

Choose Wisely: An Extensive Evaluation of Model Selection for Anomaly Detection in Time Series

Emmanouil Sylligardos

Thesis submitted in partial fulfillment of the requirements for the
Masters' of Science degree in Computer Science and Engineering

University of Crete
School of Sciences and Engineering
Computer Science Department
Voutes University Campus, 700 13 Heraklion, Crete, Greece

Thesis Advisor: Prof. *Panos Trahanias*

This work has been performed at the University of Crete, School of Sciences and Engineering, Computer Science Department.

The work has been supported by the Foundation for Research and Technology - Hellas (FORTH), Institute of Computer Science (ICS).

UNIVERSITY OF CRETE
COMPUTER SCIENCE DEPARTMENT

**Choose Wisely: An Extensive Evaluation of Model Selection for
Anomaly Detection in Time Series**

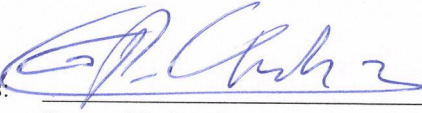
Thesis submitted by
Emmanouil Sylligardos
in partial fulfillment of the requirements for the
Masters' of Science degree in Computer Science

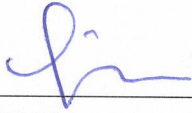
THESIS APPROVAL

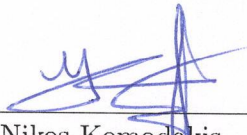
Author:


Emmanouil Sylligardos

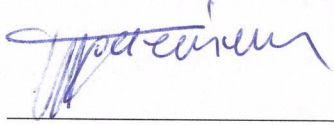
Committee approvals:


Panos Trahanias
Professor, Thesis Supervisor


Themis Palpanas
Professor, Committee Member


Nikos Komodakis
Assistant Professor, Committee Member

Departmental approval:


Polyvios Pratikakis
Associate Professor, Director of Graduate Studies

Heraklion, June 2023

Choose Wisely: An Extensive Evaluation of Model Selection for Anomaly Detection in Time Series

Abstract

Anomaly detection is a fundamental task for time-series analysis with important implications for the downstream performance of many applications. Despite increasing academic interest and the large number of methods proposed in the literature, recent benchmark and evaluation studies demonstrated that no overall best anomaly detection methods exist when applied to very heterogeneous time series datasets. This lack of a universally superior method poses a significant challenge for practitioners who need to select the most appropriate technique for their specific datasets. To overcome this limitation, this thesis proposes a model selection pipeline that can automatically determine the best anomaly detection technique based on the characteristics of the time series data. By leveraging time series classification algorithms for model selection, the goal is to provide a scalable and viable solution to solve anomaly detection over highly diverse time series collected from various domains.

Existing AutoML solutions are not directly applicable to time series anomaly detection, and no evaluation of time series-based approaches for model selection currently exists. Accordingly, we compare 16 different classifiers over 1800 time series, representing a diverse range of datasets. By comparing the performance of these classifiers, the study provides the first comprehensive evaluation of time series classification as a model selection approach for anomaly detection. The results demonstrate that model selection methods outperform individual anomaly detection methods while maintaining execution times in the same order of magnitude. This evaluation serves as a crucial first step in demonstrating the accuracy and efficiency of time series classification algorithms for anomaly detection, setting a strong baseline that can guide the model selection step in general AutoML pipelines.

The findings of this evaluation have significant implications for the field of time series anomaly detection. The demonstrated superiority of model selection methods over individual anomaly detection techniques highlights the importance of selecting the most appropriate method based on time series characteristics. By capitalizing on the strengths of different anomaly detection methods, practitioners can enhance the overall performance of their anomaly detection systems. Moreover, the evaluation serves as a benchmark for comparing and selecting time series classification algorithms for model selection purposes in anomaly detection tasks. This comprehensive study not only provides valuable insights into the effectiveness of various classifiers, but also establishes a foundation for further research and development in automated model selection approaches for time series anomaly detection. Ultimately, the proposed model selection method and the experimental evaluation contribute to advancing the state-of-the-art in time series analysis and enable more accurate and efficient anomaly detection in diverse application domains.

Επιλέξτε προσεκτικά: Μια Εκτενής Αξιολόγηση Επιλογής Μοντέλων για την Ανίχνευση Ανωμαλιών σε Χρονοσειρές

Περίληψη

Η ανίχνευση ανωμαλιών είναι μια θεμελιώδης εργασία για την ανάλυση χρονοσειρών με σημαντικές επιπτώσεις στην επίδοση πολλών εφαρμογών. Παρά το αυξανόμενο ακαδημαϊκό ενδιαφέρον και τον μεγάλο αριθμό μεθόδων που προτείνονται στη βιβλιογραφία, πρόσφατες μελέτες έδειξαν ότι δεν υπάρχουν καθολικά βέλτιστες μέθοδοι ανίχνευσης ανωμαλιών όταν εφαρμόζονται σε ετερογενή σύνολα χρονοσειρών. Αυτή η έλλειψη μιας καθολικά βέλτιστης μεθόδου αποτελεί σημαντική πρόκληση για όσους πρέπει να επιλέξουν την καταλληλότερη τεχνική για τα δεδομένα τους. Για να ξεπεραστεί αυτός ο περιορισμός, η παρούσα διατριβή προτείνει μία μέθοδο επιλογής μοντέλων που μπορεί να επιλέξει αυτόματα την καλύτερη τεχνική ανίχνευσης ανωμαλιών με βάση τα χαρακτηριστικά των χρονοσειρών. Αξιοποιώντας αλγορίθμους ταξινόμησης χρονοσειρών, ο στόχος είναι μια επεκτάσιμη και εφικτή λύση για την επίλυση της ανίχνευσης ανωμαλιών σε εξαιρετικά διαφορετικές χρονοσειρές από διάφορους τομείς.

Οι υπάρχουσες λύσεις αυτόματης μηχανικής μάθησης δεν είναι άμεσα εφαρμόσιμες στην ανίχνευση ανωμαλιών για χρονοσειρές, και επί του παρόντος δεν υπάρχει αξιολόγηση προσεγγίσεων επιλογής μοντέλων βασισμένων σε χρονοσειρές. Ως εκ τούτου, συγκρίνουμε 16 διαφορετικούς ταξινομητές σε 1800 χρονοσειρές, που αντιπροσωπεύουν διάφορα και ποικίλα σύνολα δεδομένων. Συγκρίνοντας τις επιδόσεις αυτών των ταξινομητών, αυτή η εργασία παρέχει την πρώτη εκτεταμένη αξιολόγηση της ταξινόμησης χρονοσειρών ως επιλογή μοντέλων για την ανίχνευση ανωμαλιών. Τα αποτελέσματα δείχνουν ότι οι μέθοδοι επιλογής μοντέλου υπερτερούν των επιμέρους μεθόδων ανίχνευσης ανωμαλιών, διατηρώντας παράλληλα χρόνους εκτέλεσης στην ίδια τάξη μεγέθους. Αυτή η αξιολόγηση χρησιμεύει ως ένα σημαντικό βήμα για να καταδείξει την αποτελεσματικότητα των αλγορίθμων ταξινόμησης για την ανίχνευση ανωμαλιών και θέτει ένα ισχυρό σημείο αναφοράς για την καθοδήγηση της επιλογής μοντέλων στις γενικές διαδικασίες της αυτόματης μηχανικής μάθησης.

Τα ευρήματα αυτής της αξιολόγησης έχουν σημαντικές επιπτώσεις στον τομέα της ανίχνευσης ανωμαλιών. Η αποδεδειγμένη υπεροχή των μεθόδων επιλογής μοντέλων υπογραμμίζει τη σημασία της επιλογής της καταλληλότερης μεθόδου με βάση τα χαρακτηριστικά της χρονοσειράς. Με την αξιοποίηση των πλεονεκτημάτων διάφορων μεθόδων ανίχνευσης ανωμαλιών, μπορεί να βελτιωθεί η συνολική απόδοση των συστημάτων ανίχνευσης ανωμαλιών. Επιπλέον, η τρέχουσα αξιολόγηση χρησιμεύει στη σύγκριση αλγορίθμων ταξινόμησης χρονοσειρών για την επιλογή μοντέλων για ανίχνευση ανωμαλιών. Αυτή η μελέτη όχι μόνο παρέχει πολύτιμες γνώσεις σχετικά με την αποτελεσματικότητα των διαφόρων ταξινομητών, αλλά θέτει και τα θεμέλια για περαιτέρω έρευνα σε αυτοματοποιημένες προσεγγίσεις επιλογής μοντέλων για την ανίχνευση ανωμαλιών. Τελικά, η προτεινόμενη μέθοδος και η πειραματική αξιολόγηση συμβάλλουν στην πρόοδο της ανάλυσης χρονοσειρών και επιτρέπουν την ακριβέστερη και αποτελεσματικότερη ανίχνευση ανωμαλιών σε διάφορους τομείς.

Acknowledgements

I would like to express my sincere gratitude to everyone who has supported me throughout my master's studies. First and foremost, I extend my heartfelt appreciation to my supervisor, Professor Panos Trahanias, for his indispensable guidance and support. Not only has he provided valuable insights for my thesis, but he has also equipped me with essential tools that will guide me throughout my entire life.

I am also grateful to Professor Themis Palpanas, who served as my supervisor during my internship at Université Paris Cité. My stay with him and his exceptional team, comprising brilliant researchers, who have now become my friends, has been a transformative experience that has enriched me both academically and socially. I am truly indebted to my co-supervisor and dear friend, postdoctoral researcher Paul Boniol, whose guidance during my stay in Paris left an indelible impact on me.

Additionally, I would like to express my appreciation to Assistant Prof. Nikos Komontakis for his positive stance to my MSc thesis and for providing valuable comments that significantly enhanced its quality. I am also grateful to Professor Panagiota Fatourou for her continuous guidance and unwavering support, which have profoundly shaped and influenced my academic journey. Furthermore, I extend my sincere thanks to Professor Evangelos Markatos for engaging in thought-provoking discussions and providing invaluable advice throughout my studies.

I extend my gratitude to the University of Crete, the Foundation for Research and Technology - Hellas (FORTH), and Université Paris Cité for providing the necessary resources and support that facilitated the successful completion of this research project. Their commitment to fostering a conducive research environment has played a pivotal role in my academic development.

Lastly, I want to express my deep gratitude to my parents and my sister for their constant support, standing by my side through thick and thin. I am also immensely grateful to my closest friends, for the beautiful company that has carried me through the most challenging moments. I would like to extend a special thank you to my dear friend and esteemed researcher, Dr. Savvas Kastanakis, whose guidance has been instrumental in my academic success.

I am profoundly grateful for the opportunities and blessings life has bestowed upon me, and any achievements I have or will attain are dedicated to them. Thank you.

*στους γονείς μου, Παντελή και Μαίρη,
που με κάνουν περήφανο*

Contents

1	Introduction	1
1.1	Motivation	1
1.2	Proposed Approach	2
1.3	Contributions	4
1.4	Publications	5
2	Background and Related Work	7
2.1	Time-Series and Anomaly Score Notations	7
2.2	Anomaly Detection Methods for Time Series	8
2.3	Limitations of Anomaly Detection Methods	8
2.3.1	Heterogeneity in anomaly types	9
2.3.2	Heterogeneity in time series structures	9
3	Motivation and Problem	11
3.1	Ensembling Detectors	11
3.2	Model Selection	12
3.3	Classification for Model Selection	12
3.4	Problem Formulation	13
3.5	Objectives	13
4	Proposed Pipeline	15
4.1	Preprocessing Step	15
4.2	Time Series Classification Approaches	17
4.2.1	Feature-based classification	17
4.2.2	Raw-based classification	18
4.2.3	Convolutional-based classification	18
4.2.4	Transformer-based classification	19
4.3	Selecting the Detector	20
5	Experimental Evaluation	23
5.1	Experimental Setup and Settings	23
5.2	Overall Evaluation	24
5.2.1	Accuracy Evaluation	24
5.2.2	Model selected distribution	26

5.2.3	Execution Time Evaluation	28
5.3	Influence of the Window Length	29
5.4	Influence of Datasets and Anomaly Types	31
5.5	Detection vs Classification Accuracy	32
5.6	Out-of-Distribution Experiments	35
6	Adecimo	39
6.1	System Overview	39
6.1.1	Overall Accuracy Frame	40
6.1.2	Overall Execution Time Frame	40
6.1.3	Interactive Exploration Frame	42
6.2	Demonstration Scenarios	42
6.2.1	Finding the best model selection method	42
6.2.2	Understanding and assessing model selection choice	42
6.2.3	Testing on your own data	43
7	Conclusions	45
7.1	Future Work	46
	Bibliography	47

List of Tables

5.1 Summary of datasets, methods, and measures.	38
---	----

List of Figures

1.1	Summary of our evaluation on the TSB-UAD benchmark [74] of model selection methods (best in blue) when compared to 12 anomaly detection methods and the Averaging Ensemble (in orange).	3
2.1	Accuracy of 12 anomaly detection methods on 4 datasets.	10
4.1	Proposed pipeline for the anomaly detection method selection . . .	16
4.2	Taxonomy of time series classification approaches used as model selection methods. We use the same color code for each class in all figures in the thesis.	20
5.1	VUS-PR and Detection time (seconds) for all model selection approaches (showing only the window length that maximizes VUS-PR for each model) over a test set of 497 series from TSB-UAD. The most accurate methods are at the top (a); the fastest methods are at the bottom (b)	25
5.2	Distribution of the selected models for five models (the best for each category) compared to the distribution of the labels (in black). Difference of distributions between time series containing (b) sequence and point anomalies, and (c) unique or multiple anomalies.	27
5.3	Execution time vs. length of model selection methods.	29
5.4	(a) Accuracy ((a.1) classification accuracy, (a.2) VUS-PR and (a.3) AUC-PR) and (b) execution time ((b.1) training time, (b.2) selection time and (b.3) detection time) versus window length ℓ	30
5.5	Correlation between accuracy and time series characteristics vs. the window length used to train the model selection methods.	31
5.6	Classification accuracy versus anomaly detection accuracy (VUS-PR) for (a) all datasets and (b) two specific datasets.	33
5.7	Out-of-distribution experiment, when model selection algorithms are trained on all but one dataset. (a) results for each dataset (when not included in the training set) and (b) average results. . .	34

5.8	VUS-PR and Detection time (seconds) for all model selection approaches (complete version) over a test set of 497 series from TSB-UAD. The most accurate methods are at the top (a); the fastest methods are at the bottom (b)	37
6.1	Summary of our system inputs and features	40
6.2	The three main ADecimo frames	41

Chapter 1

Introduction

Extensive collections of time-dependent measurements have become a reality in every scientific domain [10, 72]. The recording of these measurements results in an ordered sequence of real-valued data points, commonly referred to as *time series*. Analyzing time series data is becoming increasingly important in virtually every scientific and industrial domain, including astronomy [50], biology [11], economics [65], energy sciences [8], engineering [87], environmental sciences [42], medicine [77], neuroscience [13], and social sciences [26]. Anomaly detection, in particular, has received ample academic and industrial attention [71, 38], and has become a significant problem that finds applications across a wide range of domains and situations. These applications share the same goal [12, 85, 95]: analyzing time series to identify observations that do not correspond to expected behavior. In practice, anomalies can correspond to [3]: (i) noise or erroneous data (e.g., broken sensors); or (ii) actual data of interest (e.g., abnormal behavior of the observed system). In both cases, detecting such types is crucial for many applications [7, 46].

1.1 Motivation

In recent years, many techniques have been proposed for time-series anomaly detection. Multiple surveys and benchmarks summarize and analyze the state-of-the-art proposed methods [14, 74, 80, 73, 54, 92, 59, 53, 57]. Such surveys and benchmarks provide a holistic view of anomaly detection methods and how they perform. Unfortunately, these benchmark and evaluation studies demonstrated that no overall best anomaly detection methods exist when applied to very heterogeneous time series (i.e., coming from very different domains). In practice, we observe that some methods outperform others on specific time series with either specific characteristics (e.g., stationary or non-stationary time series) or anomalies (e.g., point-based or sequence-based anomalies).

To overcome the above limitation, ensembling solutions have been proposed [5] that consist of running all existing anomaly detection methods and averaging all

anomaly scores. Figure 1.1 shows that this solution (in orange) is outperforming all individual existing techniques in the TSB-UAD benchmark [74]. Nevertheless, as shown in Figure 1.1, such solutions require running all methods, resulting in an excessive cost that is not feasible in practice.

Therefore, the only scalable and viable solution to solve anomaly detection over very different time series from various domains is to propose a model selection method that will select, based on time series characteristics, the best anomaly detection method to run. This topic has been tackled in several recent research works related to AutoML (Automated Machine Learning) for the general case of anomaly detection [99, 97]. Nevertheless, existing AutoML solutions require (i) a universal objective function among models, which is not applicable to anomaly detection methods; (ii) a predefined set of features, which is difficult to obtain for time series due to varying lengths and the lack of standardized featurization solutions; (iii) running multiple anomaly detection methods several times, which is prohibitively expensive in practice; or (iv) labeled anomalies, which (in contrast to classification tasks) are difficult to obtain. Therefore, more work is needed in order to render AutoML solutions applicable to time-series anomaly detection.

1.2 Proposed Approach

Towards that direction, we cast the model selection problem for anomaly detection in time series as a time series classification problem. The objective is to train a time series classification model on time series for which we know in advance which anomaly detection method is the best. However, the lack of a benchmark with labeled time series has been a limiting factor for training robust model selection models (this only changed very recently [74, 80, 55]). Therefore, there exists no experimental evaluation that measures the effectiveness of classification methods for the task of model selection for time series anomaly detection. Though, such an evaluation is very important for determining which time series classification methods are accurate as model selection methods, and which solutions should be considered in unsupervised settings (i.e., using model selection approaches on time series from domains that were not included in the training set). These results would help the design and effectiveness of general AutoML pipelines for time series.

