TSGBench: Time Series Generation Benchmark

Yihao Ang
National University of Singapore
NUS Research Institute in Chongqing
yihao_ang@comp.nus.edu.sg

Qiang Huang*
National University of Singapore
huangq@comp.nus.edu.sg

Yifan Bao
National University of Singapore
yifan_bao@comp.nus.edu.sg

Anthony K. H. Tung
National University of Singapore
atung@comp.nus.edu.sg

Zhiyong Huang
National University of Singapore
NUS Research Institute in Chongqing
huangzy@comp.nus.edu.sg

ABSTRACT

Synthetic Time Series Generation (TSG) is crucial in a range of applications, including data augmentation, anomaly detection, and privacy preservation. Although significant strides have been made in this field, existing methods exhibit three key limitations: (1) They often benchmark against similar model types, constraining a holistic view of performance capabilities. (2) The use of specialized synthetic and private datasets introduces biases and hampers generalizability. (3) Ambiguous evaluation measures, often tied to custom networks or downstream tasks, hinder consistent and fair comparison.

To overcome these limitations, we introduce TSGBench, the inaugural Time Series Generation Benchmark, designed for a unified and comprehensive assessment of TSG methods. It comprises three modules: (1) a curated collection of publicly available, real-world datasets tailored for TSG, together with a standardized preprocessing pipeline; (2) a comprehensive evaluation measures suite including vanilla measures, new distance-based assessments, and visualization tools; (3) a pioneering generalization test rooted in Domain Adaptation (DA), compatible with all methods. We have conducted comprehensive experiments using TSGBench across a spectrum of ten real-world datasets from diverse domains, utilizing ten advanced TSG methods and twelve evaluation measures. The results highlight the reliability and efficacy of TSGBench in evaluating TSG methods. Crucially, TSGBench delivers a statistical analysis of the performance rankings of these methods, illuminating their varying performance across different datasets and measures and offering nuanced insights into the effectiveness of each method.

PVLDB Reference Format:
Yihao Ang, Qiang Huang, Yifan Bao, Anthony K. H. Tung, and Zhiyong Huang. TSGBench: Time Series Generation Benchmark. PVLDB, 17(3): 305 - 318, 2023.
doi:10.14778/3632093.3632097

PVLDB Artifact Availability:
The source code, data, and other artifacts have been made available at https://github.com/YihaoAng/TSGBench/.

*Qiang Huang is the corresponding author.

Proceedings of the VLDB Endowment, Vol. 17, No. 3 ISSN 2150-8097.
doi:10.14778/3632093.3632097

1 INTRODUCTION

Within the myriad of tasks centered on time series, synthetic Time Series Generation (TSG) stands out as a burgeoning area of focus due to growing demands in data augmentation [68], anomaly detection [3, 8], privacy protection [40], and domain transfer [7]. TSG aims to produce time series akin to the original, preserving temporal dependencies and dimensional correlations while ensuring the generated time series remains useful for various downstream tasks like classification [16, 49] and forecasting [11, 84].

Towards this end, numerous methods have emerged to generate synthetic time series. Their overarching objective is to develop a generative model that accurately captures the features and dependencies inherent in the input time series, thereby generating new time series that maintain both utility and statistical characteristics. For instance, many representative methods [20, 66, 74, 81, 87] utilize the power of Generative Adversarial Networks (GANs), integrating distinctive time series architectures to adeptly capture temporal and dimensional dependencies. Others harness Variational AutoEncoders (VAEs) [15, 48, 50] to strike a balance between data fidelity and the statistical consistency of latent space, thereby enhancing interpretability. Additionally, some recent approaches employ flow-based models [2, 38] to provide explicit likelihood modeling, facilitating effective optimization.

1.1 Motivations

Despite the strides made by these pioneering studies, TSG lags relative to other time series tasks, with three primary limitations (**L1–L3**) emerging prominently:

L1: A comprehensive taxonomy and comparative analysis of various methods are lacking, limiting a holistic view of performance capabilities. When selecting baselines for comparison, researchers often choose models from their methodological realm, missing a broad performance comparison across various methodologies. For example, GT-GAN [38] and AEC-GAN [81] predominantly consider advanced GAN-based methods, overlooking potential contenders like VAE-based approaches. Furthermore, some methods [37, 66, 89] mainly target specific downstream tasks such as missing value imputation or forecasting, hindering the application of the generated time series to other tasks.

L2: Inconsistent dataset selection and preprocessing introduce biases and curb generalizability. First, the dataset choice greatly influences generation outcomes. Different from other time series tasks, the datasets for TSG are diverse due to their lack of

label constraints, broadening data source possibilities. Yet, many studies resort to private and/or synthetic datasets for validation [15, 74, 87]. Private datasets, while varied, hinder reproducibility due to their inaccessibility. Moreover, synthetic datasets, albeit straightforward, may oversimplify real-world complexities, risking biased performance evaluations [85, 87]. Second, preprocessing choices significantly impact generation results. For instance, many methods [35, 38, 87] use a fixed 24-unit sliding window to segment time series, which often inadequately represent their full periods and temporal patterns. Practices like normalizing time series to a range of $[0, 1]$, either before or after applying the sliding window, can lead to discrepancies in results [87]. In addition, datasets such as MIMIC III [39] and PhysioNet 2012 [25], though public, are raw and require meticulous preprocessing to address issues like missing values or anomalies, adding another layer of variability.

L3: Ambiguous evaluation measures hinder a uniform and fair comparison. Evaluating the generated time series hinges on three foundational principles: diversity, fidelity, and usefulness [87]. Diversity gauges how closely the generated series distribution mirrors the original; fidelity examines the similarity between generated and real series; and usefulness assesses the generated series' practical utility in predictive tasks. The choice of evaluation measures crucially impacts method efficacy determination. Most studies [15, 37, 87], however, choose just a subset of these measures, introducing biases in performance evaluation. In addition to quantitative evaluation for TSG, many methods [20, 40] utilize downstream tasks, like classification and forecasting within the "Train on Synthetic, Test on Real" (TSTR) scheme, to showcase utility. These tasks, though popular, come with inherent challenges. For example, some datasets lack classification labels, and short sequences may be unsuitable for forecasting. To counter this, alternative unsupervised approaches [20, 22, 89] like interpolation and clustering have emerged, eliminating the need for external labels. However, they may introduce biases due to their dependency on post-hoc models in time series data [38, 48].

To address these limitations, a comprehensive benchmark for TSG is crucial. It is worth noting that the time series community has introduced a plethora of benchmarks and surveys in areas like databases [29, 41], classification [4, 13, 33], forecasting [5, 55, 56], clustering [36, 69], and anomaly detection [34, 47, 63, 64]. These benchmarks have been pivotal in driving advancements in their respective fields. Nevertheless, there is a noticeable gap when it comes to the field of TSG despite its growing significance. Some notable contributions include Yan et al. [86]'s framework for benchmarking electronic health record generation and Nikitin et al. [61]'s open-source tool aimed at enhancing time series and fostering the use of generative models. A recent study [28] also provides an overview of prevalent evaluation measures for TSG, complemented with an evaluation pipeline. Yet, to our knowledge, a holistic survey or benchmark dedicated to TSG is still absent.

1.2 Our Contributions

In light of these limitations and the evident gap for a dedicated TSG standard, we introduce TSGBench, an open-sourced benchmark designed to standardize comparative assessments of emerging methodologies. It offers robust evaluations and in-depth analyses

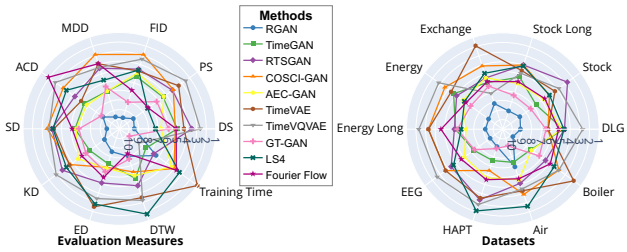


Figure 1: Method ranking across ten evaluation measures and ten datasets.

of principle methods, establishing a foundational benchmark for future investigations. The key contributions of TSGBench (**C1–C4**) are highlighted as follows:

C1: We provide a new taxonomy of various TSG methods based on three foundational generative models. Regarding **L1**, we establish a taxonomy for systematic comparison and categorization (**§3**). Specifically, we categorize TSG methods based on three foundational generative models, offering a new perspective for understanding existing research. On this basis, we compare and analyze the model they used, make connections if different methods use similar techniques, and elaborate upon the inheritance and improvement of relative methods.

C2: We introduce a standard pipeline for dataset selection and preprocessing. In terms of **L2**, we incorporate ten publicly available, real-world datasets from various application domains into TSGBench to enhance reproducibility and mitigate biases or oversimplification in our evaluations. These datasets vary in terms of training sample size, sequence length, and time series dimensions, thereby contributing to a more comprehensive analysis. Moreover, we introduce a standardized pipeline for preprocessing raw time series datasets for TSG (**§4.1**).

C3: We design a suite of diverse yet robust measures to make a comprehensive assessment of the TSG. For **L3**, we provide a suite of twelve evaluation measures tailored for TSG. This suite encompasses five facets, i.e., model-based, feature-based, distance-based measures, training efficiency, and visualization. This allows for a standardized yet thorough comparison (**§4.2**). Further, we introduce a novel generalization test based on Domain Adaptation (DA) to evaluate the generalization capabilities of TSG methods. Notably, we define three scenarios (i.e., single DA, cross DA, and reference DA) that align with real-world applications (**§4.3**).

C4: We conduct systematic evaluations for TSG methods. As part of the benchmark development, we present TSGBench in detail (**§5**) and have conducted preliminary yet systematic experiments to assess the performance of ten representative TSG methods (**§6**). The results demonstrate the ability of TSGBench to assess the efficacy of various TSG methods, illuminating their strengths and weaknesses from different perspectives. The generalization test using DA sheds light on domain shift scenarios in TSG. In addition, we provide statistical analysis of method ranking, examining the consistency of different techniques across ten evaluation measures and ten datasets (Figure 1). Our findings underscore the reliability and efficacy of TSGBench in evaluating TSG methods.

2 PRELIMINARIES

2.1 Problem Definition

Suppose that a time series $\boldsymbol{T}$ with N ($N \geq 1$) individual series of length L is denoted as a matrix, i.e., $\boldsymbol{T} = (\boldsymbol{s}_1, \cdots, \boldsymbol{s}_N)^\mathsf{T}$, where each individual series $\boldsymbol{s}_i$ can be represented as an L-dimensional vector, i.e., $\boldsymbol{s}_i = (x_{i,1}, \cdots, x_{i,L})$, and each $x_{i,j}$ corresponds to a single time point t_j of $\boldsymbol{s}_i$. We denote $p(\boldsymbol{s}_1, \cdots, \boldsymbol{s}_N)$ as the real distribution of a given time series $\boldsymbol{T}$. The goal of Time Series Generation (TSG) is to create a synthetic time series $\boldsymbol{T}^{gen} = (\boldsymbol{s}_1^{gen}, \cdots, \boldsymbol{s}_N^{gen})$ such that its distribution $q(\boldsymbol{s}_1^{gen}, \cdots, \boldsymbol{s}_N^{gen})$ is similar to $p(\boldsymbol{s}_1, \cdots, \boldsymbol{s}_N)$, and $\boldsymbol{T}^{gen}$ and $\boldsymbol{T}$ exhibit consistent statistical properties and patterns. Table 1 summarizes the frequently used notations in this work.

