



UNIVERSIDADE
ESTADUAL DE LONDRINA

RICARDO PETRI SILVA

ADVANCES IN TIME SERIES ANALYSIS:
SEGMENTATION, PREDICTION, AND UNSUPERVISED
TUNING OF DRIFT DETECTORS

LONDRINA

2024

RICARDO PETRI SILVA

**ADVANCES IN TIME SERIES ANALYSIS:
SEGMENTATION, PREDICTION, AND UNSUPERVISED
TUNING OF DRIFT DETECTORS**

Thesis presented to the Doctoral Program in
Electrical Engineering at State University of
Londrina as a partial requirement to obtain
the title of Doctor in Electrical Engineering

LONDRINA

2024

Ficha de identificação da obra elaborada pelo autor, através do Programa de Geração Automática do Sistema de Bibliotecas da UEL

S586a Silva, Ricardo Petri.
Advances in time series analysis : segmentation, prediction, and unsupervised tuning of drift detectors / Ricardo Petri Silva. - Londrina, 2026.
88 f. : il.

Orientador: Leonimer Flávio de Melo.
Tese (Doutorado em Engenharia Elétrica) - Universidade Estadual de Londrina, Centro de Tecnologia e Urbanismo, Programa de Pós-Graduação em Engenharia Elétrica, 2026.
Inclui bibliografia.

1. time series segmentation - Tese. 2. stationarity analysis - Tese. 3. time series prediction improvement - Tese. 4. size reduction in time series - Tese. I. Melo, Leonimer Flávio de. II. Universidade Estadual de Londrina. Centro de Tecnologia e Urbanismo. Programa de Pós-Graduação em Engenharia Elétrica. III. Título.

CDU 62

RICARDO PETRI SILVA

**ADVANCES IN TIME SERIES ANALYSIS:
SEGMENTATION, PREDICTION, AND UNSUPERVISED
TUNING OF DRIFT DETECTORS**

Thesis presented to the Doctoral Program in
Electrical Engineering at State University of
Londrina as a partial requirement to obtain
the title of Doctor in Electrical Engineering

EXAMINATION BOARD

Supervisor: Leonimer Flávio de Melo
Universidade Estadual de Londrina

Internal member: Alan Salvany Felinto
Universidade Estadual de Londrina

Internal member: Daniel dos Santos Kaster
Universidade Estadual de Londrina

External member: Gustavo Taiji Naozuka
Laboratório Nacional de Computação
Científica

External member: Rodrigo Capobianco
Guido
Universidade Estadual Paulista

*This work is dedicated to adult children
who, when they were small, dreamed of
becoming scientists.*

ACKNOWLEDGEMENTS

At first, I would like to thank my parents Sergio Alves da Silva and Maria Elizete Petri Silva, for all the support and encouragement offered and, mainly, for teaching me be the person I am today. I owe them all the conquests acquired, as they made it possible for all of this to become a reality. I would also like to thank my brother Lucas Petri Silva who also contributed and supported my entire trajectory. Undoubtedly, the support of all of them was essential for me to achieve my goals without faltering.

I would like to express my profound gratitude to my girlfriend, Nayra Kawana Turini, whose unwavering support and constant motivation have been invaluable to me. Nayra, your fervent belief in my dreams has made it possible for them all to come true. Thank you for being my tireless and inspiring companion.

I thank everyone involved in this stage of my life, from the employees of the State University of Londrina to the professors I had more contact with. In a special way, I would like to thank my supervisors Sylvio Barbon Junior and Leonimer Flávio de Melo, who, in addition to accepting me as a supervisor, encouraged and supported me throughout the development of my research, giving me confidence and encouraging me to carry out research and solve problems.

Finally, I would like to thank all the friendships built throughout this journey, from graduation in the Computer Science course to the present moment. These friendships were extremely important as we experienced so many moments of joy and difficult situations, but we always managed to overcome all obstacles.

*“To gain knowledge, add things every day.
To gain wisdom, eliminate things every
day.”
(Lao-Tsé)*

SILVA, R. P.. **Advances in Time Series Analysis: Segmentation, Prediction, and Unsupervised Tuning of Drift Detectors**. 2024. 88p. Thesis (Doctorate in Electrical Engineering) – State University of Londrina, Londrina, 2024.

ABSTRACT

Time series analysis proves highly useful and widely applicable, enabling reliable forecasting and informed decision-making across various fields. This thesis describes the fundamental importance of time series analysis and introduces advanced techniques aimed at enhancing the structure of time series data to improve predictive accuracy. Given the challenges associated with concept drifts—where the statistical properties of a signal change unpredictably—this study also emphasizes the need for innovative, unsupervised approaches to optimize drift detectors. The thesis presents two novel segmentation methods designed to refine data processing based on stationarity analysis, thus facilitating more accurate forecasts. Additionally, it develops a new method to enhance drift detectors using unsupervised approach, reducing the need for manual tuning and deep prior knowledge of the data. These contributions significantly advance the field of time series analysis, providing robust, adaptable, and efficient solutions for applications where precise predictions are vital. Our results demonstrate that the segmentation process enhances the prediction process for both Long Short-term Memory and Temporal Convolutional Networks. Nonetheless, our unsupervised tuning outperforms the default configuration in drift detectors in most evaluated cases.

Keywords: time series segmentation; stationarity analysis; time series prediction improvement; size reduction in time series; unsupervised tuning of drift detectors; concept drifts

LIST OF ILLUSTRATIONS

Figure 1 – LSTM cell.	29
Figure 2 – Methodology framework of Change Detector Segmentation.	39
Figure 3 – Methodology proposal of ADF-based Segmentation.	41
Figure 4 – TS size reduction according to segmentation methods ADF, ADW, PH, and OSTs. (a) Database A; (b) database B; (c) database C; (d) database D; (e) database E; (f) database F; (g) database G; (h) database H; (i) database I; (j) database J.	50
Figure 5 – Segmented samples of dataset A (chunk from 500 to 560 samples). ADF removed 2 samples, PH and ADW removed a single sample, and OSTs removed 17. The stationarity-based segmenters considered the range from 515 to 543 as stable and without the need for segmentation, while the OSTs segmenter performed quite a lot of segmentation at these points.	51
Figure 6 – Segmented samples of dataset C (chunk from 520 to 580 samples). ADF removed 19 samples, PH removed 2 samples, ADW removed a single sample and OSTs removed 18. Unlike the case of dataset A, the ADF segmenter performed many segmentations like the OSTs, but in different regions, while the other segmenters considered the region as stable.	52
Figure 7 – Comparison of the RMSE values obtained by segmentation techniques for Naive predictor according to the Nemenyi test. Groups that are not significantly different ($\alpha = 0.05$ and $CD = 1.04$) are connected.	53
Figure 8 – Comparison of the RMSE values obtained by segmentation techniques for ARIMA predictor according to the Nemenyi test. Groups that are not significantly different ($\alpha = 0.05$ and $CD = 1.03$) are connected.	53
Figure 9 – Comparison of the RMSE values obtained by segmentation techniques for LSTM predictor according to the Nemenyi test. Groups that are not significantly different ($\alpha = 0.05$ and $CD = 1.04$) are connected.	54
Figure 10 – Comparison of the RMSE values obtained by segmentation techniques for TCN predictor according to the Nemenyi test. Groups that are not significantly different ($\alpha = 0.05$ and $CD = 1.03$) are connected.	54
Figure 11 – Correlation between the stationarity (based on ADF Test) and performance (rRMSE) obtained by segmentation methods.	55
Figure 12 – Correlation between the stationarity (based on ADF Test) and dimension reduction delivered by segmentation methods.	56
Figure 13 – Unsupervised Tuning Methodology.	58

Figure 14 – Random signal generated from CSG. Initial amplitude = 10, final amplitude = 60.	64
Figure 15 – Random signal generated from CSG. Initial amplitude = 10, final amplitude = 60.	64
Figure 16 – Boxplot of low amplitudes variation without perturbations on signal experiments.	72
Figure 17 – Boxplot of low amplitudes variation with perturbations on signal experiments.	72
Figure 18 – Boxplot of high amplitudes variation without perturbations on signal experiments.	72
Figure 19 – Boxplot of high amplitudes variation with perturbations on signal experiments.	73
Figure 20 – Boxplot of our tuning process vs Random Search.	74
Figure 21 – Page-Hinkley without perturbations.	75
Figure 22 – HDDM _A without perturbations.	75
Figure 23 – HDDM _W without perturbations.	76
Figure 24 – KSWIN without perturbations.	76
Figure 25 – Page-Hinkley with perturbations.	77
Figure 26 – HDDM _A with perturbations.	77
Figure 27 – HDDM _W with perturbations.	78
Figure 28 – KSWIN with perturbations.	78
Figure 29 – Friedman’s statistical test on experiment J, size = 5000.	79
Figure 30 – Friedman’s statistical test on experiment O, size = 5000.	79

LIST OF TABLES

Table 1	– Comparison of TS segmentation techniques in the literature.	32
Table 2	– Intervals used for experimentation from PV, MDT, and MS databases. .	43
Table 3	– Hyperparameters experimented for tuning.	46
Table 4	– Global RMSE results from predictive techniques.	47
Table 5	– Relative RMSE of each segmentation technique referring to the original base across four different predictive techniques using the ADF-based method, ADW, and PH (i.e., instances of our segmentation framework), and OSTS segmentation method. Lower errors are in bold and worst average cases are underlined.	48
Table 6	– Signal generation configurations.	65
Table 7	– Drift detector parameter values.	68

LIST OF ABBREVIATIONS AND ACRONYMS

1. ADF - Augmented Dickey-Fuller
2. ADW - ADWIN
3. AI - Artificial Intelligence
4. ARIMA - Auto Regressive Integrated Moving Average
5. BISO - Bigger in Smaller out
6. BI - Bigger in
7. SO - Smaller out
8. CD - Critical Difference
9. CDS - Change Detector Segmentation
10. CNN - Convolutional Neural Network
11. CSG - Configurable Signal Generator
12. CUSUM - Cumulative Sum Control Chart
13. DD - Drift Detector
14. DL - Deep Learning
15. FN - False Negatives
16. FP - False Positives
17. FTDD - Fisher Test Drift Detector
18. GBM - Gradient Boosting Machines
19. LSTM - Long Short-Term Memory
20. MDT - Minimum Daily Temperature
21. ML - Machine Learning
22. MS - Monthly Sunspot
23. PH - Page-Hinkley
24. PV - Photovoltaic Plant

- 25. RMSE - Root Mean Squared Error
- 26. RNN - Recurrent Neural Network
- 27. RQ - Research Question
- 28. RS - Random Search
- 29. TCN - Temporal Convolutional Network
- 30. TP - True Positives
- 31. TS - Time Series
- 32. WSTD - Wilcoxon Rank Sum Test Drift Detector

CONTENTS

1	INTRODUCTION	15
1.1	Hypotheses	16
1.1.1	Time Series Segmentation	17
1.1.2	Drift Detectors Optimization	17
1.2	Contributions	18
1.3	Thesis Structure	19
2	PRELIMINARIES	21
2.1	Time Series	21
2.1.1	Time Series Forecasting	22
2.1.2	Stationarity in Time Series Analysis	24
2.1.3	Augmented Dickey–Fuller Test	24
2.2	Concept Drift	25
2.2.1	Concept Drift in Time Series	26
2.2.2	Concept Drift Detector	26
2.3	Machine Learning and Time Series Analysis	27
2.3.1	Ensemble Learning Techniques	27
2.3.2	Advanced Deep Learning Models for Time Series	27
2.3.3	Long Short-Term Memory	28
2.3.4	Temporal Convolutional Network	29
3	LITERATURE REVIEW	30
3.1	Overview	30
3.2	Background on Time Series Analysis	30
3.2.1	Current Limitations	31
3.3	Background on Concept Drift and Drift Detectors	32
3.4	Background on Tuning for Drift Detectors	33
3.4.1	Current Limitations	34
4	TIME SERIES SEGMENTATION	35
4.1	Overview	35
4.2	Proposed Approach: Time Series Segmentation Based on Sta- tionarity	35
4.2.1	Change Detector Segmentation Framework	36
4.2.1.1	Global ADF Test	37
4.2.1.2	Windowing	37

4.2.1.3	Local ADF Test	37
4.2.1.4	Thresholding	37
4.2.1.5	Tuning	37
4.2.1.6	Change Detectors	38
4.2.1.7	Removing Samples	38
4.2.2	ADF-Based Segmentation	39
4.3	Experimental Study	41
4.3.1	Databases	42
4.3.2	Change Detectors	43
4.3.3	Metrics	44
4.3.4	Prediction Techniques	44
4.3.5	OSTS Method	45
4.3.6	Experimental Results	45
4.3.7	Predictions Results	46
4.3.8	Time Series Size Reduction	49
4.3.9	Sample Segmentation	49
4.3.10	Statistical Analysis	52
4.3.11	Stationarity Impact on Segmentation	55
4.4	Conclusions	57
5	UNSUPERVISED TUNING FOR DRIFT DETECTORS . . .	58
5.1	Step 1	59
5.2	Step 2	59
5.3	Step 3	59
5.4	Step 4	60
5.5	Step 5	60
5.6	Materials and Methods	61
5.6.1	Configurable Signal Generator (CSG)	61
5.6.2	Experimental setup	63
5.6.3	Drift Detectors	65
5.6.3.1	Page-Hinkley	65
5.6.3.2	Page-Hinkley hyperparameters	66
5.6.3.3	ADWIN	66
5.6.3.4	ADWIN hyperparameters	66
5.6.3.5	Drift Detection Method	67
5.6.3.6	DDM hyperparameters	67
5.6.3.7	HDDM _A and HDDM _W	67
5.6.3.8	HDDM _A and HDDM _W hyperparameters	67
5.6.3.9	KSWIN	68
5.6.4	Evaluation metrics	68

5.6.5	Unsupervised Tuning Algorithm	69
5.7	Results and discussion	70
5.7.1	Unsupervised Tuning vs Random Search	73
5.7.2	Number of samples impact	74
5.7.3	Statistical analysis	79
5.8	Conclusions	80
6	FINAL CONCLUSION	81
	REFERÊNCIAS	83

1 INTRODUCTION

In the rapidly evolving field of data science, the study of Time Series (TS) analysis holds a important role, driven by its wide-ranging applications across diverse domains [1, 2, 3, 4]. A TS can also be represented as a sequence of signals, effectively capturing the dynamic variations over time within various systems. This perspective is particularly useful in fields such as signal processing, where data from sensors or communications are interpreted as signals that vary over time. Additionally, TS data is fundamental in representing data streams in online applications, where continuous, real-time data flow is relevant. For instance, in financial trading platforms, social media feeds, or Internet of Things (IoT) devices, the incoming data is processed as a stream. In these contexts, TS analysis is essential for extracting meaningful patterns, detecting anomalies, and making predictions based on the evolving nature of the data, thereby enabling proactive and informed decision-making in real-time environments.

TS analysis, at its core, involves the examination of data points collected or recorded at successive time intervals. This methodological approach enables researchers and practitioners to discern patterns, trends, and seasonal variations within data, facilitating a deeper understanding of temporal dynamics. One of the most compelling motivations for studying TS analysis is its broad applicability across various sectors.

Traditional statistical models increasingly struggle to keep pace with the complexities and scale introduced by the advent of big data. The sheer volume, velocity, and variety of data generated in modern contexts—ranging from social media interactions and e-commerce to IoT devices and real-time sensors—demand more sophisticated approaches. Traditional models often fall short due to their inherent limitations in handling large-scale data efficiently, their need for assumptions about data distribution and independence, and their lack of adaptability to rapid changes and non-linear relationships within the data.

In contrast, modern computational techniques, such as Machine Learning (ML) and Deep Learning (DL), have risen to prominence as they inherently cater to these complexities. These methods are equipped to automatically extract patterns from vast datasets without explicit programming for each task, adapt to new trends in real-time, and manage unstructured data like images, video, and text more effectively. Consequently, as the data landscape evolves to encompass more complex and high-dimensional data, the reliance on more advanced analytics and predictive models becomes not just beneficial but essential for extracting valuable insights and making informed decisions in the era of big data.

Some literature review highlighted some techniques capable of handling large datasets. It is possible to examine TS patterns and create predictions of future samples, as discussed in Mahalakshmi et al. [5]. Models based on ML, e.g., Long short-term memory (LSTM) and Temporal Convolutional Network (TCN), have shown promising results, [6, 7], as an alternative to statistical models. The ability to accurately forecast future events based on historical data is invaluable, offering a competitive advantage in various industries. For instance, in retail, forecasting demand for products can optimize inventory management, reducing costs and increasing customer satisfaction. In transportation, predicting traffic patterns can enhance route planning and reduce congestion.

However, even the most sophisticated prediction models based on DL are not immune to the challenges posed by concept drift. This phenomenon occurs when the statistical properties of the target variable, which the model is trying to predict, change over time. This phenomenon can degrade the performance of models as the assumptions under which they were initially trained no longer hold true. For instance, changes in consumer behavior, economic shifts, or new market regulations can all induce concept drift that affects models trained on historical data.

The importance of addressing concept drift cannot be overstated, especially in dynamic environments where decision-making relies heavily on accurate and timely predictions. Failure to manage concept drift effectively can lead to significant predictive errors, resulting in financial losses or missed opportunities. Understanding and adapting a TS analysis to concept drift is essential for maintaining the accuracy and reliability of predictive models.

1.1 Hypotheses

To address emerging challenges in TS analysis, two hypotheses have been developed. The first hypothesis involves the segmentation of a TS with the aim of removing malicious samples that tend to represent concept drift. This approach posits that by identifying and isolating segments of the data that are significantly different in behavior or statistical properties from the rest, one can effectively cleanse the dataset, removing these outliers that may skew the model’s learning and prediction capabilities. This method not only helps in maintaining the integrity and relevance of the model but also enhances its performance by focusing on more stable and representative data segments.

The second hypothesis proposes optimizing existing Drift Detectors (DDs) in the literature through a user-independent approach, employing an unsupervised methodology. This strategy is designed to reduce the necessity for users to have an in-depth understanding of the specific environment they are analyzing. By leveraging unsupervised learning techniques, the optimized DDs can autonomously identify and adapt to changes in the

TS or data stream without predefined labels or extensive manual adjustments. This approach broadens the applicability of DD mechanisms, making them more accessible and effective across various domains and data scenarios where expert knowledge is limited or unavailable.

1.1.1 Time Series Segmentation

We developed a hypothesis for scenarios where the temporal samples of a TS show non-stationary variability; that is, it does not manifest stability in the behavior of its samples over time with a constant average. As far as we know, no study concentrated on TS segmentation resorted to the use of stationary analysis. Usually, segmentation processes are based on static thresholds [8], classical techniques such as sliding window assessments [9], and even complex DL procedures [10]. Thus, we consider our hypothesis the first proposal considering stationarity to perform TS segmentation, taking advantage of this important feature when describing a TS. It is based on TS preprocessing, which can lead to improvements in new samples prediction and even a size reduction for further storage. This preprocessing consists of segmenting data from the samples that present a non-stationary behavior over time. The advantage of this approach lies in the simplicity of stationary analysis to accurately identify samples to be segmented without relying on more sophisticated techniques, such as DL methods.

1.1.2 Drift Detectors Optimization

DDs are specialized algorithms designed to quickly and accurately identify concept drift instances in time-sensitive situations [11]. These detectors assess key performance metrics of a signal, such as predictive accuracy or error rates, over consecutive time intervals. Their main goal is to promptly identify substantial changes in signals, indicating drifts, anomalies, trends, or other critical events, to facilitate timely corrective actions.

There is a clear need in the literature for the optimization of DDs through tuning techniques that are independent of user knowledge [12, 13, 14]. Despite the advancements in this domain, the prevalent tuning techniques largely depend on supervised learning paradigms [15], which may not align with the practicalities of real-world implementations. The challenge of acquiring labeled data for the purpose of training and evaluating DDs presents a significant obstacle, rendering the process not only impractical but also resource-intensive. This limitation hinders detailed detector parameter optimization, potentially resulting in suboptimal detection and adaptive model performance. In this context, our research aims to address this autonomous tuning process. To achieve this goal, we propose an innovative unsupervised learning approach to fine-tune DDs.

