# VADETIS: An Explainable Evaluator for Anomaly Detection Techniques

Abdelouahab Khelifati*, Mourad Khayati*, Philippe Cudré-Mauroux*, Adrian Hänni†, Qian Liu‡, Manfred Hauswirth‡
*University of Fribourg, Switzerland, {firstname.lastname}@unifr.ch
†University of Bern, Switzerland, adrian.haenni@gmail.com
‡TU Berlin, Germany, {firstname.lastname}@tu-berlin.de

*Abstract*—**Anomaly detection is a fundamental problem that consists of identifying irregular patterns that do not conform to the expected behavior of a system or the generated data. Many anomaly detection techniques have been proposed for time series data. However, selecting the most suitable detection method remains challenging as the proposed techniques widely vary in performance. The appropriate choice of a detection method impacts many properties of mission-critical applications such as in monitoring a patient's health, where anomalies are inevitable but need to be detected securely. In this demo, we present a new evaluator that allows to peruse the performance of several anomaly detection techniques and supports practitioners in understanding the behavior and (dis-)advantages of each technique for a given dataset. In a simple and well-structured way, practitioners can specify the desired anomaly detection setup, and our system would tune the parameters of each technique and analyze their properties in an easily understandable report. The tool also allows recommending the most appropriate technique for each anomaly type and evaluation metric.**

## I. Introduction

Time series are becoming prevalent in many domains thanks to recent advances in the Internet of Things (IoT). Sensors in the IoT are, however, prone to failure, malfunction, and malicious attacks. As a result, the produced data often contains patterns that exhibit properties different from normal instances, i.e., anomalies. Detecting these anomalies helps to find less reliable data that needs to be cleansed, which in turn significantly improves downstream applications such as classification [1]. The detection can also reveal interesting latent phenomena such as frauds, data leakage or diseases [2].

The detection of anomalies is a well-studied problem, and several methods have been proposed to address it. However, there is a lack of practical systems for analyzing these approaches and their performance, making it very cumbersome for the user to pick the most suitable detection algorithm for a given problem. This problem occurs for at least three reasons.

First, existing techniques widely vary depending on their underlying approach and the type of anomalies they can detect. One class of methods builds a representation of "correct" data[1], then considers anomalies the instances that do not conform to this representation. These methods are often applied to detect continuous subsequences of anomalies [3]. The second class of techniques explicitly separates anomalous points in the datasets, making them suitable for scattered anomalies such as single-point outliers [4]. The choice of the appropriate method is difficult in real-world time series, which include both types of anomalies [2].

Next, anomaly detection techniques often assume specific properties of data to achieve good performance. A large body of techniques assumes temporal continuity, i.e., the main features of the data vary over time only a little, which constrains their applicability. For example, statistical methods can detect anomalies accurately on seasonal instances but perform poorly on non-periodic data [2]. In contrast, methods based on isolation [4] can be accurate on non-periodic data but may fail to distinguish normal seasonal peaks from anomalies. Real-time applications, such as IoT-based sports monitoring, produce time series that can be both periodic and bursty, rendering the choice of the technique very challenging.

The third reason why it is challenging to select a suitable detection technique is the lack of a unified metric for performance evaluation. Classification-based metrics, e.g., F1-score, measure the performance of a detection method as a binary classifier. Entropy-based metrics, such as Mutual Information, quantify the amount of information portrayed by an instance, assuming anomalies carry less information. Distance-based metrics, such as Root Mean Squared Error (RMSE), compute the distance between the ground-truth and the detection labels, assuming anomaly instances have a higher distance from the normal data. To evaluate the accuracy of a detection method, it can be essential to compare the detection results using several performance metrics simultaneously.

To solve this issue, we introduce VADETIS (Validator for Anomaly Detection in Time Series), a new graphical tool and engine to evaluate anomaly detection. Our system implements several anomaly detection methods and evaluation metrics, and allows to generate synthetic time series with multiple types, scales, and frequencies of anomalies. Users can train, tune, and compare the anomaly detection on different time series. VADETIS allows to recommend, for a specific dataset, the most accurate detection technique using a performance metric. This work was motivated by the need for such a comprehensive tool to assist our recent time series' repair benchmarks [5], [6]. Our tool helps identify suitable anomaly-aware techniques for the recovery of missing values.

[1]The definition of correctness depends on the problem under investigation itself and is outside the scope of this paper.