2.2 Scope Illustration

To ensure our initial benchmark is both focused and comprehensive, we employ specific constraints for TSGBench.

Scope of Methods. We restrict our attention to generation methods designed for general-purpose time series. Although certain methods showcase efficacy in specific domains, they lack the flexibility and adaptability needed for broader applications. Thus, these specialized methods are excluded from our evaluation. Moreover, while fine-tuning hyperparameters could yield superior results for each method, we opted not to engage in such optimization. This decision was made to maintain consistency and fairness in comparisons and to adhere to the resource and time constraints inherent in a comprehensive benchmarking study.

Scope of Datasets. To make a holistic evaluation of different generation methods, we select public time series datasets that are both diverse and representative across application domains. Moreover, as we target the generation methods, we consider purely time series as the input. Thus, edge cases, such as those involving the input of a missing data-position matrix or the causality determined by causal graphs, are not incorporated.

Scope of Evaluation Measures. We mainly adhere to the widely accepted "Train on Synthetic, Test on Real" (TSTR) scheme [20, 35, 38, 40, 50, 66, 77, 87, 89]. It assesses the synthetic time series' relevance for real-world applications. While alternative schemes like "Train on Real, Test on Synthetic" (TRTS) [74, 76] do exist, their likeness to TSTR and infrequent use render them non-essential for our purposes. Our evaluation suite, illustrated in Figure 4, is meticulously crafted using well-recognized measures from TSG literature. We have omitted measures specific to some TSG studies, like cosine similarity [51], due to their limited prevalence for extensive benchmarking. Additionally, TSGBench does not use specific downstream tasks as separate evaluation metrics. This is because the included measures, notably discriminative and predictive scores, cover the main goals of time series downstream tasks, such as classification and forecasting. Furthermore, some tasks, like anomaly detection, require extra data and ground truth, which would complicate the evaluation and surpass the designated scope of TSGBench.

3 TSG OVERVIEW

This section provides a taxonomy and comprehensive analysis of TSG methods. We first explore three foundational generative models: GAN, VAE, and Flow-based models (**§3.1**). We then delve into notable TSG methods stemming from these models (**§3.2**).

Table 1: List of frequently used notations.

Symbol	Description
$\boldsymbol{T}$, N	A time series $\boldsymbol{T}$ with N individual series
$\boldsymbol{s}_i$, L	An individual series $\boldsymbol{s}_i$ of length L
l	The sequence length for partitioning
R	The number of sub-matrices, where $R = L - l + 1$
$\boldsymbol{T}_r$	A sub-matrix $\boldsymbol{T}_r$ with N individual series of length l
$\boldsymbol{T}_s^{tr}$, $\boldsymbol{T}_s^{te}$	The training (or historical) data and the test data from a source time series
$\boldsymbol{T}_t^{his}$,$\boldsymbol{T}_t^{gen}$,$\boldsymbol{T}_t^{gt}$	The historical, generated, and ground truth data from a target time series

3.1 Generative Models for TSG

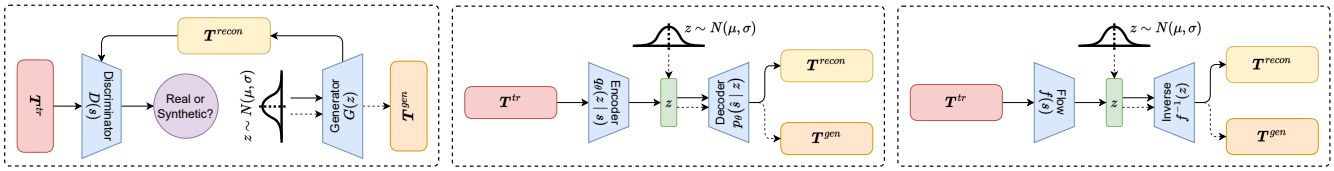
Generative models are designed to learn the intricate patterns and temporal dependencies in time series datasets, allowing for the generation of new time series that reflect the original data's statistical properties. We next describe three foundational generative models for TSG, with their architectural nuances visualized in Figure 2.

Generative Adversarial Networks (GANs). A typical GAN model [26], as depicted in Figure 2(a), comprises a generator G and a discriminator D. During training, G produces a synthetic time series that D attempts to distinguish from real series. This adversarial cycle continues until D cannot reliably differentiate between the two. For generation, only G is used to create a new time series.

Numerous algorithms employ GAN-based models for TSG. They often incorporate specialized neural networks like RNN, LSTM, and Transformer to capture the sequential and temporal intricacies of time series data [20, 51, 58, 66, 77, 87]. Some models utilize novel metrics or loss functions for better alignment with specific temporal patterns [59, 60]. Others enhance traditional GANs with additional modules, such as extra discriminators, classification layers, error correction, and data augmentation to generate time series with specific temporal attributes [37, 74, 81]. While GAN-based models are effective in generating time series, they can be challenging to train and are often resource and time-intensive [38].

Variational AutoEncoders (VAEs). A standard VAE model [44, 45], as shown in Figure 2(b), contains an encoder and a decoder. The encoder $q_\theta(\boldsymbol{z}|\boldsymbol{s})$ transforms input time series $\boldsymbol{s}$ into a latent representation $\boldsymbol{z}$, capturing essential features and generating parameters (e.g., mean μ and variance σ) that model the inherent uncertainty and variability of the time series. The decoder $p_\theta(\hat{\boldsymbol{s}}|\boldsymbol{z})$ reconstructs time series $\hat{\boldsymbol{s}}$ from $\boldsymbol{z}$, preserving the model's capacity to regenerate the temporal patterns. The training phase focuses on minimizing reconstruction loss and the divergence between the learned and a prior standard Gaussian distribution. For generation, the decoder draws from the latent space to create a synthetic time series.

While there are fewer studies on VAE-based methods for TSG compared to GANs, they effectively leverage variational inference to capture the complex temporal aspects of time series data [15, 48, 50]. The benefits of VAE models lie in their interpretability and training efficiency. Additionally, the structured and informative



(a) GAN-based models. (b) VAE-based models. (c) Flow-based models.

Figure 2: Three foundational generative models in popular TSG methods (solid arrow: training, dashed arrow: generation).

latent representations from VAEs can be used for other time series tasks, including representation learning [22] and imputation [21].

Flow-based Models. GANs and VAEs do not directly model the probability density function of time series due to the computational challenge of covering all possible latent representation z values. Flow-based models [17, 18, 43], as shown in Figure 2(c), address this by using invertible transformations f's to explicitly learn data distributions. They have been increasingly employed in TSG, often utilizing explicit likelihood models or Ordinary Differential Equations (ODEs) [2, 10, 38, 42, 73]. Their architectures, typically featuring coupling layers, allow for a computable Jacobian determinant and reversibility. This is further enhanced by specific transformation techniques crucial for modeling complex data distributions.

3.2 Taxonomy of TSG Methods

We detail ten representative TSG methods (**A1–A10**) grounded in the three foundational generative models. Table 2 provides a summary of these methods, including their backbone models.

Pure GAN-based Methods. Early efforts [20, 58, 68] blended vanilla GAN architectures from image generation with neural networks like RNN and LSTM tailored for sequential data. Subsequent studies have focused on pioneering techniques to adapt to time series data and boost performance.

- **A1: RGAN [20].** RGAN is a pioneering work that utilizes the GANs for TSG based on RNNs. It is inspired by the maximum mean discrepancy [27] and tries to measure the statistical difference between generated and practical time series.
- **A2: TimeGAN [87].** TimeGAN considers temporal dependencies within GANs, by simultaneously learning to encode features, generate representations, and iterate across time. It delivers advanced performance and has become a benchmark model for subsequent methods.
- **A3: RTSGAN [66].** RTSGAN combines an autoencoder into GANs and centers on generating time series with varying lengths and handling missing data.
- **A4: COSCI-GAN [74].** COSCI-GAN is devised to explicitly model the complex dynamical patterns across every series, which favors channel/feature relationship preservation.
- **A5: AEC-GAN [81].** AEC-GAN aims to generate long time series with distribution shifts and bias amplification via an error correction module that corrects bias in previously generated data and introduces adversarial samples.

Pure VAE-based Methods. VAE-based methods often exploit variational inference to capture the temporal features effectively. They are generally efficient and have the potential interpretability.

Table 2: Summary of popular TSG methods with their backbone models and specialties (TS: Time Series).

Time	Method	Model	Specialty
2016	C-RNN-GAN [58]	GAN	Music
2017	RGAN [20]	GAN	General (w/ Medical) TS
2018	T-CGAN [68]	GAN	Irregular TS
2019	WaveGAN [19]	GAN	Audio
2019	TimeGAN [87]	GAN	General TS
2020	TSGAN [76]	GAN	General TS
2020	DoppelGANger [53]	GAN	General TS
2020	SigCWGAN [60]	GAN	Long Financial TS
2020	Quant GANs [82]	GAN	Long Financial TS
2020	COT-GAN [85]	GAN	TS and Video
2021	Sig-WGAN [59]	GAN	Financial TS
2021	TimeGCI [35]	GAN	General TS
2021	RTSGAN [66]	GAN	General (w/ Incomplete) TS
2022	PSA-GAN [37]	GAN	General (w/ Forecasting) TS
2022	CEGEN [70]	GAN	General TS
2022	TTS-GAN [51]	GAN	General TS
2022	TsT-GAN [77]	GAN	General TS
2022	COSCI-GAN [74]	GAN	General TS
2023	AEC-GAN [81]	GAN	Long TS
2023	TT-AAE [54]	GAN	General TS
2021	TimeVAE [15]	VAE	General TS
2023	CRVAE [50]	VAE	Medical TS & Causal Discovery
2023	TimeVQVAE [48]	VAE	General TS
2018	Neural ODE [10]	ODE + RNN	General TS
2019	ODE-RNN [73]	ODE + RNN	Irregular TS
2021	Neural SDE [42]	ODE + GAN	General TS
2022	GT-GAN [38]	ODE + GAN	General (w/ Irregular) TS
2023	LS4 [89]	ODE + VAE	General (w/ Forecasting) TS
2020	CTFP [14]	Flow	General TS
2021	Fourier Flow [2]	Flow	General TS
2023	TSGM [52]	SGM	General TS

- **A6: TimeVAE [15].** TimeVAE extends VAEs to the general-purpose TSG. It builds on convolution and enhances interpretability through time series decomposition.
- **A7: TimeVQVAE [48].** It employs the STFT to decompose input time series into low-frequency and high-frequency components. Then, it integrates Vector Quantization with VAEs [79] for enhanced modeling of these components, preserving both the general shape and specific details of the time series.

Mixed-Type Methods. Recent TSG advancements have explored mixed-type methods, merging flow-based models with techniques like DFT or ODEs, or integrating them with GANs or VAEs.

