EasyAD: A Demonstration of Automated Solutions for Time-Series Anomaly Detection

Qinghua Liu
The Ohio State University
Columbus, OH, USA
liu.11085@osu.edu

Seunghak Lee
Meta
Menlo Park, CA, USA
seunghak@meta.com

John Paparrizos
The Ohio State University
Columbus, OH, USA
paparrizos.1@osu.edu

ABSTRACT

Despite the recent focus on time-series anomaly detection, the effectiveness of the proposed anomaly detectors is restricted to specific domains. A model that performs well on one dataset may not perform well on another. Therefore, how to develop automated solutions for anomaly detection for a particular dataset has emerged as a pressing issue. However, there is a noticeable gap in the literature regarding providing a comprehensive review of the ongoing efforts toward automated solutions for selecting or generating scores in an automated manner. Conducting a meta-analysis of proposed methods is challenging due to: (i) their evaluation across limited datasets; (ii) different assumptions on application scenarios; and (iii) the absence of evaluations for out-of-distribution performance. Motivated by the limitations above, we introduce the EasyAD, a modular web engine designed to facilitate the exploration of the first comprehensive benchmark for automated time-series anomaly detection. The EasyAD engine enables rigorous statistical analysis of 20 automated methods and 70 of their variants across the TSB-AD benchmark, a recently curated, heterogeneous dataset spanning nine application domains. The engine supports a two-dimensional evaluation framework, incorporating both accuracy and runtime performance. Our engine allows users to assess the performance of various methods per dataset and per instance, which offers fine-grained analysis per time series. Furthermore, the engine accommodates the processing of user-uploaded data, enabling users to experiment with different model selection strategies on their own datasets.

PVLDB Reference Format:
Qinghua Liu, Seunghak Lee, and John Paparrizos. EasyAD: A Demonstration of Automated Solutions for Time-Series Anomaly Detection. PVLDB, 18(12): 5431 - 5434, 2025.
doi:10.14778/3750601.3750689

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://easyad.streamlit.app.

1 INTRODUCTION

Time-series anomaly detection, which describes the process of analyzing an instance to identify abnormal patterns, has become critical across multiple scientific fields and industries [5, 22]. Recent years have witnessed a surge in the development of anomaly detection algorithms. Previous studies have evaluated the performance of these methods across different datasets [15, 24, 27]. These investigations have consistently highlighted the absence of a one-size-fits-all anomaly detector. Despite the vast amount of anomaly detection models, a critical question remains: *How can we automate time-series anomaly detection by selecting, ensembling, or generating models?*

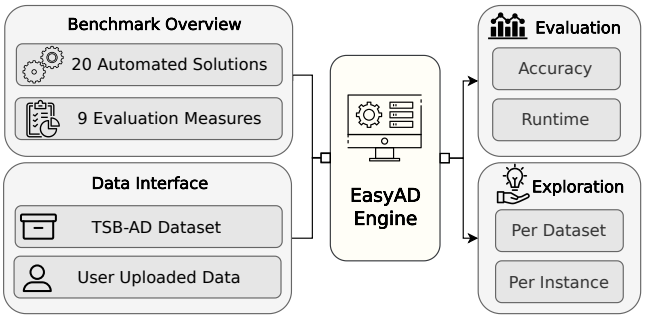


Figure 1: Overview of EasyAD Engine.

There have been some attempts made to address such challenges [10, 18, 29]. However, these studies exhibit certain limitations. Specifically, Ma *et al.* [18] provide an evaluation of unsupervised model selection for anomaly detection, yet their analysis primarily revolves around internal evaluation methodologies. Goswami *et al.* [10] target the time series scenario; however, their approach also confines itself to internal evaluation. Conversely, Sylligardos *et al.* [29] explore pretraining-based techniques, but their study is only focused on model selection via pretraining-based classifiers. The variability in datasets and different assumptions regarding application scenarios in these studies present a significant challenge when attempting to conduct a meta-analysis of their empirical performance. What is more, the efficacy of automated solutions in the context of time series datasets remains insufficiently validated.

