TAB: Unified Benchmarking of Time Series Anomaly Detection Methods

Xiangfei Qiu
East China Normal University, China

Zhe Li
East China Normal University, China

Wanghui Qiu
East China Normal University, China

Shiyan Hu
East China Normal University, China

Lekui Zhou
Huawei Cloud Availability Engineering Lab, China

Xingjian Wu
East China Normal University, China

Zhengyu Li
East China Normal University, China

Chenjuan Guo
East China Normal University, China

Aoying Zhou
East China Normal University, China

Zhenli Sheng
Huawei Cloud Availability Engineering Lab, China

Jilin Hu
East China Normal University, China ✉

Christian S. Jensen
Aalborg University, Denmark

Bin Yang
East China Normal University, China

ABSTRACT

Time series anomaly detection (TSAD) plays an important role in many domains such as finance, transportation, and healthcare. With the ongoing instrumentation of reality, more time series data will be available, leading also to growing demands for TSAD. While many TSAD methods already exist, new and better methods are still desirable. However, effective progress hinges on the availability of reliable means of evaluating new methods and comparing them with existing methods. We address deficiencies in current evaluation procedures related to datasets and experimental settings and protocols. Specifically, we propose a new time series anomaly detection benchmark, called TAB. First, TAB encompasses 29 public multivariate datasets and 1,635 univariate time series from different domains to facilitate more comprehensive evaluations on diverse datasets. Second, TAB covers a variety of TSAD methods, including Non-learning, Machine learning, Deep learning, LLM-based, and Time-series pre-trained methods. Third, TAB features a unified and automated evaluation pipeline that enables fair and easy evaluation of TSAD methods. Finally, we employ TAB to evaluate existing TSAD methods and report on the outcomes, thereby offering a deeper insight into the performance of these methods.

PVLDB Reference Format:
Xiangfei Qiu, Zhe Li, Wanghui Qiu, Shiyan Hu, Lekui Zhou, Xingjian Wu, Zhengyu Li, Chenjuan Guo, Aoying Zhou, Zhenli Sheng, Jilin Hu, Christian S. Jensen and Bin Yang. TAB: Unified Benchmarking of Time Series Anomaly Detection Methods. PVLDB, 18(9): 2775 - 2789, 2025.
doi:10.14778/3746405.3746407

PVLDB Artifact Availability:

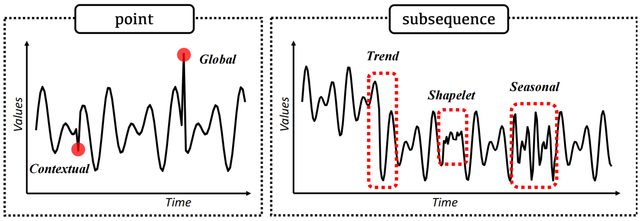


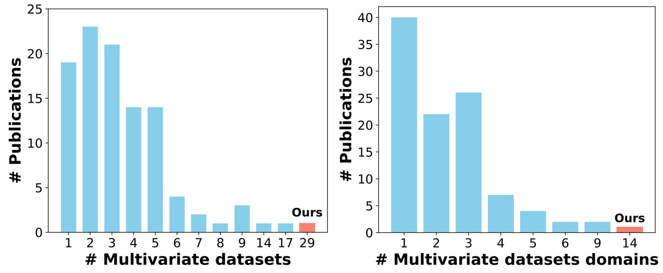
Figure 1: Types of time series anomalies.

The source code, data, and/or other artifacts have been made available at https://github.com/decisionintelligence/TAB.

1 INTRODUCTION

Due to the ongoing digitalization and instrumentation of processes with sensors, time series data is collected in numerous settings and fuels an increasing number of applications [59, 63, 79, 117, 119, 130]. In recent years, there has been significant progress in time series analysis, with key tasks such as forecasting [39, 60, 77, 120, 129, 131, 132], classification [13, 15, 126], and imputation [28, 104, 106, 108, 109], among others [40, 105, 107, 110], gaining attention. Among these, Time Series Anomaly Detection (TSAD), which aims to identify data in time series that derive from different processes than the intended processes, stands out as a critical studied task. The objective is thus to invent a *detector* that can detect the data that deviates from the expected or normal behavior [14, 38, 102]. For example, financial firms aim to identify unusual transactions in time to prevent financial fraud, and manufacturing companies are eager to predict the impending malfunction of devices so that they can be replaced before they fail. Time series anomalies can be classified into **point** and **subsequence** anomalies [51, 135]. As is shown in Figure 1, **point** anomalies can be further classified as *contextual* or *global* anomalies, while **subsequence** can be classified as *trend*, *shapelet*, or *seasonal* anomalies.

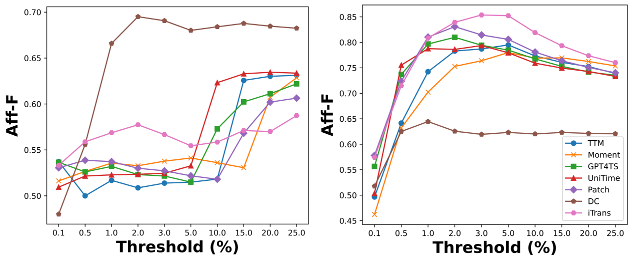
Proceedings of the VLDB Endowment, Vol. 18, No. 9 ISSN 2150-8097.
doi:10.14778/3746405.3746407



(a) Number of datasets

(b) Number of datasets domains

Figure 2: limits in multivariate datasets.



(a) SMAP

(b) PSM

Figure 3: Threshold problem description.

Recently, a variety of TSAD methods have been proposed. These methods include non-learning methods [12, 30, 127], machine learning methods [56, 81, 86], deep learning methods [103, 123, 125], LLM-based methods [62, 140], and Time-series pre-trained (TS pre-trained) methods [29, 31, 65]. According to the degree of supervision, TSAD methods can also be divided into unsupervised, semi-supervised, and supervised learning methods [14]. More specifically, unsupervised learning methods do not require labeled anomalies for training, semi-supervised learning methods are trained on time series without anomalous data points, and supervised learning methods are trained on time series with all data points labeled as normal or abnormal. In reality, anomalies are rare, resulting in labeled data being rare, so supervised learning methods are seldom used in TSAD [85]. Therefore, we focus on the evaluation of unsupervised and semi-supervised methods.

Time series include different types of anomalies, and it is difficult to invent a method that is good at detecting all types of anomalies [85]. We have surveyed a total of 100 studies[1] that are published in recent years, yet we found that the existing works fall short as follows.

First, *few of the most recent studies include a performance study that covers a broad variety of datasets.* Figure 2a shows the numbers of multivariate datasets included in the existing TSAD studies. We observe that more than half of the studies include at most four datasets, and only one study covers 17 datasets. When considering the domains of datasets, we find that the number of domains is also limited—see Figure 2b. The limited numbers of datasets and domains serve to question whether the proposed methods were evaluated comprehensively [19].

Second, *few of the most recent studies adopt consistent experimental settings and evaluation protocols, thus failing to offer reliable results to real-world applications.* Specifically, inconsistencies are observed in relation to the following aspects: (i) *Inconsistent Dataset Splitting.* Some methods use a validation set from the testing set [123, 125], while others use a validation set from the training set [102, 115], resulting in inconsistent dataset splitting approaches. (ii) *Drop-last Issue.* To accelerate the testing phase, it is common to split the data into batches. Unless all methods use the same batch size, discarding the last incomplete batch with fewer sample instances than the batch size is unfair, as the actual usage length of the testing set is inconsistent. [78, 90]. (iii) *Point Adjustment.* This is a technique for modifying predictions to complement the point-wise F1-scores. Any anomaly window with at least one correctly predicted time step is considered predicted correctly [122]. However, even random methods have a good chance to predict at least one point in larger anomaly windows, the effect being that they can easily reach the performance of complex methods or even outperform them [47]. Additionally, threshold-independent evaluation metrics such as *AUC-ROC* should not be recalculated after point adjustment. (iv) *Threshold Problem.* When calculating some label-based metrics (e.g., *precision*, *recall*), methods that output anomaly scores need to select a threshold to convert anomaly scores into anomaly labels. Then, as is shown in Figure 3, when the threshold changes, the performance can change considerably. Therefore, different threshold choices during the evaluation of the same dataset can impact performance comparisons. (v) *Inconsistent Algorithm Outputs.* For instance, TimesNet [115] chooses overlapping windows during testing, while DCdetector [125], Anomaly Transformer [123], and others choose non-overlapping windows. This difference in output lengths for different methods on the same dataset affects the direct comparison of method performance.

Third, *recent studies have rarely systematically evaluated the performance of foundation methods, particularly time series pre-trained methods in TSAD.* As mentioned above, the types of anomalies in time series are diverse, encompassing point anomalies (global and contextual), subsequence anomalies (seasonal, trend, and shapelet), as well as mixed types. Recall the "No-Free-Lunch Theorem," stating that improving a method for one aspect leads to deterioration of another aspect [114]. This implies that no single TSAD method is universally best for all time series and anomaly types. Therefore, it is challenging to select a suitable method for a given time series. Foundation methods which include LLM-based and time series pre-trained methods are widely recognized for their strong generalization capabilities and may help address the limitations of specific methods[2], which often struggle with generalization and fail to handle all types of anomaly data effectively. However, recent studies have seldom conducted systematic evaluations of the performance of these foundational methods, nor have they extensively compared their practical effectiveness with more specific methods.

[1] For details on the reviewed studies, please refer to: https://github.com/decisionintelligence/TAB/blob/main/docs/paper_statistics.csv.

[2] Most existing TSAD methods require training on specific datasets before they can perform inference on corresponding datasets. In this paper, these methods are referred to as "specific methods," to distinguish them from "foundation methods."

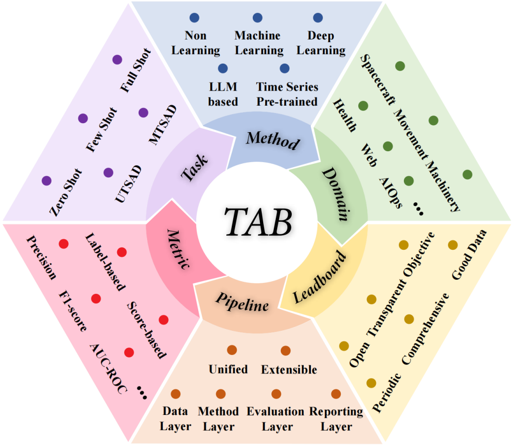


Figure 4: Key characteristics of TAB.

To address the three challenges covered above, we propose a **T**ime series **A**nomaly detection **B**enchmark (TAB). As shown in Figure 4, TAB has the following key characteristics:

(1) Comprehensive datasets (to address **Challenge 1**): TAB collects and filters 29 public multivariate datasets and 1,635 univariate time series, spanning diverse domains.

(2) Unified and extensible pipeline (to address **Challenge 2**): To enable fair and efficient comparison of both univariate and multivariate TSAD methods, TAB establishes a unified, user-friendly, and extensible evaluation pipeline, enabling different methods to undergo consistent performance evaluation.

(3) Broad coverage of existing methods (to address **Challenge 3**): TAB covers state-of-the-art TSAD methods, including Non-learning, Machine learning, Deep learning, LLM-based, and Time-series pre-trained methods.

(4) Multi evaluation strategies and tasks: TAB is specifically designed to further improve fair comparisons in TSAD, include univariate time series anomaly detection (UTSAD) and multivariate time series anomaly detection (MTSAD). Besides, it integrates zero-shot, few-shot, and full-shot evaluation strategies, facilitating improved assessment of method performance.

(5) Diversified evaluation metrics: TAB covers Label-based and Score-based metrics to ensure a comprehensive assessment of model performance.

(6) Continuously updated leaderboard: TAB has also launched an online TSAD leaderboard that covers LLM-based and time series pre-training methods. The leaderboard is open and transparent, and we welcome everyone to participate actively.

Based on the experiments conducted using TAB, we make the following key observations: (1) Non learning and machine learning methods exhibit strong performance on the UTSAD task, with KMeans, DWT, S2G, and, OCSVM performing the best. (2) On the MTSAD task, the deep learning and the machine learning methods demonstrate clear advantages, with methods such as TsNet, CATCH, DAGMM, KNN, and KMeans exhibiting excellent performance. (3) Global point anomalies are the easiest to detect, while mixed anomalies are the most challenging. (4) Deep learning as well as foundation methods are more suitable than other methods for finding point anomalies, while non learning and machine learning methods are better suited for finding subsequence anomalies. (5) Density based methods are appropriate for identifying trend anomalies, and distance based methods are suitable for identifying shapelet anomalies. (6) The application of foundation methods in TSAD needs further development, as their current performance remains relatively poor.

In summary, we make the following main contributions:

- We present TAB, a unified and extensible TSAD evaluation benchmark. TAB streamlines the process of evaluating TSAD methods and enables fair comparison of methods based on a unified experimental setup. Besides, all datasets and code are available at https://github.com/decisionintelligence/TAB. We have also launched an online leaderboard https://decisionintelligence.github.io/OpenTS/Benchmarks/overview/ that covers time series pre-trained and LLM-based methods.
- We survey recent TSAD studies, among which we select 29 multivariate datasets and 1,635 univariate time series. These datasets span various domains, and we classify them based on their characteristics and anomaly types. Besides, they are stored in a unified format for consistency and ease of use.
- We cover state-of-the-art time series anomaly detection methods, including non-learning, machine learning, deep learning, LLM-based, and time-series pre-trained methods.
- We use TAB to assess the performance of 40 multivariate TSAD methods on 29 multivariate datasets and 46 univariate TSAD methods on 1,635 univariate time series, covering diverse evaluation strategies and metrics. This provides deeper insights into the performance of these methods.