- **A8: Fourier Flows [2].** It uses DFT [62] to analyze the time series in the frequency domain and applies a sequence of data-dependent spectral filters to learn their distributions.

- **A9: GT-GAN [38].** GT-GAN is tailored for dealing with both regular and irregular time series, employing Continuous Time Flow Processes (CTFP) [14] for its generator and GRU-ODE for the discriminator;
- **A10: LS4 [89].** LS4 draws from deep-state space models and incorporates stochastic latent variables to enhance the model's capacity and leverage the training objectives from VAEs.

4 TSGBENCH

To address the challenges and potential biases in dataset and evaluation selection, we have distilled the best practices from pertinent research and established TSGBench, a benchmark tailored for systematically assessing TSG methods. Its architecture is visualized in Figure 3, which encompasses three key modules: (1) a meticulous set of ten public, real-world time series datasets with a standardized preprocessing pipeline (**§4.1**); (2) a comprehensive suite of twelve evaluation measures customized for TSG (**§4.2**); (3) an innovative generation test using Domain Adaption (DA) for TSG (**§4.3**).

4.1 Dataset Selection and Preprocessing

Dataset Selection. We use only publicly available, real-world datasets to ensure reproducibility and sidestep biases or oversimplification in our evaluations. Importantly, our aim is *not* to amass an exhaustive dataset collection but to curate a diverse set that spans multiple domains and features varied data statistics and distributions. We identify and collect ten datasets (**D1–D10**) published over the past two decades. Table 3 summarizes their statistics. Below, we provide a brief description of each dataset.

- **D1: Dodgers Loop Game (DLG) [32].** It consists of loop sensor data from the Glendale on-ramp for the 101 North freeway in Los Angeles.
- **D2: Stock [87].** It comprises daily historical Google stock data from 2004 to 2019, including volume and high, low, opening, closing, and adjusted closing prices.
- **D3: Stock Long [87].** It is identical to the Stock dataset but with a sequence length of 125.
- **D4: Exchange [46].** It contains the daily exchange rates of eight countries (i.e., Australia, Britain, Canada, Switzerland, China, Japan, New Zealand, and Singapore) from 1990 to 2016.
- **D5: Energy [9].** It includes information on appliance's energy use in a low-energy building.
- **D6: Energy Long [9]**. It is identical to the Energy dataset but with a sequence length of 125.
- **D7: EEG [72].** It is with the measurements derived from ElectroEncephaloGraphy (EEG) data captured by Emotiv EEG Neuroheadset. It helps to understand brainwave patterns, especially those under specific cognitive conditions or stimuli.
- **D8: HAPT [71].** It comprises recordings of 30 subjects performing activities of daily living captured via waist-mounted smartphones with embedded inertial sensors.
- **D9: Air [88].** It has air quality, meteorological, and weather forecast data from 4 major Chinese cities: Beijing, Tianjin, Guangzhou, and Shenzhen from 2014/05/01 to 2015/04/30.
- **D10: Boiler [7].** It collects sensor data from three boilers from 2014/03/24 to 2016/11/30 to monitor the operating states.

Table 3: The statistics of the ten datasets.

Datasets	R	l	N	Domain
DLG [32]	246	14	20	Traffic
Stock [87]	3,294	24	6	Financial
Stock Long [87]	3,204	125	6	Financial
Exchange [46]	6,715	125	8	Financial
Energy [9]	17,739	24	28	Appliances
Energy Long [9]	17,649	125	28	Appliances
EEG [72]	13,366	128	14	Medical
HAPT [71]	1,514	128	6	Medical
Air [88]	7,731	168	6	Sensor
Boiler [7]	80,935	192	11	Industrial

Preprocessing Pipeline. While these ten datasets are publicly accessible, few TSG methods have evaluated a majority of them. To circumvent these issues, we introduce a standardized pipeline for preprocessing the raw time series datasets tailored for TSG.

To generate time series in a brief span while preserving meaningful structures, we first follow [35, 87] and segment the long time series $\boldsymbol{T}$ into shorter sub-matrices $\{\boldsymbol{T}_1, \boldsymbol{T}_2, \boldsymbol{T}_3, \cdots\}$. With a specified sequence length l and a stride of 1, we convert $\boldsymbol{T}$ into R overlapping sub-matrices $\{\boldsymbol{T}_r\}_{1 \le r \le R}$, where $R = L - l + 1$ and each $\boldsymbol{T}_r$ has the same l. To determine the value of l, we employ autocorrelation functions [65], ensuring that each $\boldsymbol{T}_r$ encompasses at least one time series period. The time series is then shuffled to approximate an i.i.d. sample distribution [87]. To assess the generalization capability of TSG methods, we divide the data into training and testing sets in a 9:1 ratio, allocating a larger portion for training and evaluation as is common in TSG methodology. Additionally, we normalize the dataset to the range of $[0, 1]$ to enhance efficiency and numerical stability, resulting in a dataset shape of (R, l, N).

4.2 Evaluation Measure Suite

Dozens of measures exist to gauge the quality of TSG methods, which typically adhere to principles like diversity, fidelity, and usefulness, as outlined in **§1.1**. We next offer a suite of twelve prevalent measures (**M1–M12**), complemented by in-depth descriptions. A summary of these measures used in TSG is depicted in Figure 4.

Model-based Measures. These measures predominantly adhere to the TSTR scheme [20, 40]. This scheme involves using the synthetically generated series to train a post-hoc neural network, which is subsequently tested on the original time series.

- **M1: Discriminative Score (DS) [87].** This measure employs a post-hoc time-series classification model with 2-layer GRUs or LSTMs to differentiate between original and generated series [87]. Each original series is labeled as *real*, while the generated series is labeled *synthetic*. Using these labels, an RNN classifier is trained. The classification error on a test set quantifies the generation model's fidelity.
- **M2: Predictive Score (PS) [87].** It involves training a post-hoc time series prediction model on synthetic data [87]. Using GRUs or LSTMs, the model predicts either the temporal vectors of each input series for the upcoming steps [35, 87] or the entire vector [38]. The model's performance is then evaluated on the original dataset using the mean absolute error.

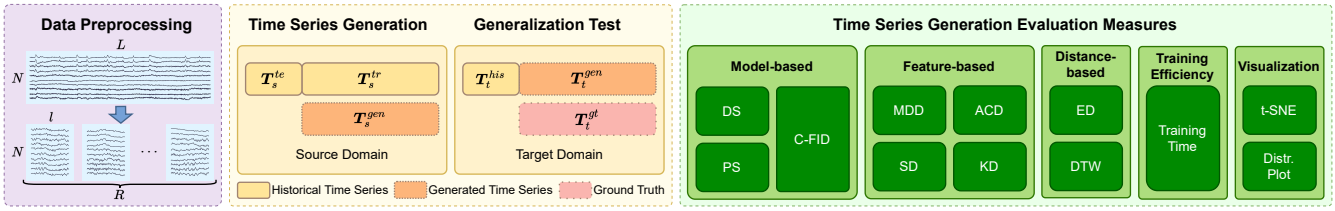


Figure 3: Overall architecture of TSGBench.

- **M3: Contextual-FID (C-FID) [37].** It extends the concept of Frechet Inception Distance (FID) [30] from image generation to TSG. It quantifies how well the synthetic time series conforms to the local context of the time series. Using the time series embeddings from [23], it learns embeddings that seamlessly blend with the local context.

Feature-based Measures. These measures are designed to capture inter-series correlations and temporal dependencies, assessing how well the generated time series preserves original characteristics. A key advantage of feature-based measures is their capacity to yield clear and deterministic results, providing an unambiguous assessment of the quality of generated time series.

- **M4: Marginal Distribution Difference (MDD) [59].** This measure computes an empirical histogram for each dimension and time step in the generated series, using the bin centers and widths from the original series. It then calculates the average absolute difference between this histogram and that of the original series across bins, assessing how closely the distributions of the original and generated series align.
- **M5: AutoCorrelation Difference (ACD) [46].** It computes the autocorrelation of both the original and generated time series, then determines their difference [46, 65]. By contrasting the autocorrelations, we could evaluate how well dependencies are maintained in the generated time series.
- **M6: Skewness Difference (SD).** Beyond ACF, we also consider the statistical measures [28, 81]. Skewness is vital for the marginal distribution of a time series, quantifying its distribution asymmetry. Given the mean (standard deviation) of the train time series $\boldsymbol{T}_s^{tr}$ as $\boldsymbol{\mu}_s^{tr}$ ($\boldsymbol{\sigma}_s^{tr}$) and the generated time series $\boldsymbol{T}_s^{gen}$ as $\boldsymbol{\mu}_s^{gen}$ ($\boldsymbol{\sigma}_s^{gen}$), we evaluate the fidelity of $\boldsymbol{T}_s^{gen}$ by computing the skewness difference between them as:

$$SD = \left| \frac{\mathbb{E}[(\boldsymbol{T}_s^{gen} - \boldsymbol{\mu}_s^{gen})^3]}{\boldsymbol{\sigma}_s^{gen3}} - \frac{\mathbb{E}[(\boldsymbol{T}_s^{tr} - \boldsymbol{\mu}_s^{tr})^3]}{\boldsymbol{\sigma}_s^{tr3}} \right|. \quad (1)$$

- **M7: Kurtosis Difference (KD).** Like skewness, kurtosis assesses the tail behavior of a distribution, revealing extreme deviations from the mean. Using notations from Equation 1, the kurtosis difference between $\boldsymbol{T}_s^{tr}$ and $\boldsymbol{T}_s^{gen}$ is calculated as:

$$KD = \left| \frac{\mathbb{E}[(\boldsymbol{T}_s^{gen} - \boldsymbol{\mu}_s^{gen})^4]}{\boldsymbol{\sigma}_s^{gen4}} - \frac{\mathbb{E}[(\boldsymbol{T}_s^{tr} - \boldsymbol{\mu}_s^{tr})^4]}{\boldsymbol{\sigma}_s^{tr4}} \right|. \quad (2)$$

Training Efficiency. Training efficiency is crucial, particularly in cases that demand rapid TSG methods or where computational resources are scarce. However, only a few studies, such as [15, 38], have been employed for evaluation in this context.

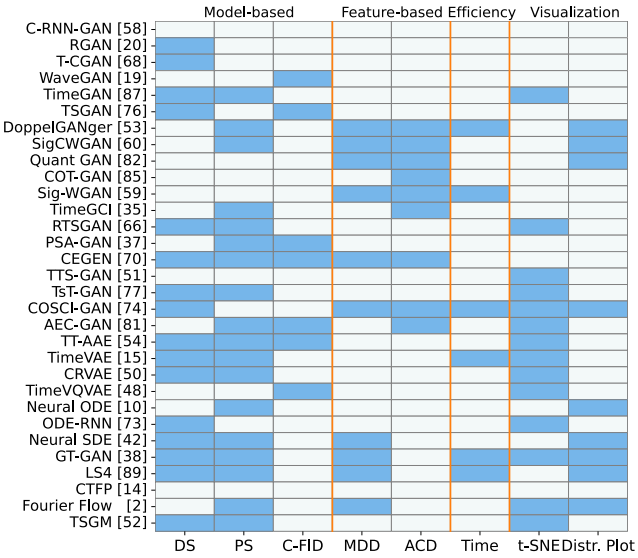


Figure 4: Summary of evaluations in popular TSG methods.