To tackle the outlined problems and gain insights into the current state of research in this domain, we introduce the EasyAD a modular web engine designed to facilitate the exploration of automated time-series anomaly detection. As illustrated in Figure 1, this system is based on TSB-AutoAD [16], the first comprehensive benchmark for automated solutions in time-series anomaly detection. It is designed not only to improve the visualization and analysis of EasyAD benchmark which encompasses 20 automated solutions with 70 variants as shown in Table 1, but also to facilitate a deeper comprehension of this domain. The system enables statistical analysis of TSB-AD dataset [17] and allows for the exploration of data uploaded by users. In terms of evaluation, the system features a two-dimensional comparison, focusing on both the effectiveness and efficiency of solutions for a holistic analysis.

Proceedings of the VLDB Endowment, Vol. 18, No. 12 ISSN 2150-8097.
doi:10.14778/3750601.3750689

Table 1: Overview of TSB-AutoAD benchmark. 'TS' indicates whether the method is proposed for the time series scenario. 'D' indicates whether it requires anomaly scores generated from the complete candidate model set. And 'S' indicates the requirement of supervision from pretraining data.

Method	Variants	TS	D	S
SATzilla [13, 31]	[ID, OOD]×2	×	×	✓
ISAC [14]	[ID, OOD]×2	×	×	✓
ARGOSMART [21, 28, 34]	[ID, OOD]×2	×	×	✓
MetaOD [33]	[ID, OOD]×2	×	×	✓
MSAD [29]	[ID, OOD]×2	✓	×	✓
UReg [19]	[ID, OOD]×2	✓	×	✓
CFact [19]	[ID, OOD]×2	✓	×	✓
CQ [20]	[XB, Silhouette, R2, ...]×10	×	✓	×
UEC [9]	[Excess-Mass, Mass-Volume]×2	×	✓	×
MC [10, 18]	[1N, 3N, 5N, .. 12N]×3	✓	✓	×
Synthetic [7, 10]	[STL-cutoff, Orig-cutoff, ...]×12	✓	✓	×
TSADAMS [10]	[Borda, MIM, ...]×6	✓	✓	×
OE [3]	[Avg, Max, AOM]×3	×	✓	×
SELECT [25]	[Vertical, Horizontal]×2	×	✓	×
IOE [35]	1	×	✓	×
HITS [18]	1	×	✓	×
AutoTSAD [26]	1	✓	✓	×
AutoOD-A [6, 11]	[Majority, Orig, Ensemble]×3	×	✓	×
AutoOD-C [6]	[Majority, Individual, Ratio, Avg]×4	×	✓	×
UADB [32]	[Orig, Mean, STD, ...]×5	×	×	×
Count: 20	**70**			

Additionally, EasyAD enables a comprehensive assessment through a global comparison that aggregates evaluations across datasets, as well as an individual, fine-grained comparison for each time series.

2 PRELIMINARY

In this section, we provide the background necessary for the rest of the paper. We first introduce the taxonomy and pipeline of automated time-series anomaly detection, followed by an overview of the evaluation framework encompassing evaluation measures and preloaded datasets within the engine.

2.1 TSB-AutoAD Benchmark

From a process-centric perspective, the works in this field can be classified into three main categories, namely model selection, ensembling, and generation. The TSB-AutoAD benchmark comprises 20 solutions ranging from the 2010s to the current state-of-the-art methods as depicted in Figure 2. **Model selection** involves identifying the best model and its corresponding hyperparameters from a set of candidate models, which is then employed for anomaly detection. Within this category, existing approaches can be broadly classified into two subgroups: meta-learning-based methods and internal evaluation methods. Meta-learning-based approaches leverage prior knowledge about the performance of various anomaly detectors on historical labeled datasets to automate model selection for new, unseen datasets. In contrast, internal evaluation methods assess model effectiveness using surrogate metrics that do not rely on external data, such as ground-truth anomaly labels.