Thus, in this work, we evaluate the performance of time series classification methods used as model selection for anomaly detection in time series. To do so, we propose a pipeline that enables any kind of time series classifier to be used for any univariate time series with different lengths. We then compare the execution time and accuracy for feature-based, traditional time series classifiers and deep learning classification algorithms. We also measure how these models perform when trained on time series of a given domain (e.g., electrocardiogram [68]) and tested on time series from a different domain (e.g., robotics sensors measurements [78]).

Overall, we compare 16 different classifiers over 1980 time series and 12 anomaly detection methods from the recent anomaly detection benchmark TSB-UAD. Thus,

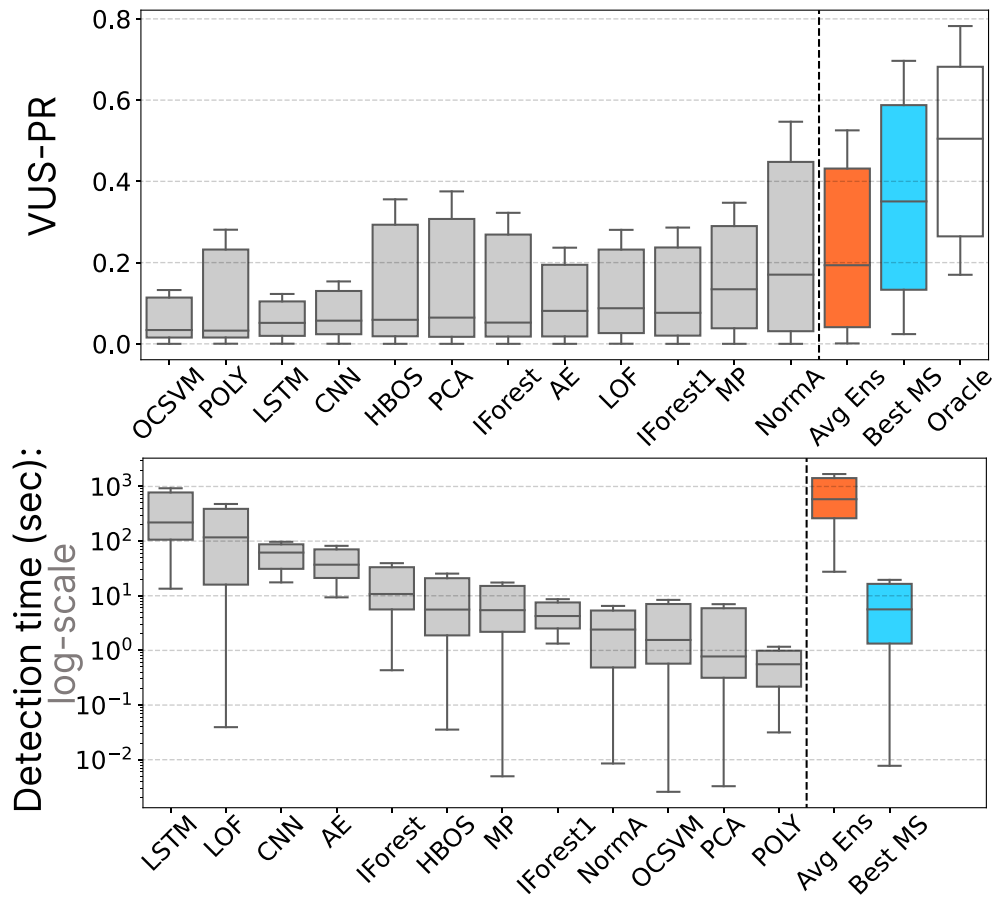


Figure 1.1: Summary of our evaluation on the TSB-UAD benchmark [74] of model selection methods (best in blue) when compared to 12 anomaly detection methods and the Averaging Ensemble (in orange).

we propose the first extensive experimental evaluation of time series classification as model selection for anomaly detection. Our results demonstrate that model selection methods outperform every single anomaly detection method while being in the same order of magnitude regarding execution time. Figure 1.1 shows a summary of our experimental evaluation, where the best model selection method (in blue in Figure 1.1) is up to $2.8\times$ more accurate than the best anomaly detection method in the TSB-UAD benchmark and $1.9\times$ more accurate than the ensembling solution mentioned above. This evaluation is the first step to demonstrate the accuracy and efficiency of time series classification algorithms for anomaly detection. It represents a strong baseline that can then be used to guide the choice of approaches for the model selection step in more general AutoML pipelines.

1.3 Contributions

Our contributions can be summarized as follows.

- We cast the model selection problem for time-series anomaly detection methods into a time-series classification problem. We describe and study the need to evaluate time series classification methods for model selection (Chapter 3).
- We introduce our novel pipeline for model selection applied to anomaly detection in time series. As this pipeline is generic, we describe how it can be used with both feature-based classification methods, traditional time series classification methods, and deep learning-based methods (Chapter 4).
- We describe our experimental framework (on top of the recent anomaly detection benchmark TSB-UAD [74]), and provide details on both anomaly detection methods and time series classification methods considered in this work (Chapter 5). We make all our material publicly available online on GitHub [15].
- We present an extensive experimental evaluation, measuring the anomaly detection accuracy and execution time (both training and inference) of model selection algorithms (Chapter 5.2). We evaluate the influence of important parameters and the relationship between classification and anomaly detection accuracy (Chapters 5.3, 5.4, and 5.5). Moreover, we measure the transferability of model selection algorithms to new types of time series by testing multiple combinations of train and test datasets that do not contain the same kinds of time series (Chapter 5.6).
- We provide an interactive Web App [16] for (a) exploring our results (Chapter 6) and (b) testing our pretrained models with your own data.

Finally, we conclude with the implications of our work and discuss possible future directions that could help improve both the accuracy and the execution time of our proposed pipeline (Chapter 7).

1.4 Publications

This research has been accepted for presentation at the *International Conference on Very Large Data Bases* (VLDB), Vancouver - Canada, Aug. 2023.

Chapter 2

Background and Related Work

We first introduce formal notations useful for the rest of the thesis (Chapter 2.1). Then, we review in detail previously proposed time-series anomaly detection methods (Chapter 2.2), and finally we discuss their limitations when applied to large heterogeneous sets of time series (Chapter 2.3).

2.1 Time-Series and Anomaly Score Notations

Time Series: A time series $T \in \mathbb{R}^n$ is a sequence of real-valued numbers $T_i \in \mathbb{R}$ $[T_1, T_2, \dots, T_n]$, where $n = |T|$ is the length of T , and T_i is the i^{th} point of T . We are typically interested in local regions of the time series, known as subsequences. A subsequence $T_{i,\ell} \in \mathbb{R}^\ell$ of a time series T is a continuous subset of the values of T of length ℓ starting at position i . Formally, $T_{i,\ell} = [T_i, T_{i+1}, \dots, T_{i+\ell-1}]$. A dataset \mathcal{D} is a set of time series. Note that the time series contained in \mathcal{D} can be of diverse lengths. We define the size of \mathcal{D} as $|\mathcal{D}|$.

Anomaly Score Sequence: For a time series $T \in \mathbb{R}^n$, an anomaly detection method (or detector) D returns an anomaly score sequence S_T . For point-based approaches (i.e., methods that return a score for each point of T), we have $S_T \in \mathbb{R}^n$. For subsequence-based approaches (i.e., methods that return a score for each subsequence of a given length ℓ), we have $S_T \in \mathbb{R}^{n-\ell}$. Overall, for subsequence-based approaches, we define $S_T = [S_{T_1}, S_{T_2}, \dots, S_{T_{n-\ell}}]$ with $S_{T_i} \in [0, 1]$. In most applications, we require the anomaly score to have the same length as the time series. Therefore, for subsequence-based approaches, we define $S_T = [S_{T_1}]_{i \in [0, \ell/2]} + [S_{T_1}, S_{T_2}, \dots, S_{T_{n-\ell}}] + [S_{T_{n-\ell}}]_{i \in [0, \ell/2]}$ with $|S_T| = |T|$.

Anomaly Detection Accuracy: For a time series $T \in \mathbb{R}^n$, an anomaly detection method (or detector) D that returns an anomaly score sequence $D(T) = S_T$ and labels $L \in [0, 1]^n$ that indicated with 0 or 1 if the points in T are normal or abnormal respectively, we define $Acc : \mathbb{R}^n, \{0, 1\}^n \rightarrow [0, 1]$ as an accuracy function for which $Acc(D(T), L)$ indicates how D is accurate (i.e., and produce an anomaly score close to 1 when the label is equal to 1) when applied on T and accordingly to L . The closer to one, the better.

2.2 Anomaly Detection Methods for Time Series

Anomaly detection in time series is a crucial task for many relevant applications. Therefore, several different methods (for diverse types of time series, or applications) have been proposed in the literature. One type of anomaly detection method is *discord-based methods*. These methods focus on the analysis of subsequences for the purpose of detecting anomalies in time series, mainly by utilizing nearest neighbor distances among subsequences [95, 82, 56, 63, 40, 27, 61].

Instead of measuring nearest neighbor distances, *proximity-based methods* focus on estimating the density of particular types of subsequences in order to either extract a normal behavior or isolate anomalies. As a subsequence can be seen as a multidimensional point (with the number of dimensions corresponding to the subsequence length), general outlier detection methods can be applied for time series anomaly detection [62, 25, 66]. Among them, Isolation Forest [62] has been shown to work particularly well for time series anomaly detection task [21]. Moreover, recent proximity-based methods dedicated to identifying abnormal subsequences in time series have been proposed. For instance, NormA, a proximity-based method that first clusters data to obtain the normal behavior [17, 19, 18, 23, 22], or Series2Graph that converts the time series into a graph to facilitate the detection of anomalies [21], has been shown to achieve strong performance.

Furthermore, *forecasting-based methods*, such as recurrent neural network-based [67] or convolutional network-based [69], have been proposed for this task. Such methods use the past values as input, predict the following one, and use the forecasting error as an anomaly score. Such methods are usually trained on time series without anomalies, or make the assumption that the anomalies are significantly less frequent than the normal behaviors.

Finally, *reconstruction-based methods*, such as autoencoder approaches [79], are trained to reconstruct the time series and use the reconstruction error as an anomaly score. As both forecasting and reconstruction-based categories detect anomalies using prediction errors (either forecasting or reconstruction error), we can group them into *prediction-based methods*.

2.3 Limitations of Anomaly Detection Methods

Recent benchmarks and experimental evaluations have been proposed in the literature [80, 73, 55]. Such benchmarks provide a large collection of time series from various domains and evaluate multiple methods belonging to the categories mentioned above. However, these experimental evaluations led to the same conclusion: no method exists that outperforms all the others on all time series from various domains. Figure 2.1, which depicts the accuracy of 12 diverse anomaly detection methods¹ on four time series datasets, illustrates the conclusion above.

¹We use 12 methods that have been employed in previous studies [74, 73]. Note that other methods and variations exist that may lead to improved results.

In Figure 2.1 (a.2), we observe that NormA is the most accurate model on the ECG dataset [68] (a time series example is depicted in Figure 2.1 (a.1)). However, Local Outlier Factor (LOF) [25], and Matrix profile (MP) [95] are significantly outperforming NormA on the MGAB dataset [86] (see Figure 2.1 (b.2)), whereas CNN [69] is outperforming NormA, LOF, and MP on the YAHOO dataset [58] (see Figure 2.1 (d.2)). The following two reasons explain this large difference in performance among datasets.

2.3.1 Heterogeneity in anomaly types

First, There are three types of time-series anomalies: *point*, *contextual*, and *collective* anomalies. *Point* anomalies refer to data points that deviate remarkably from the rest of the data. Similarly, *Contextual* anomalies refer to data points within the expected range of the distribution (in contrast to point anomalies) but deviate from the expected data distribution, given a specific context (e.g., a window). For instance, Figure 2.1 (d.1) illustrates a time series from the YAHOO dataset with a *Contextual* anomaly. The value of the anomalies is inside the range of normal values, but is abnormal in the context of the distribution of values of the surrounding point. For this particular types of anomalies, *reconstruction* and *forecasting*-based methods are particularly accurate (as shown in Figure 2.1 (d.2))

Collective anomalies refer to sequences of points that do not repeat a typical (previously observed) pattern. The first two categories, namely, point and contextual anomalies, are referred to as *point-based* anomalies, whereas *collective* anomalies are referred to as *subsequence* anomalies. For instance, Figure 2.1 (a.1), (b.1), and (c.1) show three time series with sequence anomalies. However, even for time series belonging to the same anomaly type categories, we observe that the most accurate models are all different.

2.3.2 Heterogeneity in time series structures

This diversity in model accuracy can be explained by other factors related to the time series structures. Indeed, on top of these categories mentioned above, the combination of them also matters.

First, we need to differentiate time series containing *single* anomalies from time series containing *multiple* anomalies. Last, the *multiple* time series category has to be divided into two subcategories, namely time series containing *multiple different* and *multiple similar* anomalies. For instance, methods based on neighbor distance computation such as LOF are very accurate in detecting *single* or *multiple different* anomalies, but less accurate for *multiple similar*. To illustrate this point, Figure 2.1 (a.2) depicts the results of 12 anomaly detection methods on the ECG dataset (that contains *multiple similar* anomalies), for which LOF accuracy is low. On the contrary, Figure 2.1 (b.2) depicts the results of the same 12 anomaly detection methods on the MGAB dataset (that contains *multiple different* anomalies), for which LOF accuracy is high.

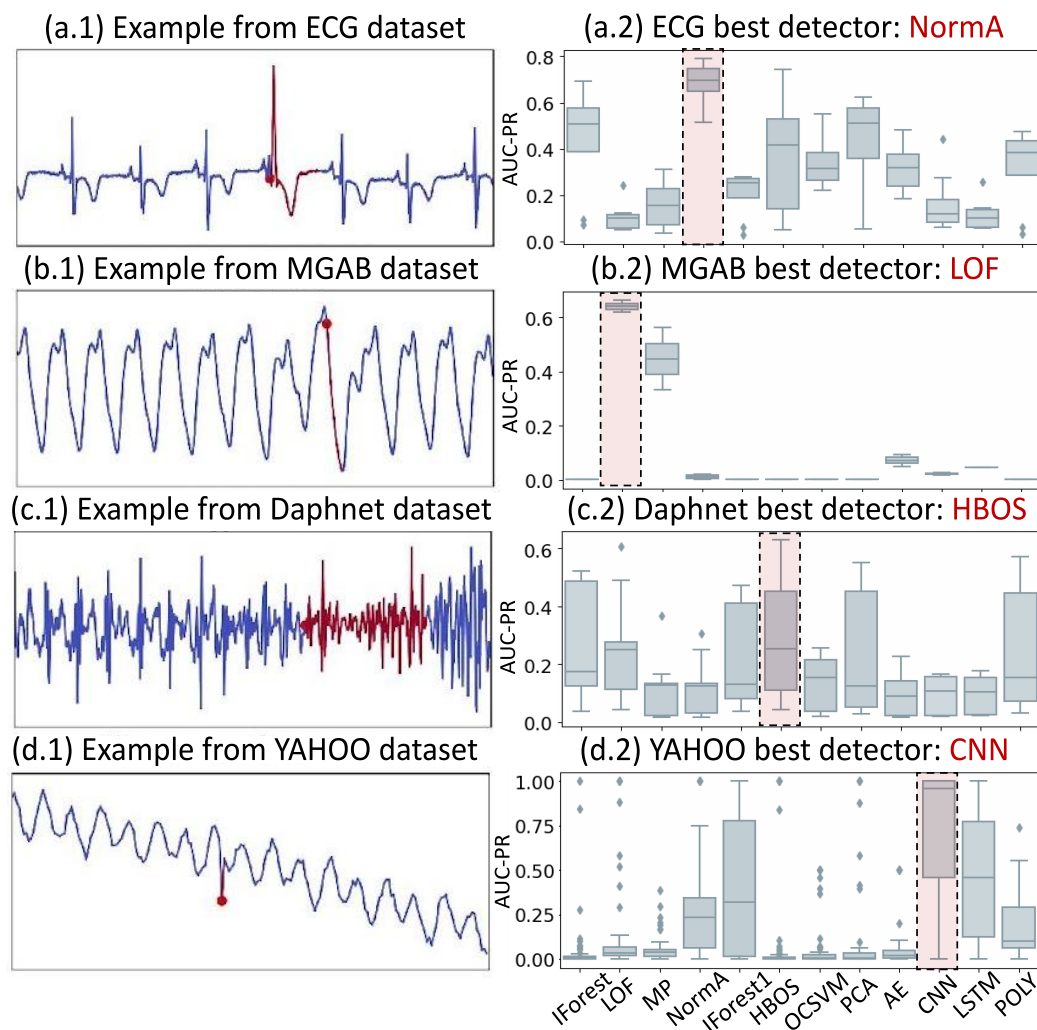


Figure 2.1: Accuracy of 12 anomaly detection methods on 4 datasets.

On top of the large variety of time series and anomaly characteristics mentioned above, time series can have distinct statistical characteristics, resulting in an even larger variability in the accuracy of anomaly detection methods. The latter can be the differences between *stationary* (i.e., with a constant distribution of values over time) and *non-stationary* (i.e., with a changing distribution of values over time) time series, or *single normality* (i.e., time series containing only one normal behavior) and *multiple normalities* (i.e., time series containing multiple normal behaviors) time series.

Chapter 3

Motivation and Problem

In this section, we describe solutions that can be applied to solve the limitations mentioned above, and we motivate the benefits of these solutions. We finally formally define the problem.