- **M8: Training Time.** It refers to the wall clock time for training a TSG method. It is a vital measure for evaluating and deploying TSG methods due to economic considerations.

Visualization. Visualization offers an intuitive and visually interpretive perspective to directly compare and contrast the structures and patterns between the original and generated time series.

- **M9: t-SNE [80].** It is a prevalent technique for succinctly visualizing the distribution of generated time series compared to the original one within a two-dimensional space.
- **M10: Distribution Plot [38].** It illuminates the difference between the input and generated time series in terms of density, spread, and central tendency to show how the generated time series closely mirrors the original's statistics.

Challenges in DS and PS. As depicted in Figure 4, DS and PS are frequently employed for evaluating TSG methods. However, they can yield unreliable evaluations for several reasons:

(1) DS and PS rely on the TSTR scheme, necessitating post-hoc model training. The inherent randomness in deep models, such as weight initialization, can cause inconsistent evaluations [38, 48]. While repeated training and averaging of results can mitigate this issue, it is time-intensive.
(2) These measures often hinge on specific network architectures and configurations, which vary across studies [37, 87], complicating the comparisons of different TSG methods.

(3) They are sensitive to dataset sizes. With smaller datasets, they may not converge effectively due to insufficient training data, compromising their reliability [15, 48].
(4) The sequence length for TSG is typically short, often just 24-time points, as seen in datasets like Stock and Energy [87]. They may not contain enough periodic information to accurately assess the TSG efficacy in real-world scenarios.

In summary, while DS and PS offer specific insights into the quality of generated time series, their lack of robustness, time-intensive nature, and difficulty adapting to varying dataset sizes and sequence lengths make them less suitable for a comprehensive and reliable evaluation of TSG methods.

Distance-based Measures. To mitigate the challenges associated with DS and PS, we propose the incorporation of two distance-based measures to provide an efficient, deterministic evaluation.

- **M11: Euclidean Distance (ED).** For each original series $\boldsymbol{s}^{tr} = (x_1, \cdots, x_l)$ and its generated series $\boldsymbol{s}^{gen} = (y_1, \cdots, y_l)$, $ED = \sqrt{\sum_{i=1}^{l}(x_i - y_i)^2}$. We take the mean of ED for all series and all samples. Given that the input time series has been preprocessed to fit within the range of $[0, 1]$, ED deterministically assesses the similarity between $\boldsymbol{s}^{gen}$ and $\boldsymbol{s}^{tr}$. It provides a value-wise comparison between the time series.
- **M12: Dynamic Time Warping (DTW) [6].** Given that ED overlooks alignment, we include DTW to capture the optimal alignment between series regardless of their pace or timing. The alignment facilitated by DTW offers insights into the predictive quality of the generated series. Moreover, as shown by [75], multi-dimensional DTW [57] can enhance downstream classification tasks, serving as a discriminative measure.

Using ED and DTW, we can efficiently and effectively evaluate the quality of generated time series, offering streamlined alternatives to DS and PS with similar evaluation goals.

4.3 Generalization Test

Motivation. The domain shift problem is a significant concern in the field of time series analysis [7, 31, 67]. For the task of TSG, most methods usually require large datasets, and many GAN-based methods (e.g., TimeGAN [87] and GT-GAN [38]) are time-intensive to train and fine-tune [15]. Nevertheless, many applications struggle to quickly accumulate sufficient data. Their efficacy may suffer in data-limited situations. Thus, evaluating the generalization capabilities of these methods on small data becomes crucial.

Domain Adaptation (DA). We introduce a novel generalization test using DA [1] to assess the generalization capabilities of TSG methods on small datasets. While DA has been widely applied in other time series tasks like classification and forecasting [7, 31, 83], its application in TSG remains underexplored. This is because typical DA tasks necessitate labels (such as class labels for classification or future values for forecasting) and aim to minimize the distribution shift between source and target domains.

We follow the conventions in DA tasks for time series data [7, 31, 67, 83], examining examples that range from monitoring patient mortality [71] to recording climate and air quality across cities [88], and assessing different machines in a factory [7]. The commonality across these examples lies in the time-dependent nature of the data, while the differences come from the unique characteristics inherent to each scenario. To illustrate, we first present a motivating example before offering a formal definition.

Example 4.1: In a factory, a machine (source domain) equipped with sensors generates operational data as a source time series $\boldsymbol{T}_s$. A TSG model G is trained on this machine's historical data $\boldsymbol{T}_s^{tr}$ and tested on its new data $\boldsymbol{T}_s^{te}$. When a new machine (target domain) with identical sensors is installed, TSG models are used to synthesize sensor readings due to limited data availability. After collecting a brief historical time series $\boldsymbol{T}_t^{his}$ from this new machine, the goal is to generate a synthetic time series $\boldsymbol{T}_t^{gen}$ reflecting its expected performance and patterns. An effective TSG model should adapt to this new machine, producing realistic synthetic data. △

In Example 4.1, the time series available are $\boldsymbol{T}_s^{tr}$, $\boldsymbol{T}_s^{te}$, and $\boldsymbol{T}_t^{his}$. To evaluate the generated time series $\boldsymbol{T}_t^{gen}$, we use a comprehensive time series $\boldsymbol{T}_t^{gt}$ from the target domain as the ground truth. Since we aim to benchmark TSG methods, we do not explore scenarios like modifying the model's architectures or hyperparameter tuning. Accordingly, we focus on the following three DA cases.

Definition 4.1 (Single DA)**:** *A TSG model G is trained using the time series $\boldsymbol{T}_s^{tr}$ from the source domain. It then generates a new time series $\boldsymbol{T}_t^{gen}$ in the target domain and evaluates the performance against $\boldsymbol{T}_t^{gt}$.*

Definition 4.2 (Cross DA)**:** *A TSG model G is trained using the time series $\boldsymbol{T}_s^{tr}$ from the source domain, alongside a small subset of time series from the target domain $\boldsymbol{T}_t^{his}$. It then generates a new time series $\boldsymbol{T}_t^{gen}$ in the target domain and assesses the performance against $\boldsymbol{T}_t^{gt}$.*

Definition 4.3 (Reference DA)**:** *A TSG model G is trained solely using a small subset of time series from the target domain $\boldsymbol{T}_t^{his}$. It then generates a new time series $\boldsymbol{T}_t^{gen}$ in the target domain and gauges the performance against $\boldsymbol{T}_t^{gt}$.*

Datasets and Evaluations. We extend three datasets HAPT [71], Air [88], and Boiler [7] for generalization testing, as they include domain information in line with conventions in time series DA tasks [7, 67]. Specifically, our dataset configurations are as follows:

- **HAPT [71].** The user is treated as the domain attribute. We randomly select User 14 as the source domain and Users 0, 23, 18, 52, and 20 as the target domains. Our evaluation targets the time series for 'walking' as it provides a more concentrated and comparable analysis of each user's walking pattern.
- **Air [88].** The city serves as the domain attribute. We randomly chose Tianjin (TJ) as the source domain and selected Beijing (BJ), Guangzhou (GZ), and Shenzhen (SZ) as the target domains. This setup enables us to investigate variations in air quality patterns across different urban settings.
- **Boiler [7].** We employ the boiler machine as the domain attribute. Boiler 1 is randomly chosen as the source domain, with Boilers 2 and 3 serving as the target domains. This configuration allows us to examine operational variations and similarities across different boiler units.

We utilize the evaluation criteria outlined in **§4.2** for assessment. DA tasks serve as a valuable lens to scrutinize the generalization capabilities of TSG methods across domains. It is pertinent to mention that for the generalization test, efficient processing of data

from the target domain is essential, making the training efficiency of a TSG method a pivotal consideration. Thus, the results from DA can enhance other time series tasks by providing synthetic data for target domains with limited sample availability.

5 EXPERIMENTAL SETUP

The primary goal of TSGBench is to establish a rigorous, standardized framework for evaluating various TSG methods. As such, we will not conduct an exhaustive evaluation, given time and resource constraints [15, 38]. Instead, we concentrate on assessing recent, prevalent methods that come with publicly available code and have exhibited state-of-the-art performance on selected datasets.

Algorithms. Our initial experiments evaluate the performance of ten pivotal TSG algorithms mentioned in **§3.2**. These algorithms, carefully selected from academic literature, include: (1) pure GAN-based methods: RGAN [20], TimeGAN [87], RTSGAN [66], COSCI-GAN [74], and AEC-GAN [81]; (2) pure VAE-based method: TimeVAE [15] and TimeVQVAE [48]; (3) mixed-type methods: GT-GAN [38], LS4 [89], and Fourier Flow [2].

Datasets. Our experiments draw on ten real-world datasets across diverse domains, as detailed in **§4.1**. Moreover, we extend HAPT, Air, and Boiler for generalization tests, as described in **§4.3**.

Evaluation Measures. We employ twelve evaluation measures outlined in **§4.2** to make a thorough and systematic assessment of the TSG algorithms. The lower value means the better performance. Specifically, for DS and PS, we adopt two layers of LSTM; for C-FID, we adopt ts2vec [23] as the backbone. For all evaluation measures, we repeat them five times and report their average results.

Experiments Environments. All experiments are conducted on a machine with Intel® Xeon® Gold 6342 CPU @ 2.80GHz, 64 GB memory, and NVIDIA GeForce RTX 3090.

Parameter Settings. For RGAN, the number of hidden units for GANs is set to $4n$. For TimeGAN, we use the suggested settings [87] and adopt three-layer GRUs for the network architectures. For RTSGAN, we adhere to its complete time series generation [66] and set $\beta_1 = 0.9$ and $\beta_2 = 0.999$. For COSCI-GAN, we set $\gamma = 5$, employ MLP-based networks for the central discriminator, and follow other hyper-parameters from [74]. For AEC-GAN, we set the context length $l_c = 4$ if $l = 16$, $l_c = 85$ if $l = 24$, $l_c = 25$ if $l = 125$, $l_c = 28$ if $l = 128$, $l_c = 56$ if $l = 168$, and $l_c = 64$ if $l = 192$, and set the generation length $l_q = l - l_c$. For TimeVAE, we set the latent dimension to 8 and the hidden layer sizes to 50, 100, and 200. For TimeVQVAE, we adopt settings from [48], with $n\_fft = 8$ and varying $max\_epochs \in \{2000, 10000\}$ for two training stages. For GT-GAN, we use regular time series from [38] and set $P_{MLE} = 2$. The absolute and relative tolerances for the generator are set to 0.001 and 0.01 for Energy and Energy Long, respectively, and 0.01 and 0.001 for other datasets. For LS4, we set the latent space dimension to 5 and configured the batch size to 512 for Air and Boiler and 1024 for the rest, optimizing GPU utilization. For Fourier Flow, which is primarily designed for individual series, we follow its guidelines [2] and adapt it for time series with $N > 1$ by using DFT [62] to each dimension. We configure the hidden size to 50 and set the number of flows to 3 for Stock and Stock Long and 5 for others.