Model ensembling aims to enhance robustness and accuracy by combining the predictions of multiple candidate models through ensemble strategies. Meanwhile, **model generation** focuses on constructing an entirely new model from the candidate set, which is then used as an anomaly detector to produce anomaly scores.

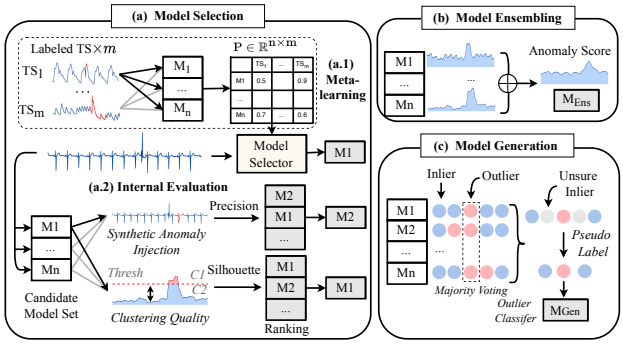


Figure 2: Overview of automated solution pipeline. We use M_1, M_2, and M_n to represent the candidate models.

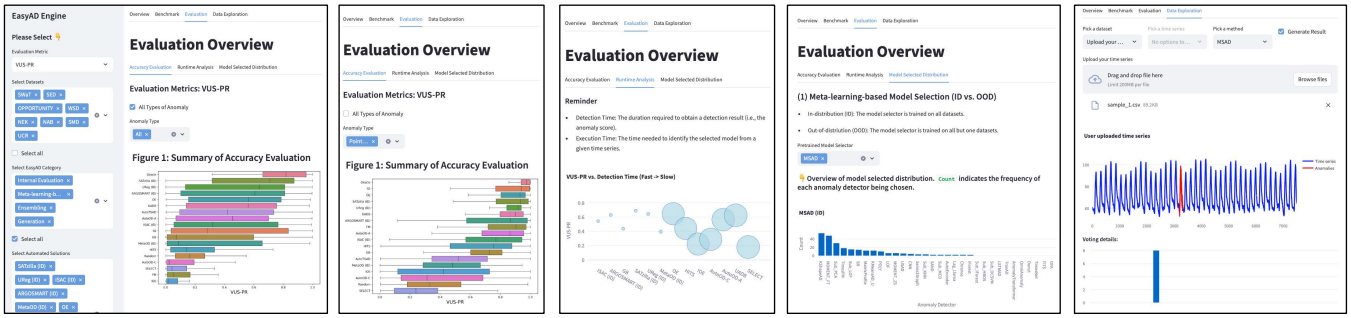
2.2 Evaluation Framework

Evaluation Measures. For accuracy evaluation, when comparing multiple solutions across multiple datasets, we use the Friedman test followed by the post-hoc Nemenyi test to determine their rankings. For point-level anomaly detection, we adopt standard metrics such as AUC-ROC, AUC-PR, and Standard-F1. To enhance completeness, we include PA-F1, a commonly used but imperfect variant that applies point adjustment [17]. Event-based-F1 [8] mitigates biases introduced by point adjustment by evaluating each anomaly segment as a single event, contributing once to the final score. To capture the sequential nature of time series, R-based-F1 [30] extends conventional metrics by incorporating existence and overlap rewards, along with a cardinality factor. Affiliation-F1 [12] further refines this by quantifying the temporal proximity between predicted and true anomaly ranges. Traditional metrics like AUC-ROC and AUC-PR, which treat all points equally, often fail to account for temporal consistency and label ambiguity. To address this, VUS-ROC and VUS-PR [4, 23] introduce a tolerance-aware framework that softens boundary definitions and employs continuous scoring, improving robustness in range-based evaluations.