This unsupervised learning approach yields numerous substantial advantages. Firstly, it amplifies the adaptability and performance of detectors across diverse data distribu-

tions which can be extended to real-world scenarios. The optimizer facilitates automatic fine-tuning of hyperparameters, alleviating the necessity for manual adjustments and expert intervention. This, in turn, broadens the applicability of the detectors. Furthermore, an efficient hyperparameter optimizer contributes to heightened generalization capabilities of DDs, ensuring robust performance on unseen data. This aspect assumes a significance in unsupervised settings where data distribution may dynamically evolve, requiring detectors to adjust seamlessly without labeled information. Additionally, an automated hyperparameter optimization framework helps mitigate overfitting or underfitting issues, thereby enhancing the overall reliability and accuracy of change detection models.

Also, we hypothesize that relying solely on default values for DDs, commonly found in prevalent literature and libraries like `scikit-multiflow` [16] and `river` [17], might not always provide the optimal configuration. Building on this idea, our methodology aims to enhance this decision-making process by using an unsupervised approach for hyperparameter tuning. This approach is grounded in the Change Detector Segmentation (CDS) framework [18]. By leveraging the principles of the CDS framework, our unsupervised tuning strategy aims to surpass the limitations of default DDs hyperparameters. It seeks a more precise configuration that aligns with the dynamics of distribution shifts in signal data.

1.2 Contributions

The main contributions of this thesis are as follows:

- **A new proposal for TS segmentation based on stationarity:** A framework was created that uses the information obtained by stationary analysis to properly adjust the DDs hyperparameter to determine which samples should be segmented. This approach enhances new samples prediction and reduces the TS size.
- **Augmented Dickey-Fuller-based Segmentation:** The Augmented Dickey-Fuller (ADF)-based Segmentation is a novel approach to TS segmentation. This method effectively reduces the number of samples required for analysis while enhancing predictive capabilities. This technique not only streamlines the data processing but also preserves the essential characteristics of the TS, ensuring that the predictive models perform optimally without compromising the temporal integrity of the data.
- **Configurable Signal Generator:** The Configurable Signal Generator (CSG) addresses a notable gap in the landscape of signal generation tools, offering a mechanism for generating signals with precisely defined change points. This capability ensures that the signal generation process is fully configurable, enabling accurate simulation and analysis of various signal behavior scenarios.

- **Unsupervised Tuning for Drift Detectors:** The unsupervised tuning for DDs enables these methods to outperform default configurations in most scenarios. This approach requires no prior knowledge from the user, enhancing the detectors' effectiveness and accessibility by automatically adjusting their settings to optimally respond to data dynamics.

1.3 Thesis Structure

The present thesis is organized as follows:

- **Chapter 2:** The second chapter of the thesis introduces the preliminaries, which contains fundamental information necessary to support the subject matter addressed about TS analysis and related topics. This foundational chapter lays out key concepts such as stationarity, concept drift, DDs, and tuning, among others. Understanding these concepts is essential as they form the underpinnings of more complex discussions on how TS data can be analyzed, interpreted, and utilized effectively. By establishing a solid theoretical base, this chapter ensures that the subsequent analyses and methodologies employed throughout the thesis are built on a comprehensive understanding of these fundamental principles.
- **Chapter 3:** The third chapter focuses on Literature Review and undertakes a comprehensive examination of the most significant works in the field of TS and signal analysis, with a focus on data prediction, segmentation, anomaly detection, concept drift, and the optimization of DDs. This meticulous review outlines the advancements in ML and DL that have enhanced predictive accuracy and computational efficiency. It also explores methods for segmenting complex TS into analytically significant segments.
- **Chapter 4:** Chapter 4 of the thesis presents the author's proposed methodology for segmenting TS, highlighting the significance of stationary analysis and providing comparative results between DL models and traditional statistical models. This comparison demonstrates that the segmentation process is a beneficial step in TS analysis. The chapter also shows how segmenting a TS into more manageable, homogeneous parts allows for more accurate modeling by both types of approaches, but particularly enhances the performance of DL models. These segments, when analyzed under the assumption of stationarity, allow for more precise predictions and robust analysis, thereby proving the utility of segmentation in breaking down complex TS data into simpler, more analyzable components.
- **Chapter 5:** Chapter 5 of the thesis introduces a study on unsupervised tuning for DDs, specifically designed to function without requiring the user to have detailed

knowledge of the signal or TS under analysis. This chapter explores how unsupervised approaches can autonomously adjust and optimize DDs, thereby simplifying the application for users who may not have in-depth expertise in TS analysis. Additionally, the chapter includes comparative analyses between this unsupervised tuning method and supervised techniques for hyperparameter optimization, such as Random Search (RS) [19]. These comparisons are relevant in validating the efficiency of the unsupervised approach. By demonstrating that unsupervised tuning can match or even surpass the performance of supervised methods, which is a gap investigation in the literature.

- **Chapter 6:** Chapter 6 of the thesis serves as the conclusion, encapsulating all topics related to TS analysis discussed throughout the thesis. It comprehensively summarizes the findings, highlighting the key discoveries and the innovative methodologies developed. The chapter also addresses the limitations encountered during the research, providing a assessment of the areas where further development is needed. Moreover, it outlines the next steps for future research, suggesting avenues to refine and enhance the field of TS analysis. These recommendations for future work are based on the current study's outcomes and the identified gaps, aiming to propel further advancements and encourage ongoing exploration in this dynamically evolving area.

2 PRELIMINARIES

In this chapter, fundamental theories that support the development of this thesis will be conducted. Furthermore, the chapter will explore the most influential findings in the field related to the thesis topic. By examining these key discoveries, we aim to draw a comprehensive picture of the current landscape of research, emphasizing how these findings have progressed over time and what they reveal about the potential future directions of the field. Additionally, this chapter will highlight some limitations of current approaches as identified in existing research.

2.1 Time Series

A TS is fundamentally a sequence of data points collected at successive points in time, typically at uniform intervals. It is represented as an ordered set $X = [x_1, x_2, \dots, x_T]$, where each x_t is the value of the series at time t , and T denotes the total number of observations within the series. This ordering reflects the temporal dynamics inherent to the dataset, making TS analysis distinct from other data analysis techniques where data ordering might not be of importance.

Definition 1 (Time Series) *A TS can be defined as a set of sequential data, ordered in time [20]. It can be collected at equally spaced time points and we use the notation x_t with ($t = \dots, -1, 0, 1, 2, \dots$), i.e., the set of observations is indexed by t , representing the time at which each observation was taken. If the data was not taken at equally spaced times, we denote it with $i = 1, 2, \dots$, and so, $(t_i - t_{i-1})$ is not necessarily equal to one [21].*

The main characteristic of a TS is its capability to exhibit different behaviors over time, such as trends, seasonality, cycles, and irregular fluctuations. Analyzing these characteristics can help in understanding the underlying patterns within the data, which is essential for effective modeling and forecasting.

1. **Trend:** The trend in a TS represents the long-term progression of the data, which can be either upward or downward. This trend component reflects the overall direction in which the series is moving over a significant period. Trends can be caused by persistent influences on the data, such as increasing population, technological advancements, or broad economic changes. In TS, identifying and modeling trends is vital because it allows for the prediction of future values by extrapolating these long-term movements. Techniques such as moving averages, linear regression, and

more sophisticated methods like Hodrick-Prescott filters are used to smooth out short-term fluctuations and highlight the underlying trends [22].

2. **Seasonality:** Seasonality refers to patterns that repeat at regular intervals over time, whether hourly, daily, weekly, monthly, or yearly. This component is driven by seasonal factors that affect the TS at specific times within a given period. For instance, retail sales increase significantly during the holiday season each year. Accurately modeling seasonality allows businesses and researchers to predict future peaks and troughs within the data, facilitating better planning and resource allocation. Seasonal decomposition of TS and methods incorporating seasonal adjustment factors, such as SARIMA (Seasonal AutoRegressive Integrated Moving Average) [23], are commonly employed to isolate and forecast seasonal effects.
3. **Cyclical Changes:** Cyclical changes are fluctuations occurring at irregular, often undetermined intervals, lasting longer than a year. Unlike seasonality, which is predictable, cyclical fluctuations are often influenced by broader economic, environmental, or political factors and do not have a fixed period. These cycles can be seen in economic data, where periods of rapid economic growth are followed by recessions. Cyclical components require complex modeling approaches, such as spectral analysis or filtering techniques like the Band-Pass filter [24], to estimate and predict because of their irregular nature.
4. **Irregular fluctuations:** Irregular fluctuations, or also considered as noise in a TS are caused by unpredictable or random variations that do not follow any identifiable seasonal or cyclical pattern. These fluctuations can result from unforeseen events such as natural disasters, market panics, or technological innovations. Modeling these components is challenging because they do not exhibit systematic behavior that can be easily predicted. Techniques such as exponential smoothing or random walk models are often used to account for random variations within the data [25], providing a way to adjust forecasts to reflect unexpected changes in the trend and seasonal components.

2.1.1 Time Series Forecasting

The essence of TS forecasting is to predict future values based on previously observed values. The forecasting task involves finding a function f that can predict the h -th future value of X , denoted as $\hat{x}_{t+h}$, based on i past observations:

$$\hat{x}_{t+h} = f(x_{t-i-1}, x_{t-i-2}, \dots, x_{t-1}, x_t) \quad (2.1)$$

Here, i represents the input window size, indicating how many past points are considered to predict the future value, and h is the forecast horizon, indicating how far into the future the prediction is aimed.

One-step-ahead Forecasting

One-step-ahead forecasting is a fundamental approach in TS analysis where the objective is to predict the very next value in the sequence, with the forecast horizon h set to one. This method is particularly favored for its straightforwardness and high precision, as it only requires the immediate past data to generate forecasts. The simplicity of this forecasting model often leads to reduced computational costs and quicker response times, which are advantageous in real-time applications such as stock trading or operational adjustments in businesses.

The primary analytical methods used in one-step-ahead forecasting include linear regression, ARIMA (Auto Regressive Integrated Moving Average), and simpler models like naive forecasting [26], where the prediction for the next period is assumed to be the same as the last observed value. These models are often refined with error correction techniques to improve accuracy, adjusting forecasts based on the residuals of previous predictions. This approach is effective in maintaining the stability and reliability of the forecasts in dynamic environments where quick decisions are fundamental.

Multi-step-ahead Forecasting

Multi-step-ahead forecasting aims to predict multiple future points in the TS, extending the forecast horizon h beyond just one future point. This method is indispensable in strategic planning and long-term decision making, where understanding the future trajectory of a series is a need. However, multi-step-ahead forecasts are more complex and less accurate than one-step-ahead forecasts due to the accumulation of forecasting errors at each step and the increasing uncertainty associated with longer prediction intervals.

To manage these challenges, advanced models such as Vector Autoregressions, state space models, and ML approaches like Recurrent Neural Networks (RNNs) are typically employed [27]. These models can capture complex interactions within the data over multiple time steps, potentially reducing the error propagation as the forecast horizon increases. Techniques like direct forecasting, where a separate model is developed for each forecast horizon, and iterative forecasting, where a one-step model is applied repeatedly to forecast further into the future, are also used depending on the specific requirements and the nature of the data.

2.1.2 Stationarity in Time Series Analysis

Stationarity is a fundamental concept in the analysis and forecasting of TS data. A TS is considered stationary if its statistical properties such as mean, variance, and autocorrelation are constant over time [28]. In simpler terms, a stationary TS has no trend, its fluctuations around its mean are roughly constant across time, and it exhibits consistent cyclical patterns unaffected by the time of observation. This concept is relevant because many of the analytical methods used for forecasting in TS assume stationarity as a prerequisite.

Definition 2 (Stationarity) *According to the definition of random processes [29], a discrete-time or continuous-time random process $X(t)$ is stationary if the joint distribution of any set of samples does not depend on the placement of the time origin. This means that the joint cumulative distribution function of $X(t_1), X(t_2), \dots, X(t_k)$ is the same as that of $X(t_1 + \tau), X(t_2 + \tau), \dots, X(t_k + \tau)$ for all time shifts τ , all k , and all choices of sample times $t_1, \dots, t_k$.*

Stationarity in TS can be broadly categorized into three types:

1. **Strict Stationarity:** A strict stationary series satisfies the mathematical condition that the joint distribution of any moments of any degree (e.g., means, variances, etc.) is the same regardless of the time at which they are measured. This definition is often too strict for practical applications, as it requires checking an infinite number of conditions.
2. **Second-order Stationarity (Weak Stationarity):** A series is second-order stationary if the mean and variance are constant over time and the covariance between two time points depends only on the distance or lag between the two points and not on the actual time at which the covariance is computed. This is the most commonly used form of stationarity in TS analysis.
3. **Trend Stationarity:** A TS that exhibits a deterministic trend but whose fluctuations around this trend are stationary. In such series, once the trend is removed, the residuals can be modeled as a stationary series.

2.1.3 Augmented Dickey–Fuller Test

The ADF test is a statistical analysis that verifies the null hypothesis that a unit root is present in a TS sample [30]. The null hypothesis can inform whether a given TS is stationary or not. It checks if the TS can be represented by a unit root with a time-dependent structure [31]. If this hypothesis is rejected, we can assume that the TS is

stationary. In other words, the statistics are not affected by time offsets. The unit root exists when the α assumes the value one, according to Equation:

$$Y_t = \alpha Y_{t-1} + \beta X_e + \epsilon, \quad (2.2)$$

where t represents the time, Y_t represents the value of the TS at time t , and X_e is an exogenous variable, which is explanatory and belongs to the TS. ϵ is a serially uncorrelated, zero-mean stochastic process with constant variance σ^2 . The ADF test assumes the α value equal to one in the Equation (2.3):

$$Y_t = c + \beta t + \alpha Y_{t-1} + \phi \Delta Y_{t-1} + \epsilon_t, \quad (2.3)$$

where c is a constant, β is the coefficient on a time trend, Y_{t-1} represents the lag order equals one, ΔY_{t-1} is the first difference of the series at time $t - 1$ and ϵ is the exogenous variable.

However, the confirmation or rejection of the hypothesis can be simply verified by analyzing the p-value of the ADF test. A p-value below a threshold determines rejection of the null hypothesis, otherwise suggesting the failure to reject the null hypothesis. This representation can be verified according to the conditions given below.

$$\delta = \begin{cases} 1, & \text{if } p\text{-value} < \gamma \\ 0, & \text{otherwise} \end{cases}, \quad (2.4)$$

where δ represents a TS stationarity and γ represents a determined threshold. We assume the stationarity of the series if the value generated by the conditional is one. A non-stationary series is obtained otherwise.

In addition to the p-value analysis, it is possible to check the ADF statistic. In this work, we refer to the ADF statistic as an ADF value. Usually, when this value reaches a positive representation it will describe a non-stationary TS. Otherwise, a negative value represents stationarity in a TS. The more negative this value is, the greater the confirmation that its statistic does not depend on the temporal offset.

2.2 Concept Drift

Concept drift refers to the phenomenon where the statistical properties of the data target variable, which the model is trying to predict, change over time in unforeseen ways. The model can refer to a variety data, such as data streams and TS. This can result in previously learned patterns becoming obsolete, leading to a degradation in model performance. Concept drift is particularly significant in dynamic environments where the data generating processes are not static but evolve over time.

2.2.1 Concept Drift in Time Series

Regarding TS data, concept drift manifests as changes in the underlying relationships between input features and the output over time. This changes could be related to the definition of irregular fluctuations. For instance, in economic TS, the influence of certain economic indicators on market trends may diminish or increase in unpredicted ways due to regulatory changes, market saturation, or shifts in consumer preferences. Some strategies are adopted for handling concept drift in TS:

1. **Detection:** The first step in managing concept drift is detecting it. This involves monitoring the model's performance over time to identify any significant drops in accuracy or effectiveness. Statistical tests, control charts, or monitoring prediction residuals can help spot changes. For instance, a sudden increase in prediction errors or changes in error distribution can signal that the model is no longer capturing the underlying patterns accurately.
2. **Adaptation:** Once drift is detected, the next step is to adapt the model to realign it with the new data patterns. This adaptation might involve retraining the model with recent data that reflects current conditions or incrementally updating the model using online learning algorithms. In some cases, adjusting the model's parameters or incorporating new variables that capture the changed influences can be necessary to accommodate the new patterns.
3. **Verification:** After adapting the model, it's necessary to verify that the changes have restored its predictive power. This involves testing the updated model against recent data to ensure it performs effectively under the new conditions. Verification helps confirm that the adaptations have successfully addressed the concept drift and that the model is now better aligned with the current data characteristics.

2.2.2 Concept Drift Detector

Concept Drift Detector or Change Detector is commonly employed in handling data streams, which consist of an infinite sequence of data denoted as $S = (x_1, y_1), (x_2, y_2), \dots, (x_t, y_t), \dots$. Here, each instance is represented by (x_t, y_t) , where x_t represents a d -dimensional vector arriving at time stamp t , and y_t denotes the class label of x_t [32]. For univariate TS, the DD algorithm analyzes each sample X of the TS at time t . It anticipates that a given point X_{t+n} , with n as a time shift, should have a distribution similar to the point X_t . Any deviation prompts an alarm. However, in unsupervised approaches, lacking y_t information complicates the DD process significantly.

DDs analyze past data patterns to predict potential changes. They are commonly deployed to detect concept drift, which means alterations in data behavior over time

[33]. Such changes are frequent in data stream analysis [34]. Calibrating these detectors involves selecting hyperparameters that align best with data behavior, often employing techniques for enhancement.

2.3 Machine Learning and Time Series Analysis

The use of ML techniques has become increasingly prevalent in handling TS data [35]. Traditional methods such as Linear Regression and Decision Trees have laid the groundwork by offering simplicity and speed, which are beneficial for quick analyses and forecasts. However, as the complexity and volume of data grow, more sophisticated models have been employed to capture nonlinear relationships and intricate patterns more effectively.

2.3.1 Ensemble Learning Techniques

Ensemble learning techniques, which combine multiple predictive models to improve accuracy and robustness, are particularly well-suited to handling the complexities of TS data [36]. These models capitalize on the strength of various individual models to produce a more accurate and stable prediction by reducing variance and bias, essential qualities for forecasting in volatile environments.

- **Random Forests:** Random Forests are an ensemble of Decision Trees, typically trained with the “bagging” method to enhance the overall prediction accuracy [37]. Each tree in the forest is built from a random subset of data points and features, making them diverse and reducing the model’s susceptibility to overfitting. In TS forecasting, Random Forests are used to predict demand and supply fluctuations by capturing nonlinear relationships between features without the need for extensive data preprocessing.
- **Gradient Boosting Machines :** Gradient Boosting Machines (GBM) are another powerful ensemble technique that builds trees sequentially, with each new tree correcting errors made by previously trained trees [38]. This method, known as ‘boosting’, focuses on reducing bias and can be particularly effective in scenarios where the prediction error needs to be minimized. GBMs are adept at handling various irregularities in data, such as non-linear dependencies and seasonality in energy data, which are common in TS forecasts.

2.3.2 Advanced Deep Learning Models for Time Series

Deep Learning (DL) has brought about a transformation in the field of TS analysis due to its capability to discern and model complex patterns and dependencies within

extensive datasets. The following items will demonstrate the most relevant techniques for TS analysis.

- **Recurrent Neural Networks and LSTM Networks:** RNNs are designed to handle sequential data, with the unique feature of memory cells that retain information across time steps. This makes them ideal for application to predicting future trends. Long Short-Term Memory networks, a special kind of RNN, are particularly effective in avoiding the long-term dependency problem, allowing them to remember information for prolonged periods. This is fundamental for accurate load forecasting, where inputs from several days, weeks, or even months ago can influence future outcomes.
- **Convolutional Neural Networks:** Though primarily known for image processing, CNNs have been adapted for TS data, where they can process sequences as one-dimensional spatial inputs. CNNs are used to detect patterns and anomalies in usage data. Their ability to automatically detect important features without the need for manual extraction makes them highly efficient and scalable for large datasets.
- **Temporal Convolutional Networks:** TCNs offer a compelling alternative to models based on recurrent architectures for handling sequence data. TCNs utilize causal convolutions—convolutions where the output at time t is convolved only with elements from time t or earlier in the previous layers. This structure ensures that the model respects the temporal directionality, meaning that future inputs do not influence past outputs.