3.1 Ensembling Detectors

The first solution is to ensemble the anomaly scores produced by all the detectors. Multiple ensembling techniques have been proposed in the literature [5] from which two main methods arise: (i) *Averaging*: the average of the anomaly scores for each timestamp, (ii) *Maximizing*: the maximum anomaly score for each timestamp (iii) *Average of Maximum*: the average of the maximum for a randomly selected subset of detectors. *Averaging* strategy is proven to be the more robust and low-risk strategy compared to the other two [5]. Formally, the *Averaging* strategy is defined as follows:

Definition 1. *Given time series T of length n and a set of detectors \mathcal{B} , Averaging strategy is defined as $Avg\ Ens = [Avg_1, Avg_2, \dots, Avg_m]$ with Avg_i (for $i \in [1, m]$) equals to $Avg_i = (1/|\mathcal{B}|) \sum_{D \in \mathcal{B}} D(T)_i$.*

In the rest of the thesis, we call the *Averaging* strategy *Averaging Ensemble* (*Avg Ens*). As depicted in Figure 1.1 (a), which shows the accuracy of detectors (in grey) and the Averaging Ensemble (in orange), we observe that such a strategy already outperforms all existing approaches. Nonetheless, such a method requires running all detectors to produce one ensembled anomaly score, resulting in a costly execution time (see Figure 1.1 (b)). In a scenario with very long time series and an increasing number of detectors to consider, such an approach is not sustainable and feasible in practice.

3.2 Model Selection

A solution to tackle the limitations mentioned above is to apply model selection based on the characteristics of the time series. The main idea is to train a model to automatically select the best detector (anomaly detection method) for a given time series. In such a case, the user only has to run one model, drastically limiting the execution time required. This topic has been tackled in several recent papers related to AutoML (Automatic Machine Learning). Recent approaches, such as MetaOD [99, 97], explored meta-learning to identify the best outlier detection algorithm on tabular datasets. These research works rely on pre-computed accuracy performances of models on a subset of datasets to essentially learn a mapping from the characteristics of a dataset to detectors' accuracy performance. Methods have been proposed to select models in an unsupervised way [44], but require running multiple models in advance, which (like Averaging Ensemble) limit the applicability due to high cost.

3.3 Classification for Model Selection

In general, for the specific case of time series, most of the work described above and future AutoML methods will rely on time series classification methods for the model selection step. In such a case, the method aims to classify time series into classes corresponding to the available anomaly detection methods. One time series must be classified into the detector class that maximizes anomaly detection accuracy. However, no existing guidelines indicate which time series classification approach can be used as model selection. Thus, there is a need to evaluate and measure the benefit that time series classification approaches can bring to the anomaly detection task.

The first step is to evaluate the potential gain in accuracy that model selection could bring. To do this, recent time series anomaly benchmarks [74, 80] can be used. We can evaluate the accuracy upper bound that model selection methods reach on such benchmarks. Thus, we define a hypothetical model called *Oracle*, which, for a given time series, always selects the correct anomaly detector to use (i.e., the most accurate). Formally, *Oracle* is defined as follows:

Definition 2. *Given a dataset \mathcal{D} composed of time series T and labels L (with the length of the time series $|T| = n$ non-constant for all time series in \mathcal{D}), and a set of detectors $\mathcal{B} = \{D_1, D_i, \dots, D_m\}$ with the number of detectors defined as $|\mathcal{B}| = m$, $Oracle(T) = \operatorname{argmax}_{D \in \mathcal{B}} \{Acc(D(T), L)\}$*

In the rest of the thesis, we call *Oracle*, the hypothetical model $Oracle(T)$ applied to all T in a given benchmark. For example, figure 1.1 shows in white the accuracy of *Oracle* applied on the TSB-UAD benchmark [74] and demonstrates that a perfect model selection method outperforms the best detector in TSB-UAD and the Averaging Ensemble by a factor of 2.5. This large improvement

in accuracy and execution time confirms the potential benefits of model selection applied for time series anomaly detection. Thus, there is a need to evaluate the performances of existing time series classification methods when used as model selection algorithms and how close such methods can get to the *Oracle*.

3.4 Problem Formulation

Therefore, based on the limitations and the motivation listed above, we can formalize the problem of model selection as follows:

Problem 1. *Given a dataset \mathcal{D} composed of time series T (with the length of the time series $|T| = n$ non-constant for all time series in \mathcal{D}) and a set of detectors $\mathcal{B} = \{D_1, D_2, \dots, D_m\}$ with the number of detectors defined as $|\mathcal{B}| = m$, we want to build a model selection method \mathcal{M} that takes a time series $T \in \mathcal{D}$ and returns a detector $D \in \mathcal{B}$ (formally $\mathcal{M} : \mathcal{D} \rightarrow \mathcal{B}$) such that, for a given time series T and corresponding label L :*

$$\mathcal{M}(T) = Oracle(T) = \operatorname{argmax}_{D \in \mathcal{B}} \left\{ Acc(D(T), L) \right\}$$

Moreover, as the input of the model \mathcal{M} is a time series (of variable length) and the output is a detector D among a finite number of detectors \mathcal{B} , the problem can be seen as a time series classification problem for which the classes are the detectors in \mathcal{B} . Therefore, the only requirement is to have computed all $Acc(D(T), L)$ for all $T \in \mathcal{D}$ and all $D \in \mathcal{B}$ and use it as a training set.

3.5 Objectives

In summary, the goal of this experimental evaluation is to answer the following questions:

- **Classification as Model selection:** How do current time series classification methods compare to individual detectors and the *Oracle*?
- **Ensembling or selecting:** Is selecting detectors automatically more accurate than ensembling them?
- **Features or Raw values:** Should we use time series features or the raw time series values to predict which detector to use?
- **Out-Of-Distribution:** What happens when the model selection approach is trained on some datasets and tested on completely new datasets? Are all the answers from the previous questions valid?

We now describe our pipeline and experimental evaluation to answer the questions listed above.

Chapter 4

Proposed Pipeline

In the current chapter, we present in detail the proposed pipeline. The latter corresponds to a sequence of preprocessing and postprocessing steps such that the inputs of the model selection algorithms are equal in length. The proposed pipeline, illustrated in Figure 4.1, is composed of the following steps: (i) **Preprocessing step**: Extraction of the subsequences of same lengths (Figure 4.1 (b)), (ii) **Prediction step**: Prediction of which detector to use for each subsequence (Figure 4.1 (c)), and (iii) **Selection step**: Majority voting among all the different prediction to select one detector only (Figure 4.1 (d)). In the following section, we describe the three steps mentioned above in detail.

4.1 Preprocessing Step

Time series classification can be performed with three different strategies: (i) treating the entire time series as one sample, (ii) dividing the time series into overlapping subsequences, (iii) dividing the time series into shifting subsequences (i.e., non-overlapping subsequences). The first strategy is straightforward, as each time series is treated as a single observation. Nevertheless, not all classifiers can handle variable-length inputs, and training such models can be computationally intensive (i.e., batches of time series cannot be treated in parallel). The second strategy involves dividing the time series into overlapping subsequences (of a given window length ℓ). Despite possible loss of information, it forces each input of the methods to be the same length (ℓ), allowing simpler and faster computation when performed in parallel. In the third strategy, we divide time series into non-overlapping subsequences (of a given length ℓ), removing redundant information in overlapping subsequences. The latter might lead to separate anomalies into multiple windows, but significantly reduces the number of inputs generated by the second strategy and significantly accelerates the training and inference time. For these reasons, as illustrated in Figure 4.1 (a) and (b), we chose the third strategy.

Thus, the time series of length $|T|$ are divided into \mathbb{T}_ℓ non-overlapping subsequences of length ℓ . When the length of the time series is not divided evenly with

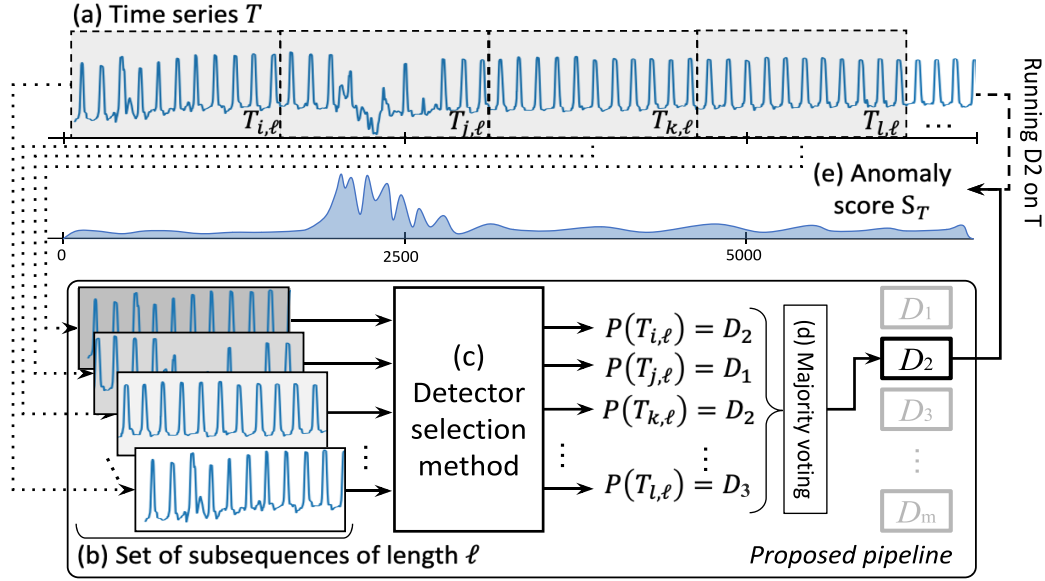


Figure 4.1: Proposed pipeline for the anomaly detection method selection

the window length ℓ , the remainder is added with an overlap between the first two windows. Formally, \mathbb{T}_l is defined as follows:

$$\mathbb{T}_l = \begin{cases} \{T_{i*\ell,\ell} | \forall i \in [0, \lceil \frac{|T|}{\ell} \rceil]\} & , \text{ if } \lceil \frac{|T|}{\ell} \rceil = \frac{|T|}{\ell} \\ \{T_{0,\ell}\} \cup \{T_{|T| - \lceil \frac{|T|}{\ell} \rceil + i*\ell,\ell} | \forall i \in [0, \lceil \frac{|T|}{\ell} \rceil]\} & , \text{ if } \lceil \frac{|T|}{\ell} \rceil < \frac{|T|}{\ell} \end{cases}$$

We expect the length ℓ to have a great impact on the anomaly detection accuracy. We thus test multiple length values and measure their influence (on accuracy and execution time) in Chapter 5.

At this point, we preprocessed the time series into subsequences of equal length. We now discuss the label (i.e., the best detector to apply) attribution. For that matter, we use the TSB-UAD benchmark [74] that contains 12 different anomaly detection methods. We compute the 12 methods for each time series and attribute the most accurate (based on AUC-PR) detector as the label. Then, the produced subsequences share the same label as the time series they originate from. This labeled dataset can be used to train classification methods and divided into the train, test, and validation sets. It is important to note that although each time series produces multiple samples (i.e., subsequences), these samples should not be mixed between train, validation, and test set. Indeed, too strong similarities between subsequences that belong to the same time series, if contained in both the train, validation and the test, can lead the classification model to overfit or create an illusion of accuracy. Therefore, we guarantee that the intersection between the train, validation, and test set, regarding which time series the corresponding subsequences originate from, is empty.

4.2 Time Series Classification Approaches

In this section, we describe the time series classifier approaches that we use as model selection methods. As many approaches have been proposed in the literature, we restrict our experimental evaluation to two main categories: (i) feature-based and (ii) raw-based methods. In addition, the second category can be divided into two sub-categories: (i) convolutional-based and (ii) transformer-based. It is worth noting that raw-based methods also utilize features for classification, although the extraction of these features is performed automatically within the network. Despite this, we classify them as raw-based due to the nature of their input. Figure 4.2 illustrates a simplified taxonomy of the methods considered, and we describe them in the following section.

4.2.1 Feature-based classification

The main idea regarding feature-based classification is to use the dataset of time series (or subsequences of time series) to create a dataset whose samples are described by features common to all samples. Using the feature-based dataset, we then employ traditional machine learning classifiers to classify each time series. We use the TSFresh [29] (Time Series FeatuRE Extraction on basis of Scalable Hypothesis tests) to extract each subsequence’s features. The latter is used for automated time series feature extraction and selection based on the FRESH algorithm [30]. More specifically, it automatically selects relevant features for a specific task. This is achieved using statistical tests, time series heuristics, and machine learning algorithms. The TSFresh package provides three options for automated feature extraction, namely (i) *comprehensive*, (ii) *efficient*, and (iii) *minimal*. The first two options provide 700 features and the latter provides only 9. For scalability reasons (the dataset transformation can reach millions of subsequences), we consider the *minimal* option in this work.

Moreover, the objective is not to evaluate Feature-based classifiers *per se*, but rather to evaluate the ability of TSFresh to extract meaningful features for time series classification (and model selection for anomaly detection, in particular). In this work, we consider the following classification approaches.

[SVC] A Support Vector Classifier (SVC) [24] is a classifier that maps instances in space in order to maximize the width of the gap between the classes. New instances are mapped into the same space and classified according to which side of the gap they fall.

[Bayes] The naive Bayes classifier [98] uses Bayes’ theorem to predict the class of a new instance based on prior probabilities and class-conditional probabilities. The prediction is made by computing the posterior probabilities for each class.

[MLP] A Multi Layer Perceptron (MLP) [48] is a fully connected (connections between every neuron) neural network.

[QDA] A Quadratic Discriminant Analysis (QDA) [41] Classifier is a linear discriminant analysis algorithm. The prediction is made by computing the discriminant functions for each class.

[AdaBoost] AdaBoost [39] is a boosting ensemble machine learning algorithm for solving classification problems. It creates a sequence of weak classifiers, where each classifier is trained on a weighted sample of the dataset. The prediction is made by combining the predictions of all classifiers, weighted by their accuracy.

[Decision Tree] A Decision Tree Classifier [51] is a tree-based method that represents a sequence of decisions based on the features of the dataset. To classify a new instance, the algorithm follows the decisions in the tree to reach a leaf node associated with a class.

[Random Forest] A Random Forest Classifier [49] is an ensemble machine learning algorithm that combines multiple decision trees, where each tree is built using a random subset of the features and a random sample of the data. The final class prediction for a new instance results from the aggregation of the predictions of all trees.

[kNN] A kNN classifier [37] is a method that classifies instances based on their distance to other instances in a training set. The algorithm assigns the new instances to the class with the most number of closest neighbors among the K nearest data points.

4.2.2 Raw-based classification

Instead of using features to perform classification, the raw values of the time series can be used. Indeed, whereas features are efficient for homogenizing time series datasets (e.g., setting a constant number of features for variable length time series), it might hide important information in the shape of consecutive values. Thus, many approaches that use raw-values time series have been proposed.

[Rocket] Among the recent raw-values methods, MiniRocket [32] is one of the state-of-the-art time series classification methods. The latter consists of a feature extraction step and a classification step. More specifically, MiniRocket works by transforming input time series using a small, fixed set of convolutional kernels and using the transformed features to train a logistic regression classifier (using stochastic gradient descent). We refer to MiniRocket as Rocket.

4.2.3 Convolutional-based classification

Convolutional-based approaches take as input raw-values of time series and have been shown to be accurate for time series classification [20].

[ConvNet] A Convolutional Neural Network (CNN) [70] is a type of deep learning neural network widely used in image recognition that is specially designed to extract patterns through data with a grid-like structure, such as images, or time series. A CNN uses convolution, where a filter is applied to a sliding window over the time series. The ConvNet architecture proposed in [91] is composed of three

stacked Convolutional blocks followed by Global Average Pooling (GAP), and a Softmax activation function. Each Convolutional block is composed of a convolutional layer (used with a kernel length of 3) followed by a batch normalization layer, and then a ReLU activation function is applied.

[ResNet] The Residual Network (ResNet) architecture [47] was introduced to address the gradient vanishing problem encountered in large CNNs [83]. A ResNet is composed of several blocks connected together with residual connections (i.e., identity mapping). For time series classification, a ResNet architecture has been proposed in [91], and has demonstrated strong classification accuracy [34]. It is the same architecture as the previously described ConvNet, with additional residual connections between convolutional blocks.

[InceptionTime] The model consists of a network using residual connections and convolutional layers with kernels of variable lengths [35]. Such a network uses three Inception blocks that replace the traditional residual blocks that we can find in a ResNet architecture. Each Inception block consists of a concatenation of convolutional layers using different sizes of filters. For each block, the time series is fed to three different 1D convolutional layers with different kernel sizes (10, 20, and 40) and one Max-Pooling layer with kernel size 3. The last step consists of concatenating the previous four layers along the channel dimension and applying a ReLU activation function to the output, followed by batch normalization. The convolutional layers have 32 filters and a stride parameter of 1.

4.2.4 Transformer-based classification

The second category, initially introduced for natural language processing and computer vision tasks [88, 33], is Transformer-based approaches. Such methods can easily be adapted for time series classification tasks, and in this work we propose SiT (Signal Transformer), an extension of a recent computer vision transformer approach [33]. SiT first starts by projecting the input to the latent space with an embedding step. After the embedding step, the input is mapped to a D dimensional space (we use $D = 256$ in the rest of the thesis) that serves as input to an encoder. For SiT, we use an encoder originally proposed for computer vision tasks [88] that consists of multiple blocks. Each block has an alternating multi-headed self-attention block and a feed-forward layer, both preceded by a normalization step and a residual connection. We now describe the different embedding steps in detail in the following paragraphs. In the experimental evaluation, we consider the SiT architecture with the four embeddings as four different methods.

