018) Generative adversarial network for synthetic time series data generation in smart grids. In: 2018 IEEE international conference on communications, control, and computing technologies for smart grids (SmartGridComm), pp 1–6. https://doi.org/10.1109/SmartGridComm.2018.8587464
25. Yoon J, Jarrett D, van der Schaar M (2019) Time-series generative adversarial networks. In: Advances in neural information processing systems, pp 5508–5518
26. Creswell A, White T, Dumoulin V, Arulkumaran K, Sengupta B, Bharath AA (2018) Generative adversarial networks: an overview. IEEE Signal Process Mag 35(1):53–65. https://doi.org/10.1109/MSP.2017.2765202
27. Radford A, Metz L, Chintala S (2016) Unsupervised representation learning with deep convolutional generative adversarial networks. CoRR, arXiv:abs/1511.06434
28. Jin Y, Zhang J, Li M, Tian Y, Zhu H, Fang Z (2017) Towards the automatic anime characters creation with generative adversarial networks. arXiv:1708.05509
29. Huang R, Zhang S, Li T, He R (2017) Beyond face rotation: Global and local perception gan for photorealistic and identity preserving frontal view synthesis. In: 2017 IEEE international conference on computer vision (ICCV), pp 2458–2467. https://doi.org/10.1109/ICCV.2017.267
30. Ma L, Jia X, Sun Q, Schiele B, Tuytelaars T, Van Gool L (2017) Pose guided person image generation. In: Proceedings of the 31st international conference on neural information processing systems, ser. NIPS'17. Red Hook, NY, USA: Curran Associates Inc., pp 405–415. https://doi.org/10.5555/3294771.3294810
31. Isola P, Zhu J-Y, Zhou T, Efros AA (2017) Image-to-image translation with conditional adversarial networks. In: 2017 IEEE conference on computer vision and pattern recognition (CVPR), pp 5967–5976. https://doi.org/10.1109/CVPR.2017.632
32. Zhu J-Y, Park T, Isola P, Efros AA (2017) Unpaired image-to-image translation using cycle-consistent adversarial networks. In: 2017 IEEE international conference on computer vision (ICCV), pp 2242–2251. https://doi.org/10.1109/ICCV.2017.244
33. Zhang H, Xu T, Li H, Zhang S, Wang X, Huang X, Metaxas DN (2017) Stackgan: text to photo-realistic image synthesis with stacked generative adversarial networks. In: 2017 IEEE international conference on computer vision (ICCV), pp 5908–5916. https://doi.org/10.1109/ICCV.2017.629
34. Sønderby C, Caballero J, Theis L, Shi W, Huszár F (2017) Amortised map inference for image super-resolution. In: International conference on learning representations. https://arxiv.org/abs/1610.04490
35. Ledig C, Theis L, Huszár F, Caballero J, Aitken AP, Tejani A, Totz J, Wang Z, Shi W (2017) Photo-realistic single image super-resolution using a generative adversarial network. In: 2017 IEEE

conference on computer vision and pattern recognition (CVPR), pp 105–114. https://doi.org/10.1109/CVPR.2017.19
36. Kupyn O, Budzan V, Mykhailych M, Mishkin D, Matas J (2018) Deblurgan: blind motion deblurring using conditional adversarial networks. In: 2018 IEEE/CVF conference on computer vision and pattern recognition, pp 8183–8192. https://doi.org/10.1109/CVPR.2018.00854
37. Haradal S, Hayashi H, Uchida S (2018) Biosignal data augmentation based on generative adversarial networks. In: 2018 40th annual international conference of the IEEE engineering in medicine and biology society (EMBC), pp 368–371. https://doi.org/10.1109/EMBC.2018.8512396
38. Simonetto L (2018) Generating spiking time series with generative adversarial networks : an application on banking transactions
39. Angryk R, Martens P, Aydin B, Kempton D, Mahajan S, Basodi S, Ahmadzadeh A, Cai X, Filali Boubrahimi S, Hamdi SM, Schuh M, Georgoulis M (2020) SWAN-SF. https://doi.org/10.7910/DVN/EBCFKM
40. Hoeksema JT, Liu Y, Hayashi K, Sun X, Schou J, Couvidat S, Norton A, Bobra M, Centeno R, Leka KD, Barnes G, Turmon M (2014) The helioseismic and magnetic imager (HMI) vector magnetic field pipeline: overview and performance. Sol Phys 289(9):3483–3530. https://doi.org/10.1007/s11207-014-0516-8
41. Bobra MG, Sun X, Hoeksema JT, Turmon M, Liu Y, Hayashi K, Barnes G, Leka K (2014) The helioseismic and magnetic imager (hmi) vector magnetic field pipeline: Sharps-space-weather hmi active region patches. Solar Phys 289(9):3549–3578. https://doi.org/10.1007/s11207-014-0529-3
42. Yeoleka A, Patel S, Talla S, Puthucode K. R, Ahmadzadeh A, Sadykov VM, Angryk RA (2021) Feature selection on a flare forecasting testbed: a comparative study of 24 methods. arXiv:2109.14770
43. Hossin M, Sulaiman M (2015) A review on evaluation metrics for data classification evaluations. Int J Data Min Knowl Manag Process 5(2):1
44. Hanssen A, Kuipers W (1965) On the relationship between the frequency of rain and various meteorological parameters: (with reference to the problem ob objective forecasting), ser. Koninkl. Nederlands Meterologisch Institut. Mededelingen en Verhandelingen. Staatsdrukkerij- en Uitgeverijbedrijf . https://books.google.com/books?id=nTZ8OgAACAAJ
45. Balch CC (2008) Updated verification of the space weather prediction center's solar energetic particle prediction model. Space Wea Int J Res Appl, 6(1). https://doi.org/10.1029/2007SW000337
46. Brownlee J (2019) Generative adversarial networks with python: deep learning generative models for image synthesis and image translation. Mach Learn Mastery. https://books.google.com/books?id=YBimDwAAQBAJ
47. Kullback S, Leibler RA (1951) On information and sufficiency. Ann Math Stat 22(1):79–86. https://doi.org/10.1214/aoms/1177729694
48. Yale A, Dash S, Dutta R, Guyon I, Pavao A, Bennett KP (2019) Privacy preserving synthetic health data. F1000Research, https://doi.org/10.7490/f1000research.1116780.1
49. Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, Corrado G. S, Davis A, Dean J, Devin M, Ghemawat S, Goodfellow I, Harp A, Irving G, Isard M, Jia Y, Jozefowicz R, Kaiser L, Kudlur M, Levenberg J, Mané D, Monga R, Moore S, Murray D, Olah C, Schuster M, Shlens J, Steiner B, Sutskever I, Talwar K, Tucker P, Vanhoucke V, Vasudevan V, Viégas F, Vinyals O, Warden P, Wattenberg M, Wicke M, Yu Y, Zheng X (2015) TensorFlow: Large-scale machine learning on heterogeneous systems. software available from tensorflow.org. [Online]. http://tensorflow.org/
50. Buitinck L, Louppe G, Blondel M, Pedregosa F, Mueller A, Grisel O, Niculae V, Prettenhofer P, Gramfort A, Grobler J, Layton R, VanderPlas J, Joly A, Holt B, Varoquaux G (2013) API design for machine learning software: experiences from the scikit-learn project. In: ECML PKDD workshop: languages for data mining and machine learning, pp 108–122
51. Hsu C-W, Chang C-C, Lin C-J et al (2003) A practical guide to support vector classification'
52. Ben-Hur A, Weston J (2010) A user's guide to support vector machines. Methods Mol Biol 609:223–39. https://doi.org/10.1007/978-1-60327-241-4_13