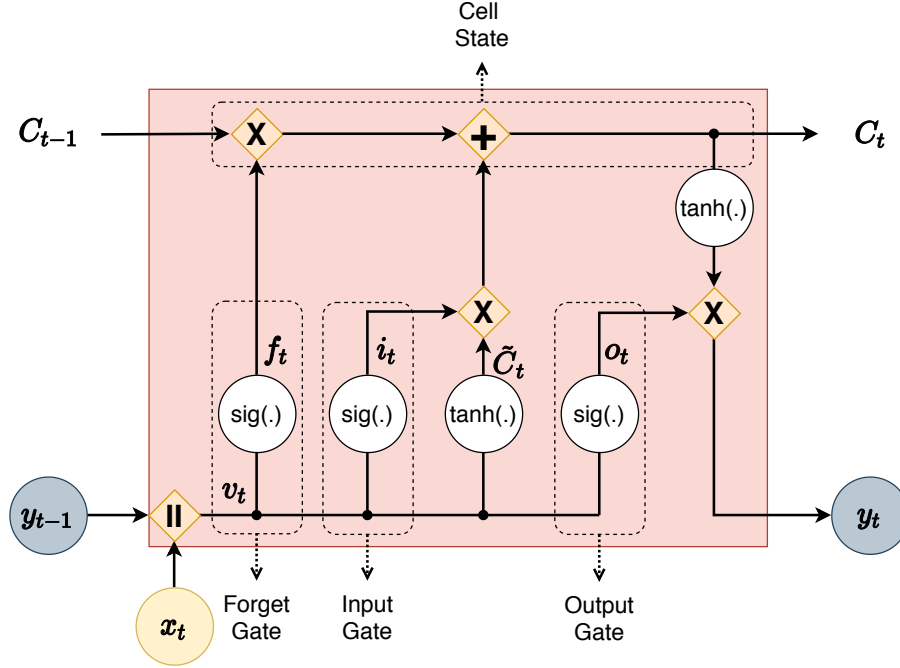
2.3.3 Long Short-Term Memory

LSTM is a classical neural network for dealing with TS. In this network, the information is propagated throughout a chain of repeating units of neural network located in the hidden layer. However, standard RNNs suffer from error backflow problems as gradient vanishing and explosion, which restrain long-term dependency learning. The first may lead to very slow learning, and the latter, to weight oscillation.

The backflow calculation drawback was one of the main motivations for the development of LSTM [39]. This network, a particular case of RNN, uses a gating mechanism for learning long-term dependencies without losing the short-term capability. These gating mechanisms are inside memory cells, which are located in the hidden layer. Each unit in traditional LSTM has the aspect presented in Fig. 1.

Hyperbolic tangent function ($\tanh(\cdot)$) transforms the values to the range from -1 to 1, and the sigmoid function ($\text{sig}(\cdot)$) to values between 0 and 1. This property makes

Figure 1 – LSTM cell.



$\text{sig}(\cdot)$ act as a gate, since the values that are transformed to 0 are forgotten by the network and the values that are transformed into 1 are kept.

Each cell is mainly composed of three gates (forget, input and output gates) and a cell state. The forget gate (f_t) is the main gate to select which information should pass forward to the next cell or be eliminated. The input gate (i_t) is used to update the cell state and to select what will be written to the cell. The output gate (o_t) decides the values that will be part of the output. The cell state (C_t) allows the gradient flow.

2.3.4 Temporal Convolutional Network

TCN is a specific CNN that has the capability of dealing with TS. For this purpose, it uses causal convolutions, which performs the convolution operation depending only on past values. This kind of convolution may be submitted to dilation. In TCN, dilated convolutions in one dimension are used seeking to explore long-term patterns. The procedure for doing this is skipping d values between the inputs of convolution.

Aiming to increase the receptive field of the network, b convolutional blocks can be stacked. Since it increases the number of parameters, the learning process becomes more complex. The parameters b , k and d are factors that define the receptive field of the network by the following Eq. 2.5. For an adequate use of TCN, the receptive field should cover past history, which, in turn, should cover seasonality.

$$\text{receptive field} = b \times k \times d_n \quad (2.5)$$

3 LITERATURE REVIEW

3.1 Overview

The literature review on TS analysis encompasses a broad range of fundamental themes for understanding and advancing the field. This overview covers areas such as TS prediction, TS segmentation, concept drift, DDs, and optimization of these detectors. TS prediction is a foundational aspect that focuses on forecasting future values based on past and present data, utilizing statistical and ML models to extract patterns. TS segmentation, meanwhile, involves dividing a TS into meaningful segments according to its intrinsic characteristics, facilitating more detailed analysis and concept drift detection. The phenomenon of concept drift in TS signifies the changes in underlying data patterns over time, which can affect the performance of predictive models. Addressing this, DDs are developed to identify and adapt to these changes, ensuring models remain accurate and relevant. Lastly, the tuning of drift detectors is important for enhancing their efficiency and sensitivity, involving techniques that adjust detection mechanisms to respond effectively to the dynamics of data flow. This literature review provides a comprehensive examination of these areas, highlighting their interconnections and the innovations that drive their evolution.

3.2 Background on Time Series Analysis

TS are explored in several types of applications and are subject of study until today. Most of these applications are focused on forecasting [40, 41] and feature extraction [42] approaches, as can be seen in the following studies.

Due to the growth of generated data volume, there is a need for its treatment. Size reduction is an important procedure before carrying out analyses. The OSTS method was proposed by Carmona-Poyato et al. [43] and consists of the segmentation of points in a TS to reduce its size. This process is based on the A* algorithm and performs interpolations using optimal polygonal approximations. This method was used for comparison in this thesis with different segmentation techniques to assess its impact on the predictive capacity of new samples in a TS.

TS segmentation from human specified breakpoint detection was introduced in Lee et al. [10]. The process is conducted by exploiting DL techniques for automatically extracting features. However, this technique was not used as a preprocessing step to predict new samples in a TS. Despite it being very interesting, it requires much data processing before applying breakpoint detection. For our approach, we explored an automatic way

of detecting these breakpoints simply from the analysis of TS stationarity.

Temporal segmentation is also an alternative as discussed by Bessec et al. [44] and Jamali et al. [8]. The former performs predictions at a day-ahead with combinations of linear regression techniques, Markov-switching models, and thresholds. The latter requires user knowledge to define the ideal parameters for the proposed segmentation method, based on the TS characteristics. Both works rely on prior knowledge and since they are applied in specific environments, it is difficult to generalize. The work addressed by Hooi et al. [45] also performs TS segmentation, but not in a temporal approach, it is based on patterns to create a vocabulary capable of identifying possible regions to perform segmentation. It is dependent on user intervention and knowledge.

Other proposals are derived from classical methods, such as the survey by Keogh et al. [9], which describes traditional representativeness techniques from the perspective of data mining. Different segmentation methods and the evaluation of their impact on representativeness in the TS are discussed; most of them are based on linear approximations. Among the methods used for comparison, there are sliding windows, bottom-up, and top-down approaches. There is also a proposal by the authors called SWAB, which consists of the union of the sliding windows and Bottom-up methods. This study shows a classic approach to segmentation in TS, where the samples to be evaluated do not depend on static information inherent in their behavior. In this thesis, techniques placed on stationarity are proposed to perform segmentation, as an alternative method to improve the predictive capacity of new samples.

Table 3.2 presents a summary of some relevant works in the literature that address the topic of TS segmentation. To our knowledge, our segmentation proposal differs from the others in Table 3.2 because it is the first to deal with segmentation to improve new sample predictions through an approach based on TS stationarity.

3.2.1 Current Limitations

According to the most recent research on TS segmentation, successful outcomes still heavily depend on the user's knowledge of the data being analyzed. Additionally, to date, no approaches have been identified that utilize stationarity as a decisive factor for driving segmentation. This omission requires more complex analyses to effectively segment TS data. Such complexities highlight the need for further research to develop methodologies that can leverage stationarity, potentially simplifying the segmentation process and enhancing its effectiveness without extensive reliance on user expertise. This gap in the literature suggests significant opportunities for advancing the field of TS analysis.

Also, this aspect of integrating stationarity to enhance segmentation and consequently improve the prediction process has not been widely investigated, indicating a fertile ground for future research. Developing methodologies that can automatically incor-

Table 1 – Comparison of TS segmentation techniques in the literature.

Reference	Segmentation Technique	Purpose of Segmentation
Carmona-Poyato et al. (2020) [43]	Based on A* algorithm with optimal polygonal approximations	Data representation reducing the dimensionality with minimum information loss
Lee et al. (2018) [10]	Unsupervised approach, based on DL	Automatic knowledge extraction
Hooi et al. (2017) [45]	BeatLex, based on patterns to match segments of the TS	Vocabulary-based approach to match segments of the TS in an intuitive and interpretable way
Bessec et al. (2015) [44]	Temporal segmentation based on hourly and seasonal segmentation	Forecast spot prices in France with double temporal segmentation
Jamali et al. (2015) [8]	Temporal segmentation based on thresholds of the TS features	Segments the changes in the vegetation TS to identify the change type and its characteristics
Keogh et al. (2004) [9]	Sliding windows, Bottom-up, Top-Down, and SWAB	Empirical comparison of TS segmentation algorithms from a data mining perspective

porate such fundamental statistical properties could significantly streamline the predictive modeling workflow. This represents a substantial opportunity to advance the field of TS analysis, potentially leading to more robust and efficient predictive models that require less user intervention and deeper domain expertise.

3.3 Background on Concept Drift and Drift Detectors

A significant study, outlined in [46], performed a comparative analysis of 14 different DDs. This investigation evaluated their accuracy in detecting actual drift instances within the dataset. Among the assessed detectors were HDDM_A, Fisher Test Drift Detector (FTDD), Wilcoxon Rank Sum Test Drift Detector (WSTD), FHDDM, and HDDM_W. An unexplored gap in this study lies in the absence of tuning techniques for the DDs, given that all were compared with their default configurations.

Santos et al. [47] discuss the concept of data streams and concept drift, where the distribution of data changes over time. Several approaches have been proposed to address concept drift, including using ensembles, focusing on recurring concepts, and concept drift detection methods. Various DD methods have been proposed, each with its own hyperparameters, and optimal values vary depending on the datasets used. The paper introduces a genetic algorithm designed to optimize DD hyperparameters. This approach aims to identify suitable configurations adaptable to diverse scenarios, facilitating a fair comparison of methods by adjusting hyperparameter values based on the dataset characteristics utilized in the experiments.

Gemaque et al. [48] introduced a taxonomy for unsupervised concept drift approaches via a systematic literature review. This review revealed two primary categories within recent scientific publications on concept drift detection: articles encompassing surveys and analyses of diverse concept drift approaches, and articles proposing unsupervised DD methods. The unsupervised and semi-supervised drift detection methods were further classified into batch-based and online-based methods. Upon analyzing the batch-based methods, it was noted that their primary distinction lies in whether they monitor significant data distribution changes across the entire arriving batch or solely on a selected instance set. The online-based methods were also examined, subcategorized based on their approach to comparing the two involved windows during drift detection.

Agrahari et al. [49] discuss data streams and the need to analyze data near real-time. Data stream mining involves detecting changes in non-stationary data distribution, which is a major challenge. Data points in a stream are generated sequentially and independently under some probability distribution. To effectively mine data streams, examples must be processed only once, learned under memory restrictions, processed in a limited amount of time, and ready to predict class labels on demand. In the context of data stream mining, one of the biggest challenges is dealing with semi-supervised or unsupervised data, which often have very few or no labeled data available for training. Or even solutions based on methods such as Meta-Learning and Active Learning [13]. This lack of labeled data makes it difficult to train supervised learning models and demands the use of unsupervised or semi-supervised methods. Therefore, developing effective unsupervised and semi-supervised methods for data stream mining is crucial for making accurate predictions and discovering valuable insights in these challenging scenarios.

Some reviews focus on the study of DDs in the streaming data environment [50]. One of the main challenges in this context is the scarcity of labeled data due to the large volume of data processed in a short period of time. However, the accuracy of the analysis requires labeled data. Therefore, there is a need to develop unsupervised or semi-supervised algorithms to address this issue and achieve high accuracy.

3.4 Background on Tuning for Drift Detectors

In the domain of Hyperparameter Optimization [51], the RS approach has been extensively studied [52]. Nevertheless, it is imperative to underscore that these methodologies predominantly operate in a supervised context, requiring validation through comparisons with labeled datasets.

The current state of tuning for DDs has reached a complex point where the necessity for more sophisticated and adaptive methodologies is evident [53]. As data streams become increasingly complex and voluminous due to the rise in real-time applications

across various industries, traditional tuning methods have struggled to keep pace. These conventional approaches, often reliant on supervised techniques, require extensive manual effort and a deep understanding of the data’s underlying patterns. However, such methods can be inflexible and time-consuming, posing significant limitations in environments where data properties can change rapidly and unpredictably. The reliance on historical data and the need for frequent re-calibration further exacerbate the challenge, as they may not effectively capture or respond to real-time changes in the data stream.

3.4.1 Current Limitations

Upon a thorough analysis, a significant gap emerges in hyperparameter tuning for DDs. Notably, prior research in this domain has largely overlooked the exploration of unsupervised tuning methods. Most studies on DDs and their optimizers still rely heavily on supervised sampling and a profound user knowledge of the underlying data, which necessitates a high level of expertise and limits the applicability in more automated settings.

Furthermore, the dependence on supervised approaches highlights a crucial limitation: the need for labeled data, which is often costly or impractical to obtain in real-time scenarios. In contrast, unsupervised methods, which could operate without pre-labeled data and extensive user input, have not been sufficiently explored, representing a significant oversight in the field.

Addressing this research gap, this study introduces an unsupervised approach for tuning hyperparameters of drift detectors. This development is particularly relevant as it promises to reduce the reliance on expert knowledge and labeled data, potentially broadening the usability and efficiency of DDs in various applications.

4 TIME SERIES SEGMENTATION

4.1 Overview

TS segmentation is a process involving the division of a continuous TS into distinct segments based on various criteria such as similarity, trends, or change points. This method facilitates a deeper understanding of the underlying structure of the data, enabling analysts and researchers to identify patterns, anomalies, and shifts. The segmentation aims to simplify the analysis of complex TS data by breaking it down into manageable, homogeneous segments. Each segment is characterized by certain properties or behaviors that are consistent within the segment but differ from those in other segments. This approach allows for a more granular analysis of the data, highlighting underlying patterns and dynamics that might not be apparent in the full, unsegmented series.

Several techniques are employed in TS segmentation, each with its advantages and applicability depending on the nature of the data and the specific objectives of the analysis. Commonly used methods include:

- **Change Point Detection:** Identifies points in the TS where statistical properties, such as mean or variance, change significantly. Techniques such as CUSUM (Cumulative Sum Control Chart) and Bayesian methods are popular for change point detection.
- **Sliding Windows:** Involves moving a window of fixed size across the TS and computing a statistic, such as the mean or variance, for each window position. Segments are identified based on changes in these statistics.
- **Dynamic Programming:** Finds the optimal segmentation of a TS by minimizing a cost function that quantifies the error or variance within segments and possibly the cost of segment changes.
- **Clustering-Based Approaches:** Utilize clustering algorithms to group similar data points into segments. Methods like k-means or hierarchical clustering can be adapted for TS data by defining appropriate distance measures.

4.2 Proposed Approach: Time Series Segmentation Based on Stationarity

The proposed approach to perform the TS segmentation can be examined in Figs. 2 and 3. They demonstrate the steps followed to identify potential samples that do not

contribute to the TS stationarity. A framework and technique to perform the segmentation of a TS are proposed. In both cases, the process begins with the stationarity analysis of a given TS, defined as the ADF test. Each segmentation proposal has its peculiarities, but both depend on change detector processes to define the samples that should be segmented. The main objective of sample removal is to obtain a better-behaved TS model, which can improve various applications, such as prediction, data compression, and feature extraction. The framework, called Change Detector Segmentation (CDS), seeks to verify the impact of stationary analysis in aiding change detector techniques. Intervals with less stationarity serve as an extra alert for the change detector techniques to perform their detection. The ADF-based segmentation consists of a stationary analysis of the TS. The framework was created because as far as we know, there are no stationarity-based TS segmenters in the literature. Another reason is that choosing hyperparameters for these techniques is not a trivial task. The difference between the CDS and ADF-based segmentation is that the latter does not rely on the windowing process to perform segmentation. The process is carried out continuously.

4.2.1 Change Detector Segmentation Framework

The CDS is a framework to perform segmentation in TS. Segmentation is based on the analysis of critical intervals in the TS alongside change detectors and stationarity analyses. Algorithm 1 describes how the segmentation points are obtained.

Algorithm 1: Change Detector Segmentation code

Input: The Time Series (TS).
Output: A list with the Segmentation Points (SP).

```

1 Pseudo-code
2 Calculates the Global ADF test  $\lambda_{global}$  over the TS.
3 Calculates the Windowing ( $W$ ), obtaining the intervals from TS.
4  $SP = []$ 
5  $i = 0$ 
6 while  $i \leq \text{length}(W)$  do
7   | Calculates the local ADF from the interval  $\lambda_i$ 
8   | if  $\lambda_{global} < \lambda_i$  then
9   |   | SP append point ( $i$ ) as a critical interval.
10  | end
11  |  $i = i + 1$ 
12 end
13 Performs the Change Detector.
14 Performs the tuning.
15 SP append the points obtained from the tuning.
```

Fig. 2 demonstrates the entire process for performing the segmentation, each step will be defined in the following subsections:

4.2.1.1 Global ADF Test

The Global ADF test obtains the stationarity value of all TS samples, and it is applied according to the definition in Section 2.1.3. It is denoted by λ_{global} . This step is done to obtain information about the general behavior of the TS. The ADF value is obtained as the output, which serves as a threshold for evaluating each point in the TS. This process can be replaced by another one that is also able to describe the behavior of the TS.

4.2.1.2 Windowing

The windowing step consists of checking cyclical behaviors in a given TS to define its size. It can be given for days, weeks, years, or even longer intervals depending on the TS. The windowing allows comparing the Local ADF test of each interval with λ_{global} , described in the Thresholding box in Fig. 2. This process is conducted to identify critical intervals, which consists of λ_i greater than λ_{global} and makes the series less stationary. The critical intervals will be evaluated by the change detectors to examine samples individually, as they have more variability in the TS.

4.2.1.3 Local ADF Test

The Local ADF test obtains the stationarity value of a given interval of the TS and is denoted by λ_i , where i represents an interval at a given time. This interval is obtained by the windowing step and will be inputted into the thresholding step.

4.2.1.4 Thresholding

The Thresholding step compares the stationarity value of a given interval λ_i with λ_{global} for decision making. This interval is considered critical when λ_i presents a value greater than λ_{global} . This configuration implies that this critical interval does not present a stationary behavior like the remaining samples in the TS. A possible problem at this stage is to consider every sample inside the critical interval to be segmented. However, it is mitigated by the tuning step.

4.2.1.5 Tuning

A tuning process was designed to make the change detectors individually examine each sample with greater emphasis on critical intervals. In addition to the use of standard hyperparameter values for the change detectors, an attempt to improve their choice was also made. The standard values were varied to fit the best configuration for each scenario. Three different approaches were considered for tuning to optimize the choice of the best hyperparameters, The Bigger in Smaller out, Bigger in, and Smaller out approaches. These

approaches allow obtaining three different hyperparameter variation results according to the available critical intervals.