In addition to the accuracy evaluation, we measure the *detection time* which refers to the duration required to obtain a detection result (i.e., the anomaly score) for a given time series. In the process of model selection, the detection time is divided into two components: *execution time*, which measures the time needed to identify the optimal model, and *detector runtime*, which is the time required for the chosen model to compute and produce the anomaly score. For model generation, where the method's output is directly the detection result, the execution time represents the detection time.

Datasets. To ensure reliable benchmarking results, we conduct our evaluation of automated solutions using the recently published, heterogeneous, and curated TSB-AD dataset [17]. Each dataset is divided into two parts: the *training set*, which enables access to ground truth anomaly labels, and the *evaluation set*, where the labels are only used for evaluation but are ignored during inference. The evaluation of automated solutions is conducted on the evaluation set, with the training set serving as the resource for supervised selection and pretraining data for meta-learning-based methods.

Baseline. We employ four types of baselines. The *Oracle* represents the theoretical upper bound for model selection, where the most accurate anomaly detector for a time series is selected based on ground truth labels. *Global Best (GB)* selects the model that exhibits



(a) Overall Accuracy Evaluation (b) Anomaly Type Analysis (c) Runtime Analysis (d) Model Selected Distribution (e) Test on your own data

Figure 3: The main frames of EasyAD Engine.

the highest overall performance (highest average ranking) across the entire evaluation set. *Supervised Selection (SS)* identifies the best anomaly detector on the training set of each dataset and then uses it for the evaluation set. In *Random*, the model is randomly selected from the candidate set for each time series and utilized for anomaly detection on that dataset.

Candidate Model Set: The candidate models serve as the base anomaly detectors from which automated solutions either select or generate final predictions. To ensure broad and representative coverage, we leverage the detection algorithms provided by the TSB-AD benchmark [17] - one of the most extensive and recent benchmarks for time-series anomaly detection, comprising 40 different algorithms across both univariate and multivariate settings.

3 SYSTEM OVERVIEW

In this section, we describe EasyAD engine [1], a modular web engine designed to facilitate the exploration of automated time-series anomaly detection. The GUI is a stand-alone web application developed using Python 3.10 and the Streamlit framework [2]. In total, the GUI consists of four frames: (i) Overview, (ii) Benchmark, (iii) Evaluation, and (iv) Data Exploration as shown in Figure 3. The Overview frame offers an overview of the engine's goals and a user manual to help get familiar with the engine. The Benchmark frame contains the pipeline and taxonomy of automated solutions, along with details on datasets and candidate model sets. Next, we provide details for the remaining frames.

Evaluation Frame. This frame is composed of three sub-frames. Illustrated in Figure 3 (a), the *Accuracy Evaluation* sub-frame features analyzing the performance variance of chosen methods across selected datasets. For illustrative purposes, this sub-frame includes a boxplot to summarize the distribution of performance of different methods, a critical diagram for evaluating rankings relative to one another and baselines, and a table that presents detailed performance metrics. Additionally, this section facilitates the exploration of performance variance under various anomaly types, such as single, multiple, point, and sequential anomalies, thus enabling a more in-depth exploration as depicted in Figure 3 (b). The second sub-frame *Runtime Analysis* in Figure 3 (c) showcases the detection and execution times of the selected methods. Within the bubble plot, a larger bubble size denotes a longer time, while a bubble's height correlates with greater accuracy. The third sub-frame, *Model Selection Distribution*, depicted in Figure 3 (d), delves into the nuances of model selection methods. For the pretraining-based model selection method, this section underscores the variability in model selection between in-distribution and out-of-distribution scenarios, followed by a pairwise accuracy comparison. For internal evaluation methods, the section highlights the frequency of different anomaly detectors being chosen throughout the evaluation process.

Data Exploration Frame. This frame is designed to conduct a detailed analysis of the efficacy of various methods across individual time series, offering a more fine-grained perspective compared to the Evaluation frame, which focuses on dataset-wise comparisons. Specifically, within this frame, users have the opportunity to delve into the evaluation results for all 20 solutions across TSB-AD. Furthermore, the frame enables the exploration of user-uploaded data by selecting the "Upload your own" option. For demonstration purposes, we have selected two best-performing methods *SATzilla* and *MSAD* as the exemplary methods. Upon uploading their data, users can opt for these methods to generate results, including the selection of the predicted model and the results of anomaly detection (comprising both the anomaly score and the identified anomalies) utilizing the chosen model, as illustrated in Figure 3 (e).