## II. Related Work

### A. Anomaly Detection Systems

Several systems have been proposed to visually explore time series data, e.g., Qetch [7], RINSE [8], PVD [9], RecovDB [10], etc. These systems typically focus on visualizing and querying time series while highlighting their key features and properties. The closest system to ours is Metro-Viz [11], which implements different anomaly detectors and metrics and performs what-if testing of anomaly detectors. The key novelty of our system resides in introducing an end-to-end framework for evaluating anomaly detection. In addition to the visual exploration of anomaly detectors and performance metrics, VADETIS offers the following salient features i) it can generate anomalous time series, ii) it supports anomaly detection on multiple series, and iii) it can recommend the optimal detector and metric. VADETIS can also be easily extended with new algorithms, datasets and metrics.

### B. Anomaly Detection Categories

**Statistical methods** use statistics such as distribution or variance to build a representation of the data. Techniques in this category assume that anomalous data instances have a lower probability of belonging to the representation area. As such, Histogram-Based Outlier Score (HBOS) [12] generates a histogram using all the values of each feature of the data. The anomaly score is inversely proportional to the heights of the columns in which each feature of the data value resides. Local Indicators of Spatial Association (LISA) is another statistical technique that measures the degree of spatial correlation at specific locations [13]. It uses the location coordinates of data instances and the weighted connections between them to detect anomalies by looking for the significant dissimilarity between instances and their physical neighbors.

**Clustering-based methods** build classes (or clusters) from the data and use them to identify as anomalies data points that do not conform to these classes, e.g., that lie outside the clusters. For example, the Gaussian mixture model (GMM) [2] assumes that all the data instances are generated from a mixture of Gaussian distributions represented by a cluster. During the detection, the probability that a data instance belongs to each of the clusters is computed, and anomalies represent instances that do not pertain to any cluster. One-Class Support Vector Machine (OC-SVM) [2] operates as a classifier by constructing a high-dimensional space using training data to separate the data instances into two classes. OC-SVM starts by assuming that the data belongs to only one class and learns the boundaries to derive a binary function that classifies instances which fall outside of this boundary as anomalies.

**Isolation-based methods** separate anomalous points in the dataset from the rest of the data, unlike previous approaches. The Isolation Forest (iForest) [4] approach builds an isolation tree (iTree) by recursively splitting the data based on a randomly selected feature and a splitting value between the maximum and minimum values of the selected feature. As anomalies are more likely to be isolated, it is more likely that they are closer to the root of an iTree. Robust Principal Component Analysis (RPCA) [2] is an isolation method based on dimensionality reduction. It computes a compact representation of a multi-dimensional dataset that maps the number of features to a lower-dimensional subspace. The detection of anomalies relies on the reconstruction error obtained from the information loss of the reduction. Since anomalies are rare, anomalous data instances have a higher reconstruction error.

## III. The VADETIS System

VADETIS is a web application implemented with the Django framework, an open-source web application framework in Python. Datasets are stored as pickled objects in order to efficiently handle large time series. In what follows, we describe the main components of our system (see Figure 1).

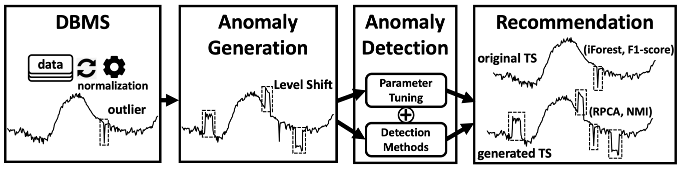


Fig. 1: Architecture Overview.

### A. Synthetic Anomaly Generation

The scarcity of anomalies creates highly unbalanced datasets, which impedes anomaly detection. To address this problem, our tool allows the contamination of the data with the most common types of anomalies [2]. It also allows controlling the relative frequency of the generated anomalies compared to all the data points. We describe below the different types of injected anomalies.

We define $X = \{x_1, \ldots, x_n\}$ as a time series, $f \in \mathbb{R}$ as a scale factor for the strength of the anomaly, $l \in \mathbb{N}^+$ as an offset for the sequence length of the anomaly, and $std$ for the standard deviation inside $X$.

**Point Outliers:** A point anomaly at a position $i$ is obtained by changing the data value with a delta $x'_i = x_i + \Delta$ such as $\Delta = f \times std$.

**Level shift:** A level shift at a position $i$ is obtained by shifting the sequence of points $x_j \in [x_{i-l}, \ldots, x_{i+l}]$ with a delta $x'_j = x_j + \Delta$ such as $\Delta = f \times std$.

**Growth Change:** A growth change at a position $i$ is obtained by increasing the values of the sequence following it $x_j \in [x_i, \ldots, x_{i+l}]$ with a growing factor. As such, for each $j \in [i, i+l]$, $x'_j = x_j + \Delta_j$ such that $\Delta_j = (j-i) \times f$. The rest of the time series $x_j \in [x_{i+l}, \ldots, x_n]$ is corrected with the delta $\Delta_{i+l}$ such that $x'_j = x_j + \Delta_{i+l}$.