The rest of the paper is structured as follows. We first review related work in Section 2. In Section 3, we classify datasets, taking into consideration both the inherent characteristics of the datasets and the different types of anomalies. In Section 4, we cover the design of TAB, and we evaluate existing TSAD methods using TAB in Section 5. Finally, we conclude in Section 6.

2 RELATED WORKS

2.1 Time Series Anomaly Detection

We distinguish among five categories of time series anomaly detection methods: non-learning (NL), machine learning (ML), deep learning (DL), LLM-based, and Time-series pre-trained (TS pre-trained) methods.

Among the non-learning methods, classical methods such as local outlier factor (LOF) identify outliers in multidimensional datasets by using the notion of local density [12]. When the distances between data points are small or the data points are densely packed in dense datasets, LOF fails to perform well. HBOS [30] calculates anomaly scores based on histograms of feature values, which makes it easy to identify global anomalies but it is not good at solving contextual anomalies. Matrix Profile [127] aims to find anomalies by measuring the distance between subsequences of the same length. If this distance of a subsequence to the most similar subsequences is large, the subsequence is likely an anomaly. In summary, non-learning methods are designed to solve a specific anomaly, falling short in different kinds of anomalies.

Table 1: Time series anomaly detection benchmark comparison.

Benchmark Property		NAB [53]	Exathlon [44]	TODS [51]	TSB-UAD [74]	TSB-AD [61]	TimeEval [111]	TimeseAD [101]	Tadpak [47]	TimeSeries -Bench [92]	ADB [138]	MTAD [57]	OrionBench [4]	Experiment -TSAD [137]	**TAB (Ours)**
Evaluation Datasets	Real-Word	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	Synthetic	✓	✗	✓	✓	✓	✓	✗	✗	✓	✓	✗	✓	✓	✓
	Univariate	✓	✗	✓	✓	✓	✓	✗	✗	✓	✓	✗	✓	✓	✓
	Multivariate	✗	✓	✓	✗	✓	✓	✓	✓	✗	✓	✓	✓	✓	✓
	Dataset taxonomy	✗	✓	✓	✓	✓	✓	✗	✗	✓	✗	✗	✗	✓	✓
Evaluation Methods	Non-learning	✓	✗	✓	✓	✓	✓	✗	✓	✓	✓	✓	✓	✓	✓
	Machine-learning	✓	✗	✓	✓	✓	✓	✗	✗	✓	✓	✓	✓	✓	✓
	Deep-learning	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	LLM-based	✗	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗	✗	✓
	TS pre-trained	✗	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗	✗	✓
Evaluation Strategies	Zero-shot	✗	✗	✗	✗	✗	✗	✗	✗	✓	✗	✗	✗	✗	✓
	Few-shot	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓
	Full-shot	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Evaluation Metrics	Score-based	✗	✓	✗	✓	✓	✓	✓	✗	✓	✓	✗	✗	✗	✓
	Label-based	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
List inconsistent phenomena		✗	✗	✗	✓	✗	✗	✓	✗	✗	✗	✓	✗	✓	✓
Accuracy, memory, and runtime trade-off		✗	✓	✗	○	○	✓	✗	✗	✓	✗	✓	○	○	✓
Method recommendations		✗	✗	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗	✓	✓
Unified and extensible pipeline		✓	○	○	✓	✓	✓	○	✗	✓	✓	○	✓	○	✓
Continuously updated leaderboard		✗	✗	✗	✗	✓	✗	✗	✗	✓	✗	✗	✓	✗	✓

✗ indicates absent, ✓ indicates present,○ indicates incomplete.

Representative works in machine learning methods include Isolation Forest (IF) [56], K-Nearest Neighbors (KNN) [81], One-Class SVM (OCSVM) [86], among others. IF assumes that anomalies have low density and can be easily separated based on tree models with a single split, while normal values have high density and require multiple splits. So, IF can process high-dimensional data efficiently but it struggles with datasets where anomalies are densely packed or with varying densities. KNN defines anomalies as the top-k data points with the largest sum of distances to their k nearest neighbors. OCSVM maps the data into a feature space according to a kernel and separates it from the origin with maximum margin.

Recently, studies have shown that learned features may perform better than human-designed features [35, 121, 133, 134, 139]. Leveraging the representation learning of deep neural networks (DNNs), many deep learning-based methods emerge. These include TimesNet [115], Anomaly Transformer [123], DCdetector [125], among others. Anomaly Transformer proposes a minimax strategy to amplify the normal-abnormal distinguishability of the Association Discrepancy and use a new Anomaly-Attention mechanism to compute the association discrepancy to identify anomalies [123]. DCdetector introduces contrastive learning into anomaly detection tasks [125]. Its objective is to create an embedding space where similar data samples (normal values) are close to each other, while dissimilar data samples (abnormal values) are far away from each other. Different from non-learning and machine learning methods, deep learning methods do not focus on a specific anomaly only, but leverage the learned latent space to distinguish different anomalies.

LLM-based methods leverage the strong representational capacity and sequential modeling capability of LLMs to capture complex patterns for time series modeling. GPT4TS [140] selectively adjust specific parameters such as positional encoding and layer normalization of LLMs, enabling the model to quickly adapt to time series while retaining most of pre-trained knowledge. Besides, methods such as UniTime [62] focuses on designing prompts, such as learnable prompts, prompt pools, and domain-specific instructions to activate time series knowledge in LLMs.

Recently, pre-training on multi-domain time series data has garnered significant attention. Reconstruction-based methods, such as UniTS [29] and Moment [31], focus on restoring the features of time series data, enabling the extraction of valuable information in an unsupervised manner. These methods primarily employ an encoder architecture. In contrast, autoregressive-based methods like Timer [65] utilize next-token prediction to learn time series representations, typically adopting a decoder architecture. Direct prediction methods, such as TTM [22], unify the training process for pre-training and downstream tasks, allowing models to demonstrate strong adaptability when transitioning to forecasting tasks.

Even though tremendous TSAD methods have been proposed, consistently evaluating and comparing them remains a challenge.

2.2 Time Series Anomaly Detection Benchmarking

In a field where new methods are proposed frequently, accurate assessment of method performance is crucial. A comprehensive benchmark is important for advancing the state-of-the-art by offering researchers the opportunity to evaluate their methods rigorously across diverse datasets [21, 54, 78, 87]. In the field of TSAD, several benchmarks have been published. However, these benchmarks fall short in different aspects—see Table 1.

First, some benchmarks consider either univariate or multivariate TSAD. NAB [53], TSB-UAD [74], and TimeSeriesBench [92] have only done excellent work in univariate TSAD. And other works, e.g., Exathlon [44], TimeseAD [101], Tadpak [47], and MTAD [57] just consider multivariate. In addition, most benchmarks fail to evaluate both real-world and synthetic datasets, and they do not perform fine-grained classification on the datasets (for example, by dataset characteristics or anomaly types in the datasets).

Second, a notable issue arises regarding the assessment of diversity of evaluation methods and strategies. Except for NAB [53], Exathlon [44], TimeseAD [101], and Tadpak [47], other benchmarks include NL, ML, and DL methods, which make the evaluation results more comprehensive. However, apart from TSB-AD [61], none of these benchmarks systematically assess the recently popular foundation methods (LLM-based and time series pre-trained methods). A fair and systematic comparison of the performance among these foundation methods, as well as their performance compared to specific methods, is crucial for advancing the future direction of TSAD. Additionally, existing benchmarks have not sufficiently focused on the generalization ability (i.e., inference capability on new or unseen data) of methods. Except for TimeSeriesBench [92], which evaluates the method's zero-shot capability, all other benchmarks focus solely on the method's full-shot ability, neglecting comparisons and evaluations of generalization abilities.

Third, some benchmarks lack extensible pipelines. Tadpak [47] does not provide a pipeline in their benchmark at all, making it hard to integrate new methods. Exathlon [44], TODS [51], TSB-UAD [74], TimeseAD [101], and MTAD [57] lack flexible interfaces for datasets, methods, and evaluation metrics, supporting only limited functionality. Consequently, these benchmarks lack sufficient extensibility, making it difficult to: i) support evaluations on new (proprietary) datasets, ii) effectively assess the performance of innovative methods using existing metrics, and iii) adapt to new evaluation metrics.

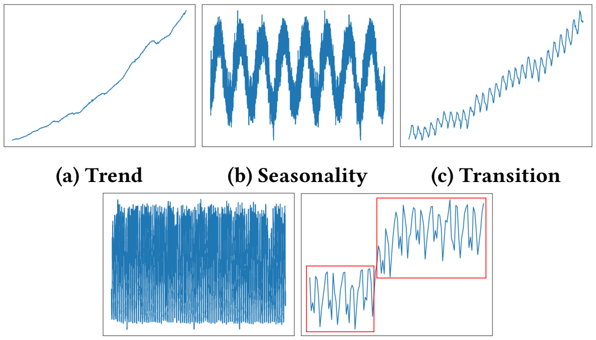
Unlike existing benchmarks, TAB provides a comprehensive evaluation of univariate and multivariate datasets, covering state-of-the-art TSAD methods, including NL, ML, DL, LLM-based, and TS pre-training methods. TAB evaluates the performance of 11 foundation methods on multivariate and univariate TSAD, and it covers three evaluation strategies: zero-shot, few-shot, and full-shot. TAB is the most comprehensive benchmark in terms of the number of foundation methods evaluated and the types of strategies covered. TAB offers comprehensive evaluation strategies and metrics, enabling a more accurate performance assessment over various methods and improvement fostering. TAB systematically organizes and unifies numerous inconsistencies found in current evaluations, analyzing the relationship among method accuracy performance, memory usage, and runtime. TAB also categorizes datasets to guide the selection of appropriate anomaly detection methods. Our goal is to deliver a fair and extensible pipeline for comprehensive assessments of diverse datasets and method performances. Additionally, TAB dynamically maintains a leaderboard, contributing consistently to the open-source community.

3 HETEROGENEITY AMONG TSAD DATASETS

This section classifies datasets by taking into consideration both the inherent characteristics of the datasets themselves (in Section 3.1) and the types of anomalies they contain (in Section 3.2).

3.1 Time Series Characteristics

In real-world scenarios, time series exhibit diverse characteristics, such as *trend*, *seasonality*, *shifting*, *transition*, and *stationarity*, which are illustrated in Figure 5. *Trend* refers to the long-term changes or patterns that occur over time and represents the change modality of the data. *Seasonality* indicates changes in a time series that recur at specific intervals, with the mean of the time series being constant and the variance being finite for all observations. *Shifting* refers to alterations in the probability distribution over time. *Transition* captures the regular and identifiable fixed features present in a time series, such as the clear manifestation of trends, periodicity, or the simultaneous presence of both seasonality and trend. *Stationarity* refers to the mean of any observation in a time series is constant and the variance is finite for all observations. Code and pseudocode for computing these characteristics are available in our code repository.



(a) Trend **(b) Seasonality** **(c) Transition**

(d) Stationarity **(e) Shifting**

Figure 5: Visualization of data with different characteristics.

3.2 Types of Time Series Anomalies

As is shown in Figure 1, time series anomalies can be categorized into point anomalies and subsequence anomalies. Point anomalies include *global anomalies* and *contextual anomalies*, while subsequence anomalies encompass *shapelet anomalies*, *seasonal anomalies*, and *trend anomalies* [51]. Additionally, there are *fixed anomalies*, which are composed of a mixture of these anomalies. Global anomalies refer to points that deviate significantly from the rest of the data points, often appearing as spikes. Next, contextual anomalies are points that deviate from their context, defined as neighboring time points within specific ranges, representing small deviations in sequential data. Shapelet anomalies involve subsequences with dissimilar basic shapelets compared to the normal shapelet component of the sequence. Seasonal anomalies exhibit unusual seasonal patterns compared to the overall seasonality, maintaining similar basic shapelets and trends. Trend anomalies involve subsequences that bring about a significant shift in the trend of the time series, resulting in a permanent change in the mean of the data. While retaining basic shapelets and seasonality of normal patterns, trend anomalies undergo drastic alterations in slopes.

4 TAB: BENCHMARK DETAILS

We proceed to cover the design of the TAB benchmark. First, we outline the process of dataset collection and filtering in TAB and their statistical informations (Section 4.1). Second, we categorize the existing methods supported (Section 4.2). Third, we cover evaluation strategies and metrics (Section 4.3). Finally, we describe the full benchmark pipeline (Section 4.4).

4.1 Datasets

We equip TAB with a robust collection of 220 multivariate time series, which are derived from 29 multivariate datasets, along with 1,635 univariate time series that come from 15 univariate datasets, sourced from community literature, providing researchers with a solid reference foundation. All datasets are formatted consistently, ensuring ease of use. This comprehensive collection spans a wide range of domains and characteristics. Detailed values of

Table 2: Statistics of multivariate datasets.