[SiT-conv] This embedding uses a single convolutional layer to map the time series into the latent space. The convolutional layer has a kernel and stride of the same length (we use a length of 16 throughout the rest of the thesis), essentially taking non-overlapping steps over the time series. Finally, the convolutional layer has D filters to match the input dimension of the SiT encoder.

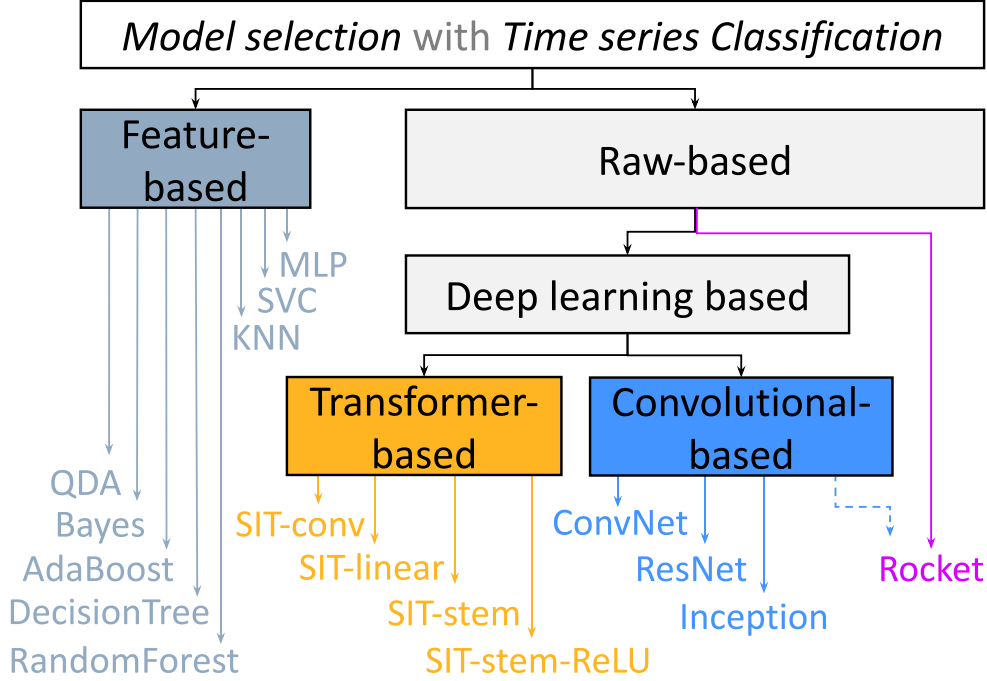


Figure 4.2: Taxonomy of time series classification approaches used as model selection methods. We use the same color code for each class in all figures in the thesis.

[SiT-linear] The linear embedding [33] splits the input time-series into non-overlapping subsequences of length l_{sit} (we use $l_{sit} = 16$ in the rest of the thesis). Then, each patch is linearly projected into D dimensions to match the input dimension of the SiT encoder.

[SiT-stem] The stem embedding [93] consists of 3 convolutional layers with a kernel length of 3, a stride length of 2, and a number of filters equal to 3, 5, and 7, respectively. These three convolutional layers are then followed by a last convolutional layer with D dimensions and a kernel and stride length equal to 1. This embedding was initially proposed to overcome unstable behavior while training because of its early visual processing step.

[SiT-stem-ReLU] Similarly to the previous embedding, the stem-ReLU embedding [90] consists of 4 convolutional layers with kernel lengths of 7, 3, 3, 8, stride lengths of 2, 1, 1, 8, and padding of 3, 1, 1, 0. The number of filters for each convolutional layer is 3, except the last one that has D filters to match the dimension SiT encoder.

4.3 Selecting the Detector

We train the time series classification methods mentioned in the previous section to predict the best detector for each subsequence (as shown in Figure 4.1 (c)).

However, there is no guarantee that the classification model selects the same detector for all subsequences. Therefore, we choose the best detector for one time series by doing a majority voting step between the predictions for every subsequence, such that the most voted detector is selected as the detector of the time series. Formally, given a classification model \mathcal{M}_{cl} applied on a given time series T subsequences \mathbb{T}_ℓ , we define $\mathcal{M}_{cl}(\mathbb{T}_\ell)$ the set of model selected for each subsequence in \mathbb{T}_ℓ . Therefore, we define the majority voting function as follows:

$$f_{MV}(T, \mathcal{M}_{cl}) = \operatorname{argmax}_{D \in \mathcal{M}_{cl}(\mathbb{T}_\ell)} \sum_{T_{i,\ell} \in \mathbb{T}_\ell} \mathbb{1}_{[\mathcal{M}_{cl}(T_{i,\ell})=D]}$$

Majority voting serves the pipeline with two significant factors, (i) it does not depend on the design of the detector and makes the pipeline easily usable for multiple different types of anomaly detection methods, and (ii) majority voting averages the predictions and reduces the impact of misclassified subsequences. To conclude, in our pipeline, the model selection method introduced in Problem 1 is the output of $f_{MV}(T, \mathcal{M}_{cl})$.

Chapter 5

Experimental Evaluation

We now describe in detail our experimental analysis. For additional information, we make all our material publicly available online [15] and provide an interactive WebApp [16] for navigating and exploring the experimental results.

5.1 Experimental Setup and Settings

Technical setup: We implemented the deep learning-based model selection methods in Python 3.5 using the PyTorch library [75]. For the feature-based approach, we used the TSFresh [29] and scikit-learn [76] libraries. We then used sktime [64] for the rocket algorithm implementation. For the anomaly detection methods, we used the implementation provided in the TSB-UAD benchmark [74]. The evaluation was conducted on a server with Intel Core i7-8750H CPU 2.20GHz x 12, with 31.3GB RAM, and Quadro P1000/PCIe/SSE2 GPU with 4.2GB RAM, and on Jean Zay cluster with Nvidia Tesla V100 SXM2 GPU with 32 GB RAM.

Datasets: For our evaluation purposes, we use the public datasets identified in the TSB-UAD benchmark [74]. The latter corresponds to 16 datasets (described in Table 5.1) proposed in the literature containing 1900 time series with labeled anomalies. Specifically, each point in every time series is labeled as normal or abnormal.

Anomaly Detection Methods: For the experimental evaluation, we select 12 different anomaly detection methods, summarized in Table 5.1. Out of these, 8 are fully unsupervised (i.e., they require no prior information on the anomalies to be detected): IForest, IForest1, LOF, MP, NormA, PCA, HBOS, and POLY. The remaining 4 methods are semi-supervised (i.e., they require some information related to normal behaviors), namely, OCSVM, AE, LSTM-AD, and CNN. For all these anomaly detection baselines, we set the parameter as described in the TSB-UAD benchmark [74].

Method Selection baselines: We then consider the method selection baseline described in Chapter 4 and summarized in Table 5.1. We first consider *feature-based* methods, that extract features using TSFresh [29] library to select the correct

anomaly detection method. We then consider rocket, state-of-the-art *time series classifier*. We also include two types of deep learning classifiers; (i) *Convolutional-based neural networks* and (ii) *Transformer-based neural networks*. Table 5.1 summarizes the different model selection methods (i.e., classifiers). In total, we consider 16 methods, trained with window lengths ℓ equal to 16, 32, 64, 128, 256, 512, 768, and 1024. In total, we trained 128 models. In the following section, we refer to a model M trained using a window length ℓ as $M-\ell$.

Parameter settings: We use the same 70/30 split of the benchmark for all the classification models. Therefore, we can compare models trained on the same training set and evaluated on the same set of time series. Then, for the feature-based methods, we set the hyperparameters of the models based on the default parameters of scikit-learn. Moreover, for rocket, we use 10000 kernels to extract the features and the logistic regression with stochastic gradient descent (computed in batches) for the classification step. Finally, for Convolutional and Transformer-based methods, we use a learning rate of 10^{-5} , with a batch size of 256 and an early stopping strategy with a maximum of 50 epochs without improvement. Moreover, we use the weighted cross-entropy loss and set the maximum number of epochs to 10,000 (with a training time limit of 20 hours).

Evaluation measures: We finally use four evaluation measures. For model selection accuracy, we use the classification accuracy (i.e., the number of anomaly detectors correctly selected divided by the total number of time series). For anomaly detection accuracy, we use both AUC-PR [31] and VUS-PR [73] (with a buffer length equal to 10 points). For execution time, we measure the *training time* (i.e., the time required to train a model selection algorithm), the *selection time* (i.e., the time a model selection approach needs to predict which detector to use), and the *detection time* (i.e., the time required to predict which detector to use, and to execute it).

5.2 Overall Evaluation

We first conduct an extensive evaluation of accuracy (classification and anomaly detection) and execution time for all model selection methods over the entire benchmark. Thus, we split the benchmark into one training and testing set. The first contains 1404 time series and the second 496. Both sets contain time series from all datasets. Therefore, the models have examples of all available domains. In Chapter 5.6, we evaluate the performance of model selection when applied to unseen (i.e., not used in the training set) datasets.

5.2.1 Accuracy Evaluation

We first analyze the accuracy of all model selection methods (using all window lengths) and compare them to the Oracle, the Averaging Ensemble method, and anomaly detection methods in the TSB-UAD benchmark.

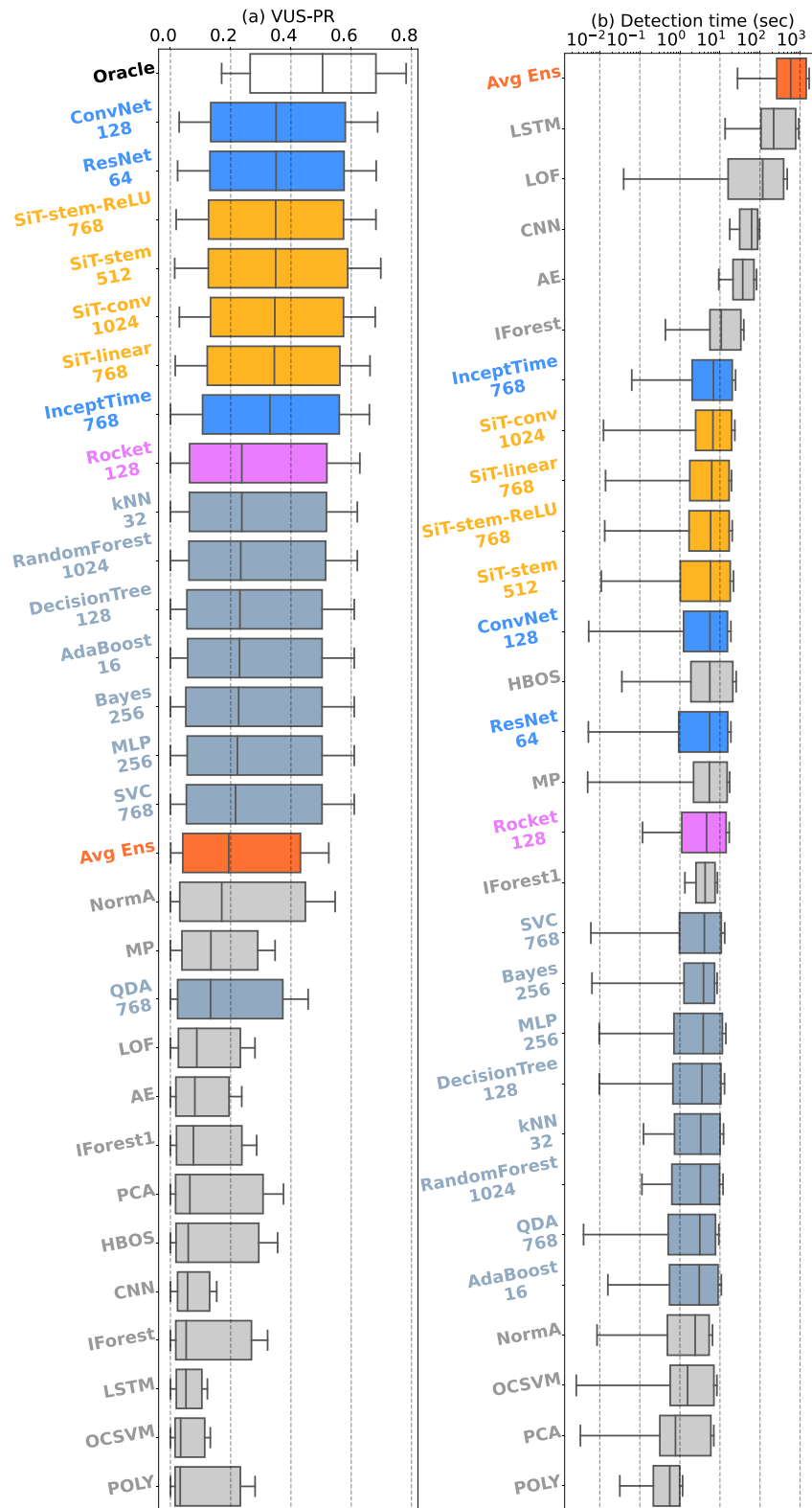


Figure 5.1: VUS-PR and Detection time (seconds) for all model selection approaches (showing only the window length that maximizes VUS-PR for each model) over a test set of 497 series from TSB-UAD. The most accurate methods are at the top (a); the fastest methods are at the bottom (b)

Figure 5.1 (a) depicts the overall VUS-PR over the entire TSB-UAD benchmark (i.e., each box-plot corresponds to 497 accuracy values for the 497 time series into the test set). The Convolutional-based approaches are in dark blue, the Transformer-based approaches are in yellow, the Feature-based approaches are in light blue, Rocket models are in violet, and the anomaly detection methods of the TSB-UAD benchmark are in light grey. The oracle is the top box plot (in white), and the Averaging Ensemble is the orange box plot. The box-plots are sorted based on the median value. In total, we compare 142 models on 497 time series, and the complete results can be found in Figure 5.8. In Figure 5.1, we depict only the models with the window length that leads to the best VUS-PR.

First, we observe that almost all model selection methods outperform the existing anomaly detection methods. We also see that most model selection methods outperform the Averaging Ensemble approach. Thus, we can conclude that model selection using time series classifiers significantly improves the state-of-the-art methods.

More interestingly, we observe a partition in the ranking of the methods. First, Convolutional and Transformer-based approaches produce equivalent accuracy values and represent the top-48 methods (see Figure 5.8). However, whereas all the Convolutional-based methods are in the top-48, a few of the Transformer-based approaches are further away in the ranking. Moreover, the first non-deep learning method is *rocket-128* (ranked 49th), followed closely by *knn* models. We also observe that the *rocket* approaches are very spread across the ranking (*rocket-128* is ranked 50th, and *rocket-16* is ranked 124th). This implies that the choice of window length strongly impacts accuracy. Overall, we note that the best selection model is two times more accurate than the best anomaly detection method in TSB-UAD.

Then, we also note that all the model selection methods are significantly less accurate than the Oracle. For example, in Figure 5.1 (a), there is a gap of 0.2 VUS-PR between the Oracle and the best model selection method. Such a gap is significant and indicates a large margin of improvement for future work. We also note that all model selection approaches produce accuracy values between 0 and 1 (as shown by each box-plot in Figure 5.1 (a)). This is caused by the large heterogeneity of individual detectors' performances (for some datasets and time series, none of the detectors are accurate). This means that no model selection method is guaranteed to perform above a given accuracy value. Making model selection more stable and robust is essential for several use cases.

5.2.2 Model selected distribution

We then inspect in detail the prediction and the detector chosen by the model selection approaches. In this section, we consider only *resnet-1024*, *convnet-128*, *sit-stem-512*, *rocket-128*, and *knn-1024*. These approaches are the best models (using either AUC-PR or VUS-PR) based on the analysis conducted in Chapter 5.2 (you may find additional information on AUC-PR evaluation in our website [16]).

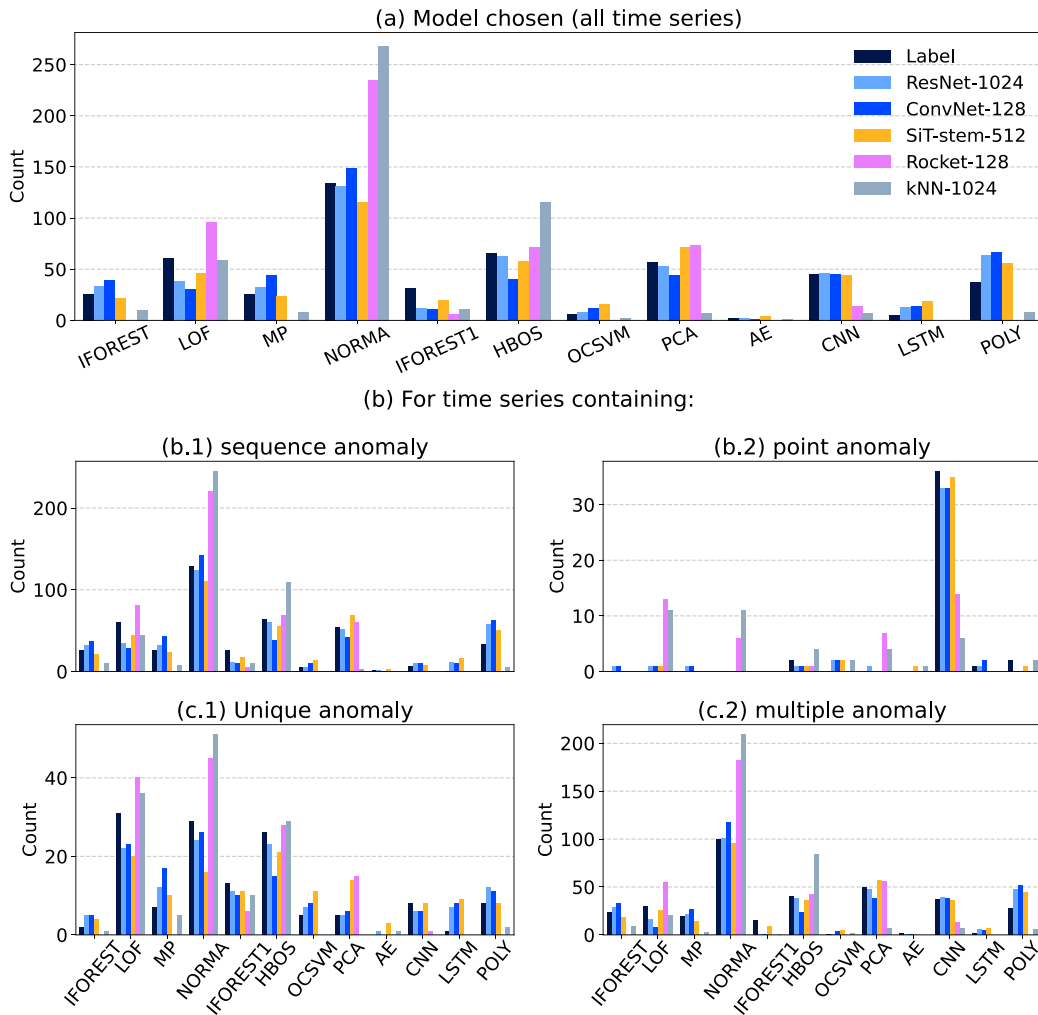


Figure 5.2: Distribution of the selected models for five models (the best for each category) compared to the distribution of the labels (in black). Difference of distributions between time series containing (b) sequence and point anomalies, and (c) unique or multiple anomalies.