4 DEMONSTRATION SCENARIOS

In this section, we present five demonstration scenarios to help users navigate through the evaluation framework and gain insights into this field. This demo has four goals: (i) providing a comprehensive overview of the current state of research in automated time series anomaly detection (Scenarios 1-2); (ii) understanding the trade-offs between accuracy and runtime, as well as the applicability of various approaches across different use cases (Scenario 3); (iii) diving into model selection method and understanding the impact of domain shifting (Scenario 4); (iv) empowering users to investigate automated solutions on individual time series, enabling direct interaction with the framework (Scenario 5).

Scenario 1: Finding the overall best automated solutions. As illustrated in Figure 3 (a), to reproduce similar overall accuracy evaluation results, users are required to access the *Evaluation* frame. Then users can select the desired evaluation measures, datasets, and automated solutions from the options presented in the left sidebar. Upon selection, the corresponding evaluation results are displayed, starting with a boxplot that summarizes the accuracy of the chosen methods, followed by a critical diagram for assessing their relative rankings. Additionally, details regarding the datasets and automated solutions are accessible within the *Benchmark* frame. This section aims to provide users with a basic understanding of the comparative

performance of various methods, including their rankings relative to each other and to baseline approaches. Furthermore, to identify the best variant within each specific category, users may select a category and then opt for 'All Variants' to determine the best one.

Scenario 2: Investigating the influence of different anomaly types. In this scenario, users can explore the performance variance under different types of anomalies by selecting the corresponding type under the *Anomaly Type* menu as depicted in Figure 3 (b). The selected anomaly types encompass include single, multiple, point, and sequence anomalies.

Scenario 3: Understanding accuracy to runtime trade-off. Following the exploration of accuracy evaluations, this scenario delves into analyzing runtime performance, as depicted in Figure 3 (c). This analysis is visualized through two bubble plots: one representing detection time and the other execution time, where a larger bubble size indicates a longer time. A noteworthy distinction emerges between meta-learning-based methods, such as *MSAD*, and internal evaluation methods, like *MC*, in terms of runtime, with the former exhibiting significantly shorter execution and detection times. However, this efficiency is not without its trade-offs. Meta-learning-based methods require training on historical labeled datasets, thereby constraining their applicability across different use cases.

Scenario 4: Exploring the model selected distribution and the effect of domain shift. This scenario focuses on the model selection methods. We first demonstrate the frequency of different anomaly detectors being chosen. By selecting different methods in the drop-down menu, the user can explore different model selected distributions in both meta-learning-based and internal evaluation methods. Furthermore, within the context of meta-learning-based model selection, we underscore the discrepancy between performance in in-distribution versus out-of-distribution scenarios, as illustrated in Figure 3 (d). A subsequent pairwise comparison elucidates the significant performance reduction in out-of-distribution cases, highlighting the impact of domain shift on model efficacy.

Scenario 5: Testing on your own data. In this scenario, users can apply automated solutions to their own data by navigating the *Data Exploration* frame and uploading their time series for testing. Upon data uploading, the initial step involves visualizing the time series, followed by employing a pre-trained model selector to determine the best anomaly detector from the candidate model set, along with providing insights into the voting process. As depicted in Figure 3 (e), the model with the highest number of votes is utilized for detecting anomalies in this time series, and ultimately, the generated anomaly score and predicted anomalies are visualized.

5 CONCLUSION

In this paper, we introduce the EasyAD, a web-based engine designed to expedite the investigation of automated solutions for time-series anomaly detection. By interacting with the demonstration system, users can explore thorough evaluations of a comprehensive collection of automated solutions on a large scale of preloaded datasets as well as their own datasets. This system equips users to grasp the current state of research within this domain and assists them in identifying practical methodologies for real-world applications while highlighting existing research challenges. We hope the interactive demo system can empower users with insights and inspire further advancements in the field.