**Distortion:** We generate a distortion at a position $i$ by adding to each observation the multiplied difference of its two previous subsequent observations. As such, for each $x_j \in [x_{i-l}, x_{i+l}]$, $x'_j = x_j + \Delta_j$ such that $\Delta_j = (x_{j-1} - x_j) \times f$. The rest of the time series remains unchanged.

In addition to anomalies, we generate missing values by setting values of the sequence around it to 0, imitating a sensor recording failure. Note that if the data is labeled, the generation only changes the normal data, because the original anomalies should be preserved.

### B. Anomaly Detection

We implemented all the techniques introduced in Section II in our tool. Each of those techniques returns either *anomaly scores* or *anomaly labels*. In the former case, a score is assigned to each data instance representing the degree to which it is considered as an anomaly. The output of such techniques is a ranked list of anomalies and a user can choose to analyze the top few anomalies or to use a domain-specific threshold to select the most relevant anomalies. In the latter case, a label (normal or anomaly) is assigned to each data instance. Before the anomaly detection is launched, the data is normalized to reduce the dominance of a variable over the other.

To further analyze the impact of time series features on anomaly detection techniques, we extended the implementation of the LISA technique to embed different types of correlations (e.g., high/low, positive/negative). In this extension, we compute anomaly scores using a sliding window, which allows taking into account local correlations. We apply Dynamic Time Warping (DTW) [14] on multiple time series in order to detect the non-linear similarities between the time series. We then compute the Pearson correlation on the DTW adjusted windows and a window is considered anomalous if its computed score is lower than a user-defined threshold $\delta$.

The performance of an anomaly detector largely depends on its parameterization (e.g. decision threshold, sliding window size, time range, training size, etc.). VADETIS tunes the parameters of each technique by evaluating a number (200 by default) of linearly distributed parameter candidates from each parameter's range and choosing the value that maximizes the detection performance. The optimal parameter values are suggested for each anomaly detector.

### C. Detection Recommendation

This component recommends the optimal detection technique for a given dataset and a performance metric. To achieve this, VADETIS compares different performance metrics (i.e., NMI, RMSE, Accuracy, AUC, $F_\beta$-Score, Precision and Recall) across the selected methods. While the component can recommend a detection method for multiple series, it is also possible to perform a customized recommendation for a single time series by comparing the performance metrics separately for each single time series in the dataset. This is especially beneficial in datasets where each time series caries different features and anomaly types (e.g. humidity and temperature). It is also useful to obtain other metrics for each technique in addition to the optimal metrics, which allows to compare the metrics and obtain deeper insight into the performance of each technique. A detailed report of the recommendation is provided for each metric once the evaluation is finished.