Dataset	Domain	Dim	Avg. AR (%)	Avg Total Length	Avg Test Length	Series Count
MSL [42]	Spacecraft	55	5.88	132,046	73,729	1
SMAP [42]	Spacecraft	25	9.72	562,800	427,617	1
SWAN [5]	Space Weather	38	23.80	120,000	60,000	1
PSM [2]	Server Machine	25	11.07	220,322	87,841	1
SMD [93]	Server Machine	38	2.08	1,416,825	708,420	1
SWAT [67]	Water treatment	51	5.78	944,919	449,919	1
GECCO [69]	Water treatment	9	1.25	138,521	69,261	1
PUMP [24]	Water treatment	44	6.54	220,302	143,401	1
Creditcard [17]	Finance	29	0.17	284,807	142,404	1
CICIDS [88]	Web	72	1.28	170,231	85,116	1
KDDcup99 [99]	Web	3	0.21	288,535	143,763	2
CalIt2 [7]	Visitors flowrate	2	4.09	5,040	2,520	1
Genesis [100]	Machinery	18	0.31	16,220	12,616	1
SKAB [45]	Machinery	8	3.65	10,505	1,100	34
GHL [25]	Machinery	19	1.13	199,001	149,653	25
CATSv2 [26]	Machinery	17	2.60	172,500	144,028	4
Daphnet [7]	Movement	9	5.75	87,724	58,483	7
OPP [83]	Movement	248	1.75	27,518	21,313	2
NYC [16]	Transport	3	0.57	17,520	4,416	1
Exathlon [44]	Application Server	multi	9.72	67,952	52,147	30
ASD [55]	Application Server	19	1.55	12,848	4,320	12
TODS [51]	Synthetic	5	1.27	25,000	5,000	12
GutenTAG [112]	Synthetic	multi	0.75	20,000	10,000	48
MITDB [1]	Health	2	2.72	141,667	106,250	6
SVDB [32]	Health	2	3.14	110,133	86,373	6
LTDB [1]	Health	2	15.57	100,000	87,285	5
DLR [128]	Health	9	2.28	23,130	11,565	1
ECG [128]	Health	32	16.27	112,209	56,105	1
TAO [94]	Climate	3	8.33	10,000	8,000	12

AR: anomaly ratio, “multi”: the dimensionality varies across time series in the dataset.

Table 3: Statistics of univariate datasets.

Dataset	Domain	Avg. AR(%)	Avg Total Length	Avg Test Length	Series Count
GAIA	AIOps	1.26	9,776.8	8,799.3	184
ECG [68]	Health	4.89	229,990.9	206,991.8	22
IOPS [82]	Web	2.15	101,430.7	91,288.3	11
KDD21 [46]	Multiple	0.58	65,902.6	47,294.0	243
MGAB [97]	Mackey-Glass	0.20	100,000.0	90,000.0	6
NAB [3]	Multiple	9.84	7,114.8	3,557.0	45
YAHOO [52]	Multiple	0.63	1,570.0	784.8	346
NASA-MSL [9]	Spacecraft	3.85	5,166.6	2,896.8	22
NASA-SMAP [9]	Spacecraft	2.20	10,538.5	7,929.7	35
Daphnet [8]	Movement	3.38	12,342.9	11,108.6	21
GHL [25]	Machinery	0.43	200,001.0	180,001.0	1
Genesis [100]	Machinery	0.31	16,220.0	14,598.0	1
OPP [83]	Movement	4.12	31,358.6	28,223.2	462
SMD [93]	Server Machine	2.63	25,810.4	23,229.9	184
SVDB [32]	Health	4.87	230,400.0	207,360.0	52

AR: anomaly ratio.

key characteristics and anomaly types for both multivariate and univariate datasets, along with their classification, are available in our code repository. This initiative enhances accessibility, tackling challenges such as inconsistent formats, varied documentation, and the time-consuming process of dataset collection.

4.1.1 Multivariate time series. Table 2 lists statistics of the 29 public multivariate datasets from 14 domains. We observe that the range of feature dimensions varies from 2 to 248 and the sequence length changes from 5,040 to 1,416,825. Some datasets contain multiple multivariate time series. For instance, the Daphnet, SKAB, Exathlon, ASD, and TODS datasets contain 7, 34, 30, 12, and 12 multivariate time series, respectively. In the experiments, we report the average evaluation metrics across all time series for each dataset.

4.1.2 Univariate time series. Table 3 summarizes statistical information of the univariate datasets. The first seven datasets (split by horizontal line) are the most commonly used and are proposed for univariate time series originally. The remaining eight datasets are transformed from multivariate time series, which is consistent with TAB-UAD [74]. Specifically, for multivariate time series, we treat it as multiple univariate time series. We then run the AD method on each univariate time series separately and retain those univariate series where at least one method achieves AUC-ROC > 0.85. To ensure data quality, we proceed to filter according to the following two rules: 1) have no anomaly; 2) have anomaly ratio > 10% [85], resulting in a total of 1,635 time series.

Due to legacy issues, we employ datasets used extensively in existing studies. However, we note that there are concerns regarding the ground-truth anomaly labels in these datasets [116]. It thus remains of interest to address such data-quality issues.

4.2 Comparison Methods

To investigate the strengths and limitations of different TSAD methods, we evaluate a large number of methods, which can be classified into non-learning (NL), machine learning (ML), deep learning (DL), LLM-based, and Time-series pre-trained (TS pre-trained) methods. Overall, **46 methods** are included for a comprehensive UTSAD comparison. S2G, LSTi, DWT, ARIMA, SARIMA, ZMS, Stat, and SR are removed from MTSAD; thus, **40 methods** are used for MTSAD comparison. For conciseness, we omit their detailed descriptions and categorize them briefly based on their distinct technical approaches, which are reported in Table 4.

4.3 Evaluation Settings

4.3.1 Evaluation strategies. TAB supports multiple evaluation strategies, including *zero-shot*, *few-shot*, and *full-shot*—see Figure 6, which support the output of either anomaly scores or anomaly labels.

(1) The zero-shot evaluation only uses the test data to evaluate the generalization ability of foundation methods to new datasets, assessing whether the method has truly learned general knowledge from vast amounts of pre-training data.

(2) The few-shot evaluation only utilizes a subset of training data and full validation data for fine-tuning, reflecting the prediction performance in low-data learning scenarios. This approach assesses whether models can effectively generalize and reason with minimal data support.

(3) The full-shot evaluation utilizes full train data and validation data for fine-tuning. It evaluates the performance when utilizing all available data, revealing its upper-bound performance.

4.3.2 Evaluation metrics. The metrics supported by TAB can be divided into two categories: Label-based and Score-based. Label-based metrics depend on threshold parameters to convert anomaly scores into anomaly labels. Score-based metrics, on the other hand, evaluate the performance of the model based on the raw anomaly scores it generates. Please refer to Table 5 for an overview of these metrics. Please note that TAB computes all metrics to provide a complete picture of each method.

Utilizing a diverse set of evaluation metrics is crucial for a comprehensive assessment of model performance. Different metrics offer insights from various perspectives, allowing for a nuanced understanding of model capabilities. In addition to the evaluation metrics provided in the table, TAB offers excellent extensibility in

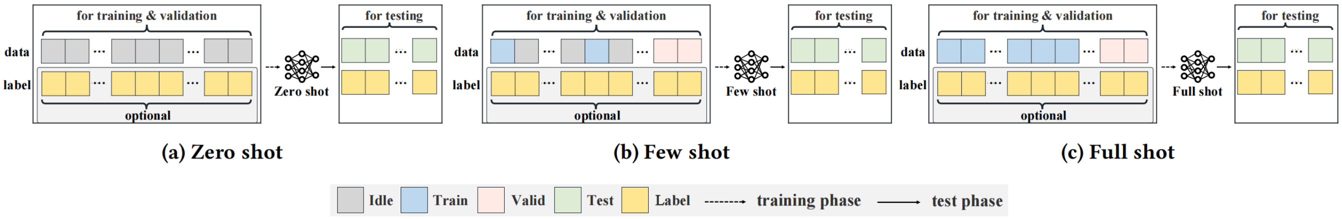


Figure 6: Evaluation strategies.

Table 4: Overview of comparison methods.

Method	Abbrev.	Area	Key Technology	UTAD	MTAD
▲ **LOF** [12]	LOF	Non Learning	Density	√	√
◓ **CBLOF** [36]	CBLOF		Clustering	√	√
● **HBOS** [30]	HBOS		Histogram	√	√
⬟ **ARIMA** [43]	ARIMA		Autoregressive + Moving Average	√	×
⬟ **SARIMA** [33]	SARIMA		Seasonal ARIMA	√	×
● **StatThreshold** [10]	Stat		Static Thresholding	√	×
● **DWT-MLEAD** [96]	DWT		Discrete Wavelet Transform	√	×
✚ **Left STAMPi** [127]	LSTi		Matrix Profile	√	×
✚ **ZMS** [10]	ZMS		Multiple z-score	√	×
⏢ **Series2Graph** [11]	S2G		Graph representation of time series	√	×
● **OCSVM** [86]	Ocsvm	Machine Learning	Classification	√	√
■ **DeepPoint** [10]	DP		Multilayer Perceptron	√	√
▲ **KNN** [81]	KNN		K-nearest Neighbors	√	√
◓ **KMeans** [124]	KMeans		K-Means	√	√
◆ **Isolation Forest** [56]	IF		Isolation Tree	√	√
◆ **EIF** [34]	EIF		Isolation Tree	√	√
■ **Spectral Residual** [82]	SR		Frequency Spectrum + Fourier Transform	√	×
● **LODA** [76]	LODA		Ensemble Learning	√	√
▼ **PCA** [91]	PCA		Dimensionality reduction + Restruction	√	√
■ **DAGMM** [142]	Dagmm	Deep Learning	Deep autoencoder + Gaussian Mixture	√	√
⬟ **Torsk** [37]	Torsk		Echo State Networks	√	√
■ **iTransformer** [64]	iTrans		Inverted Transformer	√	√
■ **TimesNet** [115]	TsNet		Temporal 2D-variations + Multi-periodicity	√	√
■ **DUET** [80]	DUET		Temporal Clustering + Channel Soft Clustering	×	√
■ **Anomaly Transformer** [123]	ATrans		Anomaly Attention + Association Discrepancy	√	√
■ **PatchTST** [72]	Patch		Channel Independent + Patchify	√	√
■ **ModernTCN** [66]	Modern		TCN with bigger receptive field	√	√
■ **TranAD** [98]	TranAD		Self-conditioning + Meta Learning + Two-Phase Adversarial Training	√	√
■ **DualTF** [70]	DualTF		Time-Frequency combination + Anomaly Transformer	√	√
■ **AE** [84]	AE		AutoEncoder	√	√
■ **VAE** [48]	VAE		AutoEncoder	√	√
■ **NLinear** [136]	NLin		Linear + Simple normalization	√	√
■ **DLinear** [136]	DLin		Linear + Decomposition	√	√
⬟ **LSTMED** [10]	LSTM		LSTM + Encoder-Decoder-based	√	√
⬢ **DCdetector** [125]	DC		Multi-scale Dual Attention + Contrastive Learning	√	√
⬢ **ContraAD** [141]	ConAD		Contrastive Learning	√	√
■ **CATCH** [118]	CATCH		Frequency Reconstruction + Channel Fusion	×	√
⬟ **GPT4TS** [140]	GPT4TS	LLM-based	Parameter-Efficient Fine-Tuning	√	√
■ **UniTime** [62]	UniTime		Domain Instructions + Masking Technique + Cross-Domain Training Strategy	√	√
⬟ **CALF** [58]	CALF		cross-modal Fine-Tuning + Parameter-Efficient Fine-Tuning	√	√
■ **LLMMixer** [49]	LLMMixer		Multi-scale time series decomposition + Prompt Embedding	√	√
⬟ **Timer** [65]	Timer	Pre-trained	Single-Series Sequence + Autoregressive	√	√
⬟ **TinyTimeMixer** [22]	TTM		Adaptive Patching + Exogenous Mixer + Resolution Prefix Tuning	√	√
⬟ **TimesFM** [18]	TimesFM		Patching + Autoregressive	√	√
■ **UniTS** [29]	UniTS		Dynamic Linear + Prompt Learning + Unified Masked Reconstruction	√	√
■ **Moment** [31]	Moment		Masking Technique + Lightweight Reconstruction Head	√	√
■ **DADA** [89]	DADA		Adaptive Bottlenecks + Dual Adversarial Decoders	√	√
⬟ **Chronos** [6]	Chronos		Time Series Tokenization + Data Augmentation + Probability Prediction	√	√

distance-based (**orange**), density-based (**red**), prediction-based (**blue**), contrast-based (**green**)
▲ proximity-based, ◓ clustering-based, ✚ discord-based, ● distribution-based, ⏢ graph-based, ◆ tree-based,
▼ encoding-based, ⬟ forecasting-based, ■ reconstruction-based, ⬢ contrast-based

its evaluation metrics, allowing users to easily and conveniently customize additional metrics to better meet specific needs.

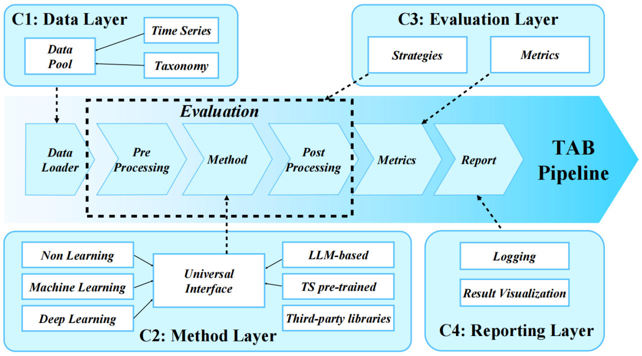


Figure 7: Pipeline of the TAB benchmark.

4.4 Unified Pipeline

To facilitate fair and comprehensive comparisons of methods, TAB proposes a unified pipeline—see Figure 7 where we first have a data layer that ensures all the input time series are in standardized formats. Thus, we can ensure the data pre-processing and splitting is the same for all the compared AD methods. Then, we have a method layer that can embed both self-implemented and third-party AD methods with a universal interface. Thus, all the compared AD methods are trained and tested in the same workflow. Finally, we have a evaluation layer that has a fixed post-processing such as changing anomaly score to anomaly label. With this layer, the results from all the AD methods are evaluated in the same way. With those three layers, we claim that TAB can ensure a fair comparison among different AD methods. Each component is detailed as follows.