6 RESULTS ANALYSIS

6.1 TSG Benchmarking

We first present the results of ten methods applied to ten real-world datasets in Figures 5 and 6.

Model-based Evaluations. As evident from the first three rows of Figure 5, TimeVQVAE, TimeVAE, RTSGAN, and COSCI-GAN consistently excel across the three model-based measures. In contrast, RGAN and TimeGAN underperform. The superior performance of VAE-based methods can be attributed to their adeptness at capturing temporal dependencies, pivotal for forecasting and representation learning tasks. Some methods like LS4, despite having high DS and PS, secure a relatively low C-FID. This indicates their satisfactory performance in the representation learning tasks. Some outliers appear in Energy and Energy Long, where almost all methods achieve DS around 0.5. This suggests while the generated time series might suffice for predictive tasks, they are easily distinguishable in classification tasks. Additionally, the large standard deviation in DS is worth noting, which will be further analyzed in **§6.3**.

Upon integrating observations from Table 3, we find that as l increases, PS will slightly drop, implying that shorter sequences present a greater challenge in generating time series. Furthermore, a relatively strong correlation appears between N and these measures, suggesting that the generation of high-dimensional time series presents a steeper challenge, leading to higher DS and PS.

Feature-based Evaluations. In Figure 5, when examining feature-based measures from rows 4 to 7, Fourier Flow delivers the best performance in ACD, while COSCI-GAN dominates in MDD and SD. This is likely due to their effectiveness in capturing the statistical properties of time series. Interestingly, within each dataset, the performance ranking across all four measures appears to be consistent. This consistency implies a robust correlation among these feature-based measures, indicating that the overall performance of a method in capturing key features is likely consistent as well.

Moreover, feature-based measures, which primarily assess the statistical similarities between original and generated data, tend to improve when $N > 10$. This observation suggests that generated time series may better match the statistical properties of the original one when dealing with high-dimensional datasets, potentially due to the rich features available for modeling.

Distance-based Evaluations. The patterns observed in the distance-based measures, shown in the 8th and 9th rows of Figure 5, differ from those in earlier measures. This divergence arises because distance-based measures direct quantification of dissimilarity between the generated and original time series. Regarding ED and DTW, VAE-based methods stand out. They effectively capture the overall trend of the original time series, preserving both value proximity (ED) and trend similarity (DTW).

An intriguing finding is that the distances between generated and authentic time series amplify as l increases. This is likely because longer sequences introduce more complex temporal dependencies, making them harder to model accurately, leading to larger divergence in values and alignment. Conversely, as N grows, these distances decrease. The reason could be that a larger set of series provides a broader range of temporal patterns and inter-series correlations, benefiting the methods' evaluation.

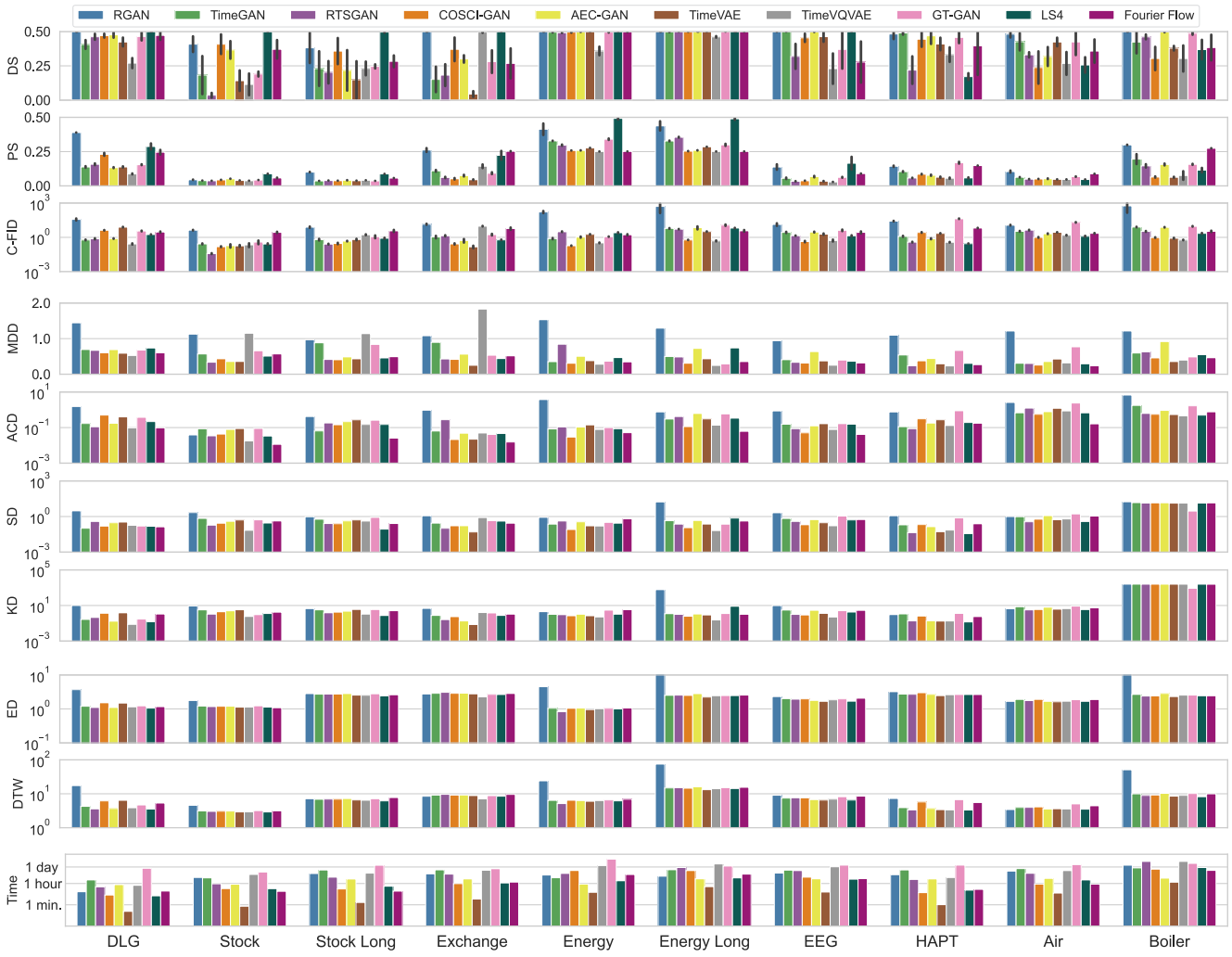


Figure 5: TSG benchmarking.

Training Efficiency. In the final row of Figure 5, we explore the training efficiency. For ease of interpretation, we categorize training times into four distinct segments: $<$ 1 minute, $<$ 1 hour, $<$ 1 day, and $\geq$ 1 day. Our findings highlight the exceptional efficiency of TimeVAE and LS4, likely due to their VAE-based structure and efficient training strategies that minimize computational demands and iterations needed for convergence. On the other hand, TimeVQVAE, while effective in modeling, presents a more time-intensive training process. The complexity of TimeVQVAE arises primarily from its STFT process, tokenization, and iterative decoding. Also, our fairness-driven decision to maintain consistent hyper-parameters across all datasets potentially escalates training times for those with larger R values. In addition, GAN-based methods generally manifest longer training times. For example, GT-GAN takes training time of more than 1 day on all datasets except Stock, DLG, and Exchange. The inherent intricacy of GANs, requiring concurrent training of generator and discriminator networks to reach equilibrium, leads to a longer convergence period.

Visualization. At last, we look into the t-SNE visualization and Distribution Plot of the generated time series in Figure 6. VAE-based methods, COSCI-GAN, and RTSGAN excel at generating time series that closely mirror the features and patterns of the original ones. Examining the subtleties among these leading methods reveals variations in their generation capabilities for different distributions. For instance, the DLG dataset, characterized by its bimodal distribution, challenges COSCI-GAN, which struggles to capture both modes accurately. Conversely, TimeVAE and LS4 perform well with the Exchange dataset with the multifaceted peak structure, indicating their innate ability to grasp multifarious distributional patterns.

Some methods, such as RGAN and GT-GAN, which might fare well on a single data distribution, cannot deal with the dramatic shift in the overall distributions (e.g., from Stock to HAPT). This suggests limitations in their ability to adapt to significantly different data distributions. We also find that some methods, e.g., RGAN, while they successfully mimic the distribution of the original time series in their generated output (e.g., Energy), stumble when it comes

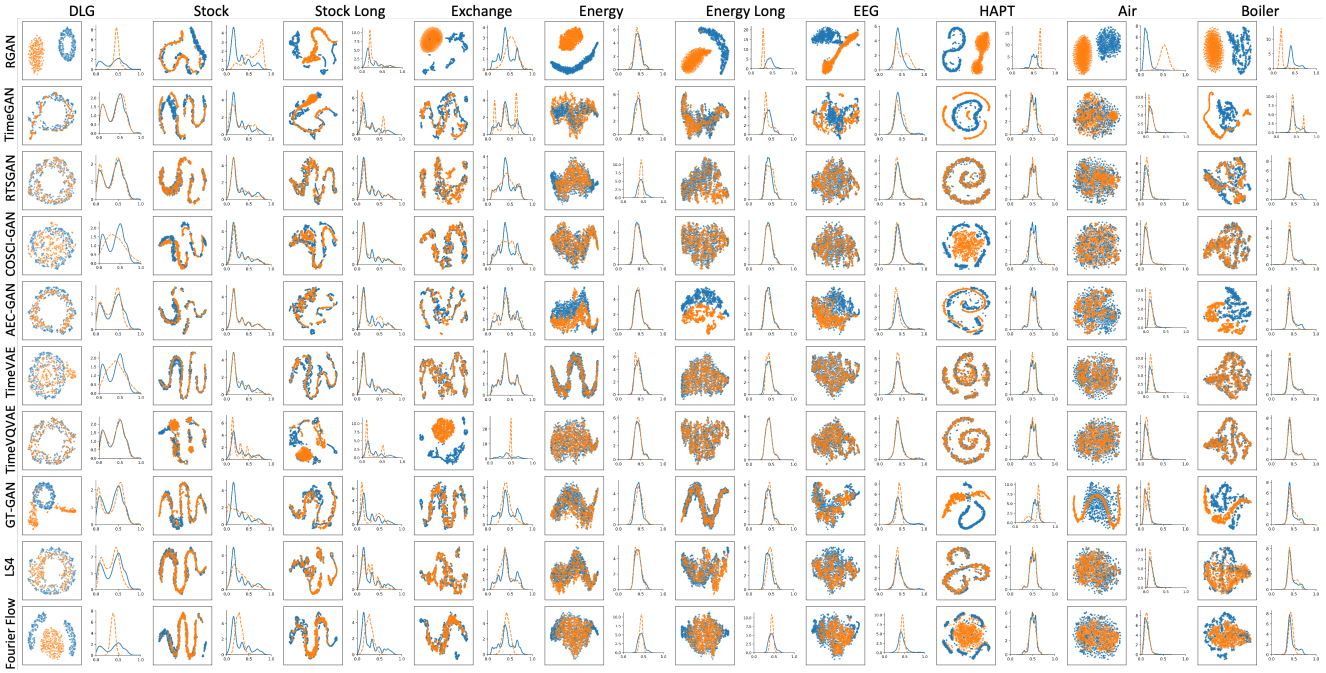


Figure 6: Visualization for TSG benchmarking by t-SNE and Distribution Plot (blue as T^{tr}, orange as T^{gen}).

to the precise matching by t-SNE. This observation aligns with the quantitative results in Figure 5, underscoring the challenges in replicating exact values of the original time series rather than overall distribution characteristics. Moreover, some methods, like TimeGAN, can partially fit the original time series but usually contain extraneous information. They may struggle to handle the inherent noise present in real-world time series.