## IV. Demonstration Scenarios

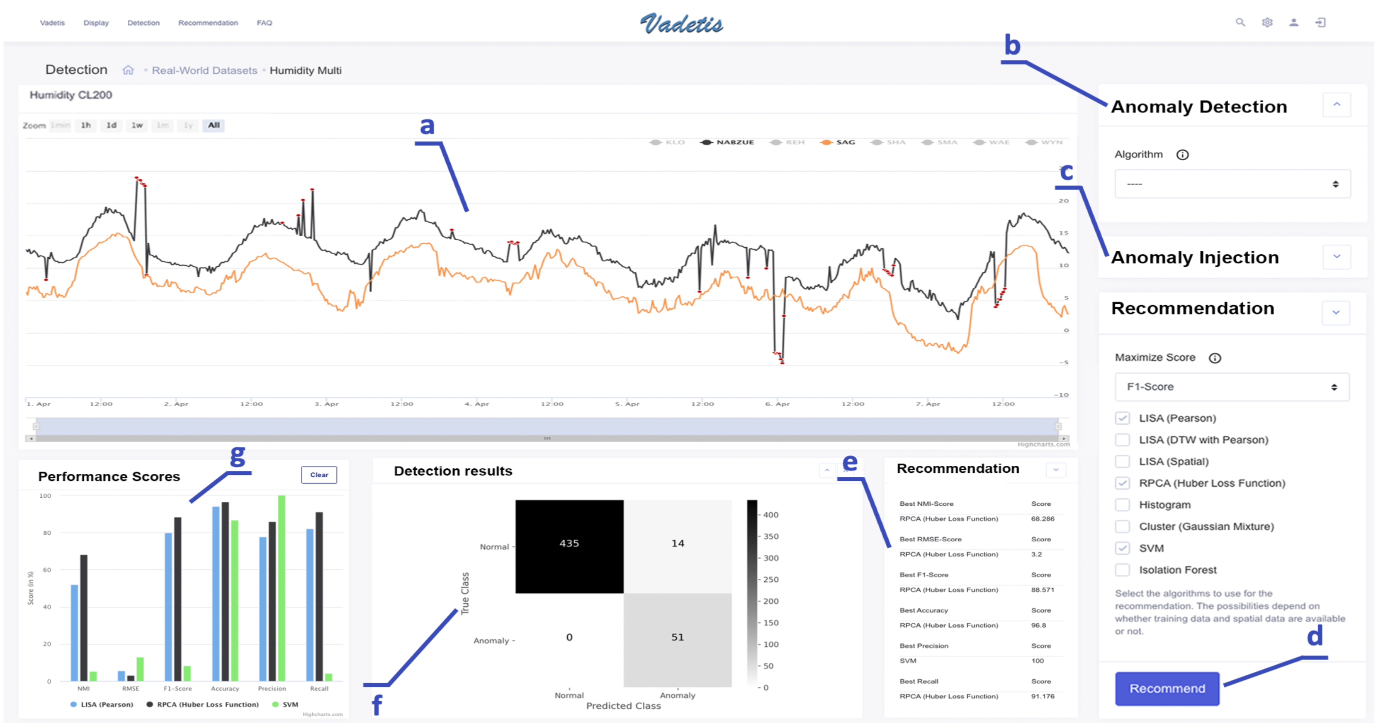
Our demo provides real-world labeled time series from a broad range of applications. We use the Yahoo S5 dataset[2], which contains seasonal time series with point outliers. We also use non-periodic humidity time series, which contain level shift and point outliers [5]. The last dataset we use is soccer, which represents the position of players during a football match [6]. The data originates from sensors located near the players' shoes and the goalkeeper's hands. Soccer dataset contains bursty and very long (15'000 position events per second) time series with multiple types of anomalies.

**Scenario 1: Anomaly detection and threshold adjustment.** In this scenario, the user can select one of the available anomaly detection techniques and define the parameters for the algorithm or proceed with the default settings. The detection will either mark anomalous data instances with training-based techniques or points in single time series (in the case of LISA). The detection is initiated by clicking the "Run" button and applied either on the selected time range or complete (set of) series. After the results have been computed, the updated chart with true positives, false positives and false negatives marked accordingly is provided. The performance metrics, confusion matrix and threshold-score plots are displayed to better understand the performance of anomaly detection. Users can also adjust the threshold level. By doing so, they can observe which data instances or points are classified correctly for changing decision boundaries. From this point, users can directly apply a different detection technique.

**Scenario 2: Synthetic anomaly generation.** In this scenario, users can alter the used time series by generating synthetic anomalies before detection. By clicking the "Inject" button the selected type of anomaly is added to the chosen time series. This step can be repeated multiple times for different types of anomalies and time series. Then, similarly to Scenario 1, the detection is initiated by clicking the "Run" button. After the results have been displayed, users can adjust the threshold for the decision boundary either with the slider or from the field to a different value and click the "Update" button. The chart, the performance metrics, confusion matrix and plots will be updated for the new decision boundary changing the markers of true positives, false positives and false negatives.

**Scenario 3: Metric-based recommendation of the anomaly detection technique.** In this scenario, users select several detection techniques they want to compare in order to obtain a recommendation. A performance metric to optimize has also to be chosen. By clicking the "Recommend" button, a detection with each selected technique is performed using default parameters. The results for NMI, RMSE, F1-Score, Accuracy, Precision and Recall are then compared in a bar chart for each of the selected techniques. Furthermore, a summary that lists the best technique for each metric is provided, and the confusion matrix and threshold-score plots are loaded for each of the used techniques. From this point,

[2]This dataset consists of time series representing the metrics of various Yahoo services. The S5 time series are highly correlated.



Algorithm: ----

Maximize Score: F1-Score

- [✓] LISA (Pearson)
- [ ] LISA (DTW with Pearson)
- [ ] LISA (Spatial)
- [✓] RPCA (Huber Loss Function)
- [ ] Histogram
- [ ] Cluster (Gaussian Mixture)
- [✓] SVM
- [ ] Isolation Forest

Select the algorithms to use for the recommendation. The possibilities depend on whether training data and spatial data are available or not.