(1) Data Layer. The data layer comprises both univariate and multivariate time series data from diverse domains. Its organization is carefully designed to categorize time series according to their distinctive characteristics, anomaly ratios, and anomaly types. To improve usability and comparability, time series data is stored in a unified format.

(2) Method Layer. The method layer is constructed elaborately to support different time series AD methods, i.e., non-learning, machine learning, and deep learning methods. To comply with other third-party time series AD libraries, e.g., TODS [50], Merlion [10], we design a simple universal interface, such that users can easily integrate third-party libraries into TAB to facilitate fast comparison.

(3) Evaluation Layer. The evaluation layer implements different evaluation strategies and metrics that are introduced in existing

Table 5: Overview of evaluation metrics supported in TAB.

Category	Metric	Abbreviation	Short summary
Label_based	Accuracy	Acc	Measures the proportion of correct predictions among the total number of predictions.
	Precision	P	Measures the proportion of true positive predictions among all positive predictions. Important when false positives are costly.
	Recall	R	Masures the proportion of true positive predictions among all actual positives. Important when false negatives are costly.
	F1-score	F1	A balanced metric when you need to account for both precision and recall.
	Range-Precision Range-Recall Range-F1-score	R-P [95] R-R [95] R-F1 [95]	These metrics' definitions consider several factors: the ratio of detected anomaly subsequences to the total number of anomalies, the ratio of detected point outliers to total point outliers, the relative position of true positives within each anomaly subsequence, and the number of fragmented prediction regions corresponding to one real anomaly subsequence.
	Affiliated-Precision Affiliated-Recall Affiliated-F1-score	Aff-P [41] Aff-R [41] Aff-F1 [41]	These metrics' definitions are the extension of the classical precision/recall/f1-score for time series anomaly detection that is local (each ground truth event is considered separately), parameter-free, and applicable generically on both point and subsequence anomalies. Besides the construction of these metrics makes them both theoretically principled and practically useful.
Score_based	AUC-PR	A-P [20]	Measures the area under the curve corresponding to Recall on the x-axis and Precision on the y-axis at various threshold settings.
	AUC-ROC	A-R [23]	Measures the area under the curve corresponding to FPR on the x-axis and TPR on the y-axis at various threshold settings.
	R-AUC-PR R-AUC-ROC	R-A-P [73] R-A-R [73]	They mitigate the issue that AUC-PR and AUC-ROC are designed for point-based anomaly detection, where each point is assigned equal weight in calculating the overall AUC. This makes them unsuitable for evaluating subsequence anomalies.
	VUS-PR VUS-ROC	V-PR [73] V-ROC [73]	VUS is an extension of the ROC and PR curves. It introduces a buffer region at the outliers' boundaries, thereby accommodating the false tolerance of labeling in the ground truth and assigning higher anomaly scores near the outlier boundaries.

studies. Moreover, it also provides customized metrics for a more comprehensive assessment of method performance.

(4) Reporting Layer. The reporting layer encompasses a logging system for tracking information to enable traceability of the experimental settings. Further, it encompasses a visualization module to facilitate a clear understanding of method performance.

Besides the pipeline is flexible in the following ways. 1) It can embed third-party libraries. 2) It supports diverse evaluation strategies that can cover existing common evaluation strategies. 3) It allows users to customize evaluation metrics and evaluation strategies. 4) It can select the dataset to be evaluated based on specified dataset characteristics or anomaly type, such as only evaluating datasets with trends. Besides the pipeline is scalable and we support large-scale parallel operations.

Moreover, TAB can comply with both CPU and GPU hardware to facilitate the evaluation in various computing environments. Additionally, it supports different modes of program execution (sequential and parallel), providing users with flexible choices.

To sum up, TAB serves as a benchmarking tool tailored for TSAD methods. We can swiftly evaluate a new method by integrating it into the model layer. Therefore, it can enhance users' understanding and assist in choosing TSAD methods suitable for specific application scenarios.

5 EXPERIMENTS

5.1 Experimental Setup

5.1.1 *Datasets and comparison methods.*

We utilize all the datasets included in TAB, which encompass 29 multivariate datasets and 1,635 univariate time series. More details are given in Tables 2 and 3. The anomaly types and characteristic types for both multivariate and univariate time series are available in our code repository.

5.1.2 *Implementation details.*

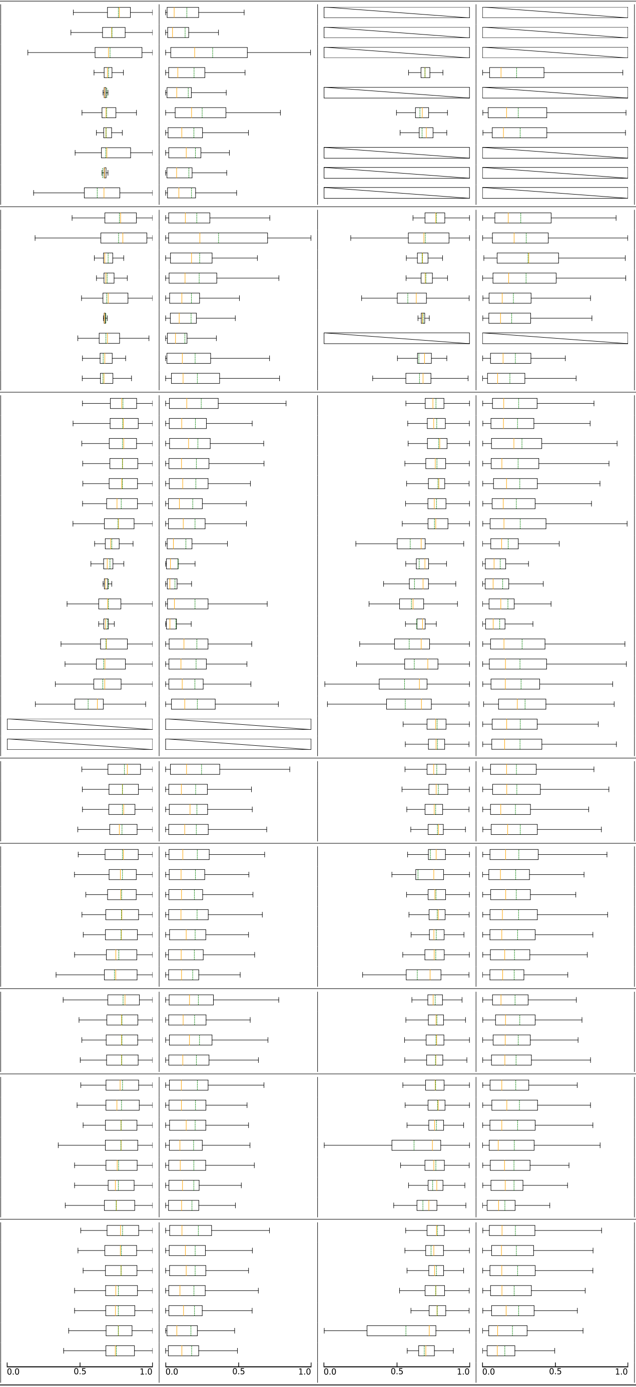
We unify all the inconsistencies mentioned in Section 1 and describe the process of hyperparameter tuning. In addition, we provide a clear explanation of the experimental environment to ensure the reproducibility of the experiments and the reliability of the results.

- **Inconsistent Dataset Splitting Issue:** Regarding dataset splitting, we follow the initial splitting provided by the raw data for both multivariate and univariate data. If the raw data does not include splitting information, we adopt a consistent splitting method, ensuring a relatively low anomaly rate in the training set. The training and validation set comprises 50% of the total data, and the test set accounts for 50%. Additionally, the last 20% of the training and validation set is extracted as the validation set. Details on "dataset level" data splitting for multivariate and univariate datasets are provided in Tables 2 and 3. The splitting details at the "time series level" are provided in the code repository.
- **Drop-last & Point Adjustment Issues:** During the testing and evaluation process, we abandon drop-last and point adjustment operations.
- **Different Thresholds Issue:** Since different TSAD datasets are sensitive to the threshold, to avoid the threshold problem as is mentioned in Section 1, we consider threshold the range [0.1, 0.5, 1, 2, 3, 5, 10, 15, 20, 25] for both multivariate and univariate time series. All the numbers are percentages. Then, we conduct metric calculations at all thresholds and report the best results.
- **Inconsistent Algorithm Outputs Issue:** If a algorithm requires the use of a rolling window during testing, we adopt a non-overlapping window rolling approach. Reasons can be found in Section 5.1.3.
- **Hyperparameter Tuning:** For each method, we adhere to the parameters specified in the original papers. Additionally, we perform a hyperparameter search over multiple sets to ensure fairness—see Table 4. We then select the best results from these evaluations to contribute to a comprehensive and fair assessment of each method. The results of evaluations are close to those reported in the original papers of each TSAD method.
- **Experiment Environment:** All experiments are conducted in Python 3.8 using PyTorch [75] and executed on an NVIDIA Tesla-A800 GPU. The training process is guided by L2 loss and utilizes the ADAM optimizer. Initially, the batch size is set to 256, and in cases of insufficient memory (OOM), it can be halved (minimum of 16). To ensure reproducibility and convenience in experimentation, both the datasets and code are available at https://github.com/decisionintelligence/TAB.

5.1.3 *Performance comparison of different post-processing methods.*

Table 7 presents the performance results of different methods using window overlapping post-processing methods and window

Table 6: Performance overview for all 48 methods: Box plots show the score distribution (mean in green and median in orange) for V-PR (VUS-PR) and Aff-F1 (Affiliated-F1) across all univariate (U) and multivariate (M) datasets, and along with training and inference time (seconds), CPU and GPU memory usage (GB). ▨ indicates that the operation is not executable.



Mode	Area	Method	U Aff-F1	U V-PR	M Aff-F1	M V-PR	Train. Time		Infer. Time		CPU		GPU	
							U	M	U	M	U	M	U	M
Full Shot	Non Learning	⛊S2G			▨	▨	0.000	▨	1.588	▨	0.720	▨	0.000	▨
		✚LSTi			▨	▨	0.000	▨	31.191	▨	0.786	▨	0.000	▨
		●DWT			▨	▨	0.000	▨	1.788	▨	0.721	▨	0.000	▨
		▲LOF					0.019	0.125	0.046	0.036	0.830	0.841	0.000	0.000
		⬟SARIMA			▨	▨	44.489	▨	1.929	▨	0.534	▨	0.000	▨
		●HBOS					3.526	3.580	0.007	0.009	0.853	0.864	0.000	0.000
		☗CBLOF					3.325	4.432	0.010	0.018	0.849	0.860	0.000	0.000
		✚ZMS			▨	▨	0.110	▨	0.063	▨	0.479	▨	0.000	▨
		⬟ARIMA			▨	▨	0.993	▨	0.710	▨	0.484	▨	0.000	▨
		●Stat			▨	▨	0.045	▨	0.015	▨	0.479	▨	0.000	▨
	Machine Learning	◆EIF					0.000	23.730	52.428	8.231	0.727	0.743	0.000	0.000
		☗KMeans					1.286	3.614	0.715	0.344	0.722	0.753	0.000	0.000
		●OCSVM					0.062	52.483	0.228	4.230	0.828	1.014	0.000	0.000
		▲KNN					0.017	0.077	1.151	0.678	0.827	0.834	0.000	0.000
		◆IF					0.556	0.222	0.080	0.038	0.481	0.490	0.000	0.000
		■DP					1.061	6.222	3.238	1.640	0.689	0.724	0.154	0.275
		■SR			▨	▨	0.056	▨	0.028	▨	0.480	▨	0.000	▨
		▼PCA					0.020	0.127	0.044	0.031	0.829	0.836	0.000	0.000
		●LODA					0.047	0.133	0.035	0.022	0.827	0.832	0.000	0.000
	Deep Learning	■iTrans					4.476	39.587	0.267	0.025	0.747	0.807	0.219	0.273
		■DLin					0.597	22.487	0.013	0.011	0.572	0.665	0.007	0.133
		■TsNet					4.089	6.343	0.252	0.126	0.918	0.989	0.426	0.505
		■Modern					2.705	3.892	0.218	0.017	0.947	0.981	0.235	0.130
		■Patch					1.449	12.001	0.027	0.027	0.675	0.800	0.206	0.523
		■NLin					0.752	21.245	0.012	0.010	0.587	0.661	0.033	0.131
		■TranAD					1.673	20.314	0.172	0.108	0.849	0.885	0.102	0.111
		■DualTF					163.936	381.367	155.892	35.221	0.981	1.176	18.564	2.797
		■ATrans					2.234	23.122	0.054	0.037	0.940	1.021	1.171	5.380
		⬢ConAD					24.001	271.416	0.216	0.133	0.987	0.994	16.199	19.917
		⬟Torsk					13.280	72.011	119.179	24.267	0.721	0.726	0.000	0.000
		⬢DC					3.854	13.655	0.044	0.029	0.957	0.949	0.274	0.821
		⬟LSTM					3.636	34.668	0.962	0.567	0.697	0.741	0.129	0.162
		■AE					1.886	70.892	0.629	0.362	0.667	0.751	0.072	0.111
		■VAE					1.814	17.637	0.941	0.452	0.683	0.743	0.065	0.112
		■DAGMM					6.838	74.846	33.441	19.729	0.828	0.847	0.109	0.111
		■CATCH	▨	▨			▨	51.000	▨	0.049	▨	1.373	▨	5.154
		■DUET	▨	▨			▨	38.778	▨	0.032	▨	1.049	▨	0.500
	LLM	⬟CALF					3.930	34.475	0.056	0.035	1.063	1.073	1.871	1.877
		⬟LLMMixer					6.698	45.499	0.079	0.060	1.069	1.080	1.213	1.211
		⬟GPT4TS					4.249	37.492	0.085	0.029	0.851	0.856	2.605	2.461
		■UniTime					6.343	393.426	0.059	0.040	0.971	1.006	2.873	8.285
	Pre-trained	⬟TimesFM					24.809	441.420	0.151	0.211	0.902	0.941	5.420	7.123
		■Dada					4.144	47.752	0.049	0.095	1.061	1.098	1.561	4.311
		■Moment					17.115	1195.658	0.094	0.102	1.144	1.165	4.291	5.201
		■UniTS					4.505	51.667	0.038	0.035	0.860	0.882	0.182	0.293
		⬟Chronos					17.714	1053.727	0.296	0.515	1.047	1.066	0.506	0.506
		⬟Timer					3.672	39.233	0.042	0.040	0.900	0.966	0.438	0.566
		⬟TTM					5.460	46.190	0.036	0.028	0.786	0.811	0.246	0.457
Few Shot	LLM	⬟GPT4TS					2.256	24.054	0.043	0.031	0.843	0.854	2.645	2.611
		■UniTime					3.511	27.959	0.060	0.033	0.968	0.995	2.537	8.197
		⬟CALF					3.022	10.602	0.052	0.040	1.008	1.064	0.604	1.416
		⬟LLMMixer					4.517	9.588	0.087	0.065	1.069	1.074	1.721	1.219
	Pre-trained	■UniTS					1.867	10.973	0.062	0.038	0.855	0.874	0.172	0.293
		⬟TimesFM					9.312	141.400	0.138	0.212	0.898	0.905	6.113	6.301
		⬟Chronos					4.440	252.742	0.342	0.474	1.040	1.046	0.506	0.506
		■Dada					2.675	13.647	0.098	0.050	1.059	1.088	0.867	4.029
		⬟Timer					1.780	8.662	0.041	0.042	0.895	0.958	0.438	0.566
		■Moment					13.525	239.935	0.098	0.101	1.142	1.147	4.271	5.201
		⬟TTM					3.144	10.436	0.037	0.032	0.785	0.821	0.205	0.457
Zero Shot	Pre-trained	■UniTS					0.000	0.000	0.362	0.354	0.769	0.773	0.129	0.108
		⬟TimesFM					0.000	0.000	1.295	2.025	0.819	0.856	0.924	1.594
		⬟Chronos					0.000	0.000	1.403	0.959	0.870	1.047	0.410	0.506
		⬟Timer					0.000	0.000	0.715	1.370	0.818	0.840	0.250	0.254
		■Moment					0.000	0.000	1.324	1.756	1.105	1.110	2.412	1.576
		■Dada					0.000	0.000	0.892	0.907	0.885	0.910	0.600	1.152
		⬟TTM					0.000	0.000	0.522	0.638	0.726	0.745	0.166	0.123