6.2 Generalization Test

In conjunction with benchmarking results, we delve deeper into a select group of methods that showcased eminent performance, with a pronounced emphasis on efficiency, especially when contemplating generalization in the nascent stages of the target domain. In particular, we employ TimeGAN as the baseline and focus on four efficient methods with leading performance, i.e., TimeVAE, COSCI-GAN, RTSGAN, and LS4. Quantitative results for the datasets HAPT, Air, and Boiler are displayed in Figure 7, where each measure is partitioned into three parts by gray dashed lines, signifying the single DA, cross DA, and reference DA, respectively.

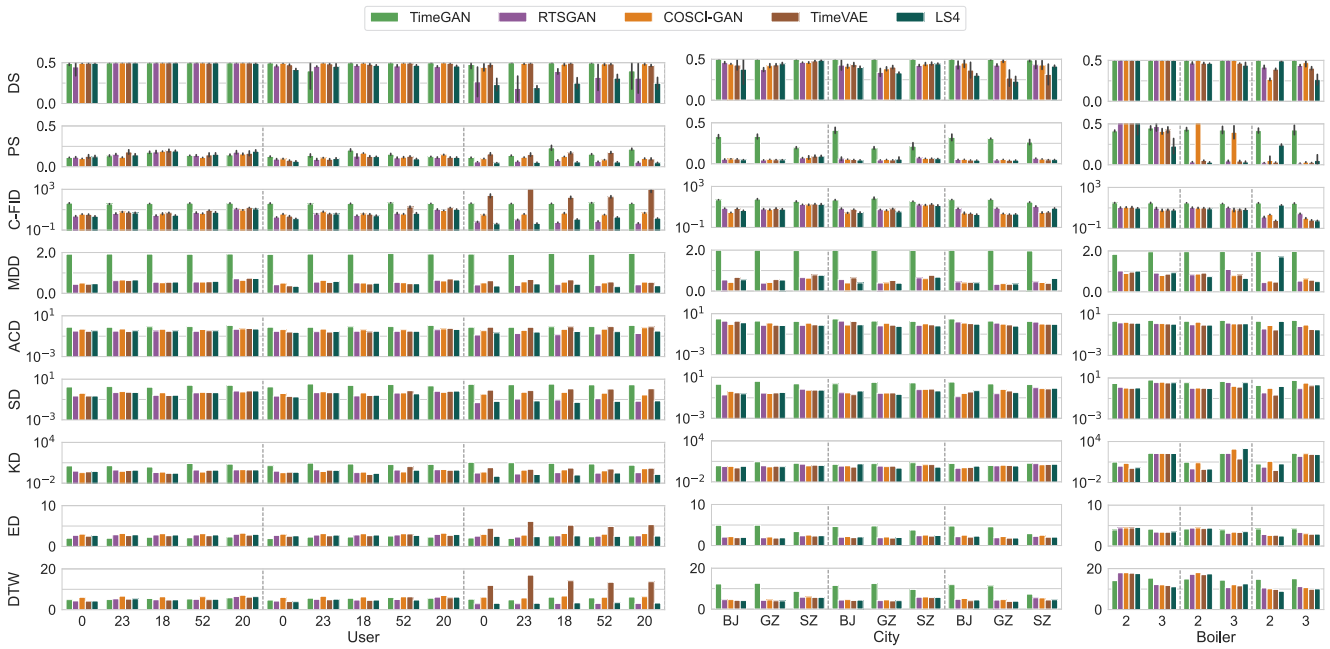
We first focus on the HAPT dataset, which consists of the most number of target domains. As depicted in Figure 7(a), certain methods like TimeGAN show a little discrepancy in their performance across single DA, cross DA, and reference DA. This suggests that these methods may struggle with generalization in many cases. For instance, TimeGAN may struggle to adapt effectively due to reasons inherent to its model design. Another noteworthy observation is the considerable standard deviation seen in DS and PS under DA tasks, barring those that exhibit the worst score (0.5). This heightened variance undermines their reliability, which will be further analyzed in **§6.3**. When examining methods with strong generalization capabilities, the performance of some algorithms (e.g., TimeVAE and COSCI-GAN) in cross DA outperforms that in reference DA. This indicates that they do not treat the smaller proportion of target domain time series $\boldsymbol{T}_t^{his}$ as noise, even when faced with multiple distributions as inputs. Conversely, methods like RTSGAN and LS4 excel in single DA, as they quickly converge when given a limited set of time series. In addition, for the target domain User 20, TimeVAE's performance in reference DA matches the level achieved in cross DA. This may be due to User 20's data distribution being relatively simpler and less noisy, facilitating quicker convergence for TimeVAE during training

As R and N grow (from HAPT to Air and Boiler), we observe a consistent pattern for the single DA task in Figures 7(b) and 7(c). In the case of the reference DA task, a larger R facilitates model convergence, yielding more competitive outcomes compared to the cross DA task. Another interesting observation is the inconsistent performance of SD, KD, and DTW measures in evaluating DA for the Boiler dataset compared to the HAPT and Air datasets. This discrepancy is due to the periodic trends present in HAPT and Air. Since SD and KD gauge data distribution and DTW assesses alignment, they are less effective for datasets lacking periodic trends.

6.3 Robustness Test for Evaluation Measures

In the course of TSG benchmarking (**§6.1**) and generalization testing (**§6.2**), we observed significant variability and inconsistency in DS and PS. Thus, we undertook additional analyses to evaluate the sensitivity and robustness of various evaluation measures.

We randomly generated 10,000 synthetic time series with $N = 5$ adhering to the sine function [87], expressed as $x_{i,j} = \sin(2\pi\eta j + \theta)$, where $\eta \sim \mathcal{U}[0, 1]$, $\theta \sim \mathcal{U}[-\pi, \pi]$, $i \in [1, 5]$, and $j \in [1, l]$. We then assessed these series using two sequence lengths, $l = 24$ and $l = 125$, employing the evaluation measures outlined in **§4.2**. The predictive



(a) HAPT (Source: User 14). (b) Air (Source: Tianjin, TJ). (c) Boiler (Source: Boiler 1).

Figure 7: Generalization test: single DA, cross DA, and reference DA (left to right in each subfigure).

score incorporated two configurations: next step forecasting [87] (denoted as PS) and entire sequence forecasting (denoted as PS (entire)) [38]. Table 4 considered two scenarios for input time series: (1) identical original and generated data, where ideal evaluation measures should be 0; and (2) time series sampled from the same sine function but with different seeds.

In Table 4, it is evident that all feature-based, distance-based measures and C-FID demonstrate the robustness and accurate reflection of the changes in input time series. Contrarily, DS and PS exhibit some inconsistencies. For instance, the scores for DS and PS from random sampling at $l = 125$ manifest lower values than their identical counterparts, a result that tends to be counter-intuitive. Also, the standard deviation of DS is notably high relative to the mean value, suggesting considerable variation. Moreover, DS appears insensitive to the increase in l, limiting its effectiveness in accurately evaluating the TSG methods' performance on longer sequences. The above observations confirm that DS and PS may be less robust than other measures, especially as input settings change. This aligns with the challenges we discussed in **§1.1**.

6.4 Ranking Analysis

Selecting suitable TSG methods is vital for handling new time series datasets. Therefore, it is crucial to understand and analyze the consistency in performance across various methods.

Method Ranking. We initially present the ranking of ten methods under two specific scenarios. First, we evaluate their performance across all datasets for each individual measure, as depicted on the left-hand side of Figure 1. Second, we examine their average ranking across all measures but constricted to each dataset included in TSGBench, as showcased on the right-hand side of Figure 1.

Across all evaluation measures and datasets, no single method consistently dominates, but TimeVQVAE, TimeVAE, COSCI-GAN, RTSGAN, and LS4 often outperform others. Specifically, TimeVAE and LS4 excel in distance-based measures and demonstrate impressive training efficiency, while COSCI-GAN and RTSGAN lead in model-based measures. In contrast, RGAN generally ranks lower in performance. A dataset-centric analysis reveals a similar pattern, with the same group of TimeVQVAE, TimeVAE, COSCI-GAN, RTSGAN, and LS4 achieving high rankings across various datasets. This consistency across different evaluation aspects further validates the robustness and reliability of TSGBench.

Statistical Validation. To statistically confirm the method ranking, we employ the Friedman test [24] along with Conover's test [12, 78] for ranking comparisons. Figure 8 presents the average rankings and the critical difference for each method.

The ten TSG methods can be segregated into four tiers. TimeVQVAE, TimeVAE, COSCI-GAN, LS4, and RTSGAN lead, followed by Fourier Flow, AEC-GAN, and TimeGAN. GT-GAN forms the third cohort, while RGAN occupies the lowest tier. This aligns with the observations from **§6.1**. The supremacy of the first-tier methods becomes unambiguously clear as they stand statistically distinct from their counterparts spanning the other three tiers. Zooming into this premier group, despite the noticeable lead of TimeVQVAE and TimeVAE, they do not statistically tower over their peers within the group. This statistical overlap is not confined to the elite group but is also exhibited in the second group. Such an overlap indicates that the performance deltas between methods within these groups are not glaringly vast. Conversely, the third and fourth groups' distinct position underscores a pronounced performance chasm, setting it distinctly apart from others.

Table 4: Robustness test on ten evaluation measures.

Input	Shape (R, l, N)	Model-based				Feature-based				Distance-based	
		DS	PS	PS (entire)	C-FID	MDD	ACD	SD	KD	ED	DTW
Identical	(10,000, 24, 5)	0.006±0.003	0.094±0.000	0.072±0.005	0.000±0.000	0.001	0.000	0.000	0.000	0.000	0.000
	(10,000, 125, 5)	0.010±0.007	0.251±0.003	0.169±0.001	0.000±0.000	0.000	0.000	0.000	0.000	0.000	0.000
Random Sampling	(10,000, 24, 5)	0.009±0.005	0.094±0.000	0.071±0.005	0.003±0.000	0.222	0.131×10^{-3}	0.009	0.007	0.653	1.689
	(10,000, 125, 5)	0.003±0.005	0.249±0.003	0.168±0.001	0.016±0.001	0.108	0.022	0.009	0.020	4.350	9.663

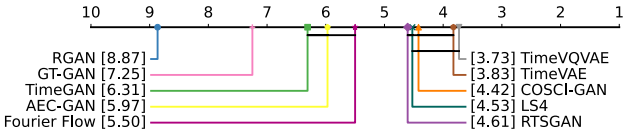


Figure 8: Critical difference diagram of TSG methods.