Figure 5.2 (a) depicts the distribution of the chosen detectors by the 5 model selection approaches mentioned above for the entire TSB-UAD benchmark. The black bar corresponds to the true labels (i.e., the best detectors). We observe from Figure 5.2 (a) that *rocket-128* and *knn-1024* are significantly overestimating the detector NormA (as well as LOF for *rocket-128* and HBOS for *knn-1024*), whereas *resnet-1024*, *convnet-128*, and *sit-stem-512* are matching the correct distribution of detectors (we observe a slight underestimation of LOF, IFOREST1 and an overestimation for POLY).

Moreover, we measure the prediction distribution differences for time series containing sequence anomalies (Figure 5.2 (b.1)) and point anomalies (Figure 5.2 (b.2)), and for time series containing only one anomaly (Figure 5.2 (c.1)) and multiple anomalies (Figure 5.2 (c.1)). We first observe that predictions of model selection methods are significantly different for time series with sequence and point anomalies. More specifically, *resnet-1024*, *convnet-128*, and *sit-stem-512* are correctly selecting the method CNN, whereas *rocket-128* and *knn-1024* are over selecting LOF and NormA for time series containing point anomalies. However, for sequence anomaly, as it represents most of the TSB-UAD benchmark, the prediction distribution is similar to the one over the entire benchmark. Moreover, the correct predictions of *resnet-1024*, *convnet-128*, and *sit-stem-512* for time series containing point anomalies are interesting, as this information is not provided in the training step. Therefore, these models found discriminant features in the time series that indicate whether it might contain a point or a sequence anomaly.

We, finally, measure the differences between the prediction distribution of model selection methods between time series containing unique and multiple anomalies. The true labels (black bars in Figure 5.2 (c.1) and (c.2)) indicate that, for unique anomalies, the best detectors are LOF, NormA, and HBOS and for multiple anomalies, the best detector is NormA. We observe that all model selection approaches select LOF, NormA, and HBOS correctly for time series containing a unique anomaly. The latter indicates that model selection methods can extract discriminant features that indicate if one time series is more likely to have multiple anomalies.

5.2.3 Execution Time Evaluation

We now discuss the execution time of model selection methods. In this section, we focus only on the detection time (i.e., the number of seconds required by a method to predict the detector to use and to run it). Figure 5.1 (b) depicts the detection time (in log scale) for each method and detector in the TSB-UAD benchmark (complete results in Figure 5.8 (b)). We first observe that the Averaging Ensemble required to run all detectors is significantly slower than the rest. Then, all model selection methods are of the same order of magnitude as the detectors. We also observe that all the deep learning methods are slower than the feature-based approaches. This is surprising because the detection time mainly depends on the chosen detector. Overall, we conclude that method selection is the only viable

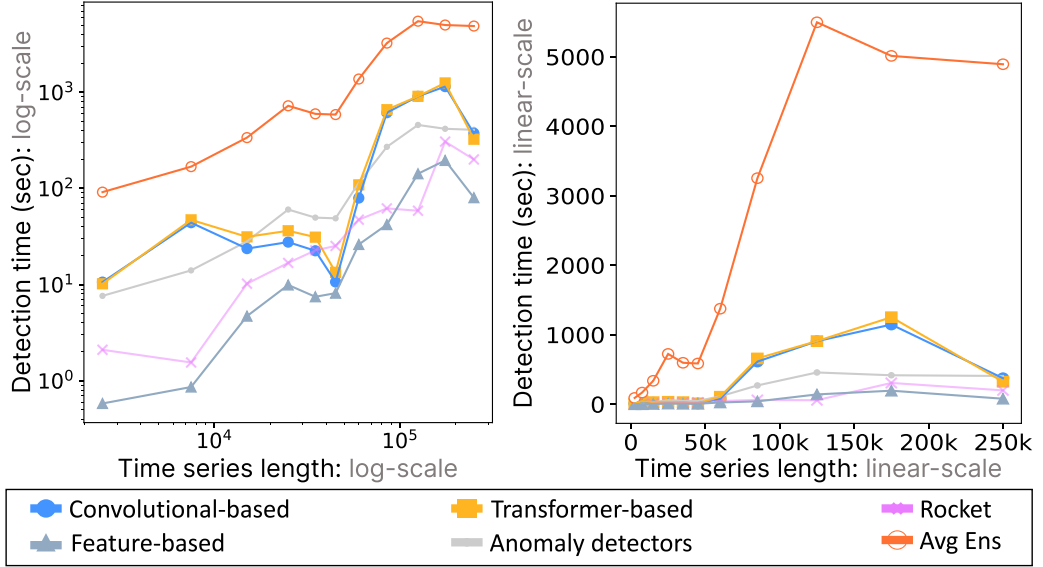


Figure 5.3: Execution time vs. length of model selection methods.

solution that outperforms the existing anomaly detection methods and can be executed in the same order of magnitude of time.

Finally, we depict in Figure 5.3 the scalability of model selection methods versus individual detectors and the Averaging Ensemble approach when the time series length increases. We observe that, on average, the execution time of model selection approaches increases similarly to the execution time of individual detectors when the time series length increases. We also observe that the Averaging Ensemble approach execution time is significantly impacted by the time series length. The latter shows the scalability issue of the Averaging Ensemble approach for very large time series.

5.3 Influence of the Window Length

In this section, we analyze the influence of the window length on classification accuracy (Figure 5.4 (a.1)), anomaly detection accuracy (Figure 5.4 (a.2) and (a.3)) and execution time (Figure 5.4 (b)). We perform the analysis per group of methods (i.e., average performances for Convolutional, Transformer, rocket, and Feature-based methods).

We first observe in Figure 5.4 (a) that Convolutional-based and Transformer-based methods outperform the best anomaly detection methods (green dash-dotted line in Figure 5.4 (a.2) and (a.3)), the Averaging Ensemble approach (orange dotted line in Figure 5.4 (a.2) and (a.3)), Rocket and Feature-based methods, whatever the length used with regard to the classification accuracy, VUS-PR, and AUC-PR. We also observe that Transformer-based approaches are less accurate for shorter

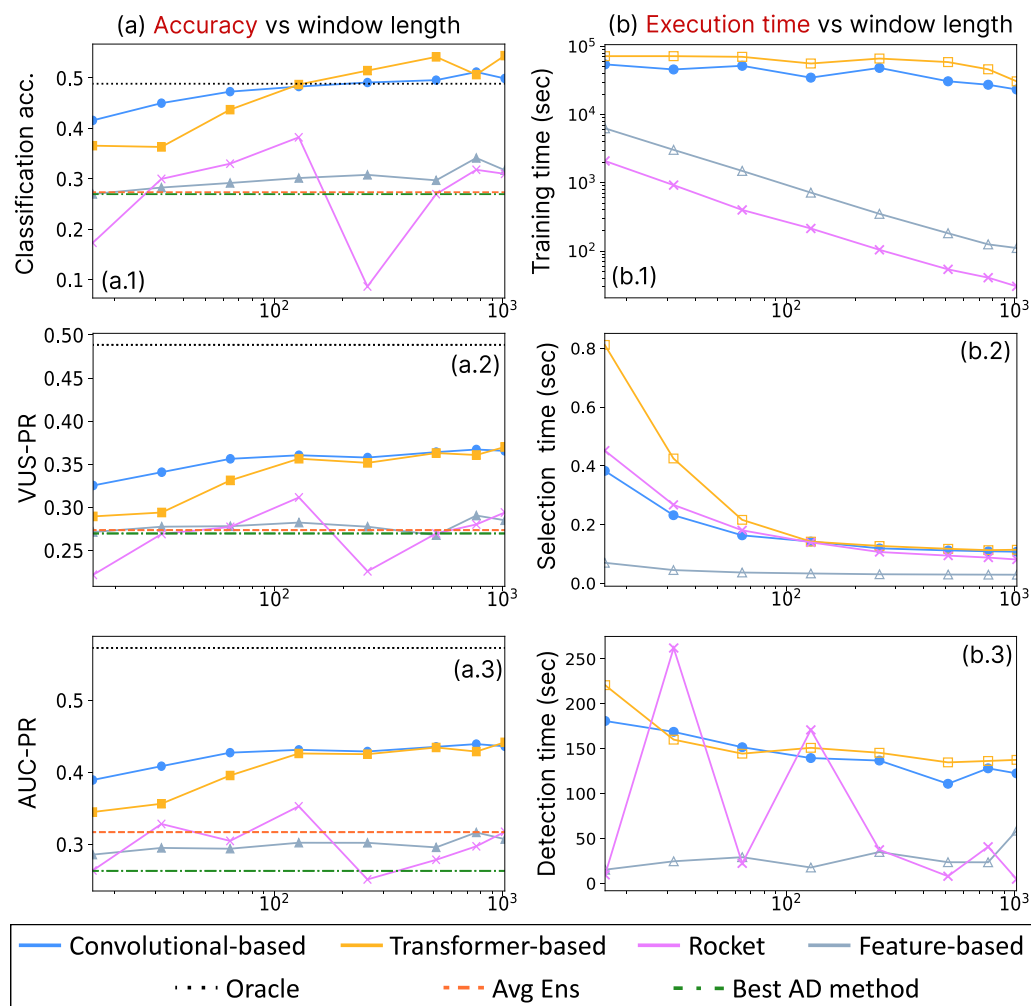


Figure 5.4: (a) Accuracy ((a.1) classification accuracy, (a.2) VUS-PR and (a.3) AUC-PR) and (b) execution time ((b.1) training time, (b.2) selection time and (b.3) detection time) versus window length ℓ .

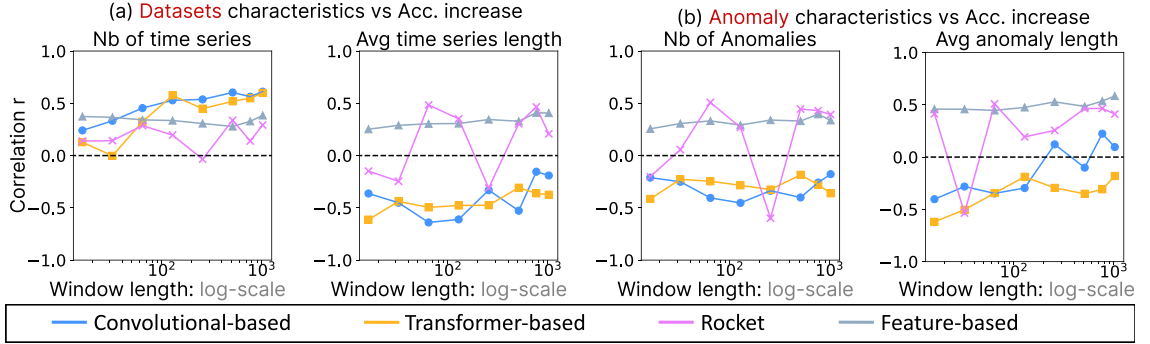


Figure 5.5: Correlation between accuracy and time series characteristics vs. the window length used to train the model selection methods.

lengths (less than 100 points), whereas, the accuracy of Convolutional-based approaches is stable regardless of the window length. Overall, Transformer and Convolutional-based approaches converge to the same anomaly detection accuracy (both for VUS-PR and AUC-PR) when the window length increases.

Furthermore, we observe that both rocket and Feature-based approaches are significantly faster to be trained than Convolutional and Transformer-based approaches (Figure 5.4 (b.1)). We make the same observation for selection time ((Figure 5.4 (b.2))). For the detection time, we observe that rocket execution time is very unstable when compared to the other approaches. The latter means that the choice of length strongly impacts the model selection performed by rocket, leading to very diverse selection and execution times.

In the general case, we can make the following two statements: (i) Large window length leads to faster selection time for the model selection process and better accuracy for Convolutional and Transformer-based approaches. (ii) Feature-based approaches are significantly faster but less accurate than Convolutional-based and Transformer based approaches, whatever the length used.

5.4 Influence of Datasets and Anomaly Types

In this section, we evaluate the influence of datasets and anomaly characteristics on model selection accuracy. We perform the analysis per group of methods (i.e., average performances for Convolutional, Transformer, Rocket, and Feature-based methods).

For this experiment, we evaluate the dataset and anomaly characteristics (i.e., the number of time series, the average length of the time series, the average number of anomalies and the average anomaly length). Figure 5.5 depicts these characteristics (x-axis) versus the average increase of accuracy (VUS-PR of the model selection method subtracted by VUS-PR of the best anomaly detection method for each dataset) for each model selection method using a given window length. For instance, if a point (one model selection method on one dataset) is positive

(above the black dotted line), thus the corresponding model is more accurate on the corresponding dataset than the best anomaly detection method selected on this same dataset. We generally observe low correlations between dataset and anomaly characteristics (i.e., $-0.6 < r < 0.6$). With such correlation values, we cannot conclude any factual statement on the impact of these characteristics and the model selection methods' performances. However, we can make the following observations.

First, Figure 5.5 (a) shows that the number of time series is impacting more substantially Convolutional and Transformer-based approaches with large window lengths. For the average time series length, only Feature-based approaches are positively impacted. On the contrary, Convolutional and Transformer-based approaches are less accurate when the average time series length is increasing. These observations imply that Convolutional and Transformer-based are more affected by the number of examples in the dataset rather than the length of each instance. In contrast, Feature-based approaches benefit from both more and large instances.

Then, Figure 5.5 (b) shows that Feature-based approach accuracy is increasing with the anomaly characteristics, whereas these characteristics either do not or negatively impact Convolutional and Transformer-based methods. More specifically, we observe that Feature-based approaches (regardless of the window length) are more accurate with time series containing large anomalies, and Convolutional-based approaches are less accurate (irrespective of the window length) when the number of anomalies increases.

We note that Rocket's correlation with the dataset and the anomaly characteristics is unstable. The latter is explained by the fact that the model prediction of Rocket is very sensitive to the window length (as described in Chapter 5.3). Thus, it is impossible to make a conclusion on Rocket's performances, datasets, and anomalies.