Terms and Conditions

Springer Nature journal content, brought to you courtesy of Springer Nature Customer Service Center GmbH (“Springer Nature”).
Springer Nature supports a reasonable amount of sharing of research papers by authors, subscribers and authorised users (“Users”), for small-scale personal, non-commercial use provided that all copyright, trade and service marks and other proprietary notices are maintained. By accessing, sharing, receiving or otherwise using the Springer Nature journal content you agree to these terms of use (“Terms”). For these purposes, Springer Nature considers academic use (by researchers and students) to be non-commercial.
These Terms are supplementary and will apply in addition to any applicable website terms and conditions, a relevant site licence or a personal subscription. These Terms will prevail over any conflict or ambiguity with regards to the relevant terms, a site licence or a personal subscription (to the extent of the conflict or ambiguity only). For Creative Commons-licensed articles, the terms of the Creative Commons license used will apply.
We collect and use personal data to provide access to the Springer Nature journal content. We may also use these personal data internally within ResearchGate and Springer Nature and as agreed share it, in an anonymised way, for purposes of tracking, analysis and reporting. We will not otherwise disclose your personal data outside the ResearchGate or the Springer Nature group of companies unless we have your permission as detailed in the Privacy Policy.
While Users may use the Springer Nature journal content for small scale, personal non-commercial use, it is important to note that Users may not:

1. use such content for the purpose of providing other users with access on a regular or large scale basis or as a means to circumvent access control;
2. use such content where to do so would be considered a criminal or statutory offence in any jurisdiction, or gives rise to civil liability, or is otherwise unlawful;
3. falsely or misleadingly imply or suggest endorsement, approval , sponsorship, or association unless explicitly agreed to by Springer Nature in writing;
4. use bots or other automated methods to access the content or redirect messages
5. override any security feature or exclusionary protocol; or
6. share the content in order to create substitute for Springer Nature products or services or a systematic database of Springer Nature journal content.

In line with the restriction against commercial use, Springer Nature does not permit the creation of a product or service that creates revenue, royalties, rent or income from our content or its inclusion as part of a paid for service or for other commercial gain. Springer Nature journal content cannot be used for inter-library loans and librarians may not upload Springer Nature journal content on a large scale into their, or any other, institutional repository.
These terms of use are reviewed regularly and may be amended at any time. Springer Nature is not obligated to publish any information or content on this website and may remove it or features or functionality at our sole discretion, at any time with or without notice. Springer Nature may revoke this licence to you at any time and remove access to any copies of the Springer Nature journal content which have been saved.
To the fullest extent permitted by law, Springer Nature makes no warranties, representations or guarantees to Users, either express or implied with respect to the Springer nature journal content and all parties disclaim and waive any implied warranties or warranties imposed by law, including merchantability or fitness for any particular purpose.
Please note that these rights do not automatically extend to content, data or other material published by Springer Nature that may be licensed from third parties.
If you would like to use or distribute our Springer Nature journal content to a wider audience or on a regular basis or in any other manner not expressly permitted by these Terms, please contact Springer Nature at

onlineservice@springernature.com