6.5 Recommendations

Finally, we provide guidelines to assist users in effectively using TSGBench. This benchmark is designed to be a beacon for navigating the intricate terrain of TSG.

Selection of TSG Methods. TSGBench serves as a toolkit to effectively discern the most appropriate TSG methods for different datasets. When confronted with a new dataset, an insightful strategy would be to juxtapose the statistical properties and distributions of the new time series datasets against those cataloged in TSGBench. This strategy offers a navigational compass, pointing users towards relevant generation techniques.

(1) As a foundational step, we advocate for users to commence with VAE-based methods (e.g., TimeVAE and LS4). Their consistent leading performance and superior computational efficiency make them go-to choices for initial exploration.
(2) In applications emphasizing autocorrelation or forecasting, such as predictive maintenance or stock market analysis, the ACD measure becomes crucial. Fourier Flow, which is recognized for maintaining temporal dependencies, is highly suitable for these scenarios. On the other hand, for capturing complex multi-variate relationships in datasets, COSCI-GAN is the recommended choice.
(3) Subsequent considerations focus on dataset size and domain specificity. For small-sized datasets, RTSGAN and LS4, which excel in single DA, are strong choices. For heterogeneous datasets, or when the goal is to generate time series for a new target domain, TimeVAE and COSCI-GAN stand out for their effectiveness in cross DA.
(4) Users can further fine-tune their method selection based on specific real-world application needs, which involves identifying the most relevant evaluation measures. In this case, Figure 1 serves as a valuable visual guide.

Selection of Evaluation Measures. When evaluating a new TSG method, a comprehensive assessment is essential. Leveraging the features of TSGBench, we offer the following guidelines to facilitate this process. This allows users to tailor their choice of evaluation measures to the specific application requirements.

(1) For applications where generated series will be used in classification or forecasting, model-based measures are advisable. Nevertheless, considering the robustness issues with DS and PS, we recommend starting with C-FID. Its prowess in gauging fidelity based on representations bestows it with the capability to augment subsequent tasks.
(2) When the goal is to emphasize the statistical attributes of the dataset, feature-based measures emerge as the preferred option, offering precise insights into the statistical nuances.
(3) In projects focusing on time series clustering, distance-based metrics assume an elevated importance due to their ability to discern subtle distinctions and similarities within the data.

Thus, users can customize their choice of evaluation measures to align closely with their research goals, ensuring more robust results. Note that not all measures tend to yield uniform outcomes consistently. Users could continually calibrate their evaluations and discern the trade-off between effectiveness and efficiency. This ensures that the chosen measures are not only reflective of the desired outcomes but also optimized for performance.

7 CONCLUSION AND FUTURE WORK

In this paper, we propose TSGBench, a groundbreaking benchmark specifically designed for TSG. This benchmark is comprehensive, featuring datasets from diverse domains, a standardized data preprocessing pipeline, a holistic evaluation suite, and a novel generalization test grounded in DA. Extensive results validate its capability to offer a unified and equitable platform for assessing the efficacy and robustness of various TSG methods. Importantly, TSGBench also sheds light on the potential of their generalization capabilities. As a collaborative resource, it promises to catalyze further advancements within the time series community.

Looking forward, our aspirations for TSGBench are multifold. We intend to continually integrate emerging TSG methods, ensuring the benchmark remains at the vanguard of advancements. Also, the addition of new datasets is on the horizon, aiming to enhance the diversity and complexity of the challenges the benchmark addresses. Lastly, we are also contemplating introducing functionalities that facilitate automatic tuning, thereby streamlining the training process and making it even more friendly for users.

ACKNOWLEDGMENTS

This research is supported by the National Research Foundation, Singapore under its Strategic Capability Research Centres Funding Initiative and the Ministry of Education, Singapore, under its MOE AcRF TIER 3 Grant (MOE-MOET32022-0001). Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore and the Ministry of Education, Singapore.

REFERENCES

[1] Ahmed Alaa, Boris Van Breugel, Evgeny S Saveliev, and Mihaela van der Schaar. 2022. How Faithful is your Synthetic Data? Sample-level Metrics for Evaluating and Auditing Generative Models. In *ICML*. 290–306.

[2] Ahmed M. Alaa, Alex James Chan, and Mihaela van der Schaar. 2021. Generative Time-series Modeling with Fourier Flows. In *ICLR*.

[3] Yihao Ang, Qiang Huang, Anthony KH Tung, and Zhiyong Huang. 2023. A Stitch in Time Saves Nine: Enabling Early Anomaly Detection with Correlation Analysis. In *ICDE*. 1832–1845.

[4] Anthony Bagnall, Hoang Anh Dau, Jason Lines, Michael Flynn, James Large, Aaron Bostrom, Paul Southam, and Eamonn Keogh. 2018. The UEA multivariate time series classification archive, 2018. *arXiv preprint arXiv:1811.00075* (2018).

[5] André Bauer, Marwin Züfle, Simon Eismann, Johannes Grohmann, Nikolas Herbst, and Samuel Kounev. 2021. Libra: A Benchmark for Time Series Forecasting Methods. In *ICPE*. 189–200.

[6] Donald J Berndt and James Clifford. 1994. Using dynamic time warping to find patterns in time series. In *KDD Workshop*. 359–370.

[7] Ruichu Cai, Jiawei Chen, Zijian Li, Wei Chen, Keli Zhang, Junjian Ye, Zhuozhang Li, Xiaoyan Yang, and Zhenjie Zhang. 2021. Time Series Domain Adaptation via Sparse Associative Structure Alignment. In *AAAI*. 6859–6867.

[8] David Campos, Tung Kieu, Chenjuan Guo, Feiteng Huang, Kai Zheng, Bin Yang, and Christian S Jensen. 2021. Unsupervised Time Series Outlier Detection with Diversity-Driven Convolutional Ensembles. *Proc. VLDB Endow.* 15, 3 (2021), 611–623.

[9] Luis Candanedo. 2017. Appliances energy prediction. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5VC8G.

[10] Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. 2018. Neural Ordinary Differential Equations. In *NeurIPS*. 6572–6583.

[11] Razvan-Gabriel Cirstea, Bin Yang, Chenjuan Guo, Tung Kieu, and Shirui Pan. 2022. Towards Spatio- Temporal Aware Traffic Time Series Forecasting. In *ICDE*. 2900–2913.

[12] William Jay Conover and Ronald L Iman. 1979. On multiple-comparisons procedures. *Los Alamos Sci. Lab. Tech. Rep. LA-7677-MS* 1 (1979), 14.

[13] Hoang Anh Dau, Anthony Bagnall, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Ann Ratanamahatana, and Eamonn Keogh. 2019. The UCR Time Series Archive. *IEEE/CAA Journal of Automatica Sinica* 6, 6 (2019), 1293–1305.

[14] Ruizhi Deng, Bo Chang, Marcus A. Brubaker, Greg Mori, and Andreas M. Lehrmann. 2020. Modeling Continuous Stochastic Processes with Dynamic Normalizing Flows. In *NeurIPS*. 7805–7815.

[15] Abhyuday Desai, Cynthia Freeman, Zuhui Wang, and Ian Beaver. 2021. TimeVAE: A Variational Auto-Encoder for Multivariate Time Series Generation. *arXiv preprint arXiv:2111.08095* (2021).

[16] Daizong Ding, Mi Zhang, Yuanmin Huang, Xudong Pan, Fuli Feng, Erling Jiang, and Min Yang. 2022. Towards backdoor attack on deep learning based time series classification. In *ICDE*. 1274–1287.

[17] Laurent Dinh, David Krueger, and Yoshua Bengio. 2014. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516* (2014).

[18] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. 2016. Density estimation using Real NVP. In *ICLR*.

[19] Chris Donahue, Julian J. McAuley, and Miller S. Puckette. 2019. Adversarial Audio Synthesis. In *ICLR*.

[20] Cristóbal Esteban, Stephanie L Hyland, and Gunnar Rätsch. 2017. Real-valued (medical) time series generation with recurrent conditional gans. *arXiv preprint arXiv:1706.02633* (2017).

[21] Vincent Fortuin, Dmitry Baranchuk, Gunnar Rätsch, and Stephan Mandt. 2020. GP-VAE: Deep Probabilistic Multivariate Time Series Imputation. In *AISTATS*. 1651–1661.

[22] Vincent Fortuin, Matthias Hüser, Francesco Locatello, Heiko Strathmann, and Gunnar Rätsch. 2019. SOM-VAE: Interpretable Discrete Representation Learning on Time Series. In *ICLR*.

[23] Jean-Yves Franceschi, Aymeric Dieuleveut, and Martin Jaggi. 2019. Unsupervised Scalable Representation Learning for Multivariate Time Series. In *NeurIPS*. 4652–4663.

[24] Milton Friedman. 1937. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *J. Amer. Statist. Assoc.* 32, 200 (1937), 675–701.

[25] Ary L Goldberger, Luis AN Amaral, Leon Glass, Jeffrey M Hausdorff, Plamen Ch Ivanov, Roger G Mark, Joseph E Mietus, George B Moody, Chung-Kang Peng, and H Eugene Stanley. 2000. PhysioBank, PhysioToolkit, and PhysioNet: components of a new research resource for complex physiologic signals. *Circulation* 101, 23 (2000), e215–e220.

[26] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *NIPS*. 2672–2680.

[27] Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander J. Smola. 2006. A Kernel Method for the Two-Sample-Problem. In *NIPS*. 513–520.

[28] Xin Dong Baoren Xiao Lei Jiang Hang Lou, Jiajie Tao and Hao Ni. 2023. Evaluation of Time Series Generative Models. https://github.com/DeepIntoStreams/Evaluation-of-Time-Series-Generative-Models.git.

[29] Yuanzhe Hao, Xiongpai Qin, Yueguo Chen, Yaru Li, Xiaoguang Sun, Yu Tao, Xiao Zhang, and Xiaoyong Du. 2021. TS-Benchmark: A Benchmark for Time Series Databases. In *ICDE*. 588–599.

[30] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. 2017. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. In *NIPS*. 6629–6640.

[31] Hailin Hu, MingJian Tang, and Chengcheng Bai. 2020. Datsing: Data augmented time series forecasting with adversarial domain adaptation. In *CIKM*. 2061–2064.

[32] Jon Hutchins. 2006. Dodgers Loop Sensor. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C51P50.

[33] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. 2019. Deep learning for time series classification: a review. *DMKD* 33, 4 (2019), 917–963.

[34] Vincent Jacob, Fei Song, Arnaud Stiegler, Bijan Rad, Yanlei Diao, and Nesime Tatbul. 2021. Exathlon: A Benchmark for Explainable Anomaly Detection over Time Series. *Proc. VLDB Endow.* 14, 11 (2021), 2613–2626.

[35] Daniel Jarrett, Ioana Bica, and Mihaela van der Schaar. 2021. Time-series Generation by Contrastive Imitation. In *NeurIPS*. 28968–28982.