Acknowledgments: All datasets and models were accessed and used solely by The Ohio State University researchers. Additionally, the associated code being released was developed solely by OSU.

REFERENCES

[1] [n.d.]. EasyAD: A Demonstration of Automated Solutions for Time-Series Anomaly Detection. https://easyad.streamlit.app. Accessed: 2025-07-14.

[2] [n.d.]. *Streamlit documentation.* https://docs.streamlit.io

[3] Charu C Aggarwal and Saket Sathe. 2015. Theoretical foundations and algorithms for outlier ensembles. *Acm sigkdd explorations newsletter* 17, 1 (2015), 24–47.

[4] Paul Boniol, Ashwin K Krishna, Marine Bruel, Qinghua Liu, Mingyi Huang, Themis Palpanas, Ruey S Tsay, Aaron Elmore, Michael J Franklin, and John Paparrizos. 2025. VUS: effective and efficient accuracy measures for time-series anomaly detection. *The VLDB Journal* 34, 3 (2025), 32.

[5] Paul Boniol, Qinghua Liu, Mingyi Huang, Themis Palpanas, and John Paparrizos. 2024. Dive into time-series anomaly detection: A decade review. *arXiv preprint arXiv:2412.20512* (2024).

[6] Lei Cao, Yizhou Yan, Yu Wang, Samuel Madden, and Elke A Rundensteiner. 2023. AutoOD: Automatic Outlier Detection. *Proceedings of the ACM on Management of Data* 1, 1 (2023), 1–27.

[7] Sourav Chatterjee, Rohan Bopardikar, Marius Guerard, Uttam Thakore, and Xiaodong Jiang. 2022. MOSPAT: AutoML based Model Selection and Parameter Tuning for Time Series Anomaly Detection. *arXiv preprint arXiv:2205.11755* (2022).

[8] Astha Garg, Wenyu Zhang, Jules Samaran, Ramasamy Savitha, and Chuan-Sheng Foo. 2021. An evaluation of anomaly detection and diagnosis in multivariate time series. *IEEE Transactions on Neural Networks and Learning Systems* 33, 6 (2021), 2508–2517.

[9] Nicolas Goix. 2016. How to evaluate the quality of unsupervised anomaly detection algorithms? *arXiv preprint arXiv:1607.01152* (2016).

[10] Mononito Goswami, Cristian Challu, Laurent Callot, Lenon Minorics, and Andrey Kan. 2023. Unsupervised Model Selection for Time-series Anomaly Detection. *ICLR* (2023).

[11] Dennis Hofmann, Peter VanNostrand, Huayi Zhang, Yizhou Yan, Lei Cao, Samuel Madden, and Elke Rundensteiner. 2022. A demonstration of AutoOD: a self-tuning anomaly detection system. *PVLDB* 15, 12 (2022), 3706–3709.

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

[13] Minqi Jiang, Chaochuan Hou, Ao Zheng, Songqiao Han, Hailiang Huang, Qingsong Wen, Xiyang Hu, and Yue Zhao. 2024. ADGym: Design Choices for Deep Anomaly Detection. *Advances in Neural Information Processing Systems* 36 (2024).

[14] Serdar Kadioglu, Yuri Malitsky, Meinolf Sellmann, and Kevin Tierney. 2010. ISAC–instance-specific algorithm configuration. In *ECAI 2010*. IOS Press, 751–756.

[15] Qinghua Liu, Paul Boniol, Themis Palpanas, and John Paparrizos. 2024. Time-series anomaly detection: Overview and new trends. *PVLDB* 17, 12 (2024), 4229–4232.

[16] Qinghua Liu, Seunghak Lee, and John Paparrizos. 2025. TSB-AutoAD: Towards Automated Solutions for Time-Series Anomaly Detection. *PVLDB* 18, 11 (2025), 4364–4379.