- **Bigger in Smaller out:** this strategy chooses the hyperparameter configuration that keeps the largest amount of data within the critical intervals and the least amount of data outside these intervals. This approach balances the selection of samples present in the critical interval with the least number of samples outside it.
- **Bigger in:** this strategy chooses the hyperparameter configuration that keeps the largest amount of data that is within the critical interval. This approach places the greatest emphasis on removing samples at critical intervals.
- **Smaller out:** this strategy chooses the hyperparameter configuration that keeps the least amount of data outside the critical interval. This approach minimizes the selection of samples outside the critical intervals.

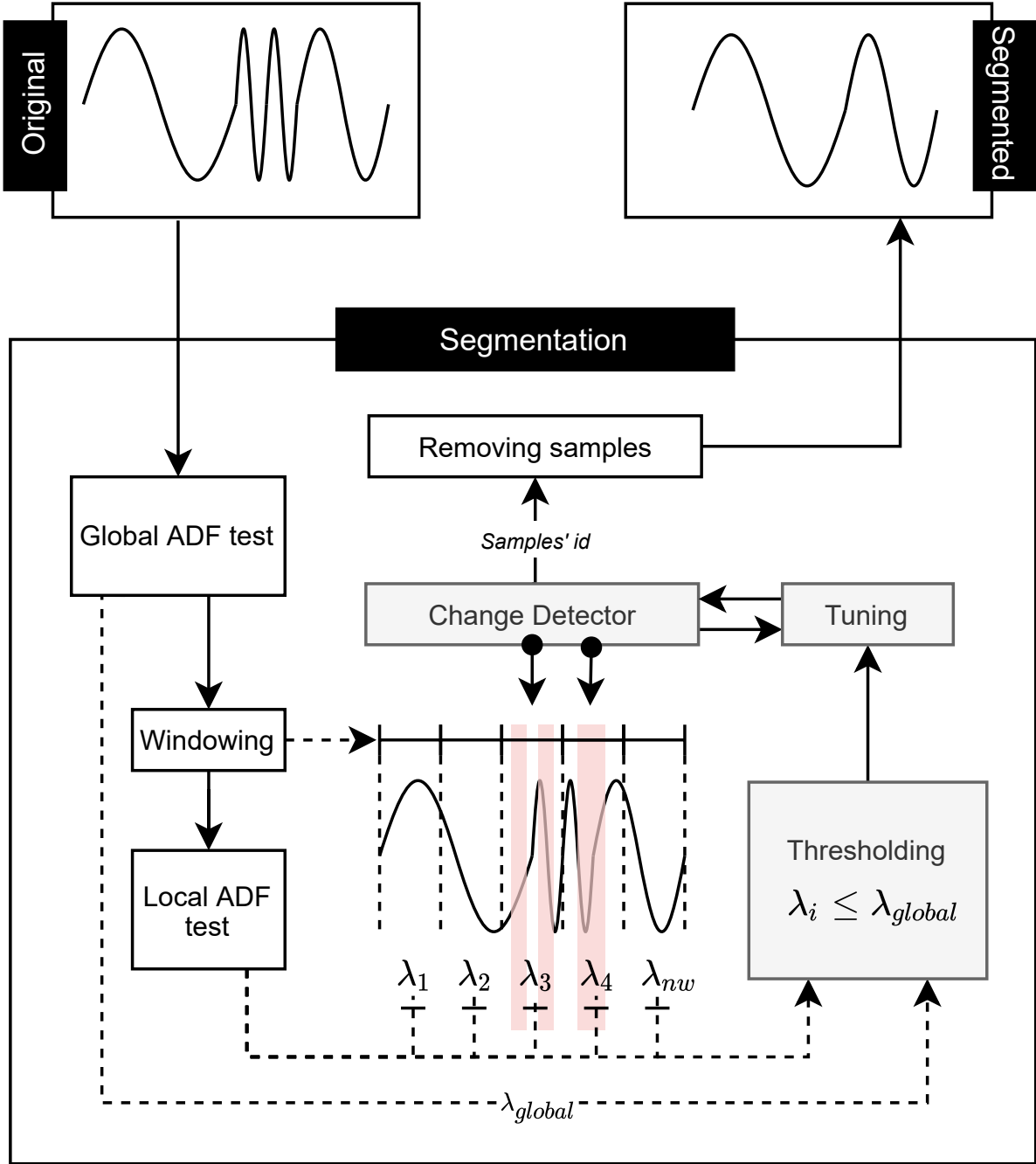
4.2.1.6 Change Detectors

Change detectors are usually applied to data streams; however, it is also possible to extend them to identify changes in TS [54]. As its name suggests, it seeks to identify changes in behavior in a data stream. This step is applied to identify the points of the TS that do not present stationary behavior. As the output, it generates a set of points to perform the removal.

4.2.1.7 Removing Samples

This step consists of removing the samples from the TS according to the points provided by the change detectors. This is the last step of the proposed framework. Neighbor points that were not removed from the TS are interpolated.

Figure 2 – Methodology framework of Change Detector Segmentation.



4.2.2 ADF-Based Segmentation

The ADF detector is also a proposal for this work. It performs the segmentation of samples in the TS based on their stationarity. Its process starts with the Grace Period, which consists of calibrating the ADF test with a predetermined interval of the TS, according to Fig. 3. It consists of separating a small number of TS samples to calibrate the ADF test, e.g., 10% of the initial TS samples. After this calibration, the Local ADF test is obtained. From this point on, the remaining samples will be evaluated to verify their stationarity continuously. For each new sample, a new ADF test value is acquired. If this

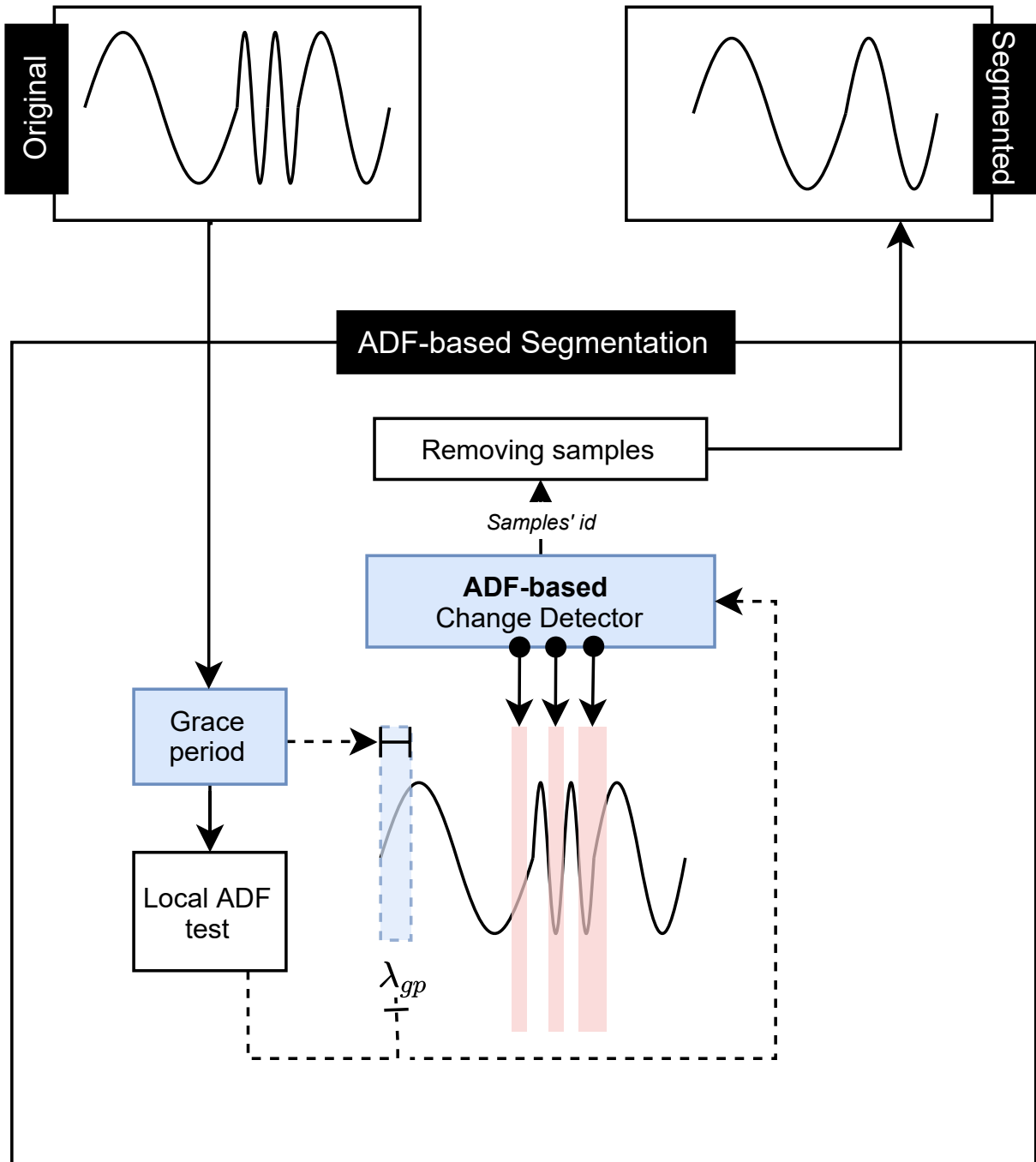
value is less than the current Local ADF test value, we understand that this sample must be kept in the TS by maintaining stationary, and the Local ADF test value is updated. However, if the sample has a greater value of ADF test than the Local ADF, it is assumed that this sample should be removed. This comparison between these two values depends on an α factor, where an acceptable error rate is applied. In this work, the values 0.05, 0.1, and 0.15 were used for α . This process seeks to ensure that the stationarity of the TS is not affected by samples that are out of normal behavior. Moreover, it can reduce the TS size. Algorithm 2 describes how the segmentation points are obtained.

Algorithm 2: ADF-based Segmentation code

Input: TS: The Time Series (TS).
Input: α : Alpha factor, to adjust the error rate.
Output: SP: A list with the Segmentation Points (SP).

- 1 **Pseudo-code**
- 2 Starts the Grace Period (GP) with 10% of TS length.
- 3 Calculates the ADF test λ_{gp} over the GP.
- 4 $\lambda_{global} = \lambda_{gp}$
- 5 $i = \text{length}(\text{GP})$
- 6 **while** $i \leq \text{length}(\text{TS})$ **do**
- 7 Calculates the ADF value of the new sample (λ_i) with the accumulated λ_{global}
- 8 **if** $\lambda_{global} - \lambda_i \geq \lambda_{global} * \alpha$ **then**
- 9 $\lambda_{global} = \lambda_i$
- 10 **end**
- 11 **else**
- 12 SP append a point (i) to be segmented
- 13 **end**
- 14 $i = i + 1$
- 15 **end**

Figure 3 – Methodology proposal of ADF-based Segmentation.



4.3 Experimental Study

The Experimental Study aims to compare the results obtained with works in the state of the art based on TS segmentation to answer the following Research Questions (RQ).

- RQ 1: Can TS segmentation based on stationarity analysis assist in the predictive process of new samples?

- RQ 2: Is stationarity-based segmentation capable of providing improvements in the aspect of prediction and reduction in TS sizes in different databases?
- RQ 3: Were the segmentation techniques proposed in this work compared with similar techniques?
- RQ 4: How correlated are stationarity and segmentation techniques?

4.3.1 Databases

The databases containing the TS used in this work are divided into three groups. The first group consists of a real database obtained by a Photovoltaic Plant (PV) installed at the State University of Londrina, Brazil. This database presents samples of power generation that started to operate in November 2019. The plant has 1020 solar panels. The samples are collected every 15 min. Thus, 96 samples are collected each day. To obtain better efficiency in capturing solar energy, the plant usually operates from 6 am to 7 pm. The sensors of the photovoltaic plant are capable of capturing 18 different features provided by solar energy. Among them, the generated power is used for analysis in this work.

The second group consists of the Minimum Daily Temperatures (MDT) database, which has the minimum daily temperatures over 10 years (1981–1990) in the city of Melbourne, Australia [55]. The third group consists of the database known as the Monthly Sunspot (MS) database [56]. This database describes a monthly count of the number of observed sunspots over 230 years (1749–1983). The experiments were separated at different intervals to verify different variations in the TS. For the first group, represented by the PV database, we have intervals starting from November, since it is the initial month of data collection and is subject to greater variation.

The experiments were separated into different training and testing intervals to verify unusual variations in the TS. For the first group, represented by the PV database, we have intervals starting from November, since it is the initial month of data collection and is subject to greater variation. For the second group, three intervals were selected. The first two intervals consist of a division of the third interval. Finally, the third group is comped by an interval of 150 years for training and 83 years for testing. The selection of these intervals was made without specific criteria. The objective is to verify the improvement in predictive capacity through the segmentation process proposed in this work.

Table 2 presents information from the databases used for experimentation with some additional information. Each database was identified by a letter from A to J for better identification in the results section 4.3.6. The ADF value column represents the stationarity of each training interval. These values demonstrate that the PV base presents well-behaved samples compared to other databases, since it has a smaller ADF

value. Furthermore, it is possible to verify the relationship of these values with the results obtained in the reduction of dataset size and predictive capacity. The exact intervals used for experimentation for each database and further information can be verified in Table 2.

Table 2 – Intervals used for experimentation from PV, MDT, and MS databases.

Identifier	Database	Train Interval	ADF Value	Test Interval
A	PV	November to December	−16.57	4 weeks January
B	PV	November to January	−19.24	March
C	PV	November to February	−21.81	March
D	PV	January to February	−16.36	4 weeks March
E	PV	February	−11.57	4 weeks March
F	PV	February	−11.57	days in March
G	MDT	1981 to 1984	−3.14	1985
H	MDT	1986 to 1989	−2.59	1990
I	MDT	1981 to 1989	−4.34	1990
J	MS	1749 to 1899	−7.04	1900 to 1983

After determining each experiment interval, the TS were submitted to segmentation techniques according to the ones proposed in this work, and their original versions were subjected to prediction techniques. This step consists of verifying whether the segmentation process improves the predictive performance of a TS, besides decreasing its size.

4.3.2 Change Detectors

The change detectors used in this work were ADWIN (ADW) [57] and Page-Hinkley (PH) [58], present in the scikit-multiflow package [16]. They were used as detectors in the change detector step of the framework to compare the efficiency and interference of a particular detector algorithm. The choices for these detectors were due to their adaptability to TS. ADW is an adaptive sliding window algorithm for detecting changes and has a delta parameter to perform its detection. The PH detector works by computing the observed values and their mean up to the current moment. It is composed of four parameters to perform its detection: the minimum number of instances before detecting change, the delta factor for the PH test, a threshold called lambda factor, and the forgetting factor, to weight the observed value and the mean.

The hyperparameters of both detectors were varied with values in percentage to values below and above the standard ones to obtain a configuration that best fits the proposed segmentation of the TS samples. For experimentation in this work, increments of 25% were used, starting at half of the standard value until double its standard value.

4.3.3 Metrics

The metric adopted for predictive gain assessment was the Root Mean Squared Error (RMSE), which is given by:

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^n \left(\frac{\hat{y}_i - y_i}{n} \right)^2} \quad (4.1)$$

where $\hat{y}_i$ represents the predicted sample and y_i represents the actual value of the sample at time i .

4.3.4 Prediction Techniques

Four predictive techniques are used in this work. The Naive, ARIMA, LSTM, and TCN methods. All of these predictors are used over the segmented and original TS.

The first predictive technique consists of a benchmark approach for TS, due to its simplicity and considerable acceptance of its results. Its operation considers that the next value to be obtained will be equal to the last observed value. Equation (4.2) shows how these values are obtained:

$$\hat{x}_{t+1} = x_t \quad (4.2)$$

where $\hat{x}_{t+1}$ represents the next value to be predicted and x_t represents the last observed value.

ARIMA is essentially exploratory and seeks to fit a model to adapt to the data structure [59]. With the aid of the autocorrelation and partial autocorrelation functions, it is possible to obtain the essence of the TS so that it can be modeled. Then, information such as trends, variations, cyclical components, and even patterns present in the TS can also be obtained [60]. This allows the description of its current pattern and prediction of future series values [61]. Autoregressive models have been used for a long time in TS analysis and are therefore considered popular and simple. However, some studies [62, 63] demonstrate that when dealing with a very large amount of data, models based on DL tend to have better results.

As a comparison, the LSTM and TCN techniques, which are based on DL, are evaluated in this work as a comparison to the classical ARIMA and Naive models. LSTM is a type of RNN. Unlike some traditional neural networks, LSTM can remember the most useful information. This is possible thanks to its architecture. The networks that comprise the LSTM are connected in the form of loops. This process allows information to persist on the network. It also has a gating mechanism for learning long-term dependencies without losing short-term capability [39]. This mechanism allows for what we call a neural network that has memory. This feature is essential when dealing with TS, as some moments tend to

be repeated over an interval of time. Another important feature is the stationary approach of this work, which allows the neural network model to adapt faster to the behavior of the TS. However, these approaches based on DL models are only interesting for problems that have a large amount of data, due to their complexity. For simpler scenarios, statistical models such as ARIMA tend to fill the need.

TCN is a special type of Convolutional Neural Network capable of handling a large amount of information [64]. This processing is done through causal convolution, which ensures the model cannot violate the order in which the data is processed. TCN uses a one-dimensional fully convolutional network architecture, where each hidden layer has the same length as the input layer, and zero padding of length [65]. As highlighted for LSTM, this approach tends to perform better for more complex problems where a statistical model is not able to determine long correspondences between the data.

Table 3 presents the hyperparameters used for experimentation. The Naive technique does not have an extra configuration as it is based only on the value of the previous sample. For the ARIMA modeling technique, auto-arima from pmdarima library was used. As for the LSTM and TCN, the values in Table 3 were used to obtain the best possible configuration for each of them.

4.3.5 OSTS Method

To answer RQ3, we compare our proposed techniques to the OSTS. The OSTS technique was proposed by Carmona-Poyato et al. [43] and consists of an approach to reduce dimensionality in a TS. In essence, it can locate the maximum and minimum points in a TS. This process is done through numerical approximation considering a point P_j and analysis of neighboring points. When the sample P_j is removed, its neighboring points are joined in a straight line. Even though its segmentation approach has no motivation for predictive improvement, it was used for comparison with the proposed techniques in this work. Its choice is also due to the extensive comparison of the OSTS approach with other segmenters. It also differs from classical approaches based on sliding windows and thresholds of statistical moments [9].

4.3.6 Experimental Results

In this subsection, the results obtained by the experiments will be presented. Tables 4 and 5 present the prediction results in terms of RMSE, demonstrating the predictive gain through TS segmentation. These results are established on the test interval defined in Table 2 using the four prediction techniques defined in the Section 4.3.4. The ADW and PH segmenters present the results of the Change Detector Segmentation Framework. These results are based on the best variation of the tuning process, defined in the Section

Table 3 – Hyperparameters experimented for tuning.

	Parameters	Experimented hyperparameters
LSTM	Number of stacked layers	1, 2, 3
	Units	32, 64, 128
	Dropout	0
TCN	Number of filters	32, 64
	Kernel	2, 3
	Dilations	[1, 4, 12, 48], [1, 2, 4, 8, 12, 24, 48], [1, 4, 16, 32], [1, 2, 4, 8, 16, 32], [1, 3, 6, 12, 24], [1, 2, 6, 12, 24], [1, 2, 4, 8, 16], [1, 4, 16], [1, 2, 4, 8], [1, 4, 8]
	Blocks	1, 2
	Dropout	0

4.2.1.5. The ADF-based segmentation is represented by ADF. Only the choice of the α factor that obtained the best result was considered.

4.3.7 Predictions Results

To answer RQ1, we have the predictive results applied to the datasets defined in this work with the proposed segmentation techniques. Table 4 presents the global error results of the new sample prediction from all databases, A to J, while Table 5 shows the relative error between them. In Table 4, we have highlighted in the last line the accumulated predictive error of all original databases, without the segmentation process. The OSTS segmenter was not able to improve the predictive capacity of the predictors in any case, except for the Naive predictor. It had the highest RMSE among the other segmentation techniques. For the PH segmenter, it obtained the second highest RMSE and did not show predictive improvement, except for the Naive predictor. As for the ADW and ADF segmenters, there were predictive improvements. In the case of ADW, only the TCN predictor was not able to present a predictive improvement; however, it obtained the smallest RMSE for the ARIMA and LSTM predictors. For the ADF segmenter, only ARIMA had no improvement; however, the smallest RMSE was obtained for the Naive and TCN predictors.