5.5 Detection vs Classification Accuracy

In this section, we analyze the relationship between the model selection methods' classification accuracy and the resulting anomaly detection accuracy. In this experiment, we consider VUS-PR as anomaly detection measures. For this experiment, we extend the definition of *Oracle* (introduced in Chapter 3) as follows:

Definition 3. *For a given dataset \mathcal{D} , we define $Oracle_{k,j}$ as a hypothetical model selection method that has a classification accuracy of $k \in [0, 1]$ and selects the j^{th} best detector (among m detectors) in cases of misclassification. Thus, $Oracle_{1,1}$ always selects the best detector, and $Oracle_{0,m}$ always selects the worst detector. Finally, we define $Oracle_{k,R}$ as the model selection method that has a classification accuracy of $k \in [0, 1]$ and randomly selects a detector in misclassification cases.*

Figure 5.6 depicts the latter comparison for all datasets (Figure 5.6 (a)), and for two specific datasets (Figure 5.6 (b)). We first observe a strong correlation

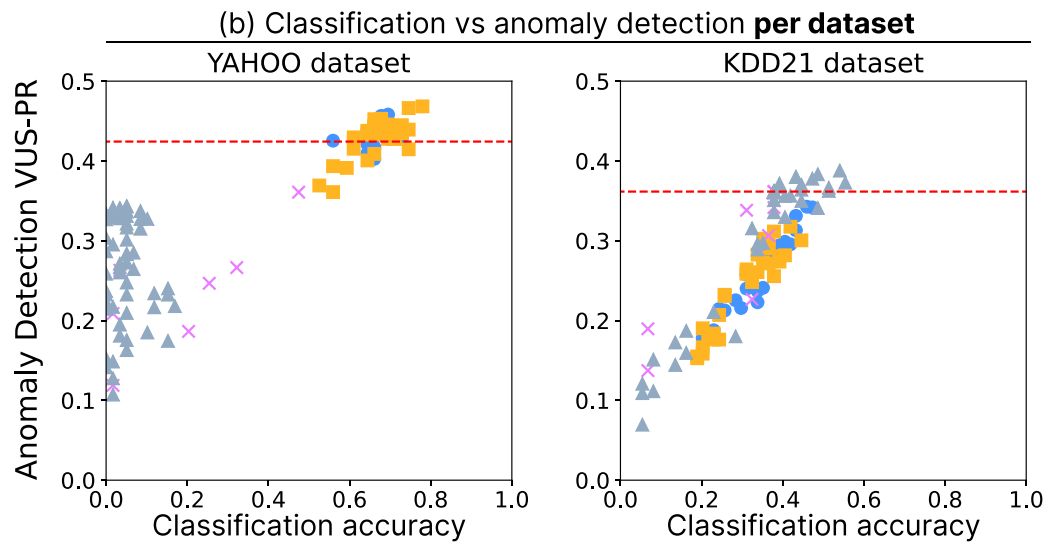
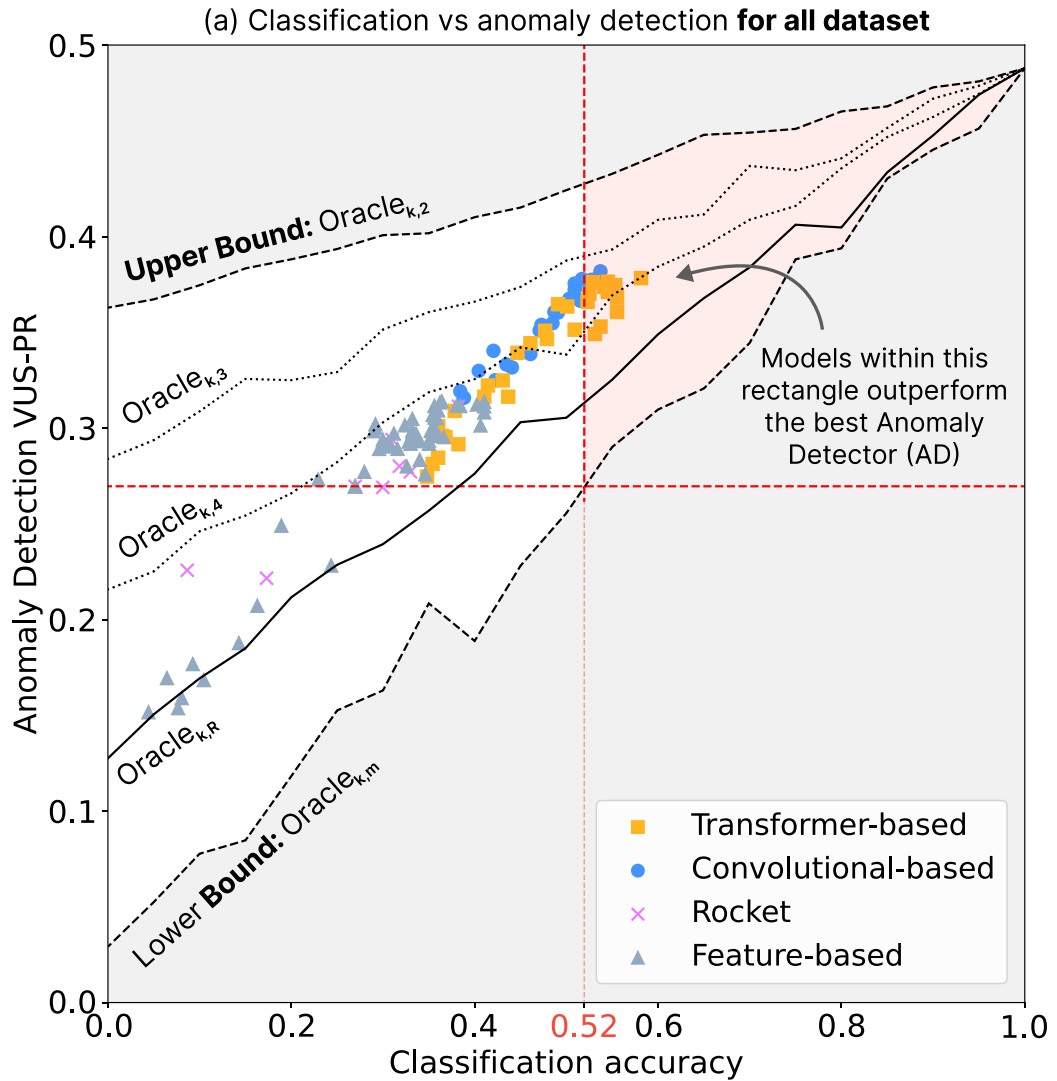


Figure 5.6: Classification accuracy versus anomaly detection accuracy (VUS-PR) for (a) all datasets and (b) two specific datasets.

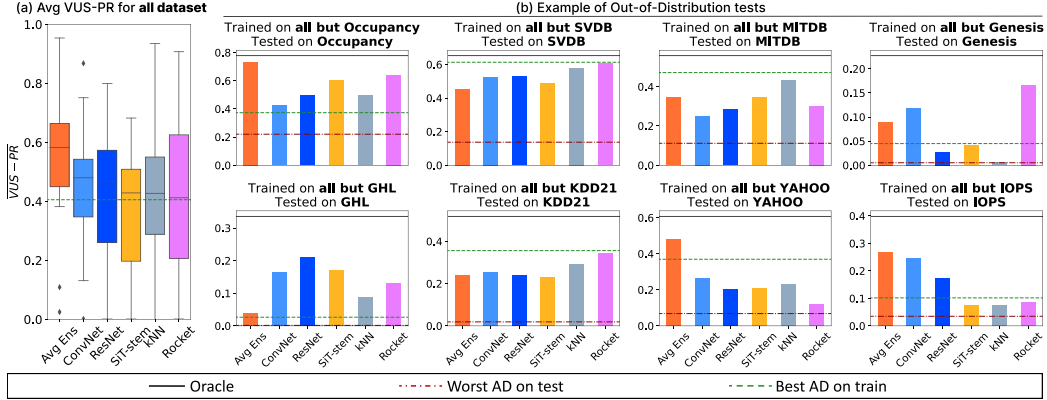


Figure 5.7: Out-of-distribution experiment, when model selection algorithms are trained on all but one dataset. (a) results for each dataset (when not included in the training set) and (b) average results.

between classification accuracy and anomaly detection accuracy for each specific dataset and, on average, for all datasets. However, methods belonging to different families (e.g., Feature-based or Transformer-based) are not performing the same. For instance, Figure 5.6 (a) shows that Feature-based approaches are not accurate for the YAHOO dataset, but are the best models for the KDD21 dataset. Overall, we observe that Convolutional-based and Transformer-based are both more accurate in classification and anomaly detection (Figure 5.6 (b)).

We also depict in Figure 5.6 (a) the lines corresponding to $Oracle_{k,2}$, $Oracle_{k,3}$, $Oracle_{k,4}$, $Oracle_{k,R}$, and $Oracle_{k,m}$. For a given classification accuracy, k , $Oracle_{k,2}$, and $Oracle_{k,m}$ correspond to the upper and lower bounds. The latter means that model selection approaches with a given classification accuracy will be within the previously mentioned upper and lower bounds for VUS-PR (i.e., in the grey zone in Figure 5.6 (a)). Therefore, for the TSB-UAD benchmark, any model selection method that has a classification accuracy above 0.53 (intersection between the two dashed red lines) is better than the current best anomaly detection method in TSB-UAD (i.e., the best AD in Figure 5.6 (b)). In our experiments, this is the case only for a few Convolutional- and Transformer-based methods.

Moreover, we compare the positions of the model selection methods with regard to $Oracle_{k,2}$, $Oracle_{k,3}$, and $Oracle_{k,R}$. We observe in Figure 5.6 (b) that almost all methods are above $Oracle_{k,R}$. The latter means that when the wrong detector is selected, the model selection methods do not select detectors randomly. Moreover, we observe that most models follow the $Oracle_{k,4}$ line. The latter indicates that the models averagely select the third-best in case of misclassification. Finally, the observations discussed above demonstrate three important statements: (i) classification accuracy can be used as a proxy for anomaly detection accuracy, and without computing the anomaly detection accuracy, we can provide an anomaly detection accuracy lower and upper bounds; (ii) the gap between the best model

selection and the top right corner of the grey zone shows that there is a significant margin for improvement for future work; (iii) the vertical gap between the models and the upper bound ($Oracle_{k,2}$) shows that there is an important margin of improvement in the prediction rank: a model with the same classification accuracy can gain up to 0.1 VUS-PR if it better selects models.

5.6 Out-of-Distribution Experiments

At this point, we tested the performances of the model selection methods when trained on a subset of the benchmark with examples from all 16 datasets available. In some cases, though, we may want to analyze time series that are not similar to any of those in the benchmark. Therefore, in this section, we measure the ability of the model selection methods to be used in an unsupervised manner (i.e., used for datasets that are not similar to the one used in the training set). We run the following experiment. We train the model selection methods on 15 datasets (70% percent of the time series for training and the other 30% for validation), and we test on the remaining one. We try all 16 possible test partitions, and (for brevity) report 4 of these tests in Figure 5.7(a). We only show the results for the best-performing model selection methods listed in Chapter 5.2.2.

Figure 5.7 (a) depicts the normalized VUS-PR (noted $\overline{VUS-PR}$) for all 16 tests: VUS-PR of 1 corresponds to the VUS-PR of the *Oracle* on each test, while 0 corresponds to the worst anomaly detection methods on each test. This figure shows that, in the unsupervised case, the Averaging Ensemble is outperforming all model selection methods, as well as the best anomaly detection method based on the accuracy performance measured on the train set (dotted green line in Figure 5.7 (a)). The latter means that, for unknown datasets, it is safer to run all existing anomaly detection methods and average their scores. Knowing that such ensembling methods are not scalable (as shown in Figure 5.1), Figure 5.7 (a) shows that ConvNet or ResNet is still a better choice than choosing the best anomaly detection method selected on train data (i.e., known data). However, kNN, Rocket, and SiT-stem are only slightly more accurate than the best anomaly detection method.

Figure 5.7 (b) depicts the average accuracy for 8 out of the 16 test (i.e., dataset not included in the training set and used for the test). We observe very different results. First, for Electrocardiograms (SVDB), none of the model selection methods and the Averaging Ensemble outperforms the best anomaly detection method (selected on the training set). But, most model selection methods or Averaging Ensemble outperform the best anomaly detection method for sensor data of different kinds (GHL and Occupancy) on the train. The latter can be explained by the fact that ECGs contain less heterogeneous behaviors (i.e., repetitive normal behavior and similar anomalies) than measurements from sensors data, and it is more likely to have in the benchmark one method that would perform very well on all time series. This is confirmed by the performance of the best anomaly detection

method being usually very close to the oracle for SVDB.

These observations lead to the following remarks: (i) there is a significant margin of improvement when using the existing time series classifiers as model selection methods in the unsupervised case; (ii) when a new dataset arrives, it is safer in the general case to use an ensembling method such as the simple average of all anomaly scores; and (iii) for heterogeneous datasets (without known and repetitive normal or abnormal patterns), classifiers as model selection (mainly convolutional-based classifiers) can be used even though similar time series are not in the training set.

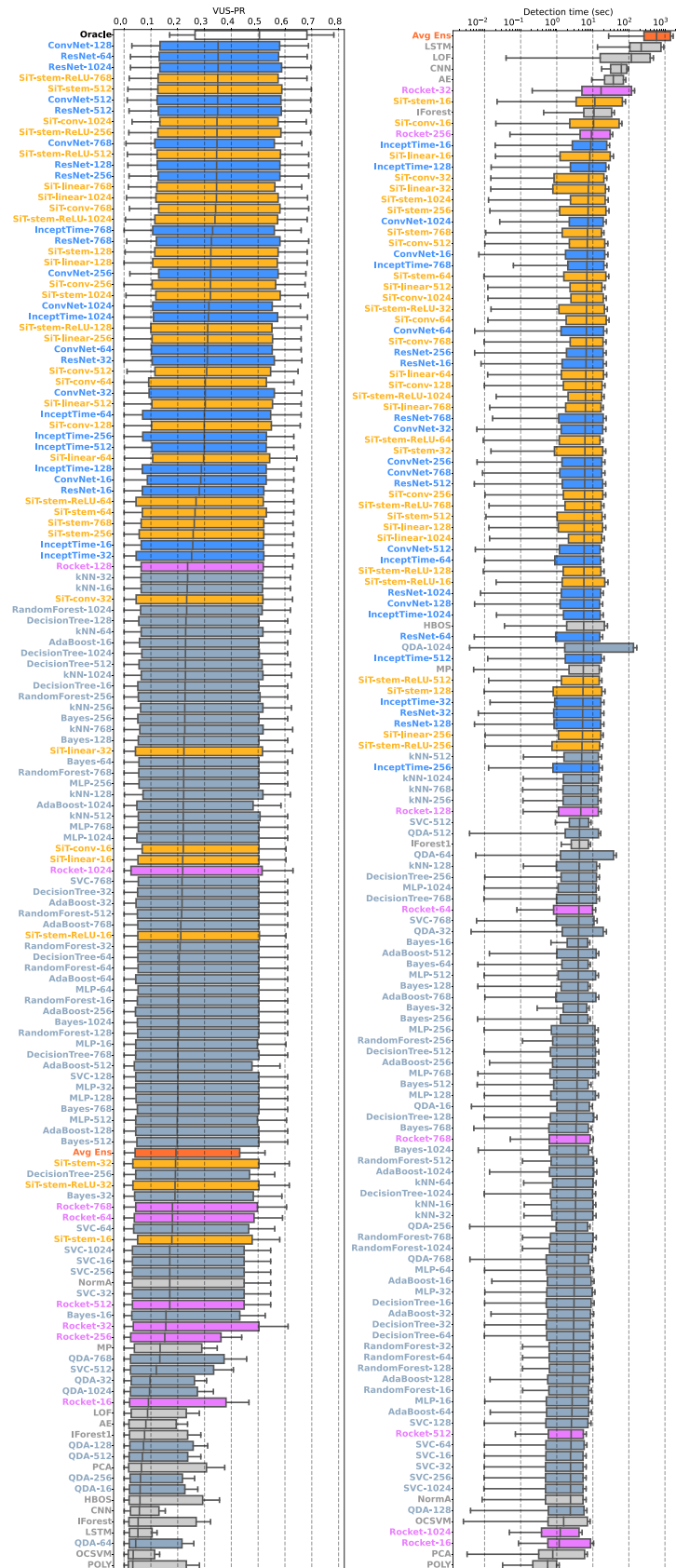


Figure 5.8: VUS-PR and Detection time (seconds) for all model selection approaches (complete version) over a test set of 497 series from TSB-UAD. The most accurate methods are at the top (a); the fastest methods are at the bottom (b)

Datasets	Description
Dodgers [52]	unusual traffic after a Dodgers game (1 time series)
ECG [68]	standard electrocardiogram dataset (52 time series)
IOPS [1]	performance indicators of a machine (58 time series)
KDD21 [55]	composite dataset released in a recent SIGKDD 2021 (250 time series)
MGAB [86]	Mackey-Glass time series with non-trivial anomalies (10 time series)
NAB [6]	Web-related real-world and artificial time series (58 time series)
SensorScope [94]	environmental data (23 time series)
YAHOO [58]	real and synthetic time series based on Yahoo production systems (367 time series)
Daphnet [9]	acceleration sensors on Parkinson’s disease patients (45 time series)
GHL [36]	Gasoil Heating Loop telemetry (126 time series)
Genesis [89]	portable pick-and-place demonstrator (6 time series)
MITDB [68]	ambulatory ECG recordings (32 time series)
OPPORTUNITY [78]	motion sensors for human activity recognition (465 time series)
Occupancy [28]	temperature, humidity, light, and CO2 of a room (10 time series)
SMD [84]	Server Machine telemetry (281 time series)
SVDB [45]	ECG recordings (115 time series)
Anomaly Detection	Description
IForest [62]	constructs binary trees based on random space splitting. The nodes (i.e., subsequences) with shorter path lengths to the root are likely to be anomalies.
IForest1 [62]	same as IForest, but each point (individually) are used as input.
LOF [25]	computes the ratio of the neighboring density to the local density.
MP [96]	detects as anomaly the subsequence with significant nearest neighbor distance.
NormA [19]	identifies the normal patterns based on clustering and calculates each point’s effective distance to the normal patterns.
PCA [4]	projects data to a lower-dimensional hyperplane, and data points with a significant distance from this plane can be identified as outliers.
AE [79]	projects data to the lower-dimensional latent space and reconstructs the data, and outliers are expected to have more evident reconstruction deviation.
LSTM-AD [67]	uses an LSTM network that from the current subsequence tries to predict the following value. The error prediction is then used to identify anomalies.
POLY [60]	fits a polynomial model that tries to predict the values of the data series from the previous subsequences. The outliers are detected by measuring the prediction error.
CNN [69]	builds a correlation between current and previous subsequences, and the outliers are detected by the deviation between the prediction and the actual value.
OCSVM [81]	is a support vector method that fits the normal training dataset and finds the normal data’s boundary.
HBOS [43]	constructs a histogram for the data and the inverse of the height of the bin is used as the outlier score of the data point.
Model Selection	Description
<i>Feature-based classifier</i>	
SVC [24]	maps training examples to points in space to maximize the gap between the two categories.
Bayes [98]	uses Bayes’ theorem to predict the class of a new data point using the posterior probabilities for each class
MLP [48]	consists of multiple layers of interconnected neurons
QDA [41]	is a discriminant analysis algorithm for classification problems
AdaBoost [39]	is a meta-algorithm using boosting technique with weak classifiers
Decision Tree [51]	is a tree-based approach that splits data points into separate leaves based on features
Random Forest [49]	is an ensemble Decision Trees fed with random samples (with replacement) of the training set and random set of features.
kNN [37]	assigns the most common class among its k nearest neighbors.
<i>Time series classifier</i>	
Rocket [32]	transforms input time series using a small set of convolutional kernels, and uses the transformed features to train a linear classifier
<i>Convolutional-based neural networks</i>	
ConvNet [91]	uses convolutional layers to automatically and adaptively learn spatial hierarchies of features from input data.
ResNet [91]	is a ConvNet with residual connections between convolutional block
Inception Time [35]	is a combination of ResNets with kernels of multiple sizes
<i>Transformer-based neural networks</i>	
SIT-conv [33]	is a transformer architecture with a convolutional layer as input
SIT-linear [33]	is a transformer architecture for which the time series are divided into non-overlapping patches and linearly projected into the embedding space
SIT-stem [93]	is a transformer architecture with convolutional layers with increasing dimensionality as input
SIT-stem-ReLU [90]	is similar to SIT-stem but with Scaled ReLU.