For training and inference time, as well as CPU and GPU memory usage, we respectively report the recorded results on the multivariate dataset NYC and the univariate dataset GAIA.
Classification from the methodological perspective: distance-based (**orange**), density-based (**red**), prediction-based (**blue**), contrast-based (**green**)
▲proximity-based, ☗clustering-based, ✚discord-based, ●distribution-based, ⛊graph-based, ◆tree-based, ▼encoding-based, ⬟forecasting-based, ■reconstruction-based, ⬢contrast-based

non overlapping post-processing methods on different datasets. We found that in most cases, the two post-processing methods do not affect the performance of the method, but there are still a small number of cases where the performance of the method is affected by the post-processing method. To keep consistent evaluation, we need to select either overlapping or non-overlapping for all the compared methods (non-learning, machine learning, and deep learning methods). However, some traditional methods cannot use window overlapping methods for anomaly detection. Moreover, non-overlapping method is widely used in existing works [123, 125]. Therefore, we finally have adopted window non-overlapping method.

Table 7: Average accuracy for 6 multivariate time series.

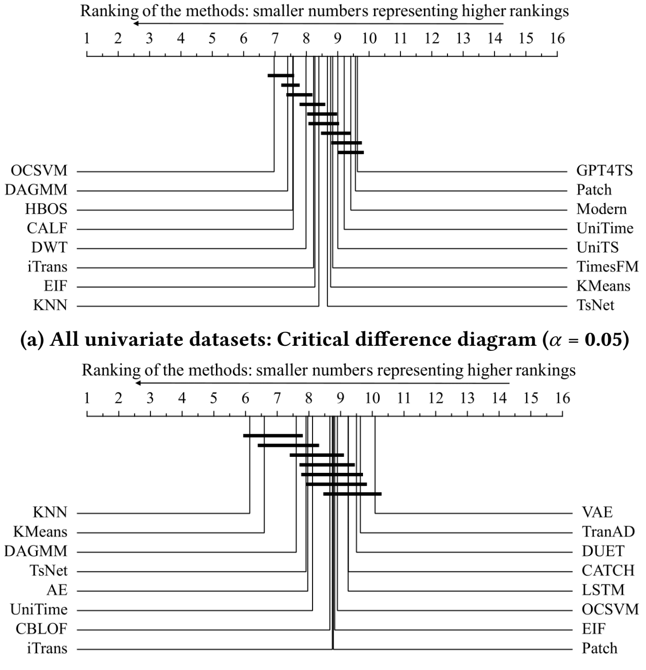
Dataset	Metric	ATrans	DC	DLin	NLin	Patch	TsNet
CalIt2	Overlap	0.483	0.499	0.761	0.694	0.791	**0.798**
	Non-overlap	0.491	0.527	0.752	0.695	**0.808**	0.771
Daphnet	Overlap	0.469	0.486	0.727	0.715	0.739	**0.773**
	Non-overlap	0.489	0.501	0.728	0.715	0.741	**0.754**
MSL	Overlap	0.494	0.502	0.626	0.594	**0.637**	0.615
	Non-overlap	0.508	0.504	0.624	0.592	**0.637**	0.613
PSM	Overlap	0.496	0.499	0.581	0.586	0.578	**0.589**
	Non-overlap	0.498	0.499	0.580	0.585	0.586	**0.592**
SKAB	Overlap	0.495	0.532	0.563	0.558	0.555	**0.592**
	Non-overlap	0.513	0.522	0.593	0.583	0.597	**0.620**
SMAP	Overlap	**0.504**	0.499	0.398	0.434	0.449	0.455
	Non-overlap	0.504	**0.516**	0.397	0.434	0.448	0.453

Bold represents the best result (AUC-ROC), double Underline represents the second best result, and single Underline represents the third best result.

5.2 Overall Performance

Due to space limitations, we only report the score distribution of two representative metrics, VUS-PR (V-PR) and Affiliated F1 (Aff-F1), in the main text. For results of other metrics, please refer: https://decisionintelligence.github.io/OpenTS/Benchmarks/overview/.

5.2.1 Univariate time series anomaly detection. The left part of Table 6 presents the results of V-PR and Aff-F1 for 46 methods across 1,635 univariate time series. We make four observations: 1) LLM-based and TS pre-trained methods achieve relatively stable average scores and lower bounds on Aff-F1, indicating that they have learned general patterns of time-series data during pre-training, leading to stable performance. Moreover, the differences across the full, few, and zero-shot settings are small, suggesting that the patterns in univariate time series are relatively simple, making them easier to fit and generalize. 2) Traditional DL methods exhibit performance closely tied to their compatibility with the datasets. For example, iTrans and DLin show higher average V-PR and Aff-F1 scores but have larger fluctuations, implying that they excel at detecting specific anomalies but perform less well on others. There are also methods, notably DualTF, ConAD, and DC, that seem to struggle with anomaly detection in univariate time series, leading to overall poor performance. 3) Machine learning (ML) and non-learning (NL) methods exhibit the best average performance in terms of the V-PR and Aff-F metrics. This not only reflects the meaning of traditional methods in theoretical foundations and practical applications but also highlights their significant value and competitiveness at solving modern problems. These results indicate that while pursuing novel methods, we should not overlook the classic methods. When combined with domain knowledge and real-world requirements, traditional methods often demonstrate remarkable performance. 4) When considering the different classes of methods, the prediction-based methods (such as CALF, iTrans, and UniTS) usually show relatively better performance than other classes on univariate TSAD tasks. Next flow the density-based methods (such as DWT, S2G, and EIF). The contrast-based methods (such as DC, ConAD) perform the weakest.



(a) All univariate datasets: Critical difference diagram (α = 0.05)

(b) All multivariate datasets: Critical difference diagram (α = 0.05)

Figure 8: Illustration of model performance (VUS-PR) on all univariate (a) and multivariate (b) datasets.

5.2.2 Multivariate time series anomaly detection. The right of Table 6 presents the results of 40 methods on the 29 public multivariate datasets. We have the following observations: 1) Under full-shot and few-shot learning scenarios, time series pre-trained models demonstrate significantly better performance compared to zero-shot learning approaches. This could be because multivariate time series are more complex, and during the pre-training phase, large models tend to capture only basic features and fail to capture the interactions between channels. As a result, fine-tuning leads to more significant improvements by allowing the model to learn these relationships more effectively. 2) Some deep learning models, such as TsNet and CATCH, appear to achieve the best performance in full-shot settings. This may be due to the significant impact of channel interactions in multivariate anomaly detection, so training a model with a reasonable number of parameters for specific datasets helps capture these interactions without overfitting, thereby outperforming even large pre-trained models in the full-shot setting. 3) While deep learning-based methods show promising results, methods based on Non-Learning and Machine Learning, such as KNN,

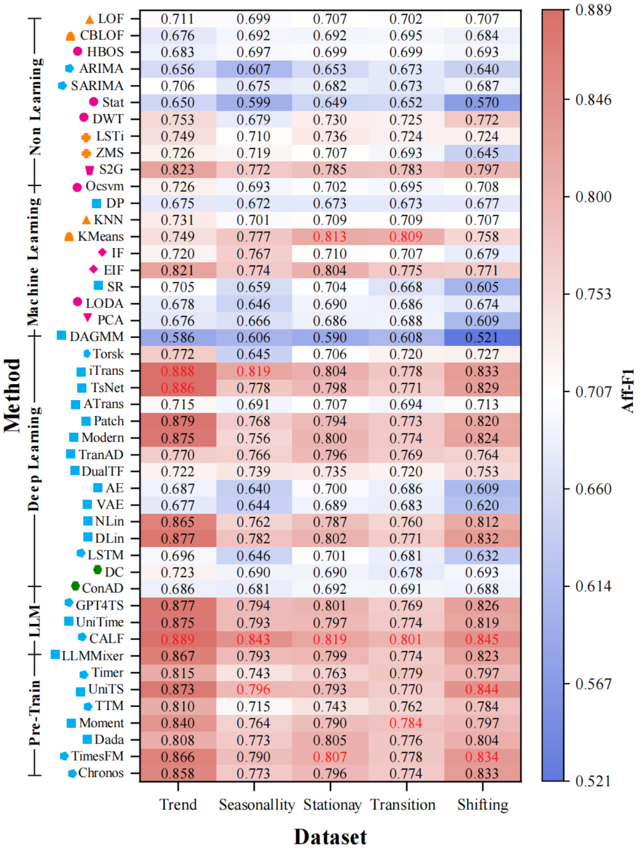


Figure 9: Classification results (Aff-F1) of univariate datasets based on characteristic.

KMeans, and OCSVM, demonstrate exceptional performance and stability in V-PR and Aff-F1. This suggests that there is still significant room for improvement in current deep learning approaches.

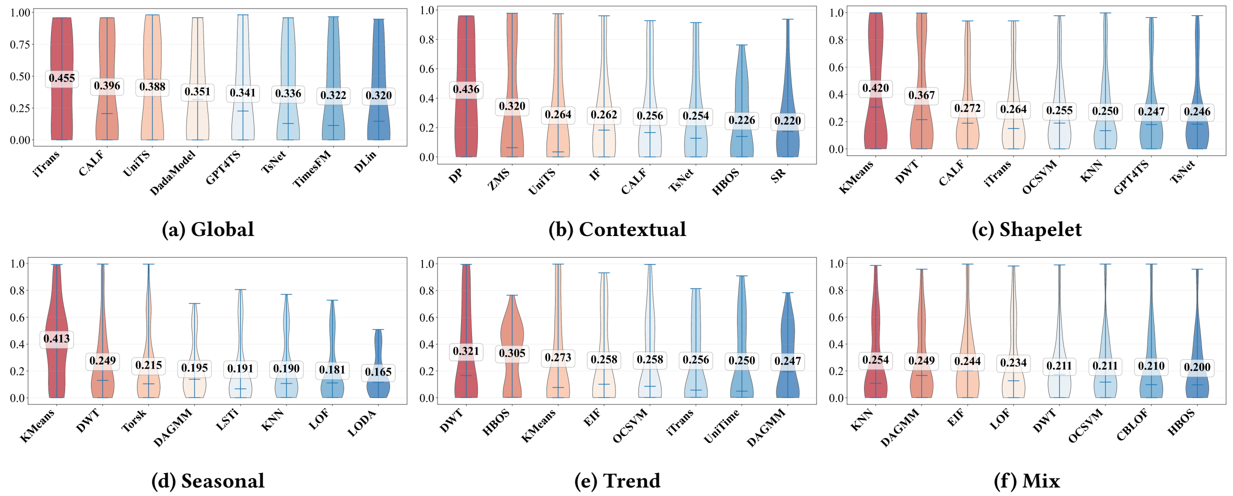
5.2.3 Statistical validation. To validate the statistical differences in the performance results for the automated solutions, we employ the Wilcoxon test [113] to obtain pairwise comparisons across datasets. Additionally, to compare multiple automated solutions across datasets, we use the Friedman test [27] followed by the post-hoc Nemenyi test [71] at a 95% confidence level. We plot the critical difference (CD) diagrams, where the numbers at the top indicate the ranking of the methods, with smaller numbers representing higher rankings. Methods without significant performance differences are connected by horizontal lines. Specifically, since we cover 48 methods, displaying them all would result in overly dense lines, which would reduce readability. Therefore, we now only present the top 16 ranked methods and have added a legend to ensure that the information is displayed more clearly and intuitively. As shown in Figure 8, the non learning and machine learning methods exhibit particularly strong performance on the univariate time series anomaly detection tasks, with OCSVM, HOBS, and DWT achieving the excellent results. On the multivariate datasets, the deep learning and machine learning methods demonstrate clear advantages, with models such as KNN, KMeans, DAGMM, TsNet, and AE showing excellent performance.