For Table 5, we have the relative RMSE between the predictive performance of the original database with the proposed segmentation techniques. Each database, A to J, is made up of four different prediction techniques. The segmentation techniques are presented in a decreasing ranking from the best result to the worst. In summary, the results that obtained a relative RMSE above one do not present a significant predictive performance. It is also possible to analyze why the original distribution did not obtain better predictive performance in any of the bases for the ARIMA and TCN predictors. Validating the results obtained by Table 4, it is verified once again that the most significant predictive results are between the LSTM and TCN predictors, promoted by the ADF and

ADW segmenters.

Table 4 – Global RMSE results from predictive techniques.

	Naive	ARIMA	LSTM	TCN
ADF	18923.65	13559.87	3193.70	3242.92
ADW	18923.66	13532.41	3177.46	3294.75
PH	18923.65	13727.95	3408.02	3844.67
OSTS	18923.65	14239.36	3482.60	3952.74
Original	20006.11	13537.78	3229.33	3276.01

Table 5 – Relative RMSE of each segmentation technique referring to the original base across four different predictive techniques using the ADF-based method, ADW, and PH (i.e., instances of our segmentation framework), and OSTS segmentation method. Lower errors are in bold and worst average cases are underlined.

Database Identifier	Segmentation Techniques	Predictive Techniques				Average
		Naive	ARIMA	LSTM	TCN	
A	ADF	0.99	0.99	0.97	0.99	0.98
	ADW	0.99	0.99	1.00	0.99	0.99
	PH	0.99	0.99	1.03	0.99	1.00
	OSTS	1.00	1.06	1.01	0.99	<u>1.01</u>
B	ADF	0.99	1.00	0.97	0.99	0.98
	ADW	1.00	0.99	0.99	0.98	0.99
	PH	0.99	0.99	0.98	0.99	0.98
	OSTS	1.08	1.12	1.00	1.02	<u>1.05</u>
C	ADF	1.00	0.98	0.94	0.83	0.93
	ADW	1.00	1.02	0.96	0.85	0.95
	PH	1.00	1.02	0.96	0.81	0.94
	OSTS	0.98	1.03	0.99	1.10	<u>1.02</u>
D	ADF	1.00	0.95	1.02	0.99	0.99
	ADW	1.00	0.98	1.03	0.97	0.99
	PH	0.99	0.98	1.01	0.90	0.97
	OSTS	1.00	1.00	1.34	1.30	<u>1.16</u>
E	ADF	0.99	0.99	1.00	0.98	0.99
	ADW	1.00	0.99	1.01	0.99	0.99
	PH	0.99	0.99	1.02	0.96	0.99
	OSTS	1.00	1.01	1.26	1.34	<u>1.15</u>
F	ADF	1.00	1.00	0.87	0.73	0.90
	ADW	0.99	0.99	0.67	0.86	0.88
	PH	0.99	1.00	0.74	1.20	0.98
	OSTS	0.99	1.03	0.93	1.45	<u>1.10</u>
G	ADF	1.00	1.00	0.99	0.96	0.98
	ADW	1.00	0.99	0.99	0.92	0.97
	PH	1.00	1.00	0.99	0.95	0.98
	OSTS	1.00	1.22	1.05	1.17	<u>1.11</u>
H	ADF	1.00	0.87	0.95	0.99	0.95
	ADW	1.00	0.99	0.95	0.99	0.98
	PH	1.00	0.93	0.96	0.96	0.96
	OSTS	1.00	1.24	1.00	1.12	<u>1.11</u>
I	ADF	1.00	1.00	0.99	0.98	0.99
	ADW	1.00	0.99	1.00	1.00	0.99
	PH	1.00	0.99	0.99	0.99	0.99
	OSTS	1.00	0.99	1.00	1.04	<u>1.00</u>
J	ADF	0.99	0.98	1.00	0.98	0.98
	ADW	0.99	0.98	1.00	0.97	0.98
	PH	0.99	1.00	1.00	1.00	0.99
	OSTS	0.99	0.99	1.22	1.13	<u>1.08</u>

In general, it can be said that the proposed segmentation techniques performed well in all experienced databases. Only in 6 cases out of 40 did the segmentation techniques not provide a better predictive capacity. Of these six, three are located in the Naive predictor, which has the worst predictive power among the other predictors. Moreover, even though

it did not obtain a better predictive performance, its result was the same as the original distribution, but with the advantage of decreasing the TS size.

4.3.8 Time Series Size Reduction

In this subsection, the effect of the segmentation process in reducing the TS size is discussed. RQ2 is answered with the analysis presented in Tables 4 and 5, in the previous subsection, and the results in Figure 4, which presents a comparison of the size reduction of the segmentation techniques for each database. These reductions are based on the LSTM predictor since it achieved the best predictive performance.

We can affirm through the analysis of Figure 4 that the size reduction of the TS is not directly related to the predictive gain. The OSTS segmentation technique was the one that obtained the biggest size reduction, but the worst result of the predictive gain, according to Tables 4 and 5. We can also affirm that the size reduction is not so considerable when comparing the other segmentation techniques, but it is still able to improve the performance gain. The ADF, ADW, and PH detectors achieved similar reductions in the PV databases, while the PH detector showed greater variability in the other databases. In addition, considering that these results were obtained using the LSTM predictor, it is possible to conclude that the ADF and ADW detectors can improve the predictive gain of new samples, while reducing the TS size.

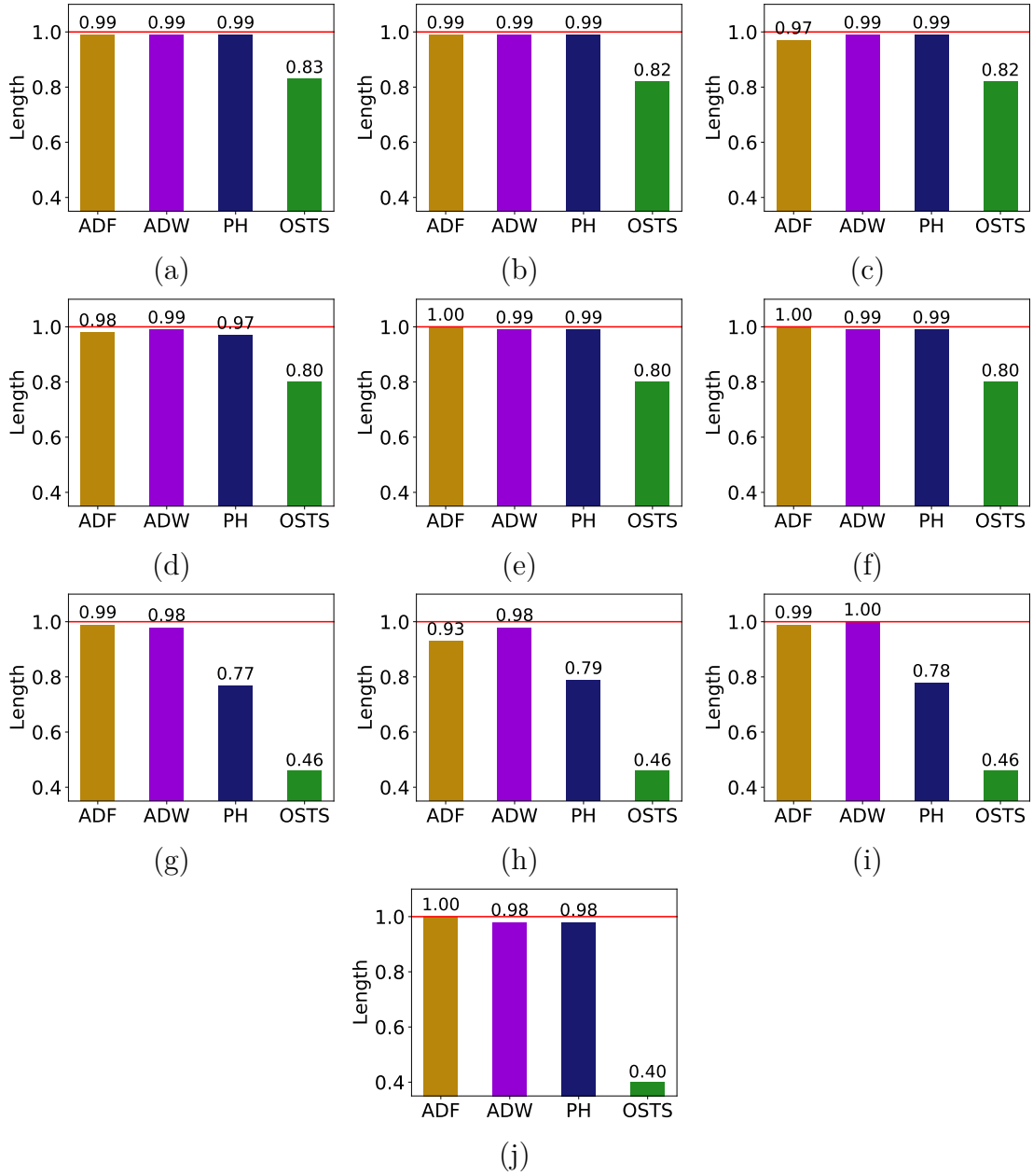
An interesting link that can be made is that the databases that had the biggest reductions were the ones with the highest ADF values in Table 2. The same is not verified in the predictive improvement (Table 5), since the cases that obtained better results are concentrated in the more stationary databases, e.g., database C.

4.3.9 Sample Segmentation

Figs. 5 and 6 present sample segmentation intervals in two distinct databases, A and C, respectively. In Fig. 5, it is possible to observe in the sample interval 500 to 515, none of the techniques presented a region to be segmented, it being considered an essential region to describe a stationary behavior. Near sample 545 a peak is seen, which was highlighted by most detectors as segmentable. The OSTS and ADW detectors acted at the moment and just before the appearance of this point, while the ADF acted at the moment of descent from this peak. The PH detector did not act to identify this region. However, the excess of points identified by the OSTS detector and the absence of others in the evaluated interval demonstrates great disparity, as the other detectors pointed out this highlighted interval with a high level of stationarity, requiring few segmentations.

Fig. 6 exhibits a behavior similar to the database analyzed above, but with emphasis on the samples verified from point 562, where there were many segmentation points by ADF. This demonstrates that the ADF detector operates differently from ADW and PH.

Figure 4 – TS size reduction according to segmentation methods ADF, ADW, PH, and OSTS. **(a)** Database A; **(b)** database B; **(c)** database C; **(d)** database D; **(e)** database E; **(f)** database F; **(g)** database G; **(h)** database H; **(i)** database I; **(j)** database J.



This interval highlighted by ADF demonstrates an important characteristic when these samples were detected and proved to be more sensitive for stationarity analysis.

Figure 5 – Segmented samples of dataset A (chunk from 500 to 560 samples). ADF removed 2 samples, PH and ADW removed a single sample, and OSTS removed 17. The stationarity-based segmenters considered the range from 515 to 543 as stable and without the need for segmentation, while the OSTS segmenter performed quite a lot of segmentation at these points.

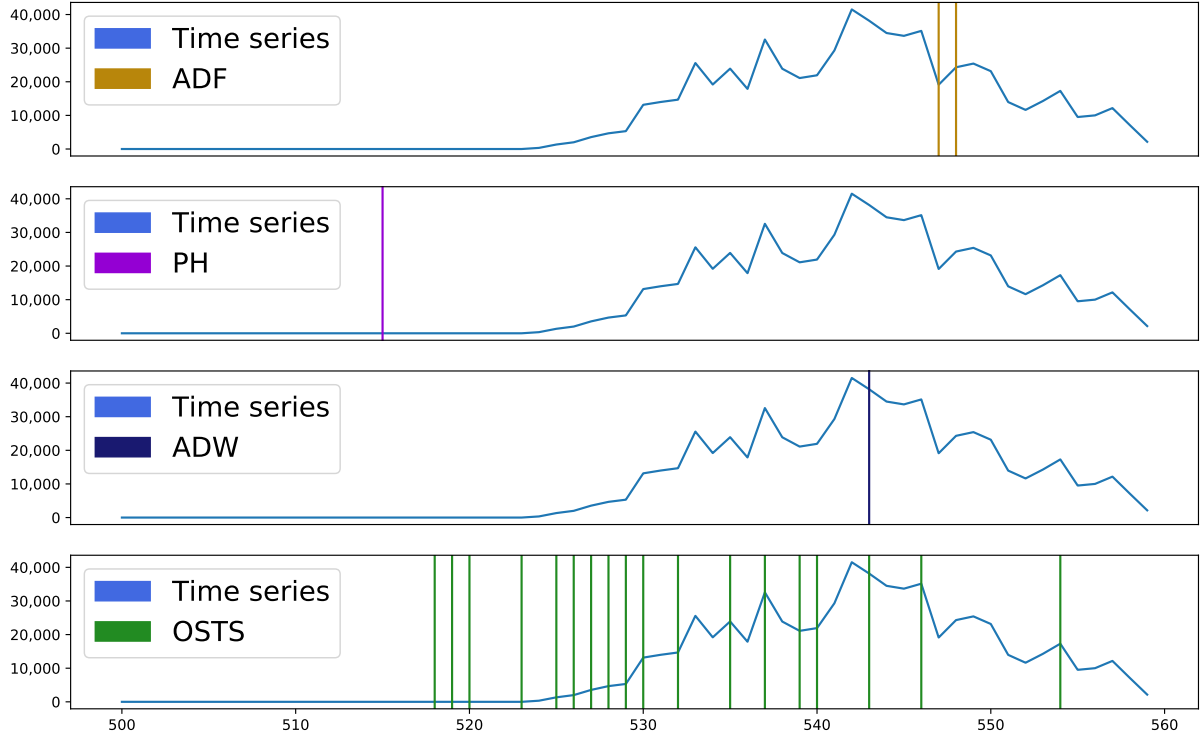
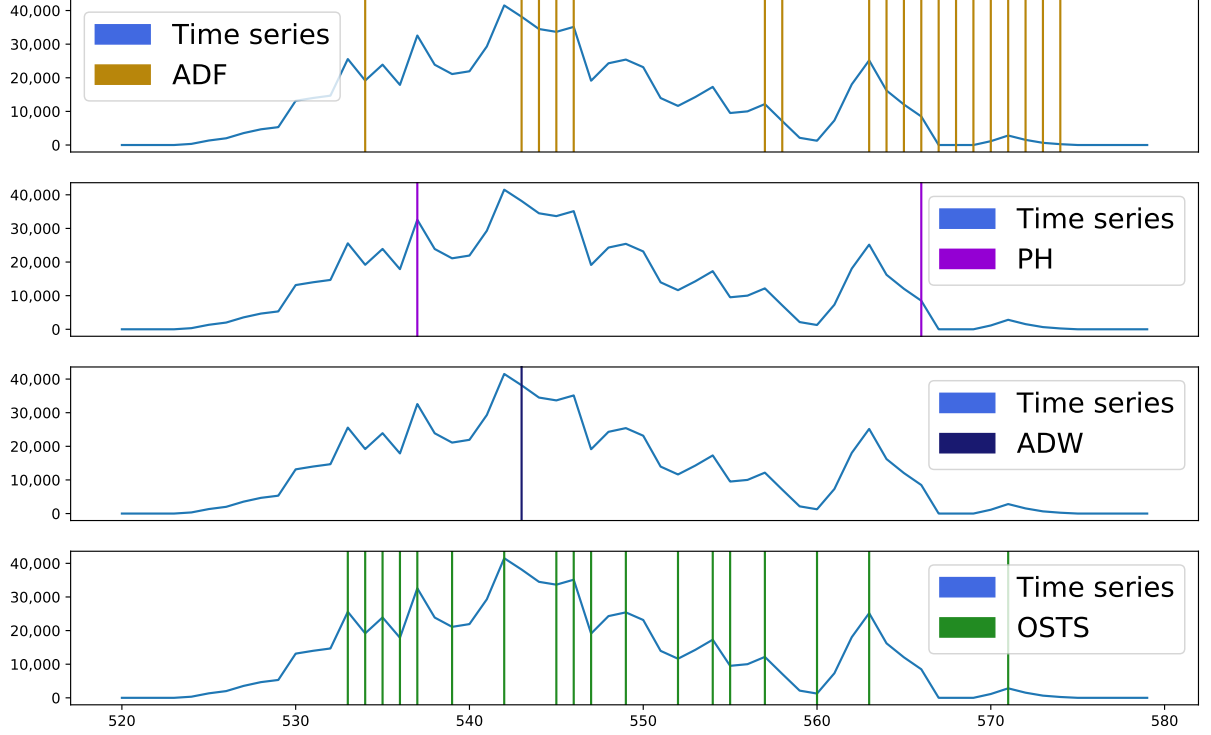


Figure 6 – Segmented samples of dataset C (chunk from 520 to 580 samples). ADF removed 19 samples, PH removed 2 samples, ADW removed a single sample and OSTS removed 18. Unlike the case of dataset A, the ADF segmenter performed many segmentations like the OSTS, but in different regions, while the other segmenters considered the region as stable.



4.3.10 Statistical Analysis

Friedman’s statistical test and the post-hoc test of Nemenyi [66] were used to verify the statistically significant difference between the performance of the segmentation techniques proposed in this work. This test was performed for each predictor described in this work and can be evaluated by Figs. 7–10. The Critical Difference (CD) allows checking when there were statistical differences between the segmenters, each diagram and algorithms’ average ranks are placed in the horizontal axis, with the best ranked to the right. For ADW segmenter, it is possible to observe that there were statistical differences compared to the original TS in all scenarios. A black line connects algorithms for which there is no significant performance difference.

Fig. 7 demonstrates no statistical difference between the ADF and OSTS techniques compared to the original TS. There was also no statistical difference between OSTS and ADW. For the Naive predictor, only the ADW and PH techniques showed statistical differences from the normal distribution.

For the ARIMA predictor in Fig. 8, the OSTS technique showed the worst performance. The PH and ADF techniques did not present statistical differences with the original distribution, only the ADW reached differences in this case. Finally, for the LSTM

and TCN predictors shown in Figs. 9 and 10, respectively, there were no statistical differences for the OSTS and PH techniques compared to the original distribution. In both cases, the ADF and ADW techniques showed better performance, with statistical similarity.

Figure 7 – Comparison of the RMSE values obtained by segmentation techniques for Naive predictor according to the Nemenyi test. Groups that are not significantly different ($\alpha = 0.05$ and $CD = 1.04$) are connected.

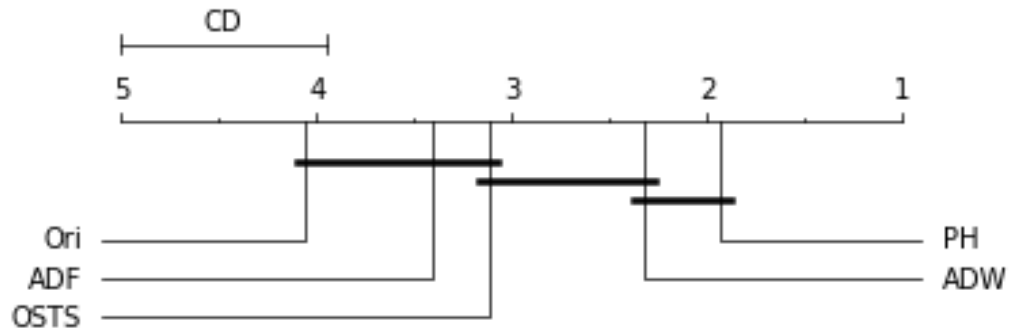


Figure 8 – Comparison of the RMSE values obtained by segmentation techniques for ARIMA predictor according to the Nemenyi test. Groups that are not significantly different ($\alpha = 0.05$ and $CD = 1.03$) are connected.