Table 5.1: Summary of datasets, methods, and measures.

Chapter 6

Adecimo

Building upon the findings presented in the preceding chapters of this thesis, we introduce ADecimo¹, a system that aims to (i) easily visualize and assess the performances of times series classification methods used as model selection for times series anomaly detection, and (ii) allow the user to use pre-trained model selection method on their own data.

6.1 System Overview

ADecimo is based on the TSB-UAD benchmark [74] and employs time series from various domains and applications. In this section, we describe ADecimo, a system that helps analysts understand the datasets, methods, and results of model selection approaches for time series anomaly detection. The GUI is a stand-alone web application developed using Python 3.6 and the Streamlit framework [2].

Figure 6.1 illustrates the inputs and features of ADecimo. The system is based on a preloaded set of datasets (16 in our demo), anomaly detection methods (12 in our demo), and accuracy evaluation measures (4 in our demo). The GUI permits interactions with these inputs. First, the user can visualize and interact with the overall experimental evaluation results (by filtering datasets or selecting only a subset of the methods). The user can also visualize the time series, the positions of the anomalies, and the model selection results. Finally, the user can upload their own time series and run our pre-trained model selection approaches.

The GUI is composed of three main frames, shown in Figure 6.2. The **Overall Accuracy** frame contains aggregate results (Figure 6.2 (A)), the **Interactive Exploration** can be used to navigate the detailed evaluation results (Figure 6.2 (C)), and the **Overall Execution Time** frame lists the running time results (Figure 6.2 (B)). In addition, a Description frame contains a brief overview of the system’s objectives, and a Datasets and Methods frame lists information on the datasets and the methods considered in our evaluation.

¹Available online: <https://adecimots.streamlit.app/>

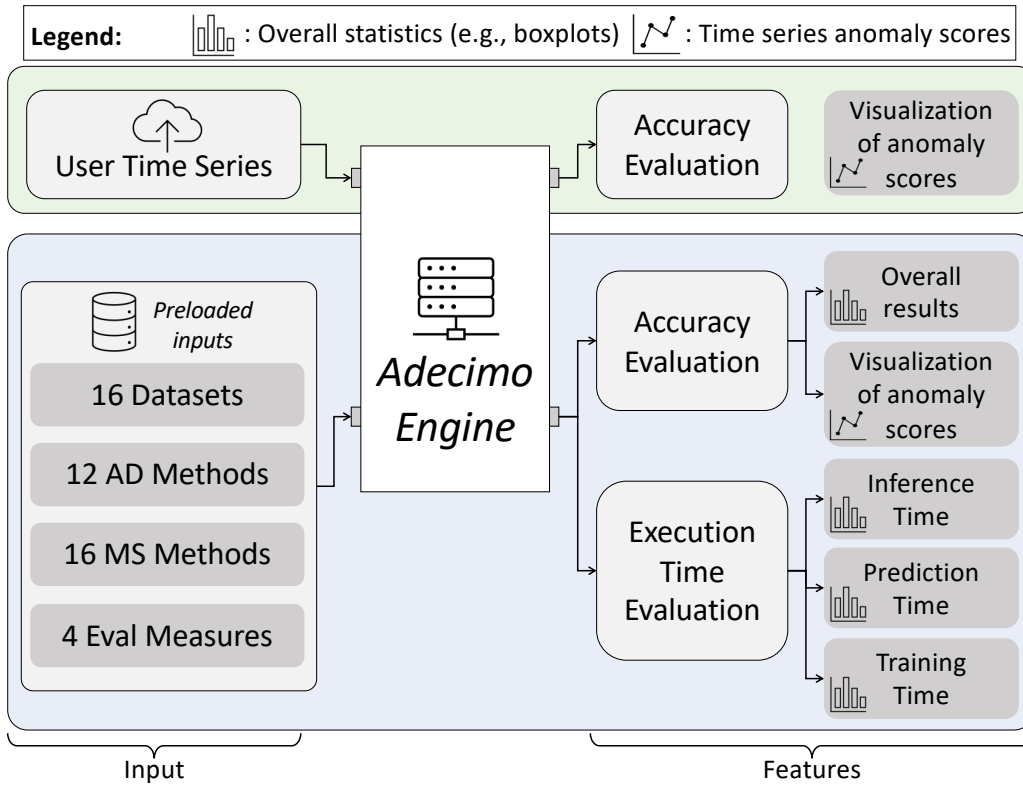


Figure 6.1: Summary of our system inputs and features

We now describe in more detail the three main frames of the GUI and the corresponding available actions.

6.1.1 Overall Accuracy Frame

The first frame depicts the overall accuracy evaluation and summarizes our results in a table (one accuracy value per time series and methods) and a boxplot (as shown in Figure 6.2 (A)). Using the sidebar on the left, the user can select which accuracy measure to use (i.e., VUS-PR or AUC-PR). Then, the user can filter based on datasets, families of methods (i.e., feature-based, convolutional-based, transformer-based, and rocket), and window length. The table and the boxplot are updated based on the user's choices.

6.1.2 Overall Execution Time Frame

The second frame depicts the overall execution time evaluation. This frame is divided into three tabs: (i) the first for training time, (ii) the second for selection time, and (iii) the third for detection time. In each tab, the results are summarized in a table (one execution time value per time series and methods) and a boxplot (as shown in Figure 6.2 (B)). The user can choose to visualize the results in a linear or logarithmic scale. Moreover, as in the first frame, the user can filter based on

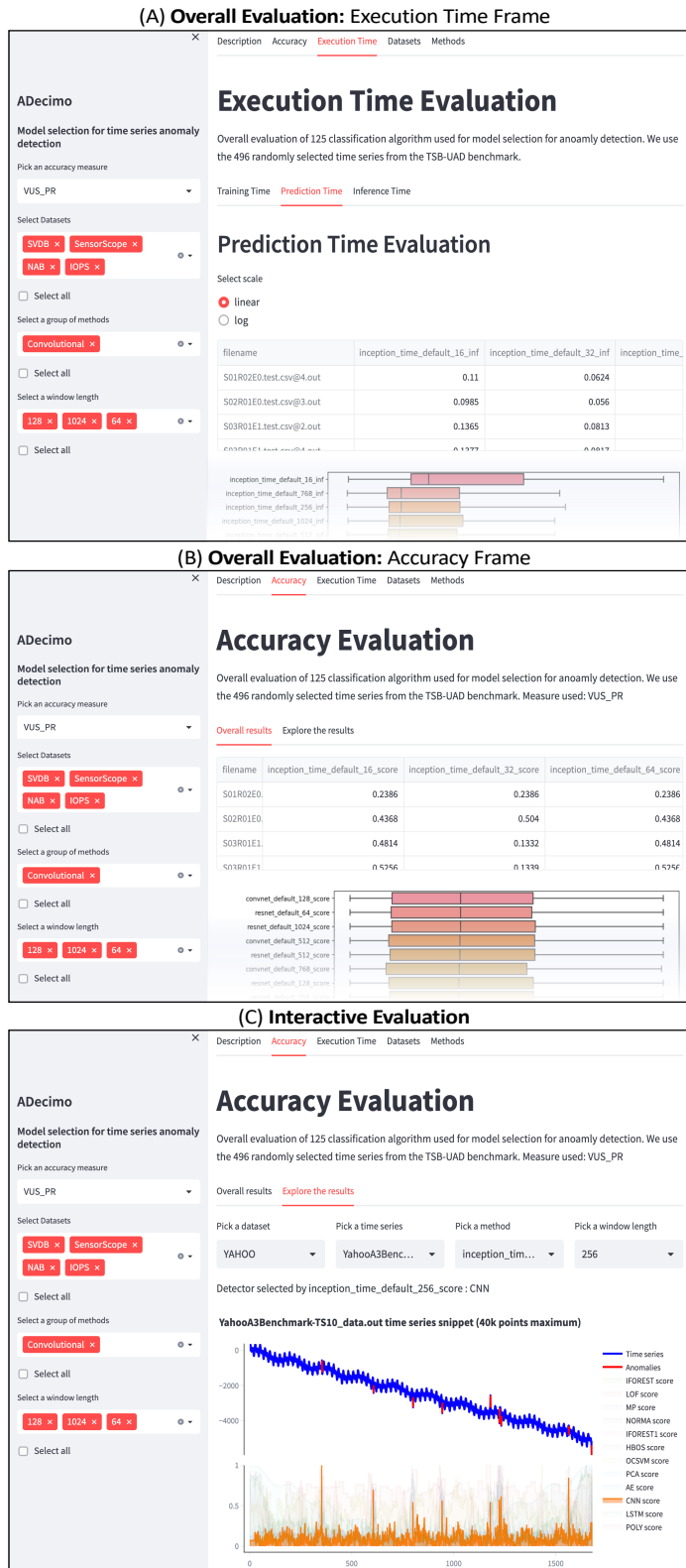


Figure 6.2: The three main ADecimo frames

datasets, families of methods, and window length.

6.1.3 Interactive Exploration Frame

Finally, in the accuracy frame, the user can click on a tab that opens the interactive Exploration frame (as shown in Figure 6.2 (C)). In this frame, the user can visualize the chosen detector and the corresponding anomaly score for each time series and model selection method. The anomaly scores of the detectors that have not been selected are also shown, but with a strong transparency ratio. The user can also select (in the time series selection drop-down) the "Upload your own time series" option. The latter will run the selected model selection methods and selected detectors on the time series uploaded by the user.

6.2 Demonstration Scenarios

This demo has three goals: (i) showcase the importance of a web application to explore large experimental results and extract meaningful insight on how much performance a user can gain using model selection on a specific use case; (ii) enable the user to interactively visualize the model selection choices for specific individual time series, models, and window length parameters; and (iii) challenge the user to test pre-trained model selection models on new (or their own) time series.

6.2.1 Finding the best model selection method

This scenario starts in frames 1 and 2 (Figure 6.2 (A) and (B)). Then, using the sidebar, we will ask the user to select datasets related to their application (e.g., medicine, environmental, or engineering). Then, the user can visit the *Datasets* frame to get more information on each dataset. Finally, the GUI will depict the most accurate (Figure 6.2 (A)) and most scalable (Figure 6.2 (B)) model selection methods (compared to existing anomaly detection methods) interactively. Thus, the user can discover if model selection methods outperform existing anomaly detection methods for a specific application, and if yes, which type of model selection approaches should be used.

6.2.2 Understanding and assessing model selection choice

In this scenario, we will ask the user to open the *Interactive Exploration* Frame (Figure 6.2 (C)). In this frame, the user can select a specific dataset based on his application of interest and a model selection. The user will then be able to select each time series in the chosen datasets and visualize both the time series and the anomaly scores of the selected anomaly detection methods based on the chosen model selection approach. For instance, in Figure 6.2 (C), the user selected one time series in YAHOO and InceptionTime methods with a window length of 256. In this specific example, the user can see that the detector selected is CNN. As the

GUI is also plotting the anomaly scores of all the remaining detectors, the user can assess if the choice made by the chosen model selection approach is correct. We will also explain the behavior of the model selection methods based on the methods' types and the window length impact.

6.2.3 Testing on your own data

In the last scenario, we will ask the user to click on the "upload your own" option in the time series selection drop-down. The user can add their own time series. Our system will run the selected model selection approach and the chosen detector. The user will then evaluate the pertinence and accuracy of model selection approaches on the new data.

Chapter 7

Conclusions

Time series anomaly detection is a challenging problem that has significant implications across various scientific, societal, and industrial domains. Despite the plethora of solutions proposed in the literature, no single method consistently outperforms others when evaluated on large and heterogeneous benchmarks. Through our extensive experimental evaluation, we have addressed the objectives outlined in Chapter 3.5. The following conclusions can be drawn from our findings:

1. **Classification as Model selection:** We have observed that time series classification methods serve as effective tools for selecting anomaly detection models. Transformer and Convolutional-based model selection methods have shown superior performance compared to individual detectors. Nevertheless, there remains a substantial gap between the best method and the *Oracle*, suggesting opportunities for future research and improvement in this direction.
2. **Ensembling or selecting:** Our evaluation has demonstrated that model selection approaches outperform ensembling methods in terms of accuracy. Moreover, model selection methods exhibit faster execution times, making them more efficient for real-world applications.
3. **Features or Raw values:** The evaluation results indicate that raw-based methods exhibit higher average accuracy compared to feature-based approaches. This suggests that leveraging the raw values of time series data can lead on average to better results.
4. **Out-Of-Distribution:** Our findings uphold the previous observations for (1) and (3) in the context of out-of-distribution time series. However, interestingly, for (2), we have observed that ensembling performs better than model selection when applied to time series that significantly differ from those in the training benchmark. This highlights the importance of considering the diversity of time series data during the selection process.

7.1 Future Work

The research conducted in this study opens up several avenues for future work in the field of model selection frameworks for anomaly detection in time series. Based on our observations and the implications of our findings, we outline the following potential directions:

1. **Improving Rank Prediction:** Enhancing the rank prediction accuracy could lead to substantial improvements in anomaly detection performance. Future work could focus on developing more advanced ranking models that accurately identify the best-performing anomaly detection techniques based on time series characteristics.
2. **Optimizing Accuracy and Execution Time Trade-off:** Model selection can be trained to strike a balance between accuracy and execution time, allowing for efficient inference of the selected anomaly detection method. Exploring techniques to optimize this trade-off could significantly enhance the overall efficiency of the model selection process.
3. **Unsupervised Settings:** The noticeable gap between the *Oracle* performance and model selection methods in unsupervised settings indicates the need for further exploration. Future research could delve into unsupervised anomaly detection techniques, leveraging advancements in unsupervised learning and self-supervised approaches to improve model selection accuracy in scenarios where labeled anomalies are not available.

By focusing on these areas, researchers can further advance the field of model selection for anomaly detection, building upon the foundation laid by this study. These future research directions hold the potential to improve the accuracy, efficiency, and applicability of time series classification as a model selection approach, ultimately enhancing anomaly detection methodologies across diverse domains and real-world applications.

Bibliography

- [1] http://iops.ai/dataset_detail/?id=10.
- [2] Streamlit documentation. <https://streamlit.io/>.
- [3] Charu C Aggarwal. An introduction to outlier analysis. In *Outlier analysis*, pages 1–34. Springer, 2017.
- [4] Charu C. Aggarwal. *Outlier Analysis*. Springer International Publishing, 2 edition, 2017.
- [5] Charu C. Aggarwal and Saket Sathe. Theoretical foundations and algorithms for outlier ensembles. *SIGKDD Explor. Newsl.*, 17(1):24–47, sep 2015.
- [6] Subutai Ahmad, Alexander Lavin, Scott Purdy, and Zuha Agha. Unsupervised real-time anomaly detection for streaming data. *Neurocomputing*, 262:134–147, 2017.
- [7] Jérôme Antoni and Pietro Borghesani. A statistical methodology for the design of condition indicators. *Mechanical Systems and Signal Processing*, 114:290–327, 2019.
- [8] Martin Bach-Andersen, Bo Rømer-Odgaard, and Ole Winther. Flexible non-linear predictive models for large-scale wind turbine diagnostics. *Wind Energy*, 20(5):753–764, 2017.
- [9] Marc Bachlin, Meir Plotnik, Daniel Roggen, Inbal Maidan, Jeffrey M. Hausdorff, Nir Giladi, and Gerhard Troster. Wearable assistant for parkinson’s disease patients with the freezing of gait symptom. *IEEE Transactions on Information Technology in Biomedicine*, 14(2):436–446, 2010.
- [10] Anthony Bagnall, Richard L. Cole, Themis Palpanas, and Kostas Zoumpatianos. Data Series Management (Dagstuhl Seminar 19282). *Dagstuhl Reports*, 9(7):24–39, 2019.
- [11] Ziv Bar-Joseph, Georg K Gerber, David K Gifford, Tommi S Jaakkola, and Itamar Simon. Continuous representations of time-series gene expression data. *Journal of Computational Biology*, 10(3-4):341–356, 2003.