Recommend

| Best NMI-Score | Score |
|---|---|
| RPCA (Huber Loss Function) | 68.286 |
| **Best RMSE-Score** | **Score** |
| RPCA (Huber Loss Function) | 3.2 |
| **Best F1-Score** | **Score** |
| RPCA (Huber Loss Function) | 88.571 |
| **Best Accuracy** | **Score** |
| RPCA (Huber Loss Function) | 96.8 |
| **Best Precision** | **Score** |
| SVM | 100 |
| **Best Recall** | **Score** |
| RPCA (Huber Loss Function) | 91.176 |

Fig. 2: VADETIS allows to (a) visualize (raw and normalized) time series with their metadata and to (b) perform anomaly detection on them. Users can (c) generate anomalies in specific time series while specifying their type, scale and frequency. They can also (d) choose a performance metric and the tool then evaluates a set of detection techniques, compares their performance, and recommends the optimal technique. The tool provides a GUI that enables displaying time series with anomalies highlighted, as well as the results of the detection/recommendation (e, f, and g).

users can request additional recommendations or adapt existing recommendations to optimize a different metric. In the future, we plan to apply pruning-based techniques to filter out irrelevant error detection algorithms and configurations [15].

## Acknowledgments

This work is funded by the Swiss National Funding under grant agreement No 169840 (ProvDS), the Deutsche Forschungsgemeinschaft (DFG) – project No 323223507 (ProvDS), and the European Research Council under grant agreement No 683253 (GraphInt).

## References

[1] A. Zhang, S. Song, J. Wang, and P. S. Yu, "Time series data cleaning: From anomaly detection to anomaly repairing," *Proc. VLDB Endow.*, vol. 10, no. 10, pp. 1046–1057, 2017.
[2] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, Jul. 2009.
[3] P. Boniol, M. Linardi, F. Roncallo, and T. Palpanas, "Sad: An unsupervised system for subsequence anomaly detection," in *2020 IEEE 36th International Conference on Data Engineering (ICDE)*. IEEE, 2020, pp. 1778–1781.
[4] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation-based anomaly detection," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 6, no. 1, pp. 1–39, 2012.
[5] M. Khayati, A. Lerner, Z. Tymchenko, and P. Cudré-Mauroux, "Mind the gap: An experimental evaluation of imputation of missing values techniques in time series," *Proc. VLDB Endow.*, vol. 13, no. 5, pp. 768–782, 2020.
[6] M. Khayati, I. Arous, Z. Tymchenko, and P. Cudré-Mauroux, "Orbits: Online recovery of missing blocks in multiple time series streams," in *Proceedings of the VLDB Endowment*, vol. 14, no. 3, 2021.
[7] M. Mannino and A. Abouzied, "Qetch: Time series querying with expressive sketches," in *Proceedings of the 2018 International Conference on Management of Data*, 2018, pp. 1741–1744.
[8] K. Zoumpatianos, S. Idreos, and T. Palpanas, "RINSE: interactive data series exploration with ADS+," *Proc. VLDB Endow.*, vol. 8, no. 12, pp. 1912–1915, 2015.
[9] L. Ramjit, Z. Kong, R. Netravali, and E. Wu, "Physical visualization design," in *Proceedings of the 2020 International Conference on Management of Data, SIGMOD Conference 2020, online conference [Portland, OR, USA], June 14-19, 2020*, D. Maier, R. Pottinger, A. Doan, W. Tan, A. Alawini, and H. Q. Ngo, Eds. ACM, 2020, pp. 2809–2812.
[10] I. Arous, M. Khayati, P. Cudré-Mauroux, Y. Zhang, M. Kersten, and S. Stalinlov, "Recovdb: Accurate and efficient missing blocks recovery for large time series," in *2019 IEEE 35th International Conference on Data Engineering (ICDE)*. IEEE, 2019, pp. 1976–1979.
[11] P. Eichmann, F. Solleza, N. Tatbul, and S. Zdonik, "Visual exploration of time series anomalies with metro-viz," in *Proceedings of the 2019 International Conference on Management of Data*, 2019, pp. 1901–1904.
[12] M. Goldstein and A. Dengel, "Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm," *KI-2012: Poster and Demo Track*, pp. 59–63, 2012.
[13] C. D. Lloyd, *Local models for spatial analysis*. CRC press, 2010.
[14] T. Rakthanmanon, B. Campana, A. Mueen, G. Batista, B. Westover, Q. Zhu, J. Zakaria, and E. Keogh, "Searching and mining trillions of time series subsequences under dynamic time warping," in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2012, pp. 262–270.
[15] M. Mahdavi, Z. Abedjan, R. Castro Fernandez, S. Madden, M. Ouzzani, M. Stonebraker, and N. Tang, "Raha: A configuration-free error detection system," in *Proceedings of the 2019 International Conference on Management of Data*, 2019, pp. 865–882.