[17] Qinghua Liu and John Paparrizos. 2024. The Elephant in the Room: Towards A Reliable Time-Series Anomaly Detection Benchmark. *NeurIPS* 37 (2024), 108231–108261.

[18] Martin Q Ma, Yue Zhao, Xiaorong Zhang, and Leman Akoglu. 2023. The need for unsupervised outlier model selection: A review and evaluation of internal evaluation strategies. *ACM SIGKDD Explorations Newsletter* 25, 1 (2023).

[19] Jose Manuel Navarro, Alexis Huet, and Dario Rossi. 2023. Meta-Learning for Fast Model Recommendation in Unsupervised Multivariate Time Series Anomaly Detection. In *AutoML Conference 2023*.

[20] Thanh Trung Nguyen, Uy Quang Nguyen, et al. 2016. An evaluation method for unsupervised anomaly detection algorithms. *Journal of Computer Science and Cybernetics* 32, 3 (2016), 259–272.

[21] Mladen Nikolić, Filip Marić, and Predrag Janičić. 2013. Simple algorithm portfolio for SAT. *Artificial Intelligence Review* 40, 4 (2013), 457–465.

[22] John Paparrizos, Paul Boniol, Qinghua Liu, and Themis Palpanas. 2025. Advances in Time-Series Anomaly Detection: Algorithms, Benchmarks, and Evaluation Measures. In *KDD*. ACM.

[23] 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. *Proceedings of the VLDB Endowment* 15, 11 (2022), 2774–2787.

[24] 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. *PVLDB* 15, 8 (2022), 1697–1711.

[25] Shebuti Rayana and Leman Akoglu. 2016. Less is more: Building selective anomaly ensembles. *Acm transactions on knowledge discovery from data (tkdd)* 10, 4 (2016), 1–33.

[26] Sebastian Schmidl, Felix Naumann, and Thorsten Papenbrock. 2024. AutoTSAD: Unsupervised Holistic Anomaly Detection for Time Series Data. *PVLDB* 17, 11 (2024), 2987–3002.

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

[28] Prabhant Singh and Joaquin Vanschoren. 2022. Meta-Learning for Unsupervised Outlier Detection with Optimal Transport. *arXiv preprint arXiv:2211.00372* (2022).

[29] Emmanouil Sylligardos, Paul Boniol, John Paparrizos, Panos Trahanias, and Themis Palpanas. 2023. Choose wisely: An extensive evaluation of model selection for anomaly detection in time series. *Proceedings of the VLDB Endowment* 16, 11 (2023), 3418–3432.

[30] Nesime Tatbul, Tae Jun Lee, Stan Zdonik, Mejbah Alam, and Justin Gottschlich. 2018. Precision and recall for time series. In *NeurIPS*. 1924–1934.

[31] Lin Xu, Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. 2008. SATzilla: portfolio-based algorithm selection for SAT. *Journal of artificial intelligence research* 32 (2008), 565–606.

[32] Hangting Ye, Zhining Liu, Xinyi Shen, Wei Cao, Shun Zheng, Xiaofan Gui, Huishuai Zhang, Yi Chang, and Jiang Bian. 2023. UADB: Unsupervised Anomaly Detection Booster. *arXiv preprint arXiv:2306.01997* (2023).

[33] Yue Zhao, Ryan Rossi, and Leman Akoglu. 2021. Automatic unsupervised outlier model selection. *NeurIPS* 34 (2021), 4489–4502.

[34] Yue Zhao, Sean Zhang, and Leman Akoglu. 2022. Toward unsupervised outlier model selection. In *ICDM*. IEEE, 773–782.

[35] Arthur Zimek, Ricardo JGB Campello, and Jörg Sander. 2014. Ensembles for unsupervised outlier detection: challenges and research questions a position paper. *Acm Sigkdd Explorations Newsletter* 15, 1 (2014), 11–22.