5.3 Method Recommendations

To study the performance of different methods over different anomaly types and data characteristics, we further classify all the univariate time series according to anomaly type and data characteristic, respectively. The quantity distribution of univariate datasets under six different anomaly categories and five time series characteristics is not uniformly distributed since the number of open-source univariate time series datasets in real scenarios is limited. But we have tried our best to make them as evenly distributed as possible under different categories. For anomaly types, we use manual expert annotation to label them. Specifically, the type of anomaly for each time series is determined by a vote of five experienced manual experts in time series anomaly detection.

5.3.1 Analysis of data characteristics. To assess performance according to different data characteristics, Figure 9 reports the average results on datasets with distinct characteristics in terms of the metric Aff-F1. We make the following observations: 1) Most methods perform better on time series with *trend*, *stationarity*, and *shifting* characteristics. Time series with *seasonality* and *transition* characteristics make it difficult to detect anomalies. 2) LLM-based and time-series pre-trained models typically exhibit good performance on most data with varying characteristics. Notably, they perform well on data with trend and shifting characteristics. Density-based methods, like LODA and PCA, perform a bit worse across most characteristics. Contrast-based methods, including DC and ConAD, exhibit relatively weak performance across all characteristics. This suggests that this type of method has poor adaptability across univariate TSAD tasks. Prediction-based methods, such as iTrans, TsNet, CALF, and UniTS, frequently perform well on data with trend and seasonality characteristics.

5.3.2 Analysis on anomaly types. To assess performance across anomaly types, we partition the time series into six sets according to their anomaly type, i.e., contextual, global, shapelet, seanonal, trend, and mix. Global and contextual anomalies are point anomalies, while trend, shapelet, and seasonal anomalies are subsequence anomalies. Figure 10 shows the VUS-PR score distribution of different methods on datasets with different anomaly types. Our observations for the different anomaly types are as follows: 1) Among the six anomaly types, the global point anomalies are the easiest to detect, with methods achieving relatively high VUS-PR values. In contrast, mixed anomalies are the most challenging to identify. 2) Deep learning methods (e.g., iTrans and DP), as well as foundation methods (LLM-based and TS pre-trained methods) like CALF and UniTS, are more suitable for finding point anomalies (including global and contextual anomalies). 3) Non learning methods (e.g., DWT, LOF and HBOS), along with machine learning methods (e.g., KMeans and KNN), are most suited for finding subsequence anomalies (including shapelet, seasonal, and trend anomalies). 4) Density-based methods such as DWT, HBOS, EIF, and OCSVM are appropriate for finding trend anomalies. 5) Distance-based methods (e.g., KMeans and KNN) are suitable for finding shapelet anomalies.



(a) Global **(b) Contextual** **(c) Shapelet**

(d) Seasonal **(e) Trend** **(f) Mix**

Figure 10: Classification of univariate datasets based on anomaly type, including the 8 algorithms with the best average performance (VUS-PR) for each category.

When given an unseen time series, five formulas in our code repository can be used to calculate the five characteristic values of the time series. Then, based on the relationship between method performance and dataset characteristics described above, users can obtain a better understanding of which method to choose. These observations aid in making more targeted choices and optimizations of methods to better adapt to diverse application scenarios and data characteristics.

5.4 Trade-off among Accuracy, Memory, and Runtime

When applying methods in real-world scenarios, it is often necessary to strike a balance among accuracy, memory, and runtime. Each of these factors plays a crucial role in determining the feasibility and practicality of a method. Table 6 presents a comparative analysis of accuracy (V-PR, Aff-F1), memory (CPU and GPU memory usage), and runtime (training and inference time) for different methods on the multivariate and univariate datasets.

It can be observed that **in terms of runtime**, specifically regarding training time, many non learning methods (S2G, LSTi, DWT, etc.) and all TS pre-trained methods under the zero-shot evaluation strategy are the fastest since they do not require training. The training times for the machine learning methods are longer than those of the non learning methods. The training times for most deep learning methods are relatively long. The training times for the TS pre-trained methods under the full-shot evaluation strategy are the longest, and the training times for LLM-based methods are only shorter than those of TS pre-trained methods. Next, the inference times of the methods in these five categories are very short, with non-learning methods being the fastest, such as HBOS and CBLOF. **In terms of memory** and specifically regarding GPU memory usage, most non learning methods and machine learning methods do not need to use the GPU, so their GPU memory usage is 0. The methods that consume the most GPU memory are the LLM-based methods because they have a huge number of parameters. Next are the TS pre-trained methods, followed by the deep learning methods. Next, the CPU memory usage of methods is similar across the five categories. Among them, the usage by the non learning methods is the lowest. Considering accuracy, memory, and runtime together, for univariate anomaly detection, some non learning methods are good choices, notably S2G, DWT, and KMeans among the machine learning methods. However, most non learning methods do not support multivariate anomaly detection, and the performance of most machine learning methods at multivariate anomaly detection is not satisfactory. For multivariate anomaly detection, if the requirements for runtime and memory are relatively lenient, lightweight deep learning methods can be chosen, notably CATCH and TsNet.

6 CONCLUSIONS

To enable accurate comparison of various TSAD methods, this paper introduces TAB, a unified time series anomaly detection benchmark. First, TAB includes 1,635 univariate time series and 29 multivariate datasets, spanning multiple domains. Second, TAB incorporates state-of-the-art time series anomaly detection methods, including non-learning, machine learning, deep learning, LLM-based, and time series pre-trained approaches. Third, TAB provides a unified and extensible evaluation pipeline for time series anomaly detection, simplifying the process of evaluating TSAD methods and enabling fair comparisons through a consistent experimental setup. Finally, we use TAB to evaluate the performance of 40 multivariate TSAD methods and 46 univariate TSAD methods on all the datasets, employing a wide range of evaluation strategies and metrics. Additionally, we have also launched an online TSAD leaderboard that covers LLM-based and time series pre-trained methods.

ACKNOWLEDGMENTS

This work was partially supported by National Natural Science Foundation of China (62472174 and 62372179) and Huawei Cloud Availability Engineering Lab. Jilin Hu is the corresponding author of the work.

REFERENCES

[1] L. Glass J. M. Hausdorff P. C. Ivanov R. G. Mark J. E. Mietus G. B. Moody C.-K. Peng A. L. Goldberger, L. A. Amaral and H. E. Stanley. 2000. Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals. *Circulation* 101, 23 (2000), e215–e220.

[2] Ahmed Abdulaal, Zhuanghua Liu, and Tomer Lancewicki. 2021. Practical approach to asynchronous multivariate time series anomaly detection and localization. In *SIGKDD*. 2485–2494.

[3] Subutai Ahmad, Alexander Lavin, Scott Purdy, and Zuha Agha. 2017. Unsupervised real-time anomaly detection for streaming data. *Neurocomputing* 262 (2017), 134–147.

[4] Sarah Alnegheimish, Laure Berti-Equille, and Kalyan Veeramachaneni. 2023. Making the End-User a Priority in Benchmarking: OrionBench for Unsupervised Time Series Anomaly Detection. *arXiv preprint arXiv:2310.17748* (2023).

[5] Rafal Angryk, Petrus Martens, Berkay Aydin, Dustin Kempton, Sushant Mahajan, Sunitha Basodi, Azim Ahmadzadeh, Xumin Cai, Soukaina Filali Boubrahimi, Shah Muhammad Hamdi, Micheal Schuh, and Manolis Georgoulis. 2020. SWAN-SF. https://doi.org/10.7910/DVN/EBCFKM

[6] Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Syndar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, Jasper Zschiegner, Danielle C. Maddix, Michael W. Mahoney, Kari Torkkola, Andrew Gordon Wilson, Michael Bohlke-Schneider, and Yuyang Wang. 2024. Chronos: Learning the Language of Time Series. *Transactions on Machine Learning Research* (2024).

[7] Arthur Asuncion and David Newman. 2007. UCI machine learning repository.

[8] Marc Bachlin, Meir Plotnik, Daniel Roggen, Inbal Maidan, Jeffrey M Hausdorff, Nir Giladi, and Gerhard Troster. 2009. Wearable assistant for Parkinson's disease patients with the freezing of gait symptom. *IEEE Transactions on Information Technology in Biomedicine* 14, 2 (2009), 436–446.

[9] Pawel Benecki, Szymon Piechaczek, Daniel Kostrzewa, and Jakub Nalepa. 2021. Detecting anomalies in spacecraft telemetry using evolutionary thresholding and LSTMs. In *GECCO*. 143–144.

[10] Aadyot Bhatnagar, Paul Kassianik, Chenghao Liu, Tian Lan, Wenzhuo Yang, Rowan Cassius, Doyen Sahoo, Devansh Arpit, Sri Subramanian, Gerald Woo, Amrita Saha, Arun Kumar Jagota, Gokulakrishnan Gopalakrishnan, Manpreet Singh, K C Krithika, Sukumar Maddineni, Daeki Cho, Bo Zong, Yingbo Zhou, Caiming Xiong, Silvio Savarese, Steven Hoi, and Huan Wang. 2021. Merlion: A Machine Learning Library for Time Series. (2021). arXiv:2109.09265 [cs.LG]

[11] Paul Boniol and Themis Palpanas. 2020. Series2Graph: Graph-based Subsequence Anomaly Detection for Time Series. *Proc. VLDB Endow.* 13, 11 (2020), 1821–1834.

[12] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. 2000. LOF: identifying density-based local outliers. In *SIGMOD*. 93–104.

[13] David Campos, Miao Zhang, Bin Yang, Tung Kieu, Chenjuan Guo, and Christian S. Jensen. 2023. LightTS: Lightweight Time Series Classification with Adaptive Ensemble Distillation. *Proc. ACM Manag. Data* 1, 2 (2023), 171:1–171:27.

[14] Varun Chandola, Arindam Banerjee, and Vipin Kumar. 2009. Anomaly detection: A survey. *ACM computing surveys* 41, 3 (2009), 1–58.

[15] Yuxuan Chen, Shanshan Huang, Yunyao Cheng, Peng Chen, Zhongwen Rao, Yang Shu, Bin Yang, Lujia Pan, and Chenjuan Guo. 2025. AimTS: Augmented Series and Image Contrastive Learning for Time Series Classification. In *ICDE*.

[16] Yuwei Cui, Chetan Surpur, Subutai Ahmad, and Jeff Hawkins. 2016. A comparative study of HTM and other neural network models for online sequence learning with streaming data. In *IJCNN*. 1530–1538.

[17] Andrea Dal Pozzolo, Olivier Caelen, Reid A Johnson, and Gianluca Bontempi. 2015. Calibrating probability with undersampling for unbalanced classification. In *SSCI*. 159–166.

[18] Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. 2024. A decoder-only foundation model for time-series forecasting. In *ICML*.

[19] 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.

[20] Jesse Davis and Mark Goadrich. 2006. The relationship between Precision-Recall and ROC curves. In *ICML*. 233–240.

[21] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *CVPR*. 248–255.

[22] Vijay Ekambaram, Arindam Jati, Pankaj Dayama, Sumanta Mukherjee, Nam Nguyen, Wesley M Gifford, Chandra Reddy, and Jayant Kalagnanam. 2024. Tiny time mixers (ttms): Fast pre-trained models for enhanced zero/few-shot forecasting of multivariate time series. In *NeurIPS*. 74147–74181.

[23] Tom Fawcett. 2006. An introduction to ROC analysis. *Pattern recognition letters* 27, 8 (2006), 861–874.

[24] Cheng Feng and Pengwei Tian. 2021. Time series anomaly detection for cyber-physical systems via neural system identification and bayesian filtering. In *SIGKDD*. 2858–2867.

[25] Pavel Filonov, Andrey Lavrentyev, and Artem Vorontsov. 2016. Multivariate industrial time series with cyber-attack simulation: Fault detection using an lstm-based predictive data model. *arXiv preprint arXiv:1612.06676* (2016).

[26] P. Fleith. 2023. Controlled anomalies time series (cats) dataset. (2023). Accessed: 2023-09-01.

[27] Milton Friedman. 1937. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the american statistical association* 32, 200 (1937), 675–701.

[28] Hongfan Gao, Wangmeng Shen, Xiangfei Qiu, Ronghui Xu, Bin Yang, and Jilin Hu. 2025. SSD-TS: Exploring the potential of linear state space models for diffusion models in time series imputation. In *SIGKDD*.

[29] Shanghua Gao, Teddy Koker, Owen Queen, Thomas Hartvigsen, Theodoros Tsiligkaridis, and Marinka Zitnik. 2024. Units: Building a unified time series model. In *NeurIPS*.

[30] Markus Goldstein and Andreas Dengel. 2012. Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm. *KI 2012 - German Conference on Artificial Intelligence: poster and demo track* 1 (2012), 59–63.