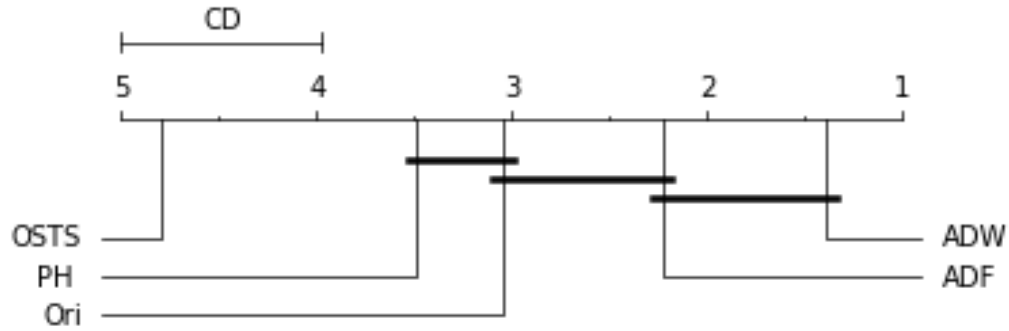


Figure 9 – Comparison of the RMSE values obtained by segmentation techniques for LSTM predictor according to the Nemenyi test. Groups that are not significantly different ($\alpha = 0.05$ and $CD = 1.04$) are connected.

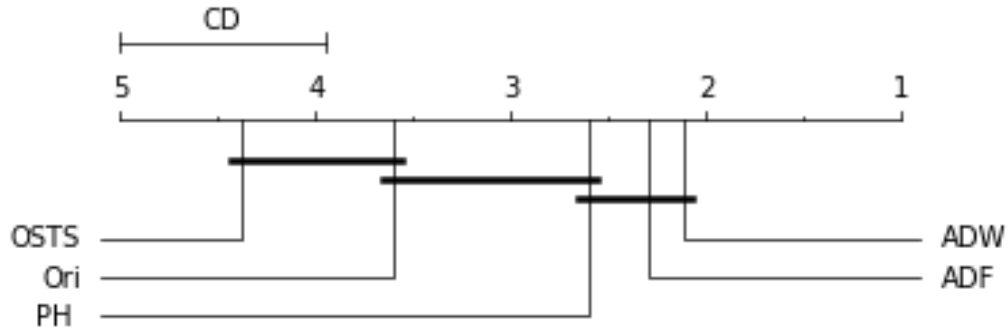
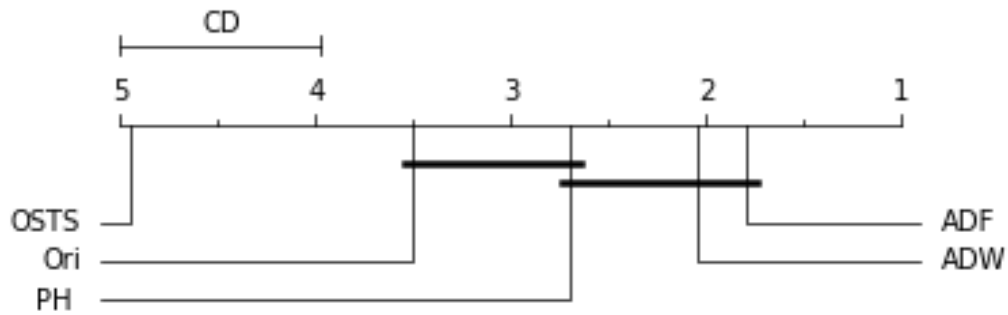


Figure 10 – Comparison of the RMSE values obtained by segmentation techniques for TCN predictor according to the Nemenyi test. Groups that are not significantly different ($\alpha = 0.05$ and $CD = 1.03$) are connected.



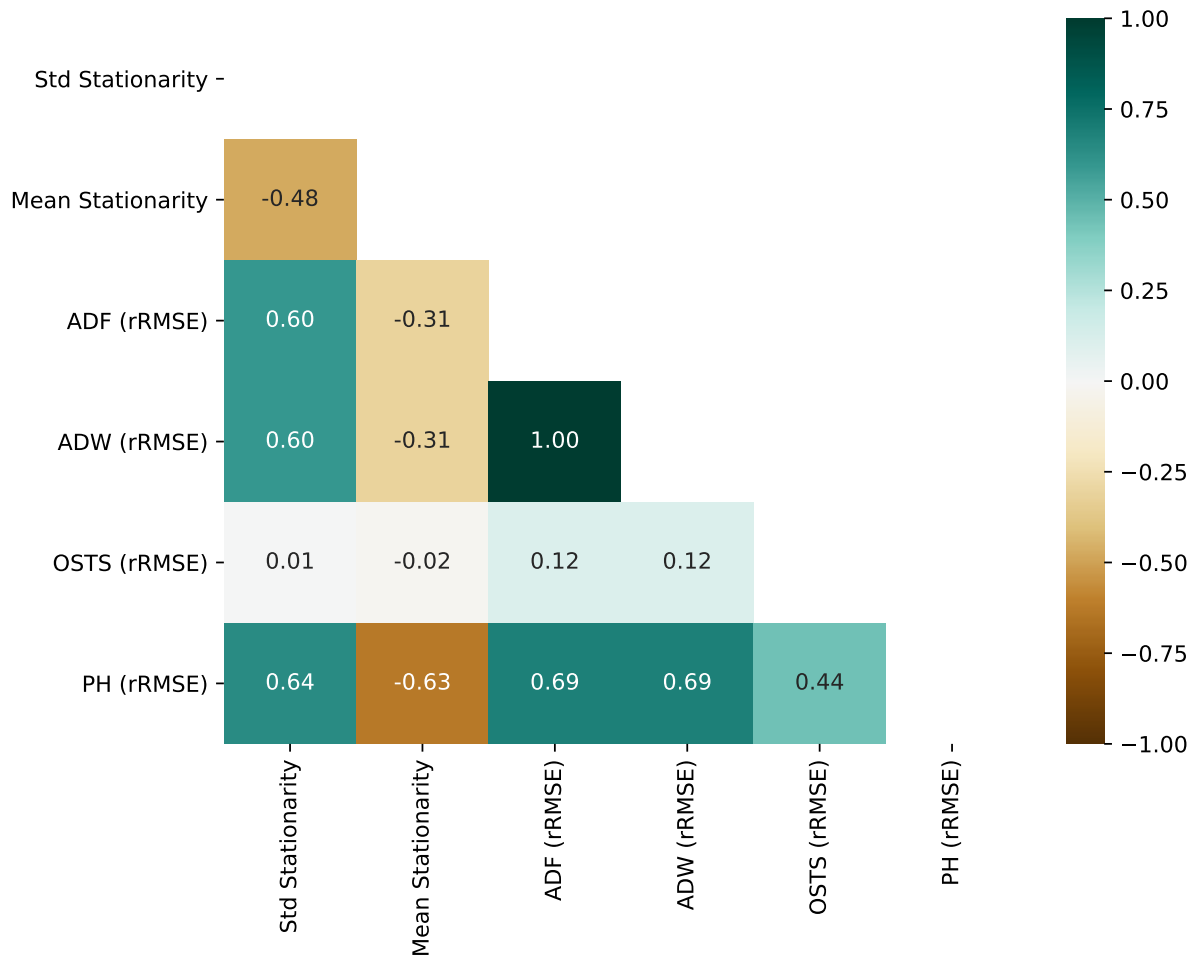
It is worth mentioning that segmentation techniques based on the analysis of stationarity of a TS are a viable technique. The advantages of this approach are the simplicity in how each sample is evaluated and the independence of information and feature extraction to perform the segmentation. In addition, we consider this as a novel approach since the segmentation of TS to improve the predictive capacity of new samples is not a task found in the literature.

Another consideration is that the segmentation process must be able to maintain the representativeness of the TS and our approach guarantees this. A large reduction in the TS size does not guarantee better performance, as seen in the results of the OSTS detector.

4.3.11 Stationarity Impact on Segmentation

One of the pillars of our hypothesis for creating a framework proposal and segmentation method is stationarity analysis. Thus, RQ4 is answered in this subsection. Extracting two stationarity descriptors based on the ADF test, in particular, the standard deviation of stationarity (Std Stationarity) and the mean value of stationarity (Mean Stationarity), we computed their correlation using the Pearson Correlation test [67]. This test is related to the errors (RMSE) obtained from the segmentation techniques proposed in this work and the OSTS method. Fig. 11 represents, as a heatmap, the correlation between stationarity and RMSE of all segmenters.

Figure 11 – Correlation between the stationarity (based on ADF Test) and performance (rRMSE) obtained by segmentation methods.



As observed in Fig. 11, OSTS RMSE does not correlate with the stationarity of the TS, reaching 0.01 and -0.02 of correlation, standard deviation, and mean. This is an expected phenomenon, since the strategy of OSTS is driven by the neighborhood of maximum and minimum local values. On the other hand, ADF, ADW, and PH presented an important correlation with the standard deviation of stationarity. More precisely, they

obtained a positive correlation, which reveals that the increase in stationarity variation in the TS leads to a less effective segmentation. However, high mean stationarity leads to boosted results from segmentation, with PH obtaining the highest correlation.

Considering the perspective of dimension reduction, we evaluated the correlation between reduction rate and stationarity, as previously, for performance. OSTS reduction capacity is inversely correlated with the stationarity mean, -0.98 . Regarding the change-based approaches, PH reduction capacity was the best correlated with an inverse correlation of -0.86 , as shown in Fig. 12. For ADF and ADW, we have the lowest correlation between reduction rate and stationarity, -0.29 and -0.38 , respectively.

Figure 12 – Correlation between the stationarity (based on ADF Test) and dimension reduction delivered by segmentation methods.



Overall, these results indicate that the stationarity of TS is correlated with the dimension reduction capacity provided by the methods. It is worth mentioning that ADW and PH followed the same framework and tuning strategy, but the way stationarity correlates to their reduction capacity and predictive performance diverges. This fact emphasizes the need for caution when selecting the change detector according to each TS pattern.

It is also worth noting that our stationary strategy to perform segmentation maintains the original TS behavior and removes only disposable samples, which makes it a less invasive method. Finally, the standard deviation of stationarity exposed how tricky the segmentation might be when reducing the predictive error of high-variated stationarity patterns.

4.4 Conclusions

The stationary approach for segmenting samples in TS has a positive impact on the issue of prediction of new samples and size reduction. According to the results, it is possible to affirm that the segmentation techniques had a better performance for the prediction algorithms based on DL. The ADW and ADF segmentation had the best performance in the DL approaches, with ADW for LSTM and ADF for TCN. However, an advantage over the functioning of the ADF over the ADW is the independence of *a priori* knowledge about the TS. The ADF segmentation analysis demands only a small number of samples to perform its grace period and start the segmentation process and has only one hyperparameter.

The largest cases of TS size reduction came from the OSTTS technique, achieving 0.40 reduction compared to the original distribution, but this result does not follow the predictive performance of the new samples. A balance between the reduction and prediction approach is best seen by the ADF and ADW segmenters.

Stationary analysis proved to be a great ally in the study of TS behavior and more studies can be directed in this respect beyond sample segmentation. For future work, we seek to improve segmentation techniques aimed at the data stream, where the segmentation would be given in real-time, as the data is obtained. Furthermore, improving the extraction of *a priori* information from the TS, makes these methods more autonomous.

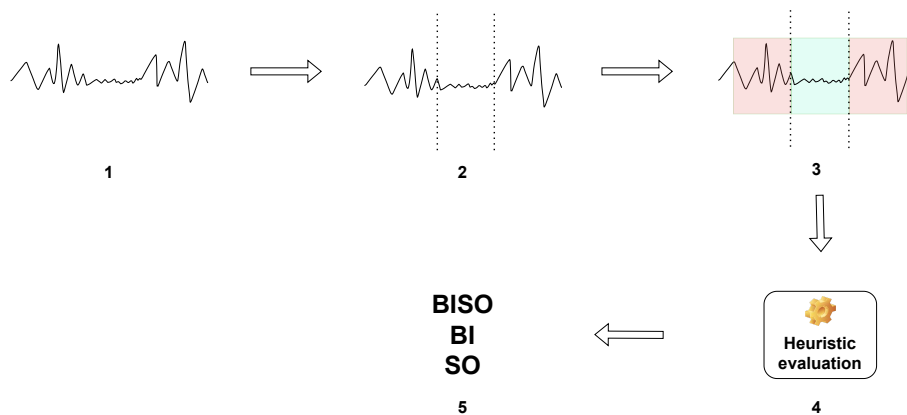
5 UNSUPERVISED TUNING FOR DRIFT DETECTORS

Our approach is oriented towards achieving unsupervised tuning for DDs. To fulfill this objective, we propose a novel combination of stationarity criteria derived from the Augmented Dickey-Fuller (ADF) test. At the core of our methodology lies an unsupervised framework called Change Detector Segmentation (CDS), which adjusts the inherent hyperparameters of DD. Importantly, the CDS framework was initially introduced to eliminate non-stationary samples in TS data. In this study, we adapt this framework to apply to any signal type, shifting its focus from segmenting samples to accurately identifying them as drifts.

Unsupervised tuning methods eliminate the need for annotated data, significantly reducing the time and resources required for data labeling. This crucial advantage speeds up the optimization process and extends the applicability of tuning to contexts with limited availability or high costs linked to acquiring labeled data, as seen in online scenarios. This addresses a key concern in ML, where access to abundant labeled data is often limited, offering a promising opportunity to improve model performance under resource constraints.

We present a comprehensive framework designed for unsupervised tuning in signal. Our proposed approach encompasses five key stages based on the CDS framework, adeptly handling the complex challenge of precisely identifying drift occurrences while optimizing the hyperparameters governing DDs. The following steps in Fig. 13 outline the methodology used in this study, enabling replication of the process:

Figure 13 – Unsupervised Tuning Methodology.



5.1 Step 1

Our investigation starts with the acquisition of a signal, predominantly presented as a TS structure, serving as the basis for evaluating and refining our approach. Within the signal's temporal evolution, concept drift becomes points of focus, triggering subsequent steps in our process.

5.2 Step 2

Step 2 encompasses segmenting the signal acquired in Step 1, which can be executed in two manners:

1. For this study, we chose to divide the signal empirically into 10 segments. This approach is advisable when the user possesses limited prior knowledge of the signal.
2. Alternatively, the signal can be segmented based on its seasonality and requires knowledge of the signal's behavior by the user. Seasonality refers to recurring cycles within the signal that display consistent behaviors over time.

Segmenting the signal serves a purpose: analyzing the entire signal may complicate statistical information analysis, reducing sensitivity to drifts. Utilizing smaller segments enhances sensitivity, facilitating drift detection.

5.3 Step 3

Step 3 defines the critical intervals, highlighted in red in Fig. 13, alongside intervals indicating normal behavior, marked in green. This distinction is established via the stationarity test performed by the ADF test. To achieve this, the following procedures are implemented:

1. Calculate the stationarity statistical value for the entire signal.
2. Calculate the stationarity statistical value for each segment.
3. Identify critical intervals as segments with a stationarity statistical value higher than that obtained in Step 1.

We propose that regions with lower stationarity statistical value are less likely to exhibit drifts compared to regions with higher values. Thus, we define critical intervals of the signal by their higher statistical values.

5.4 Step 4

The preceding steps are crucial for implementing unsupervised tuning. Moving on to Step 4, we initiate our Heuristic evaluation. The intervals derived from the segmented signal guide us in configuring any DD appropriately. At this stage, users can set a range of values for each DD hyperparameter and employ our heuristics to identify the best choice. Alternatively, users can use values empirically tested in this study. Specifically, we explore four decay and growth percentages relative to each detector's default values. The decay percentages are 95, 90, 75, 50, and 10, while the growth percentages are 105, 110, 125, 150, and 1000. Our aim is to categorize values showing slight and significant deviations from default values to improve signal generalization.

For example, if a specific hyperparameter has a default value of 10, the values used in this study to explore a wider range of configurations will be as follows:

- Decay values: 9.5, 9, 7.5, 5, and 1.
- Growth values: 10.5, 11, 12.5, 15, and 100.

These values represent adjustments to the default value for the hyperparameter under consideration.

5.5 Step 5

Step 5 employs our heuristics to select the optimal hyperparameters, offering a specific combination that satisfies the following definitions:

- **Bigger in Smaller out (BISO):** This heuristic selects the hyperparameter configuration that detects the highest number of drifts within the critical intervals while minimizing detections outside of these intervals. BISO aims for balanced sensitivity, focusing on placing the majority of drifts within the critical intervals while reducing detections in non-critical intervals. Consequently, BISO is expected to effectively identify both abrupt and gradual drifts.
- **Bigger in (BI):** This heuristic aims to select the hyperparameter configuration that detects the highest number of drifts within the critical intervals. BI is more sensitive, potentially generating more false positives compared to BISO. BI primarily excels in identifying abrupt drifts.
- **Smaller out (SO):** In contrast to the previous heuristics, SO prioritizes the hyperparameter configuration that detects the lowest number of drifts outside the critical intervals. However, its selective nature increases the possibility of false negatives due to filtering out minor variations.

Our proposal orchestrates a methodical process that combines unsupervised tuning and drift identification within a signal. Through this innovative framework, we aim to tackle the challenges inherent in drift detection, thereby advancing unsupervised learning paradigms.

Additionally, our approach outlines a systematic steps integrating unsupervised tuning and drift identification within a signal, shedding light on the complexities associated with drifts. It offers a solution when users lack prior knowledge about the analyzed signal, allowing them to explore hyperparameter configurations surpassing the default settings.

It is crucial to emphasize that our unsupervised tuning extends its applicability to a wide range of DDs, regardless of their specific characteristics.

5.6 Materials and Methods

To validate our proposed approach in this study, it is crucial to determine if our unsupervised tuning proves to be a feasible alternative compared to the default configuration of DDs. This validation hinges on demonstrating that our proposed approach consistently outperforms the default configuration. To accomplish this, we will outline the details of our experimental setup.

To empirically study drift phenomena, it was essential to create a reliable and carefully annotated synthetic database. In this regard, the Configurable Signal Generator played a crucial role, facilitating the generation of artificial signals tailored to populate our designated database. This deliberate use of synthetic data injection was intended to cultivate a dataset characterized by the necessary qualities of precision and reliability, thus providing an ideal basis for the systematic execution of our experimental investigations.

To conduct the experiment, acquiring signals with labeled concept drift samples—representing normal and concept drift-specific instances—was indispensable. However, the absence of tools providing such information was evident. Consequently, this work developed the Configurable Signal Generator, ensuring the mentioned specifications.

5.6.1 Configurable Signal Generator (CSG)

Upon a comprehensive review of the current state of available datasets, it becomes apparent that the majority lack sufficient instances of concept drift or fail to provide proper labels indicating the occurrence of such shifts, as discussed in [68]. This deficiency presents a substantial obstacle for researchers and practitioners aiming to develop and evaluate models that can effectively handle concept drifts in real-world scenarios.

Labeled datasets play a pivotal role in training and validating ML models. Howe-

ver, in the context of concept drift, the scarcity of labeled data reflecting dynamic changes in the underlying data distribution hampers the progress in developing adaptive and resilient models. The absence of such datasets limits our ability to assess the generalization and adaptability of models to changing environments.

To bridge this gap, the CSG, introduced in this study, serves as an innovative tool designed to simplify the creation of synthetic signals with adjustable hyperparameters and labeled samples. Its purpose is to enable the generation of signals with diverse statistical distributions. The main goal is to empower users by providing precise control over the signal’s amplitude and the introduction of perturbations based on their preferences.

This signal generator operates based on five key hyperparameters, each for customizing the artificial signal to meet specific requirements:

1. Number of Samples: this hyperparameter enables users to specify the granularity of the generated signal by defining the number of discrete data points or samples. This configuration is important to achieve the desired temporal or spatial resolution in the resulting signal.
2. Initial and Final Amplitude Values: this feature provides users with the flexibility to set the starting and ending amplitude levels of the signal. It is instrumental in modulating the signal’s intensity over a defined time span or spatial region. Additionally, it facilitates the creation of signals with varying amplitudes, such as linear ramps or amplitude-modulated signals.
3. Minimum and Maximum Perturbation: perturbations introduce deviations from the ideal or nominal signal, incorporating stochastic elements into the generated data. By specifying the minimum and maximum perturbation values, users have precise control over the level of randomness or noise added to the signal. This capability allows for the creation of signals that simulate real-world concept drift scenarios with inherent variability and uncertainty.