- [12] V. Barnett and T. Lewis. *Outliers in Statistical Data*. John Wiley and Sons, Inc., 1994.
- [13] Bharat B Biswal, Maarten Mennes, Xi-Nian Zuo, Suril Gohel, Clare Kelly, Steve M Smith, Christian F Beckmann, Jonathan S Adelstein, Randy L Buckner, Stan Colcombe, et al. Toward discovery science of human brain function. *Proceedings of the National Academy of Sciences*, 107(10):4734–4739, 2010.
- [14] Ane Blázquez-García, Angel Conde, Usue Mori, and Jose A Lozano. A review on outlier/anomaly detection in time series data. *ACM Computing Surveys (CSUR)*, 54(3):1–33, 2021.
- [15] P. Boniol and E. Sylligardos. Our open-source code for this paper. <https://github.com/boniolp/MSAD>, 2023.
- [16] P. Boniol and E. Sylligardos. Our website. <https://adecimots.streamlit.app/>, 2023.
- [17] Paul Boniol, Michele Linardi, Federico Roncallo, and Themis Palpanas. Automated anomaly detection in large sequences. In *2020 IEEE 36th International Conference on Data Engineering (ICDE)*, pages 1834–1837, 2020.
- [18] Paul Boniol, Michele Linardi, Federico Roncallo, and Themis Palpanas. SAD: an unsupervised system for subsequence anomaly detection. In *ICDE*, 2020.
- [19] Paul Boniol, Michele Linardi, Federico Roncallo, Themis Palpanas, Mohammed Meftah, and Emmanuel Remy. Unsupervised and scalable subsequence anomaly detection in large data series. *The VLDB Journal*, March 2021.
- [20] Paul Boniol, Mohammed Meftah, Emmanuel Remy, and Themis Palpanas. dcam: Dimension-wise class activation map for explaining multivariate data series classification. In Zachary Ives, Angela Bonifati, and Amr El Abbadi, editors, *SIGMOD '22: International Conference on Management of Data, Philadelphia, PA, USA, June 12 - 17, 2022*, pages 1175–1189. ACM, 2022.
- [21] Paul Boniol and Themis Palpanas. Series2graph: Graph-based subsequence anomaly detection for time series. *Proc. VLDB Endow.*, 13(12):1821–1834, July 2020.
- [22] Paul Boniol, John Paparrizos, Themis Palpanas, and Michael J Franklin. Sand in action: subsequence anomaly detection for streams. *Proceedings of the VLDB Endowment*, 14(12), 2021.
- [23] Paul Boniol, John Paparrizos, Themis Palpanas, and Michael J Franklin. Sand: streaming subsequence anomaly detection, 2021.

- [24] Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory, COLT '92*, page 144–152, New York, NY, USA, 1992. Association for Computing Machinery.
- [25] Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. Lof: Identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, SIGMOD '00*, pages 93–104, New York, NY, USA, 2000. ACM.
- [26] Peter J Brockwell and Richard A Davis. *Introduction to time series and forecasting*. springer, 2016.
- [27] Yingyi Bu, Oscar Tat-Wing Leung, Ada Wai-Chee Fu, Eamonn J. Keogh, Jian Pei, and Sam Meshkin. Wat: Finding top-k discords in time series database. In *SDM*, 2007.
- [28] Luis M. Candanedo and Véronique Feldheim. Accurate occupancy detection of an office room from light, temperature, humidity and co2 measurements using statistical learning models. *Energy and Buildings*, 112:28–39, 2016.
- [29] Maximilian Christ, Nils Braun, Julius Neuffer, and Andreas W. Kempa-Liehr. Time series feature extraction on basis of scalable hypothesis tests (tsfresh – a python package). *Neurocomputing*, 307:72–77, 2018.
- [30] Maximilian Christ, Andreas W Kempa-Liehr, and Michael Feindt. Distributed and parallel time series feature extraction for industrial big data applications. *arXiv preprint arXiv:1610.07717*, 2016.
- [31] Jesse Davis and Mark Goadrich. The relationship between precision-recall and roc curves. In *Proceedings of the 23rd International Conference on Machine Learning, ICML '06*, page 233–240, New York, NY, USA, 2006. Association for Computing Machinery.
- [32] Angus Dempster, Daniel F Schmidt, and Geoffrey I Webb. Minirocket: A very fast (almost) deterministic transform for time series classification. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pages 248–257, 2021.
- [33] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.
- [34] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Deep learning for time series classification: a review. *Data mining and knowledge discovery*, 33(4):917–963, 2019.

- [35] Hassan Ismail Fawaz, Benjamin Lucas, Germain Forestier, Charlotte Pelletier, Daniel F Schmidt, Jonathan Weber, Geoffrey I Webb, Lhassane Idoumghar, Pierre-Alain Muller, and François Petitjean. Inceptiontime: Finding alexnet for time series classification. *Data Mining and Knowledge Discovery*, 34(6):1936–1962, 2020.
- [36] Pavel Filonov, Andrey Lavrentyev, and Artem Vorontsov. Multivariate industrial time series with cyber-attack simulation: Fault detection using an lstm-based predictive data model, 2016.
- [37] Evelyn Fix and Joseph L. Hodges. Discriminatory analysis - nonparametric discrimination: Consistency properties. *International Statistical Review*, 57:238, 1989.
- [38] Anthony J Fox. Outliers in time series. *Journal of the Royal Statistical Society: Series B (Methodological)*, 34(3):350–363, 1972.
- [39] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *Proceedings of the Second European Conference on Computational Learning Theory, EuroCOLT '95*, page 23–37, Berlin, Heidelberg, 1995. Springer-Verlag.
- [40] Ada Wai-chee Fu, Oscar Tat-Wing Leung, Eamonn Keogh, and Jessica Lin. Finding time series discords based on haar transform. In *Proceedings of the Second International Conference on Advanced Data Mining and Applications, ADMA'06*, page 31–41, Berlin, Heidelberg, 2006. Springer-Verlag.
- [41] Seymour Geisser. Posterior odds for multivariate normal classifications. *Journal of the royal statistical society series b-methodological*, 26:69–76, 1964.
- [42] Steve Goddard, Sherri K Harms, Stephen E Reichenbach, Tsegaye Tadesse, and William J Waltman. Geospatial decision support for drought risk management. *Communications of the ACM*, 46(1):35–37, 2003.
- [43] Markus Goldstein and Andreas Dengel. Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm. *KI-2012: poster and demo track*, 9, 2012.
- [44] Mononito Goswami, Cristian Challu, Laurent Callot, Lenon Minorics, and Andrey Kan. Unsupervised model selection for time-series anomaly detection, 2022.
- [45] Scott David Greenwald. *Improved detection and classification of arrhythmias in noise-corrupted electrocardiograms using contextual information*. Thesis, Massachusetts Institute of Technology, 1990. Accepted: 2005-10-07T20:45:22Z.

- [46] Medina Hadjem, Farid Naït-Abdesselam, and Ashfaq Khokhar. St-segment and t-wave anomalies prediction in an ecg data using rusboost. In *2016 IEEE 18th International Conference on e-Health Networking, Applications and Services (Healthcom)*, pages 1–6, 2016.
- [47] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.
- [48] Geoffrey E. Hinton. Connectionist learning procedures. *Artif. Intell.*, 40:185–234, 1989.
- [49] Tin Kam Ho. Random decision forests. In *Proceedings of 3rd International Conference on Document Analysis and Recognition*, volume 1, pages 278–282 vol.1, 1995.
- [50] Pablo Huijse, Pablo A Estevez, Pavlos Protopapas, Jose C Principe, and Pablo Zegers. Computational intelligence challenges and applications on large-scale astronomical time series databases. *IEEE Computational Intelligence Magazine*, 9(3):27–39, 2014.
- [51] Earl B. Hunt, J Marin, and Philip J. Stone. Experiments in induction. 1966.
- [52] Alexander Ihler, Jon Hutchins, and Padhraic Smyth. Adaptive event detection with time-varying poisson processes. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, page 207–216, New York, NY, USA, 2006. Association for Computing Machinery.
- [53] Vincent Jacob, Fei Song, Arnaud Stiegler, Bijan Rad, Yanlei Diao, and Nesime Tatbul. Exathlon: A benchmark for explainable anomaly detection over time series. *arXiv preprint arXiv:2010.05073*, 2020.
- [54] Vincent Jacob, Fei Song, Arnaud Stiegler, Bijan Rad, Yanlei Diao, and Nesime Tatbul. Exathlon: A benchmark for explainable anomaly detection over time series. *Proc. VLDB Endow.*, 14(11):2613–2626, oct 2021.
- [55] E. Keogh, T. Dutta Roy, U. Naik, and A Agrawal. Multi-dataset Time-Series Anomaly Detection Competition 2021. <https://compete.hexagon-ml.com/practice/competition/39/>, 2021.
- [56] Eamonn J. Keogh, Stefano Lonardi, Chotirat Ratanamahatana, Li Wei, Sanghee Lee, and John C. Handley. Compression-based data mining of sequential data. *Data Mining and Knowledge Discovery*, 14:99–129, 2006.
- [57] Siwon Kim, Kukjin Choi, Hyun-Soo Choi, Byunghan Lee, and Sungroh Yoon. Towards a rigorous evaluation of time-series anomaly detection. In *AAAI Conference on Artificial Intelligence*, 2021.

- [58] N. Laptev, S. Amizadeh, and Y. Billawala. S5 - A Labeled Anomaly Detection Dataset, version 1.0(16M), March 2015.
- [59] Alexander Lavin and Subutai Ahmad. Evaluating real-time anomaly detection algorithms – the numenta anomaly benchmark. In *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, pages 38–44, 2015.
- [60] Zhi Li, Hong Ma, and Yongbing Mei. A Unifying Method for Outlier and Change Detection from Data Streams Based on Local Polynomial Fitting. In Zhi-Hua Zhou, Hang Li, and Qiang Yang, editors, *Advances in Knowledge Discovery and Data Mining*, Lecture Notes in Computer Science, pages 150–161, Berlin, Heidelberg, 2007. Springer.
- [61] Michele Linardi, Yan Zhu, Themis Palpanas, and Eamonn J. Keogh. Matrix profile goes MAD: variable-length motif and discord discovery in data series. *Data Min. Knowl. Discov.*, 34(4):1022–1071, 2020.
- [62] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation Forest. In *2008 Eighth IEEE International Conference on Data Mining*, pages 413–422, December 2008. ISSN: 2374-8486.
- [63] Yubao Liu, Xiuwei Chen, and Fei Wang. Efficient Detection of Discords for Time Series Stream. *Advances in Data and Web Management*, pages 629–634, 2009.
- [64] Markus Löning, A. Bagnall, Sajaysurya Ganesh, Viktor Kazakov, Jason Lines, and Franz J. Király. sktime: A unified interface for machine learning with time series. *ArXiv*, abs/1909.07872, 2019.
- [65] Helmut Lütkepohl, Markus Krätzig, and Peter CB Phillips. *Applied time series econometrics*. Cambridge university press, 2004.
- [66] Haoran Ma, Benyamin Ghogh, Maria N. Samad, Dongyu Zheng, and Mark Crowley. Isolation mondrian forest for batch and online anomaly detection. In *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 3051–3058, 2020.
- [67] Pankaj Malhotra, L. Vig, Gautam M. Shroff, and Puneet Agarwal. Long Short Term Memory Networks for Anomaly Detection in Time Series. In *ESANN*, 2015.
- [68] G.B. Moody and R.G. Mark. The impact of the mit-bih arrhythmia database. *IEEE Engineering in Medicine and Biology Magazine*, 20(3):45–50, 2001.
- [69] M. Munir, S. A. Siddiqui, A. Dengel, and S. Ahmed. Deepant: A deep learning approach for unsupervised anomaly detection in time series. *IEEE Access*, 7:1991–2005, 2019.

- [70] Keiron O’Shea and Ryan Nash. An introduction to convolutional neural networks. *CoRR*, abs/1511.08458, 2015.
- [71] ES Page. On problems in which a change in a parameter occurs at an unknown point. *Biometrika*, 44(1/2):248–252, 1957.
- [72] Themis Palpanas and Volker Beckmann. Report on the first and second interdisciplinary time series analysis workshop (itisa). *SIGMOD Rec.*, 48(3):36–40, December 2019.
- [73] John Paparrizos, Paul Boniol, Themis Palpanas, Ruey S. Tsay, Aaron Elmore, and Michael J. Franklin. Volume under the surface: A new accuracy evaluation measure for time-series anomaly detection. *Proc. VLDB Endow.*, 15(11), 2022.
- [74] John Paparrizos, Yuhao Kang, Paul Boniol, Ruey S. Tsay, Themis Palpanas, and Michael J. Franklin. Tsb-uad: An end-to-end benchmark suite for univariate time-series anomaly detection. *Proc. VLDB Endow.*, 15(8), 2022.
- [75] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, volume 32, 2019.
- [76] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(85):2825–2830, 2011.
- [77] Joshua S Richman and J Randall Moorman. Physiological time-series analysis using approximate entropy and sample entropy. *American Journal of Physiology-Heart and Circulatory Physiology*, 278(6):H2039–H2049, 2000.
- [78] Daniel Roggen, Alberto Calatroni, Mirco Rossi, Thomas Holleczeck, Kilian Förster, Gerhard Tröster, Paul Lukowicz, David Bannach, Gerald Pirkl, Alois Ferscha, Jakob Doppler, Clemens Holzmann, Marc Kurz, Gerald Holl, Ricardo Chavarriaga, Hesam Sagha, Hamidreza Bayati, Marco Creatura, and José del R. Millàn. Collecting complex activity datasets in highly rich networked sensor environments. In *2010 Seventh International Conference on Networked Sensing Systems (INSS)*, pages 233–240, 2010.
- [79] Mayu Sakurada and Takehisa Yairi. Anomaly detection using autoencoders with nonlinear dimensionality reduction. In *Proceedings of the MLSDA 2014*

- 2nd Workshop on Machine Learning for Sensory Data Analysis*, MLSDA'14, page 4–11, New York, NY, USA, 2014. Association for Computing Machinery.
- [80] Sebastian Schmidl, Phillip Wenig, and Thorsten Papenbrock. Anomaly detection in time series: A comprehensive evaluation. *Proc. VLDB Endow.*, 15(9):1779–1797, jul 2022.
- [81] Bernhard Schölkopf, Robert Williamson, Alex Smola, John Shawe-Taylor, and John Platt. Support vector method for novelty detection. In *Proceedings of the 12th International Conference on Neural Information Processing Systems*, NIPS'99, pages 582–588, Cambridge, MA, USA, November 1999. MIT Press.
- [82] Pavel Senin, Jessica Lin, Xing Wang, Tim Oates, Sunil Gandhi, Arnold P. Boedihardjo, Crystal Chen, and Susan Frankenstein. Time series anomaly discovery with grammar-based compression. In *EDBT*, 2015.
- [83] K Simonyan and A Zisserman. Very deep convolutional networks for large-scale image recognition. pages 1–14. Computational and Biological Learning Society, 2015.
- [84] Ya Su, Youjian Zhao, Chenhao Niu, Rong Liu, Wei Sun, and Dan Pei. Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '19, page 2828–2837, New York, NY, USA, 2019. Association for Computing Machinery.
- [85] S. Subramaniam, T. Palpanas, D. Papadopoulos, V. Kalogeraki, and D. Gunopulos. Online outlier detection in sensor data using non-parametric models. In *Proceedings of the 32nd International Conference on Very Large Data Bases*, VLDB '06, page 187–198. VLDB Endowment, 2006.
- [86] Markus Thill, Wolfgang Konen, and Thomas Bäck. MarkusThill/MGAB: The Mackey-Glass Anomaly Benchmark, <https://doi.org/10.5281/zenodo.3762385>, April 2020.
- [87] Kuniaki Uehara and Mitsuomi Shimada. Extraction of primitive motion and discovery of association rules from human motion data. In *Progress in Discovery Science*, pages 338–348. 2002.
- [88] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
- [89] Alexander von Birgelen and Oliver Niggemann. *Anomaly Detection and Localization for Cyber-Physical Production Systems with Self-Organizing Maps*, pages 55–71. Springer Berlin Heidelberg, Berlin, Heidelberg, 2018.

- [90] Pichao Wang, Xue Wang, Hao Luo, Jingkai Zhou, Zhipeng Zhou, Fan Wang, Hao Li, and Rong Jin. Scaled relu matters for training vision transformers. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 2495–2503, 2022.
- [91] Zhiguang Wang, Weizhong Yan, and Tim Oates. Time series classification from scratch with deep neural networks: A strong baseline. *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 1578–1585, 2017.
- [92] Renjie Wu and Eamonn J Keogh. Current time series anomaly detection benchmarks are flawed and are creating the illusion of progress. *arXiv preprint arXiv:2009.13807*, 2020.
- [93] Tete Xiao, Mannat Singh, Eric Mintun, Trevor Darrell, Piotr Dollár, and Ross Girshick. Early convolutions help transformers see better. *Advances in Neural Information Processing Systems*, 34:30392–30400, 2021.
- [94] Yuan Yao, Abhishek Sharma, Leana Golubchik, and Ramesh Govindan. On-line anomaly detection for sensor systems: A simple and efficient approach. *Performance Evaluation*, 67(11):1059–1075, 2010. Performance 2010.
- [95] Chin-Chia Michael Yeh, Yan Zhu, Liudmila Ulanova, Nurjahan Begum, Yifei Ding, Hoang Anh Dau, Diego Furtado Silva, Abdullah Mueen, and Eamonn Keogh. Matrix profile i: All pairs similarity joins for time series: A unifying view that includes motifs, discords and shapelets. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 1317–1322, 2016.
- [96] Chin-Chia Michael Yeh, Yan Zhu, Liudmila Ulanova, Nurjahan Begum, Yifei Ding, Hoang Anh Dau, Zachary Zimmerman, Diego Furtado Silva, Abdullah Mueen, and Eamonn Keogh. Time series joins, motifs, discords and shapelets: a unifying view that exploits the matrix profile. *Data Mining and Knowledge Discovery*, 32(1):83–123, January 2018.
- [97] Yuanxiang Ying, Juanyong Duan, Chunlei Wang, Yujing Wang, Congrui Huang, and Bixiong Xu. Automated model selection for time-series anomaly detection, 2020.
- [98] Harry Zhang. The optimality of naive bayes. In *The Florida AI Research Society*, 2004.
- [99] Yue Zhao, Ryan Rossi, and Leman Akoglu. Automatic unsupervised outlier model selection. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 4489–4502. Curran Associates, Inc., 2021.