[31] Mononito Goswami, Konrad Szafer, Arjun Choudhry, Yifu Cai, Shuo Li, and Artur Dubrawski. 2024. MOMENT: A Family of Open Time-series Foundation Models. In *ICML*.

[32] S. D. Greenwald. 1990. *Improved detection and classification of arrhythmias in noise-corrupted electrocardiograms using contextual information.*

[33] Ralf Greis, T Reis, and C Nguyen. 2018. Comparing prediction methods in anomaly detection: an industrial evaluation. In *Proceedings of the Workshop on Mining and Learning from Time Series*.

[34] Sahand Hariri, Matias Carrasco Kind, and Robert J Brunner. 2019. Extended isolation forest. *IEEE transactions on knowledge and data engineering* 33, 4 (2019), 1479–1489.

[35] Yangfan He, Jianhui Wang, Kun Li, Yijin Wang, Li Sun, Jun Yin, Miao Zhang, and Xueqian Wang. 2025. Enhancing Intent Understanding for Ambiguous Prompts through Human-Machine Co-Adaptation. *arXiv preprint arXiv:2501.15167* (2025).

[36] Zengyou He, Xiaofei Xu, and Shengchun Deng. 2003. Discovering cluster-based local outliers. *Pattern recognition letters* 24, 9-10 (2003), 1641–1650.

[37] Niklas Heim and James E Avery. 2019. Adaptive anomaly detection in chaotic time series with a spatially aware echo state network. *arXiv preprint arXiv:1909.01709* (2019).

[38] Shiyan Hu, Kai Zhao, Xiangfei Qiu, Yang Shu, Jilin Hu, Bin Yang, and Chenjuan Guo. 2024. MultiRC: Joint Learning for Time Series Anomaly Prediction and Detection with Multi-scale Reconstructive Contrast. *arXiv preprint arXiv:2410.15997* (2024).

[39] Qihe Huang, Lei Shen, Ruixin Zhang, Jiahuan Cheng, Shouhong Ding, Zhengyang Zhou, and Yang Wang. 2024. HDMixer: Hierarchical Dependency with Extendable Patch for Multivariate Time Series Forecasting. In *AAAI*. 12608–12616.

[40] Qihe Huang, Lei Shen, Ruixin Zhang, Shouhong Ding, Binwu Wang, Zhengyang Zhou, and Yang Wang. 2023. CrossGNN: Confronting Noisy Multivariate Time Series Via Cross Interaction Refinement. In *NeurIPS*. 46885–46902.

[41] Alexis Huet, Jose Manuel Navarro, and Dario Rossi. 2022. Local evaluation of time series anomaly detection algorithms. In *SIGKDD*. 635–645.

[42] Kyle Hundman, Valentino Constantinou, Christopher Laporte, Ian Colwell, and Tom Soderstrom. 2018. Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding. In *SIGKDD*. 387–395.

[43] Rob J Hyndman and George Athanasopoulos. 2018. *Forecasting: principles and practice.*

[44] Vincent Jacob, Fei Song, Arnaud Stiegler, Bijan Rad, Yanlei Diao, and Nesime Tatbul. 2020. Exathlon: A benchmark for explainable anomaly detection over time series. *arXiv preprint arXiv:2010.05073* (2020).

[45] Iurii D. Katser and Vyacheslav O. Kozitsin. 2020. Skoltech Anomaly Benchmark (SKAB).

[46] Dutta Roy T. Naik U. & Agrawal Keogh, E. 2021. Multi-dataset Time-Series Anomaly Detection Competition. In *SIGKDD*.

[47] Siwon Kim, Kukjin Choi, Hyun-Soo Choi, Byunghan Lee, and Sungroh Yoon. 2022. Towards a rigorous evaluation of time-series anomaly detection. In *AAAI*. 7194–7201.

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

[49] Md Kowsher, Md. Shohanur Islam Sobuj, Nusrat Jahan Prottasha, E. Alejandro Alanis, Ozlem Ozmen Garibay, and Niloofar Yousefi. 2024. LLM-Mixer: Multiscale Mixing in LLMs for Time Series Forecasting.

[50] Kwei-Herng Lai, Daochen Zha, Guanchu Wang, Junjie Xu, Yue Zhao, Devesh Kumar, Yile Chen, Purav Zumkhawaka, Minyang Wan, Diego Martinez, and Xia Hu. 2021. TODS: An Automated Time Series Outlier Detection System. *Proceedings of the AAAI Conference on Artificial Intelligence* (2021), 16060–16062.

[51] Kwei-Herng Lai, Daochen Zha, Junjie Xu, Yue Zhao, Guanchu Wang, and Xia Hu. 2021. Revisiting time series outlier detection: Definitions and benchmarks. In *Proceedings of the conference on neural information processing systems datasets and benchmarks track (round 1)*.

[52] N Laptev, S Amizadeh, and Y Billawala. 2015. S5-A labeled anomaly detection dataset, version 1.0 (16M).

[53] Alexander Lavin and Subutai Ahmad. 2015. Evaluating real-time anomaly detection algorithms–the Numenta anomaly benchmark. In *ICMLA*. 38–44.

[54] Zhe Li, Xiangfei Qiu, Peng Chen, Yihang Wang, Hanyin Cheng, Yang Shu, Jilin Hu, Chenjuan Guo, Aoying Zhou, Qingsong Wen, et al. 2025. TSFM-Bench: A Comprehensive and Unified Benchmark of Foundation Models for Time Series Forecasting. In *SIGKDD*.

[55] Zhihan Li, Youjian Zhao, Jiaqi Han, Ya Su, Rui Jiao, Xidao Wen, and Dan Pei. 2021. Multivariate time series anomaly detection and interpretation using hierarchical inter-metric and temporal embedding. In *SIGKDD*. 3220–3230.

[56] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. 2008. Isolation forest. In *ICDM*. 413–422.

[57] Jinyang Liu, Wenwei Gu, Zhuangbin Chen, Yichen Li, Yuxin Su, and Michael R Lyu. 2024. MTAD: Tools and Benchmarks for Multivariate Time Series Anomaly Detection. *arXiv preprint arXiv:2401.06175* (2024).

[58] Peiyuan Liu, Hang Guo, Tao Dai, Naiqi Li, Jigang Bao, Xudong Ren, Yong Jiang, and Shu-Tao Xia. 2025. CALF: Aligning LLMs for Time Series Forecasting via Cross-modal Fine-Tuning. In *AAAI*, Vol. 39. 18915–18923.

[59] Peiyuan Liu, Beiliang Wu, Yifan Hu, Naiqi Li, Tao Dai, Jigang Bao, and Shu-Tao Xia. 2025. TimeBridge: Non-Stationarity Matters for Long-term Time Series Forecasting. In *ICML*.

[60] Peiyuan Liu, Beiliang Wu, Naiqi Li, Tao Dai, Fengmao Lei, Jigang Bao, Yong Jiang, and Shu-Tao Xia. 2023. WFTNet: Exploiting Global and Local Periodicity in Long-term Time Series Forecasting. *ICASSP* (2023).

[61] Qinghua Liu and John Paparrizos. [n.d.]. The Elephant in the Room: Towards A Reliable Time-Series Anomaly Detection Benchmark. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

[62] Xu Liu, Junfeng Hu, Yuan Li, Shizhe Diao, Yuxuan Liang, Bryan Hooi, and Roger Zimmermann. 2024. Unitime: A language-empowered unified model for cross-domain time series forecasting. In *WWW*.

[63] Xvyuan Liu, Xiangfei Qiu, Xingjian Wu, Zhengyu Li, Chenjuan Guo, Jilin Hu, and Bin Yang. 2025. Rethinking Irregular Time Series Forecasting: A Simple yet Effective Baseline. *arXiv preprint arXiv:2505.11250* (2025).

[64] Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. 2024. iTransformer: Inverted Transformers Are Effective for Time Series Forecasting. In *ICLR*.

[65] Yong Liu, Haoran Zhang, Chenyu Li, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. 2024. Timer: Generative pre-trained transformers are large time series models. In *ICML*.

[66] Donghao Luo and Xue Wang. 2024. Moderntcn: A modern pure convolution structure for general time series analysis. In *ICLR*.

[67] Aditya P Mathur and Nils Ole Tippenhauer. 2016. SWaT: A water treatment testbed for research and training on ICS security. In *CySWater*. 31–36.

[68] George B Moody and Roger G Mark. 2001. The impact of the MIT-BIH arrhythmia database. *IEEE engineering in medicine and biology magazine* 20, 3 (2001), 45–50.

[69] Steffen Moritz, Frederik Rehbach, Sowmya Chandrasekaran, Margarita Rebolledo, and Thomas Bartz-Beielstein. 2018. Gecco industrial challenge 2018 dataset: A water quality dataset for the internet of things: Online anomaly detection for drinking water quality'competition at the genetic and evolutionary computation conference 2018, kyoto, japan. (2018).

[70] Youngeun Nam, Susik Yoon, Yooju Shin, Minyoung Bae, Hwanjun Song, Jae-Gil Lee, and Byung Suk Lee. 2024. Breaking the Time-Frequency Granularity Discrepancy in Time-Series Anomaly Detection. In *WWW*. ACM, 4204–4215.

[71] Peter Bjorn Nemenyi. 1963. *Distribution-free multiple comparisons.* Princeton University.

[72] Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. 2023. A Time Series is Worth 64 Words: Long-term Forecasting with Transformers. In *ICLR*.

[73] 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.

[74] 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.

[75] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, Vol. 32.

[76] Tomáš Pevnỳ. 2016. Loda: Lightweight on-line detector of anomalies. *Machine Learning* 102 (2016), 275–304.

[77] Xiangfei Qiu, Hanyin Cheng, Xingjian Wu, Jilin Hu, and Chenjuan Guo. 2025. A Comprehensive Survey of Deep Learning for Multivariate Time Series Forecasting: A Channel Strategy Perspective. *arXiv preprint arXiv:2502.10721* (2025).

[78] Xiangfei Qiu, Jilin Hu, Lekui Zhou, Xingjian Wu, Junyang Du, Buang Zhang, Chenjuan Guo, Aoying Zhou, Christian S. Jensen, Zhenli Sheng, and Bin Yang. 2024. TFB: Towards Comprehensive and Fair Benchmarking of Time Series Forecasting Methods. In *Proc. VLDB Endow.* 2363–2377.

[79] Xiangfei Qiu, Xiuwen Li, Ruiyang Pang, Zhicheng Pan, Xingjian Wu, Liu Yang, Jilin Hu, Yang Shu, Xuesong Lu, Chengcheng Yang, Chenjuan Guo, Aoying Zhou, Christian S. Jensen, and Bin Yang. 2025. EasyTime: Time Series Forecasting Made Easy. In *ICDE*.

[80] Xiangfei Qiu, Xingjian Wu, Yan Lin, Chenjuan Guo, Jilin Hu, and Bin Yang. 2025. DUET: Dual Clustering Enhanced Multivariate Time Series Forecasting. In *SIGKDD*. 1185–1196.

[81] Sridhar Ramaswamy, Rajeev Rastogi, and Kyuseok Shim. 2000. Efficient algorithms for mining outliers from large data sets. In *Proceedings of the 2000 ACM SIGMOD international conference on management of data*. 427–438.

[82] Hansheng Ren, Bixiong Xu, Yujing Wang, Chao Yi, Congrui Huang, Xiaoyu Kou, Tony Xing, Mao Yang, Jie Tong, and Qi Zhang. 2019. Time-series anomaly detection service at microsoft. In *SIGKDD*. 3009–3017.

[83] Daniel Roggen, Alberto Calatroni, Mirco Rossi, Thomas Holleczek, Kilian Förster, Gerhard Tröster, Paul Lukowicz, David Bannach, Gerald Pirkl, Alois Ferscha, et al. 2010. Collecting complex activity datasets in highly rich networked sensor environments. In *INSS*. 233–240.

[84] Mayu Sakurada and Takehisa Yairi. 2014. Anomaly detection using autoencoders with nonlinear dimensionality reduction. In *MLSDA*. 4–11.

[85] Sebastian Schmidl, Phillip Wenig, and Thorsten Papenbrock. 2022. Anomaly detection in time series: a comprehensive evaluation. *Proc. VLDB Endow.* 15, 9 (2022), 1779–1797.

[86] Bernhard Schölkopf, Robert C Williamson, Alex Smola, John Shawe-Taylor, and John Platt. 1999. Support vector method for novelty detection. In *NeurIPS*, Vol. 12.

[87] Zezhi Shao, Fei Wang, Yongjun Xu, Wei Wei, Chengqing Yu, Zhao Zhang, Di Yao, Guangyin Jin, Xin Cao, Gao Cong, et al. 2023. Exploring Progress in Multivariate Time Series Forecasting: Comprehensive Benchmarking and Heterogeneity Analysis. *arXiv preprint arXiv:2310.06119* (2023).

[88] Iman Sharafaldin, Arash Habibi Lashkari, and Ali A Ghorbani. 2018. Toward generating a new intrusion detection dataset and intrusion traffic characterization. *ICISSp* 1 (2018), 108–116.

[89] Qichao Shentu, Beibu Li, Kai Zhao, Yang Shu, Zhongwen Rao, Lujia Pan, Bin Yang, and Chenjuan Guo. 2025. Towards a General Time Series Anomaly Detector with Adaptive Bottlenecks and Dual Adversarial Decoders. In *ICLR*.

[90] Jingzhe Shi, Qinwei Ma, Huan Ma, and Lei Li. 2024. Scaling Law for Time Series Forecasting. *arXiv preprint arXiv:2405.15124* (2024).