Distinguished from prevailing signal generators [69], the distinctiveness of the CSG arises from its ability to introduce stochastic perturbation at various junctures within the signal’s trajectory. Notably, the intervals demarcating these instances of perturbation are recorded and provided as an output, rendering the CSG an invaluable asset within the signal generation paradigm.

The CSG sets itself apart from existing signal generators by its unique capability to introduce labeled random disturbances at different points along the signal’s path. What makes it particularly noteworthy is that it records and outputs the specific intervals where these disturbances occur. Other signal generators commonly lack the explicit demarcation

of regions where perturbations are introduced. This drawback becomes especially significant when trying to pinpoint the exact areas in the signal's path that have encountered perturbations, hindering a detailed understanding of the perturbation dynamics. This feature makes the CSG an invaluable asset in the field of signal generation.

A pseudocode outlining the implementation of the CSG is available in Algorithm 3. The function returns the generated signal and a list of indices corresponding to the positions where perturbations occurred, simulating concept drift in this generator.

Algorithm 3: Generate Random Stream

Input: *samples*, represents the desired number of samples to generate for the signal.
initial_amplitude, indicates the initial value of the signal's amplitude.
final_amplitude, indicates the final value of the signal's amplitude.
min_perturbation, represents the minimum perturbation value to be applied.
max_perturbation, indicates the maximum perturbation value to be applied.
Output: *signal*, *perturbation_indices*, generates final signal and the perturbation indices.

```

1 Pseudo-code
2 signal = initial_amplitude + random(samples) * final_amplitude
3 intervals = obtain_intervals(signal)
4 perturbation_indices = []
5 while i ≤ length(intervals) do
6   perturbation = min_perturbation + random(samples) * max_perturbation
7   signal[i] = signal[i] * perturbation
8   perturbation_indices.add(i) i = i + 1
9 end
10 return signal, perturbation_indices

```

In this study, we consider the points where perturbations were introduced as representations of concept drift. These marked points will act as indicators, enabling us to evaluate whether our unsupervised tuning approach can surpass the default DD configuration.

Figs. 14 and 15 depict outputs generated by the CSG. In these figures, it is evident that each generated signal exhibits a random behavior with distinct characteristics. This simulation aims to emulate the representation of real-world data, capturing the inherent variability and unpredictability often observed in practical datasets.

5.6.2 Experimental setup

We utilized various configurations during signal generation to mimic natural signals and demonstrate the effectiveness of the proposed unsupervised tuning approach for DDs.

Figure 14 – Random signal generated from CSG. Initial amplitude = 10, final amplitude = 60.

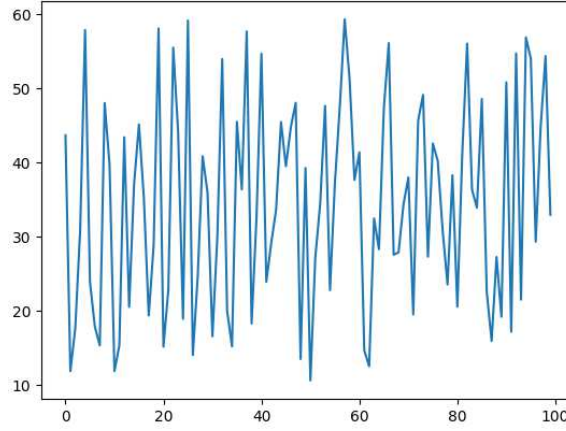


Figure 15 – Random signal generated from CSG. Initial amplitude = 10, final amplitude = 60.

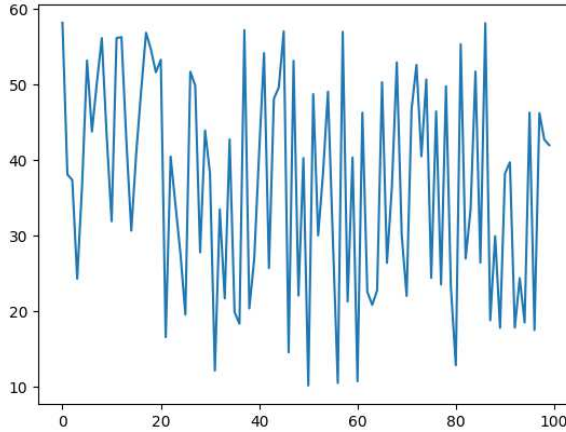


Table 6 showcases the diverse experimental setups involving the generation of random perturbations for each signal. The Signal column denotes the number of signals used for the experiments. Each signal is generated with random distributions. The Size column indicates the sample sizes for the signals, set at 100, 1000, and 5000 samples, respectively.

Additionally, the Amplitude column details the amplitude variation for each configuration. This variation is calculated by determining the difference between the final and initial amplitudes set during signal creation. The Perturbation column quantifies the degree to which the signal's samples were perturbed, with higher values indicating increased variability. Here, perturbation values of 1.5 and 2 are considered as low variations, while values of 4 and 10 are categorized as high variations. A perturbation value of 1 means signals without any perturbations.

Table 6 – Signal generation configurations.

	Signal	Size	Amplitude	Perturbation
A	10	[100, 1000, 5000]	10	1
B	10	[100, 1000, 5000]	10	1.5
C	10	[100, 1000, 5000]	10	2
D	10	[100, 1000, 5000]	10	4
E	10	[100, 1000, 5000]	10	10
F	10	[100, 1000, 5000]	30	1
G	10	[100, 1000, 5000]	30	1.5
H	10	[100, 1000, 5000]	30	2
I	10	[100, 1000, 5000]	30	4
J	10	[100, 1000, 5000]	30	10
K	10	[100, 1000, 5000]	90	1
L	10	[100, 1000, 5000]	90	1.5
M	10	[100, 1000, 5000]	90	2
N	10	[100, 1000, 5000]	90	4
O	10	[100, 1000, 5000]	90	10

To assess the efficacy of the unsupervised tuning process, we evaluated various DDs: PH, ADWIN, DDM, HDDM_A, HDDM_W, and KSWIN. Our evaluation aimed to validate the performance of the default configuration by examining the accuracy of each configuration in identifying drifts. Initially, we visualized the Correct Detection of each DD for every configuration using boxplots. Next, we analyzed how detection was influenced by the number of samples in the signal. Finally, we employed Friedman’s statistical test to determine any significant statistical differences between the techniques.

We used the default configuration as our starting point. Additionally, we conducted a comparative analysis against the RS tuning strategy, which acts as a fundamental benchmark. RS aims to identify the best hyperparameter configuration within a predefined range by employing a random selection process rather than exhaustively exploring all possible configurations [70]. It is important to note that RS assumes a supervised context and may not thoroughly explore the entire configuration space.

5.6.3 Drift Detectors

The selected DDs for experimentation in this study will be delineated herein, with particular emphasis on the hyperparameters associated with each of them, which will be evaluated during the unsupervised tuning process.

5.6.3.1 Page-Hinkley

The Page-Hinkley (PH) detector operates by maintaining simultaneous averages of the ongoing analysis moment, focusing solely on capturing detections without necessitating alert generation. It leverages the Page-Hinkley test [58] to discern deviations

within the data. The pivotal determiner in the drift detection process is the parameter λ , serving as a threshold value. The instantiation of a detection is prompted when the prevailing mean surpasses the stipulated λ value, thus activating the detection mechanism.

5.6.3.2 Page-Hinkley hyperparameters

There are several key hyperparameters to consider in PH detector. First, the hyperparameter m_i refers to the minimum number of instances that must be evaluated before the detector can accurately identify a change. This hyperparameter is critical for ensuring that the detector is reliable and produces accurate results. Another important hyperparameter is delta (δ), which represents the delta factor for the Page-Hinkley test. This hyperparameter plays a key role in determining the sensitivity of the detector to changes in the data. The change detection threshold (t) is another critical hyperparameter in the PH detector. This threshold is used to determine when a change has occurred in the data, based on the magnitude of the change relative to the threshold. Finally, the alpha (α) hyperparameter is the forgetting factor used to weight the observed value and the mean. This hyperparameter is important for ensuring that the detector can adapt to changes in the data over time, while still maintaining a stable baseline for comparison. By carefully adjusting these hyperparameters, it is possible to build a PH detector that is highly effective at detecting changes in a wide range of datasets.

5.6.3.3 ADWIN

ADWIN is a change detection algorithm proposed by Bifet and Gavalda [57]. Its objective is to monitor a data stream and maintain statistical information about it. The algorithm uses a sliding window approach, where small portions of the data stream are analyzed. The size of the window is predefined, and statistical information such as the mean and variance are calculated for each window. The ADWIN algorithm then compares the mean of two adjacent windows, and if the difference between the two means is greater than a predefined threshold value referred to as delta (δ), it is considered as a change point or drift in the data stream. ADWIN does not issue alerts for every change detected but rather keeps track of the changes and adjusts the window size accordingly to adapt to the new data distribution.

5.6.3.4 ADWIN hyperparameters

In the assessment of ADWIN, the focus was directed solely towards the δ hyperparameter, meaning the confidence level required for initiating a contraction of the sliding window size. The complementary hyperparameter associated with the sliding window dimensions, denoted as W , was not subject to explicit tuning. This restraint was exercised

owing to ADWIN’s intrinsic capacity for autonomous adaptation, ensuring the real-time adjustment of the window size in response to the continuous influx of data.

5.6.3.5 Drift Detection Method

To detect significant changes in data patterns, data monitoring systems frequently use error rate and standard deviation analysis [71]. When the error rate increases beyond a certain threshold, it is assumed that a change has been detected. This approach is known as the Drift Detection Method (DDM). The DDM is commonly used in ML applications to track model performance and detect changes in data distribution over time.

5.6.3.6 DDM hyperparameters

The DDM detector has a hyperparameter denoted as m_i , which specifies the minimum number of samples that need to be analyzed before a change can be detected. In addition to m_i , DDM also has two other hyperparameters, w_l and o_{cl} , which adjust the confidence levels for the warning and out-of-control signals, respectively. Specifically, w_l sets the threshold for raising a warning signal, while o_{cl} sets the threshold for raising an out-of-control signal.

5.6.3.7 HDDM_A and HDDM_W

Both HDDM_A and HDDM_W detectors [72, 73] are based on Hoeffding’s bounds inequality, which is a statistical theory used to determine the minimum number of observations needed to estimate a distribution. These detectors estimate concept drift using different statistics.

HDDM_A estimates concept drift through the mean and has three possible output states: stable, warning, and drift. The drift state is the most relevant because it indicates a significant change in the data stream. HDDM_W, on the other hand, uses the Exponentially Weighted Moving Average (EWMA) statistic as a change estimator. Like HDDM_A, it also has three possible output states: stable, warning, and drift.

5.6.3.8 HDDM_A and HDDM_W hyperparameters

The HDDM_A and HDDM_W detectors have similar parameterizations, with one notable difference: HDDM_A has an additional hyperparameter called lambda option (l_o), which controls the weight given to recent data. Specifically, smaller values of l_o correspond to less weight given to recent data. Both detectors share two hyperparameters: d_c , which sets the confidence level for detecting a drift, and w_c , which sets the confidence level for issuing a warning.

5.6.3.9 KSWIN

KSWIN is a change detector proposed by Kolmogorov-Smirnov, named after its basis in the Kolmogorov-Smirnov (KS) statistical test [74]. It operates using a sliding window, hence the name KSWIN. The detector identifies concept drift by comparing the distance of the empirical cumulative data distribution of different statistical windows with the last identified concept drift.

Santos et al. [75] employed a differential evolution algorithm, a technique that emulates the principles of natural selection to optimize solutions in multidimensional problem spaces, in order to derive optimal configurations for DDs. Their investigation encompassed an array of detectors, leading to the observation that the DDM in conjunction with the Naive Bayes classifier exhibited superior performance. The research additionally presented a comprehensive tabular representation enumerating the hyperparameters associated with each individual detector.

To enhance the clarity of hyperparameterization patterns across detectors, a refinement was introduced to Table 7 based on the insights gained from Santos et al.’s work. The modified representation of the table has been thoughtfully curated to exclusively encompass the detectors examined within this current study. This strategic choice ensures that the pertinent hyperparameter details are highlighted without undue complexity, thereby facilitating a succinct and illuminating overview of the studied detectors’ configurations.

The “range variation” column encompasses a spectrum of values that were meticulously assessed during the course of the unsupervised tuning process. This process involved an exhaustive exploration of potential values within this specified range to discern the optimal hyperparameter configuration for each detector. The ultimate values derived from the tuning process were rounded to ensure the detector’s operation was not compromised.

Table 7 – Drift detector parameter values.

Detector	p1 default (range variation)	p2 default (range variation)	p3 default (range variation)	p4 default (range variation)
PH	$m_i = 30$ [3, 300]	$\delta = 0.005$ [0.0005, 0.05]	$t = 50$ [5, 500]	$\alpha = 0.9999$ [0.99999, 0.999]
ADWIN	$\delta = 0.002$ [0.0002, 0.02]	N/A	N/A	N/A
DDM	$m_i = 30$ [3, 300]	$w_l = 2.0$ [0.2, 20]	$o_d = 3.0$ [0.3, 30]	N/A
HDDM _A	$d = 0.001$ [0.0001, 0.01]	$w = 0.005$ [0.0005, 0.05]	$t = 1$ [0.1, 10]	N/A
HDDM _W	$d = 0.001$ [0.0001, 0.01]	$w = 0.005$ [0.0005, 0.05]	$t = 1$ [0.1, 10]	$\lambda = 0.05$ [0.005, 0.5]
KSWIN	$\alpha = 0.005$ [0.0005, 0.05]	$w = 100$ [10, 1000]	$ss = 30$ [3, 300]	N/A

5.6.4 Evaluation metrics

The experimental validation used metrics to evaluate DD performance. One of these metrics, “Correct Detection”, inspired by the confusion matrix [76], calculates the difference between the number of accurately identified concept drift (True Positives - TP) and the non-drift samples mistakenly classified as concept drift (False Positives - FP).

This difference is divided by the total number of concept drift in the analyzed signal, represented as N . Equation (5.1) outlines the calculation for this value.

$$\text{Correct Detection} = \frac{\text{TP} - \text{FP}}{N} \times 100 \quad (5.1)$$

Other metrics such as Accuracy were not a valid choice due to the natural imbalance in this application, where the amount of normal signals far outweighs those classified as concept drifts. Precision was also deemed inappropriate since only TP with FP would be evaluated, without taking into account the actual quantity of concept drifts present in the signal. Similarly, conventional metrics like Recall suffer from the same limitation, as they only consider TP with False Negatives (FN).

It is important to note that the outcome in Equation (5.1) is presented as a percentage. When the number of FP equals or surpasses the TP, the result becomes 0%. Also, the parameter N represents the count of intervals with perturbations in the analyzed signal. TP indicates correct detections where the DDs identified a drift within intervals genuinely containing a perturbation. It is relevant to mention that each interval receives only one count, regardless of the number of detections within it.

We also used Friedman’s statistical test alongside the Nemenyi post-hoc test to evaluate the statistical differences between the different configurations of DDs. The Friedman’s test provides an unbiased evaluation of the techniques by considering both mean differences and ranking order. This is crucial when performance metrics show substantial variation across techniques. Additionally, it determines if there are statistically significant differences in performance between the techniques.

5.6.5 Unsupervised Tuning Algorithm

To enhance the comprehension of our proposed approach, Algorithm 4 outlines the necessary steps to conduct the unsupervised tuning process for any DD that incorporates a set of hyperparameters. In Algorithm 4, the *CDS* method is applied to our adaptation of the Change Detector Segmentation framework, allowing us to obtain critical intervals in the signal. Subsequently, the *drift_detector* method generalizes the application of our unsupervised approach to any DD. Following this, we derive the heuristics *BISO*, *BI*, and *SO* through the *compare_drift* method based on the drift detections and the assessment of critical intervals. Finally, our Correct Detection metric is calculated based on the accuracy of the detections according to our proposed architecture. The output of this method is precisely the heuristic with the configurations of *BISO*, *BI*, and *SO*, along with the Correct Detection metric for each of them.

It is essential to underscore that the functionality of our unsupervised tuning proposal does not rely on the input *perturbation_indices*, as it is designed to operate

effectively on unlabeled datasets. The inclusion of *perturbation_indices* in this pseudocode is solely for the purpose of validating the metric Correct Detection that evaluates the performance viability of the technique.

Algorithm 4: Unsupervised tuning

Input: *signal*, refers to the signal for which the existence of concept drift will be evaluated.
perturbation_indices, indicates the labeled perturbations in the signal.
dd_hyperparameters_percentages, represents the percentages of hyperparameters to be evaluated.
Output: BISO,
 BI,
 SO,
 correct_detection

```

1 Pseudo-code
2 while  $i \leq \text{length}(\text{dd\_hyperparameters\_percentages})$  do
3    $\text{intervals} = \text{CDS}(\text{signal})$ 
4    $\text{detections} = \text{drift\_detector}(\text{signal}, \text{dd\_hyperparameters\_percentages})$ 
5    $\text{BISO} = \text{compare\_drifts}(\text{detections}, \text{intervals})$ 
6    $\text{BI} = \text{compare\_drifts}(\text{detections}, \text{intervals})$ 
7    $\text{SO} = \text{compare\_drifts}(\text{detections}, \text{intervals})$ 
8 end
9  $\text{correct\_detection} = \text{metrics}(\text{BISO}, \text{BI}, \text{SO}, \text{perturbation\_indices})$ 
10 return BISO, BI, SO, correct_detection

```

5.7 Results and discussion

This study aims to address specific Research Questions (RQ) concerning signal drift identification via unsupervised tuning. The RQs are outlined as follows:

1. Can unsupervised tuning for DDs based on stationary analysis improve the drift identification process?
2. What is the most impactful factor in detecting drifts in signals?

To address RQ1, we assess six different DDs and illustrate the outcomes for four of them in Figs. 16-19 using a boxplot analysis. These figures portray the Correct Detection level achieved by each hyperparameter configuration in drift detection. Higher Correct Detection percentages mean superior performance. The ADWIN and DDM detectors were omitted from the analysis as they consistently yielded a 0% result, regardless of tuning. The aim of these evaluations is to analyze various signal behaviors amid concept drift presence.

Figs. 16 and 17 present outcomes associated with low amplitude variations, representing items A to J in Table 6. Fig. 16 displays results without perturbations in the signal, pertaining to items A and F in Table 6. In contrast, Fig. 17 includes perturbations, corresponding to items B, C, D, E, G, H, I, J in Table 6. These boxplots compare the default configuration of each detector against the BISO, BI, and SO tuning heuristics proposed in this study.

Figs. 18 to 19 depict outcomes obtained from high amplitude variations using the same configurations as those utilized for low amplitude variations. Fig. 18 demonstrates results from high amplitude variations without perturbations, referring to item K in Table 6. Conversely, Fig. 19 exhibits the results from high amplitude variations with perturbations, relating to items L, M, N, and O in Table 6.

In Fig. 16, it is evident that the BISO and BI configurations outperformed the default configuration across all DDs, except for BI in the HDDM_W detector. Fig. 17 shows a similar trend, except in the case of the HDDM_W detector, where the BISO heuristic performed worse than the default configuration. However, the SO heuristic stood out in this scenario, indicating that in instances of increased signal variability and low amplitudes, a less sensitive heuristic is more appropriate. Another notable observation is that the PH detector was more affected by the introduction of perturbations compared to the HDDM detectors, as inferred from the Correct Detection metric.

In Fig. 18, the BI detector demonstrated exceptional performance across most cases, failing to improve drift detection only for the HDDM_A detector. Fig. 19 depicts a similar pattern, where the BI heuristic closely matched the default configuration and underperformed only in the HDDM_W detector. Notably, the BI heuristic was the sole approach to achieve results above 0% for the KSWIN detector, which is an interesting observation.