[36] Ali Javed, Byung Suk Lee, and Donna M Rizzo. 2020. A Benchmark Study on Time Series Clustering. *Machine Learning with Applications* 1 (2020), 100001.

[37] Paul Jeha, Michael Bohlke-Schneider, Pedro Mercado, Shubham Kapoor, Rajbir Singh Nirwan, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. 2021. PSA-GAN: Progressive self attention GANs for synthetic time series. In *ICLR*.

[38] Jinsung Jeon, Jeonghak Kim, Haryong Song, Seunghyeon Cho, and Noseong Park. 2022. GT-GAN: General Purpose Time Series Synthesis with Generative Adversarial Networks. In *NeurIPS*. 36999–37010.

[39] Alistair EW Johnson, Tom J Pollard, Lu Shen, Li-wei H Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. 2016. MIMIC-III, a freely accessible critical care database. *Scientific Data* 3, 1 (2016), 1–9.

[40] James Jordon, Jinsung Yoon, and Mihaela Van Der Schaar. 2018. PATE-GAN: Generating synthetic data with differential privacy guarantees. In *ICLR*.

[41] Eamonn Keogh and Shruti Kasetty. 2002. On the need for time series data mining benchmarks: a survey and empirical demonstration. In *KDD*. 102–111.

[42] Patrick Kidger, James Foster, Xuechen Li, and Terry J Lyons. 2021. Neural SDEs as Infinite-Dimensional GANs. In *ICML*. 5453–5463.

[43] Diederik P. Kingma and Prafulla Dhariwal. 2018. Glow: Generative Flow with Invertible 1x1 Convolutions. In *NeurIPS*. 10236–10245.

[44] Diederik P Kingma, Danilo J Rezende, Shakir Mohamed, and Max Welling. 2014. Semi-supervised learning with deep generative models. In *NIPS*. 3581–3589.

[45] Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013).

[46] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. 2018. Modeling Long-and Short-Term Temporal Patterns with Deep Neural Networks. In *SIGIR*. 95–104.

[47] Kwei-Herng Lai, Daochen Zha, Junjie Xu, Yue Zhao, Guanchu Wang, and Xia Hu. 2021. Revisiting Time Series Outlier Detection: Definitions and Benchmarks. In *NeurIPS datasets and benchmarks track (round 1)*.

[48] Daesoo Lee, Sara Malacarne, and Erlend Aune. 2023. Vector Quantized Time Series Generation with a Bidirectional Prior Model. In *AISTATS*. 7665–7693.

[49] Guozhong Li, Byron Choi, Jianliang Xu, Sourav S Bhowmick, Daphne Ngar-yin Mah, and Grace LH Wong. 2022. IPS: Instance Profile for Shapelet Discovery for Time Series Classification. In *ICDE*. 1781–1793.

[50] Hongming Li, Shujian Yu, and Jose Principe. 2023. Causal Recurrent Variational Autoencoder for Medical Time Series Generation. In *AAAI*. 8562–8570.

[51] Xiaomin Li, Vangelis Metsis, Huangyingrui Wang, and Anne Hee Hiong Ngu. 2022. Tts-gan: A transformer-based time-series generative adversarial network. In *AIME*. 133–143.

[52] Haksoo Lim, Minjung Kim, Sewon Park, and Noseong Park. 2023. Regular Time-series Generation using SGM. *arXiv preprint arXiv:2301.08518* (2023).

[53] Zinan Lin, Alankar Jain, Chen Wang, Giulia Fanti, and Vyas Sekar. 2020. Using GANs for Sharing Networked Time Series Data: Challenges, Initial Promise, and Open Questions. In *IMC*. 464–483.

[54] Yuansan Liu, Sudanthi Wijewickrema, Ang Li, and James Bailey. 2022. Time-Transformer AAE: Connecting Temporal Convolutional Networks and Transformer for Time Series Generation. (2022).

[55] Spyros Makridakis, Evangelos Spiliotis, et al. 2021. The M5 competition and the future of human expertise in forecasting. *Foresight: The International Journal of Applied Forecasting* 60 (2021), 33–37.

[56] Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. 2018. The M4 Competition: Results, findings, conclusion and way forward. *International Journal of Forecasting* 34, 4 (2018), 802–808.

[57] Wannes Meert, Kilian Hendrickx, Toon Van Craenendonck, Pieter Robberechts, Hendrik Blockeel, and Jesse Davis. 2020. *DTAIDistance*.

[58] Olof Mogren. 2016. C-RNN-GAN: A continuous recurrent neural network with adversarial training. In *Constructive Machine Learning Workshop (CML) at NIPS 2016*. 1.

[59] Hao Ni, Lukasz Szpruch, Marc Sabate-Vidales, Baoren Xiao, Magnus Wiese, and Shujian Liao. 2021. Sig-Wasserstein GANs for time series generation. In *Proceedings of the Second ACM International Conference on AI in Finance*. 1–8.

[60] Hao Ni, Lukasz Szpruch, Magnus Wiese, Shujian Liao, and Baoren Xiao. 2020. Conditional Sig-Wasserstein GANs for Time Series Generation. *arXiv preprint arXiv:2006.05421* (2020).

[61] Alexander Nikitin, Letizia Iannucci, and Samuel Kaski. 2023. TSGM: A Flexible Framework for Generative Modeling of Synthetic Time Series. *arXiv preprint arXiv:2305.11567* (2023).

[62] Alan V Oppenheim. 1999. *Discrete-time signal processing*. Pearson Education India.

[63] John Paparrizos, Paul Boniol, Themis Palpanas, Ruey S Tsay, Aaron Elmore, and Michael J Franklin. 2022. Volume under the surface: a new accuracy evaluation measure for time-series anomaly detection. *Proc. VLDB Endow.* 15, 11 (2022), 2774–2787.

[64] John Paparrizos, Yuhao Kang, Paul Boniol, Ruey S Tsay, Themis Palpanas, and Michael J Franklin. 2022. TSB-UAD: an end-to-end benchmark suite for univariate time-series anomaly detection. *Proc. VLDB Endow.* 15, 8 (2022), 1697–1711.

[65] Emanuel Parzen. 1963. On spectral analysis with missing observations and amplitude modulation. *Sankhyā: The Indian Journal of Statistics, Series A* (1963), 383–392.

[66] Hengzhi Pei, Kan Ren, Yuqing Yang, Chang Liu, Tao Qin, and Dongsheng Li. 2021. Towards generating real-world time series data. In *ICDM*. 469–478.

[67] Mohamed Ragab, Emadeldeen Eldele, Wee Ling Tan, Chuan-Sheng Foo, Zhenghua Chen, Min Wu, Chee-Keong Kwoh, and Xiaoli Li. 2023. ADATIME: A Benchmarking Suite for Domain Adaptation on Time Series Data. *TKDD* (2023), 1–18.

[68] Giorgia Ramponi, Pavlos Protopapas, Marco Brambilla, and Ryan Janssen. 2018. T-CGAN: Conditional Generative Adversarial Network for Data Augmentation in Noisy Time Series with Irregular Sampling. *arXiv preprint arXiv:1811.08295* (2018).

[69] Sangeeta Rani and Geeta Sikka. 2012. Recent Techniques of Clustering of Time Series Data: A Survey. *International Journal of Computer Applications* 52, 15 (2012).

[70] Carl Remlinger, Joseph Mikael, and Romuald Elie. 2022. Conditional Loss and Deep Euler Scheme for Time Series Generation. In *AAAI*, Vol. 36. 8098–8105.

[71] Jorge Reyes-Ortiz, Davide Anguita, Alessandro Ghio, Luca Oneto, and Xavier Parra. 2012. Human Activity Recognition Using Smartphones. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C54S4K.

[72] Oliver Roesler. 2013. EEG Eye State. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C57G7J.

[73] Yulia Rubanova, Ricky T. Q. Chen, and David K Duvenaud. 2019. Latent Ordinary Differential Equations for Irregularly-Sampled Time Series. In *NeurIPS*. 5320–5330.

[74] Ali Seyfi, Jean-François Rajotte, and Raymond T. Ng. 2022. Generating multivariate time series with COmmon Source CoordInated GAN (COSCI-GAN). In *NeurIPS*. 32777–32788.

[75] Mohammad Shokoohi-Yekta, Bing Hu, Hongxia Jin, Jun Wang, and Eamonn Keogh. 2017. Generalizing DTW to the multi-dimensional case requires an adaptive approach. *DMKD* 31 (2017), 1–31.

[76] Kaleb E Smith and Anthony O Smith. 2020. Conditional GAN for timeseries generation. *arXiv preprint arXiv:2006.16477* (2020).

[77] Padmanaba Srinivasan and William J Knottenbelt. 2022. Time-series Transformer Generative Adversarial Networks. *arXiv preprint arXiv:2205.11164* (2022).

[78] Maksim Terpilowski. 2019. scikit-posthocs: Pairwise multiple comparison tests in Python. *The Journal of Open Source Software* 4, 36 (2019), 1169.

[79] Aäron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. 2017. Neural Discrete Representation Learning. In *NIPS*. 6306–6315.

[80] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing Data using t-SNE. *JMLR* 9, 11 (2008).

[81] Lei Wang, Liang Zeng, and Jian Li. 2023. AEC-GAN: Adversarial Error Correction GANs for Auto-Regressive Long Time-Series Generation. In *AAAI*. 10140–10148.

[82] Magnus Wiese, Robert Knobloch, Ralf Korn, and Peter Kretschmer. 2020. Quant GANs: deep generation of financial time series. *Quantitative Finance* 20, 9 (2020), 1419–1440.

[83] Garrett Wilson, Janardhan Rao Doppa, and Diane J Cook. 2020. Multi-source deep domain adaptation with weak supervision for time-series sensor data. In *KDD*. 1768–1778.

[84] Xinle Wu, Dalin Zhang, Chenjuan Guo, Chaoyang He, Bin Yang, and Christian S Jensen. 2021. AutoCTS: Automated correlated time series forecasting. *Proc. VLDB Endow.* 15, 4 (2021), 971–983.

[85] Tianlin Xu, Li Kevin Wenliang, Michael Munn, and Beatrice Acciaio. 2020. COT-GAN: Generating Sequential Data via Causal Optimal Transport. In *NeurIPS*. 8798–8809.

[86] Chao Yan, Yao Yan, Zhiyu Wan, Ziqi Zhang, Larsson Omberg, Justin Guinney, Sean D Mooney, and Bradley A Malin. 2022. A Multifaceted benchmarking of synthetic electronic health record generation models. *Nature Communications* 13, 1 (2022), 7609.

[87] Jinsung Yoon, Daniel Jarrett, and Mihaela van der Schaar. 2019. Time-series Generative Adversarial Networks. In *NeurIPS*. 5509–5519.

[88] Yu Zheng, Xiuwen Yi, Ming Li, Ruiyuan Li, Zhangqing Shan, Eric Chang, and Tianrui Li. 2015. Forecasting fine-grained air quality based on big data. In *KDD*. 2267–2276.

[89] Linqi Zhou, Michael Poli, Winnie Xu, Stefano Massaroli, and Stefano Ermon. 2023. Deep Latent State Space Models for Time-Series Generation. In *ICML*. 42625–42643.