[91] Mei-Ling Shyu, Shu-Ching Chen, Kanoksri Sarinnapakorn, and LiWu Chang. 2003. A novel anomaly detection scheme based on principal component classifier. In *Proceedings of the IEEE foundations and new directions of data mining workshop*. 172–179.

[92] Haotian Si, Jianhui Li, Changhua Pei, Hang Cui, Jingwen Yang, Yongqian Sun, Shenglin Zhang, Jingjing Li, Haiming Zhang, Jing Han, et al. 2024. Timeseriesbench: An industrial-grade benchmark for time series anomaly detection models. *arXiv preprint arXiv:2402.10802* (2024).

[93] Ya Su, Youjian Zhao, Chenhao Niu, Rong Liu, Wei Sun, and Dan Pei. 2019. Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In *SIGKDD*. 2828–2837.

[94] Tao. 2023. https://www.pmel.noaa.gov/

[95] Nesime Tatbul, Tae Jun Lee, Stan Zdonik, Mejbah Alam, and Justin Gottschlich. 2018. Precision and recall for time series. *NeurIPS* 31 (2018).

[96] Markus Thill, Wolfgang Konen, and Thomas Bäck. 2017. Time series anomaly detection with discrete wavelet transforms and maximum likelihood estimation. In *ITISE*, Vol. 2. 11–23.

[97] Markus Thill, Wolfgang Konen, and Thomas Bäck. 2020. MarkusThill/MGAB: The Mackey-Glass Anomaly Benchmark. *Version v1. 0.1. Zenodo. doi* 10 (2020).

[98] Shreshth Tuli, Giuliano Casale, and Nicholas R Jennings. 2022. Tranad: Deep transformer networks for anomaly detection in multivariate time series data. *arXiv preprint arXiv:2201.07284* (2022).

[99] UCI Knowledge Discovery in Databases Archive. 1999. KDD Cup 1999 Data. https://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html. Accessed: 2025-02-25.

[100] Alexander von Birgelen and Oliver Niggemann. 2018. Anomaly detection and localization for cyber-physical production systems with self-organizing maps. *IMPROVE-Innovative Modelling Approaches for Production Systems to Raise Validatable Efficiency: Intelligent Methods for the Factory of the Future* (2018), 55–71.

[101] Dennis Wagner, Tobias Michels, Florian CF Schulz, Arjun Nair, Maja Rudolph, and Marius Kloft. 2023. TimeseAD: Benchmarking deep multivariate time-series anomaly detection. *Transactions on Machine Learning Research* (2023).

[102] Chengsen Wang, Zirui Zhuang, Qi Qi, Jingyu Wang, Xingyu Wang, Haifeng Sun, and Jianxin Liao. 2023. Drift doesn't Matter: Dynamic Decomposition with Diffusion Reconstruction for Unstable Multivariate Time Series Anomaly Detection. In *NeurIPS*.

[103] Chengsen Wang, Zirui Zhuang, Qi Qi, Jingyu Wang, Xingyu Wang, Haifeng Sun, and Jianxin Liao. 2024. Drift doesn't matter: dynamic decomposition with diffusion reconstruction for unstable multivariate time series anomaly detection. In *NeurIPS*, Vol. 36.

[104] Hao Wang, Zhichao Chen, Zhaoran Liu, Haozhe Li, Degui Yang, Xinggao Liu, and Haoxuan Li. 2024. Entire Space Counterfactual Learning for Reliable Content Recommendations. *IEEE Trans. Autom. Sci. Eng.* (2024), 1–12.

[105] Hao Wang, Zhichao Chen, Zhaoran Liu, Haozhe Li, Degui Yang, Xinggao Liu, and Haoxuan Li. 2024. Entire space counterfactual learning for reliable content recommendations. *IEEE Transactions on Information Forensics and Security* (2024).

[106] Hao Wang, Zhichao Chen, Zhaoran Liu, Licheng Pan, Hu Xu, Yilin Liao, Haozhe Li, and Xinggao Liu. 2024. SPOT-I: Similarity Preserved Optimal Transport for Industrial IoT Data Imputation. *IEEE Transactions on Industrial Informatics* (2024).

[107] Hao Wang, Zhichao Chen, Honglei Zhang, Zhengnan Li, Licheng Pan, Haoxuan Li, and Mingming Gong. 2025. Debiased Recommendation via Wasserstein Causal Balancing. *ACM Transactions on Information Systems* (2025).

[108] Hao Wang, Haoxuan Li, Xu Chen, Mingming Gong, Zhichao Chen, et al. 2025. Optimal Transport for Time Series Imputation. In *ICLR*.

[109] Hao Wang, Xinggao Liu, Zhaoran Liu, Haozhe Li, Yilin Liao, Yuxin Huang, and Zhichao Chen. 2024. Lspt-d: Local similarity preserved transport for direct industrial data imputation. *IEEE Transactions on Automation Science and Engineering* (2024).

[110] Hao Wang, Licheng Pan, Zhichao Chen, Degui Yang, Sen Zhang, Yifei Yang, Xinggao Liu, Haoxuan Li, and Dacheng Tao. 2025. FreDF: Learning to Forecast in the Frequency Domain. In *ICLR*.

[111] Phillip Wenig, Sebastian Schmidl, and Thorsten Papenbrock. 2022. TimeEval: A benchmarking toolkit for time series anomaly detection algorithms. *Proc. VLDB Endow.* 15, 12 (2022), 3678–3681.

[112] Phillip Wenig, Sebastian Schmidl, and Thorsten Papenbrock. 2022. TimeEval: A Benchmarking Toolkit for Time Series Anomaly Detection Algorithms. *Proc. VLDB Endow.* 15, 12 (2022), 3678–3681.

[113] Frank Wilcoxon. 1992. Individual comparisons by ranking methods. In *Breakthroughs in statistics: Methodology and distribution*. Springer, 196–202.

[114] David H Wolpert and William G Macready. 1997. No free lunch theorems for optimization. *IEEE transactions on evolutionary computation* 1, 1 (1997), 67–82.

[115] Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. 2023. TimesNet: Temporal 2D-Variation Modeling for General Time Series Analysis. In *ICLR*.

[116] Renjie Wu and Eamonn Keogh. 2021. Current time series anomaly detection benchmarks are flawed and are creating the illusion of progress. *IEEE Transactions on Knowledge and Data Engineering* (2021).

[117] Xingjian Wu, Xiangfei Qiu, Hongfan Gao, Jilin Hu, Bin Yang, and Chenjuan Guo. 2025. K^2VAE: A Koopman-Kalman Enhanced Variational AutoEncoder for Probabilistic Time Series Forecasting. In *ICML*.

[118] Xingjian Wu, Xiangfei Qiu, Zhengyu Li, Yihang Wang, Jilin Hu, Chenjuan Guo, Hui Xiong, and Bin Yang. 2025. CATCH: Channel-Aware multivariate Time Series Anomaly Detection via Frequency Patching. In *ICLR*.

[119] Xinle Wu, Xingjian Wu, Bin Yang, Lekui Zhou, Chenjuan Guo, Xiangfei Qiu, Jilin Hu, Zhenli Sheng, and Christian S. Jensen. 2024. AutoCTS++: zero-shot joint neural architecture and hyperparameter search for correlated time series forecasting. *VLDB J.* 33, 5 (2024), 1743–1770.

[120] Yuhan Wu, Xiyu Meng, Yang He, Junru Zhang, Haowen Zhang, Yabo Dong, and Dongming Lu. 2024. Multi-view Self-Supervised Contrastive Learning for Multivariate Time Series. In *ACM MM*. 9582–9590.

[121] Yuhan Wu, Xiyu Meng, Huajin Hu, Junru Zhang, Yabo Dong, and Dongming Lu. 2025. Affirm: Interactive mamba with adaptive fourier filters for long-term time series forecasting. In *AAAI*, Vol. 39. 21599–21607.

[122] Haowen Xu, Wenxiao Chen, Nengwen Zhao, Zeyan Li, Jiahao Bu, Zhihan Li, Ying Liu, Youjian Zhao, Dan Pei, Yang Feng, et al. 2018. Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. In *Proceedings of the 2018 world wide web conference*. 187–196.

[123] Jiehui Xu, Haixu Wu, Jianmin Wang, and Mingsheng Long. 2021. Anomaly Transformer: Time Series Anomaly Detection with Association Discrepancy. In *ICLR*.

[124] Takehisa Yairi, Yoshikiyo Kato, and Koichi Hori. 2001. Fault detection by mining association rules from house-keeping data. In *i-SAIRAS*, Vol. 3.

[125] Yiyuan Yang, Chaoli Zhang, Tian Zhou, Qingsong Wen, and Liang Sun. 2023. Dcdetector: Dual attention contrastive representation learning for time series anomaly detection. In *SIGKDD*. 3033–3045.

[126] Yuanyuan Yao, Hailiang Jie, Lu Chen, Tianyi Li, Yunjun Gao, and Shiting Wen. 2024. TSec: An Efficient and Effective Framework for Time Series Classification. In *ICDE*. 1394–1406.

[127] Chin-Chia Michael Yeh, Yan Zhu, Liudmila Ulanova, Nurjahan Begum, Yifei Ding, Hoang Anh Dau, Diego Furtado Silva, Abdullah Mueen, and Eamonn Keogh. 2016. Matrix profile I: all pairs similarity joins for time series: a unifying view that includes motifs, discords and shapelets. In *Proceedings of the 2016 IEEE international conference on data mining*. 1317–1322.

[128] Susik Yoon, Jae-Gil Lee, and Byung Suk Lee. 2020. Ultrafast Local Outlier Detection from a Data Stream with Stationary Region Skipping. In *SIGKDD*. 1181–1191.

[129] Chengqing Yu, Fei Wang, Zezhi Shao, Tangwen Qian, Zhao Zhang, Wei Wei, Zhulin An, Qi Wang, and Yongjun Xu. 2025. GinAR+: A Robust End-To-End Framework for Multivariate Time Series Forecasting with Missing Values. *IEEE Transactions on Knowledge and Data Engineering* (2025), 1–14.

[130] Chengqing Yu, Fei Wang, Zezhi Shao, Tangwen Qian, Zhao Zhang, Wei Wei, and Yongjun Xu. 2024. Ginar: An end-to-end multivariate time series forecasting model suitable for variable missing. In *SIGKDD*. 3989–4000.

[131] Chengqing Yu, Fei Wang, Zezhi Shao, Tao Sun, Lin Wu, and Yongjun Xu. 2023. DSformer: A Double Sampling Transformer for Multivariate Time Series Long-term Prediction. In *CIKM*. 3062–3072.

[132] Chengqing Yu, Fei Wang, Chuanguang Yang, Zezhi Shao, Tao Sun, Tangwen Qian, Wei Wei, Zhulin An, and Yongjun Xu. 2025. Merlin: Multi-View Representation Learning for Robust Multivariate Time Series Forecasting with Unfixed Missing Rates. *arXiv preprint arXiv:2506.12459* (2025).

[133] Xinlei Yu, Ahmed Elazab, Ruiquan Ge, Hui Jin, Xinchen Jiang, Gangyong Jia, Qing Wu, Qinglei Shi, and Changmiao Wang. 2024. ICH-SCNet: Intracerebral Hemorrhage Segmentation and Prognosis Classification Network Using CLIP-guided SAM mechanism. In *BIBM*. 2795–2800.

[134] Xinlei Yu, Ahmed Elazab, Ruiquan Ge, Jichao Zhu, Lingyan Zhang, Gangyong Jia, Qing Wu, Xiang Wan, Lihua Li, and Changmiao Wang. 2025. ICH-PRNet: a cross-modal intracerebral haemorrhage prognostic prediction method using joint-attention interaction mechanism. *Neural Networks* 184 (2025), 107096.

[135] Zahra Zamanzadeh Darban, Geoffrey I Webb, Shirui Pan, Charu Aggarwal, and Mahsa Salehi. 2024. Deep learning for time series anomaly detection: A survey. *Comput. Surveys* 57, 1 (2024), 1–42.

[136] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. 2023. Are transformers effective for time series forecasting?. In *AAAI*, Vol. 37. 11121–11128.

[137] Aoqian Zhang, Shuqing Deng, Dongping Cui, Ye Yuan, and Guoren Wang. 2023. An experimental evaluation of anomaly detection in time series. *Proc. VLDB Endow.* 17, 3 (2023), 483–496.

[138] Chaoli Zhang, Yingying Zhang, Lanshu Peng, Qingsong Wen, Yiyuan Yang, Chongjiong Fan, Minqi Jiang, Lunting Fan, and Liang Sun. 2024. Advancing Multivariate Time Series Anomaly Detection: A Comprehensive Benchmark with Real-World Data from Alibaba Cloud. In *CIKM*. 5410–5414.

[139] Quan Zhang, Xiaoyu Liu, Wei Li, Hanting Chen, Junchao Liu, Jie Hu, Zhiwei Xiong, Chun Yuan, and Yunhe Wang. 2024. Distilling semantic priors from sam to efficient image restoration models. In *CVPR*. 25409–25419.

[140] Tian Zhou, Peisong Niu, Liang Sun, Rong Jin, et al. 2024. One fits all: Power general time series analysis by pretrained lm. In *NeurIPS*.

[141] Zhihao Zhuang, Yingying Zhang, Kai Zhao, Chenjuan Guo, Bin Yang, Qingsong Wen, and Lunting Fan. 2025. Noise Matters: Cross Contrastive Learning for Flink Anomaly Detection. *Proc. VLDB Endow.* (2025).

[142] Bo Zong, Qi Song, Martin Renqiang Min, Wei Cheng, Cristian Lumezanu, Daeki Cho, and Haifeng Chen. 2018. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In *ICLR*.