The DDs results showcase promising median performance following the unsupervised tuning process. Upon analyzing the figures, it becomes apparent that the primary factor influencing a decrease in Correct Detection of DDs is the presence of signal perturbation, which addresses RQ2. Notably, for the PH detector, it is clear that unsupervised tuning using the BISO and BI approaches consistently delivered better results across all scenarios, slightly favoring the BI approach. This validates our RQ1.

To precisely identify drifts, it is vital to detect them within their actual occurrence range. Any identification beyond this range leads to a FP. Remarkably, the BI configuration showcased the most favorable performance in the conducted assessments. We carried out experiments employing various signal sizes, amplitudes, and perturbation levels to ensure the method's applicability across different scenarios and to simulate the behavior seen in natural phenomena.

Figure 16 – Boxplot of low amplitudes variation without perturbations on signal experiments.

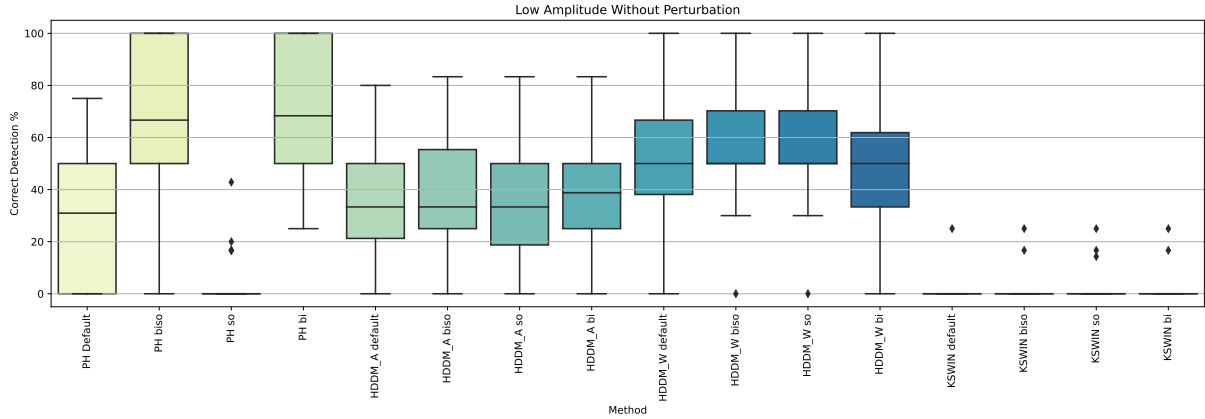


Figure 17 – Boxplot of low amplitudes variation with perturbations on signal experiments.

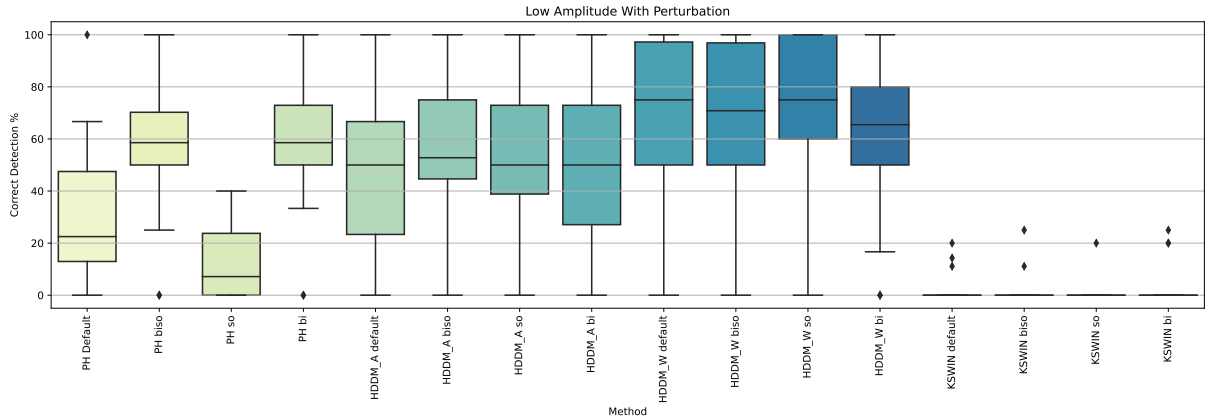
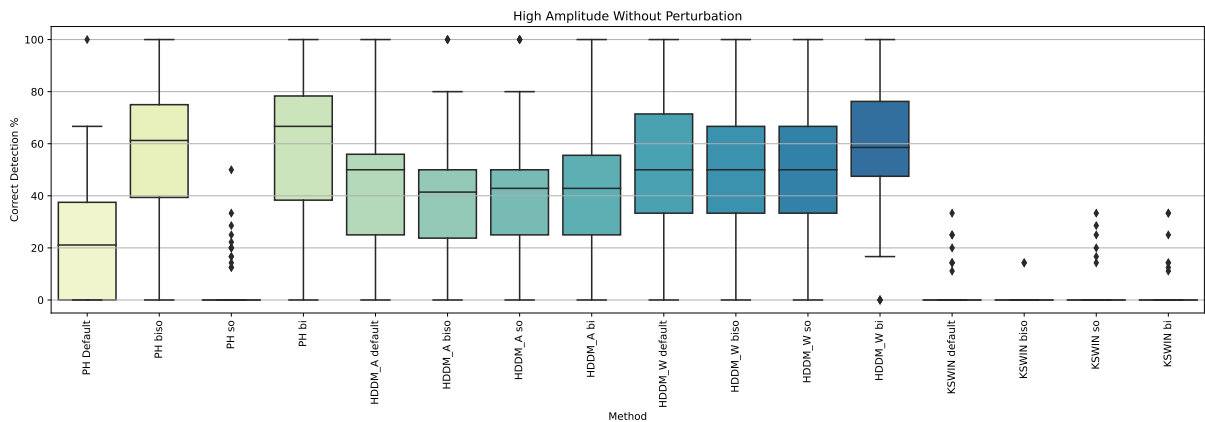
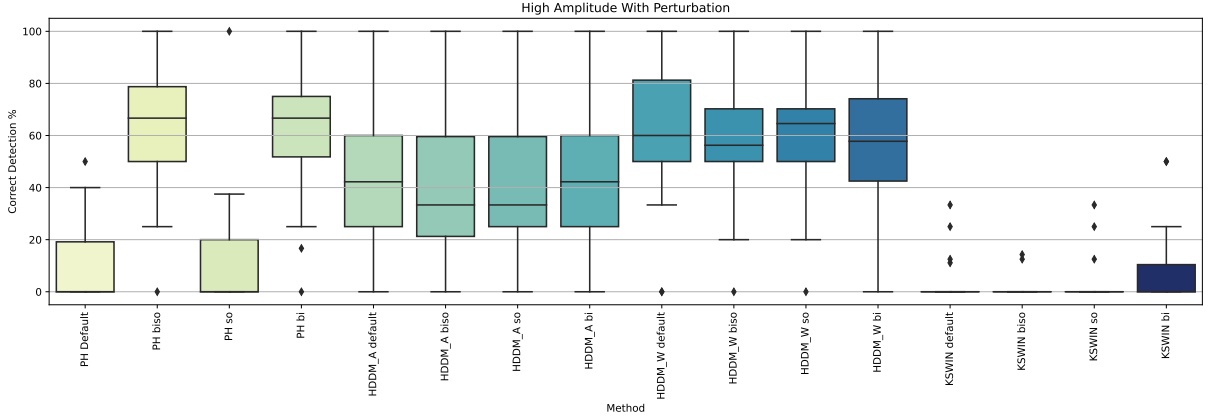


Figure 18 – Boxplot of high amplitudes variation without perturbations on signal experiments.



Overall, our experiments highlighted the proposed method's viability as an alternative for configuring DDs. Notably, the SO heuristic displayed comparatively weaker

Figure 19 – Boxplot of high amplitudes variation with perturbations on signal experiments.



performance among the heuristics, mainly due to its conservative design. It focuses on identifying the fewest drifts outside critical intervals to reduce false positives, potentially missing actual drifts occurring within these regions.

One potential explanation for KSWIN frequently yielding 0% results could be mis-configured hyperparameters. For instance, if the window size does not align well with the data’s change rate, subtle concept drift might not be detected. Similarly, setting a high change threshold could cause the algorithm to overlook changes below that threshold. ADWIN and DDM detectors might have faced similar issues due to their operational characteristics. Additionally, sudden or large changes in data magnitude could pose challenges for detection, especially using a sliding window approach to calculate concept drift statistics.

5.7.1 Unsupervised Tuning vs Random Search

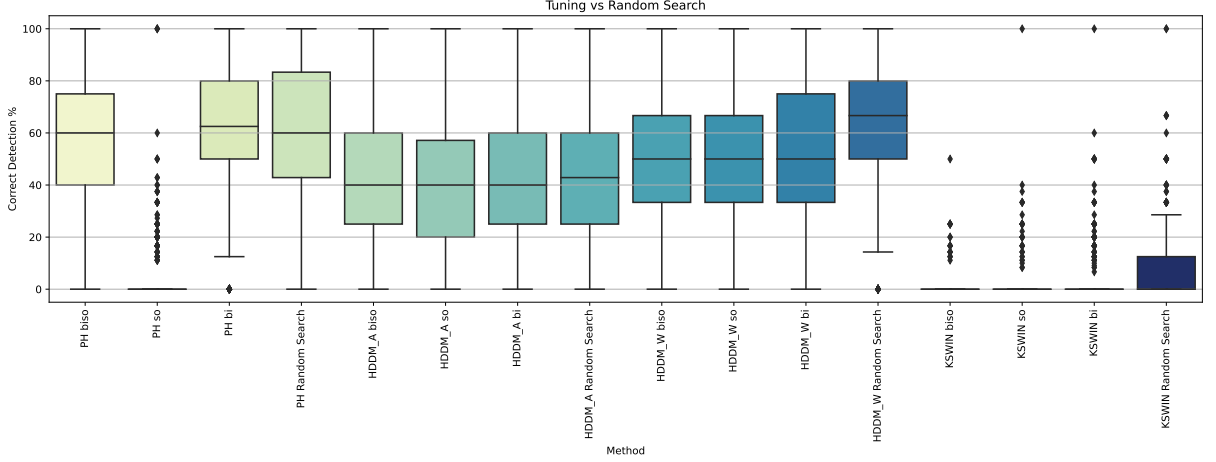
In light of the current gap in the literature pertaining to unsupervised tuning approaches, a pragmatic decision has been made to introduce RS [19, 52] as a competitor in our study. Unsupervised tuning, in its nuanced complexities, poses a unique set of challenges that, unfortunately, has not been extensively addressed in existing research. The absence of established methodologies and benchmarks for unsupervised tuning strategies underscores the need for an adaptable and exploratory approach.

Fig. 20 presents a comparison between our unsupervised tuning method and supervised RS tuning. RS is known for accurately defining hyperparameters but relies on a supervised approach, which might not always be feasible in real-world scenarios.

Given that these experiments were intentionally designed with deliberate data perturbations, all configurations listed in Table 6 were utilized. As expected, the RS method showed superior results across most test cases compared to our tuning approach.

However, our method notably outperformed the supervised approach specifically for the PH detector. Moreover, in all other experiments, our method consistently matched the results generated by the RS method. This comparison validates the effectiveness of our proposed tuning approach and supports RQ1 based on the achieved outcomes.

Figure 20 – Boxplot of our tuning process vs Random Search.



5.7.2 Number of samples impact

For a more thorough validation of RQ2 and to analyze how the number of samples affects the accurate detection of drifts, we conducted a new assessment. The first group, Figs. 21-24, illustrates this impact on DDs performance by comparing three detection configurations: default, RS, and BI. They display artificial signals generated by the CSG without perturbations, while the second group, Figs. 25-28, shows signals generated by the CSG with perturbations. This evaluation encompassed all instances detailed in Table 6.

In the first group, we noticed a tendency: as the number of samples increased, Correct Detection also tended to rise. This trend might be due to the detectors retaining more statistical information about the signal being evaluated. A similar trend was observed in the second group, albeit with a lower percentage of correct detections. Additionally, the default configuration outperformed other setups only for the HDDM_A detector.

The results from second group for the PH detector show that despite RS benefiting from its supervised approach, the BI heuristic achieved better Correct Detection. Also, the BI heuristic consistently outperformed the default configuration in most cases for the DDs.

The findings across these two groups reaffirm that the presence of perturbations in the signal significantly reduces the Correct Detection of DDs, further supporting RQ2, highlighting that the foremost influencing factor is the presence of these perturbations.

Figure 21 – Page-Hinkley without perturbations.

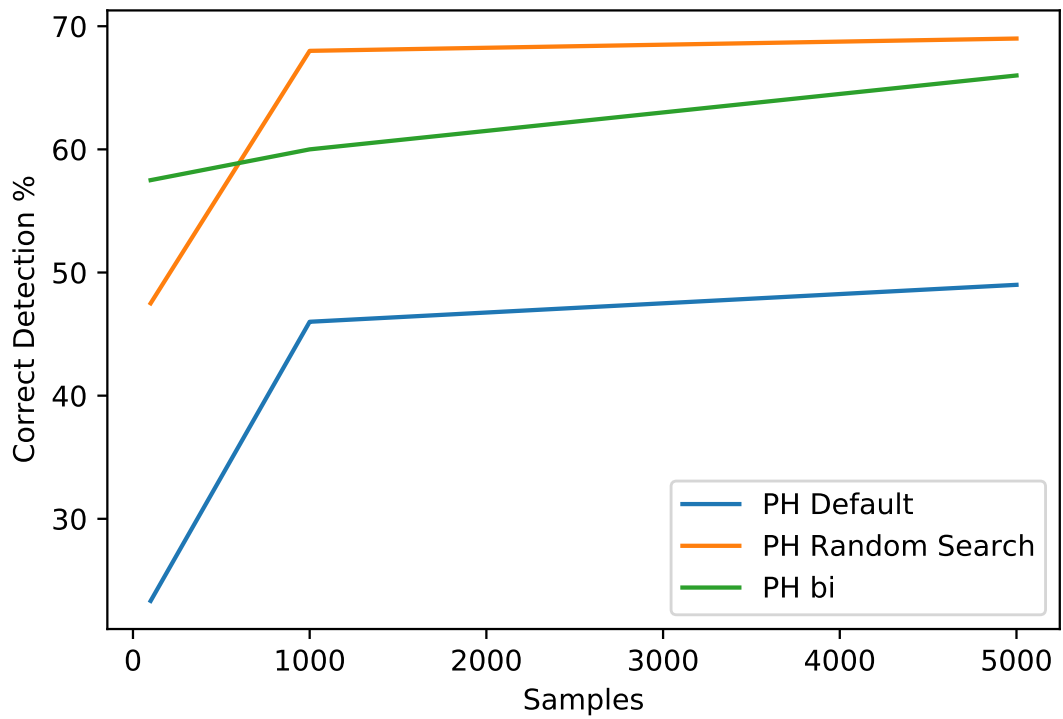
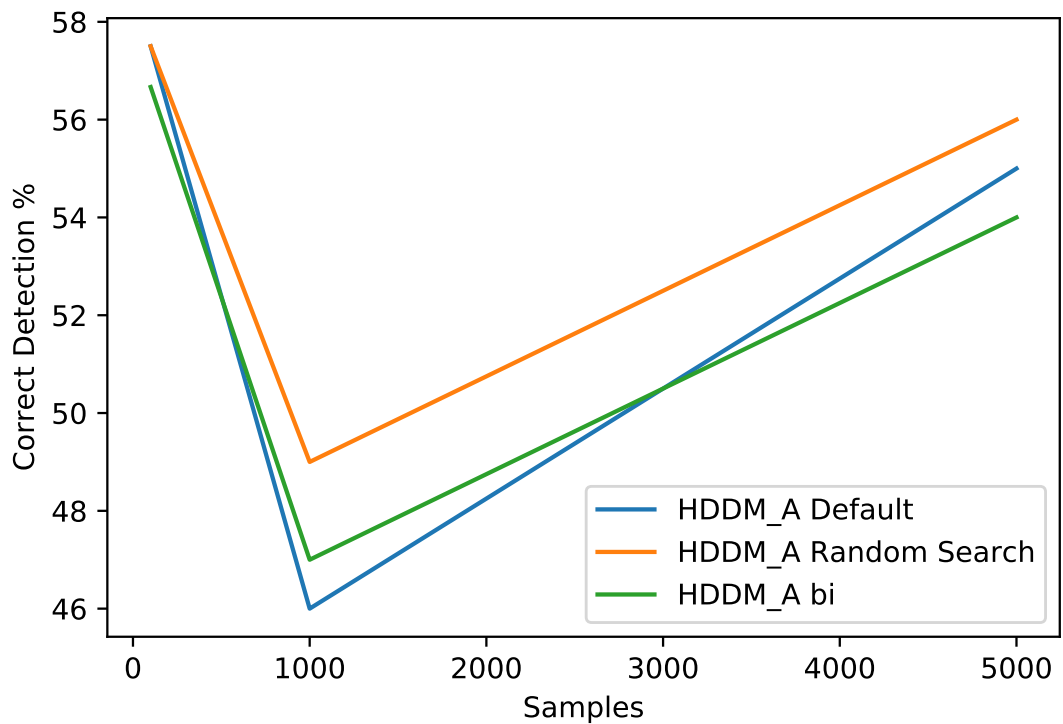
Figure 22 – HDDM_A without perturbations.

Figure 23 – HDDM_W without perturbations.

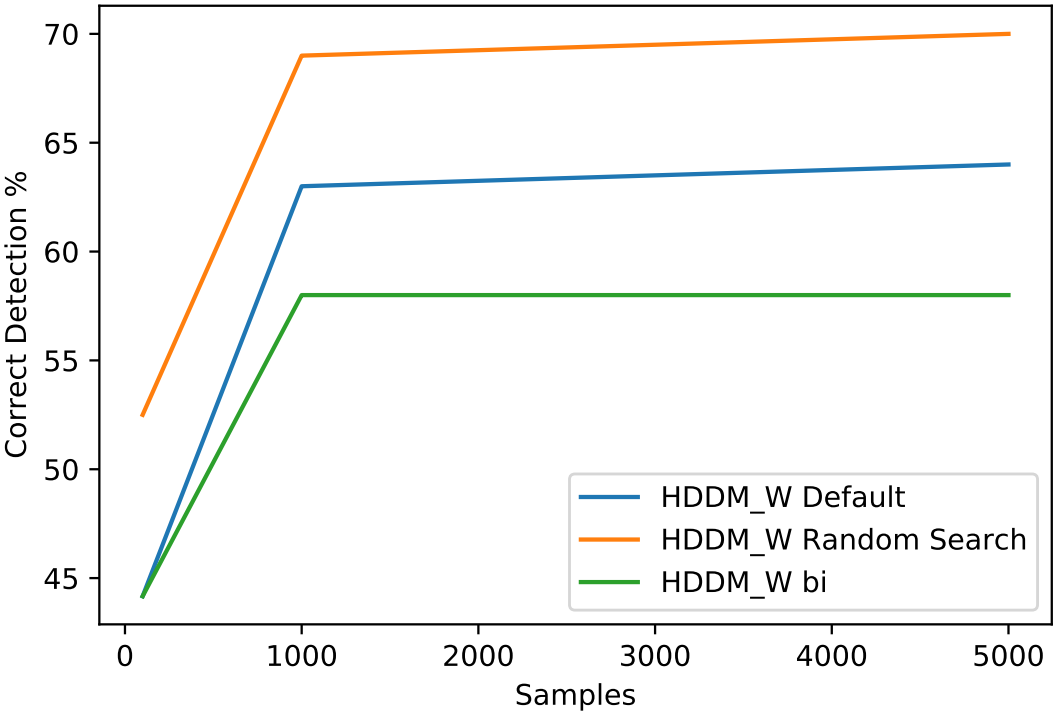


Figure 24 – KSWIN without perturbations.

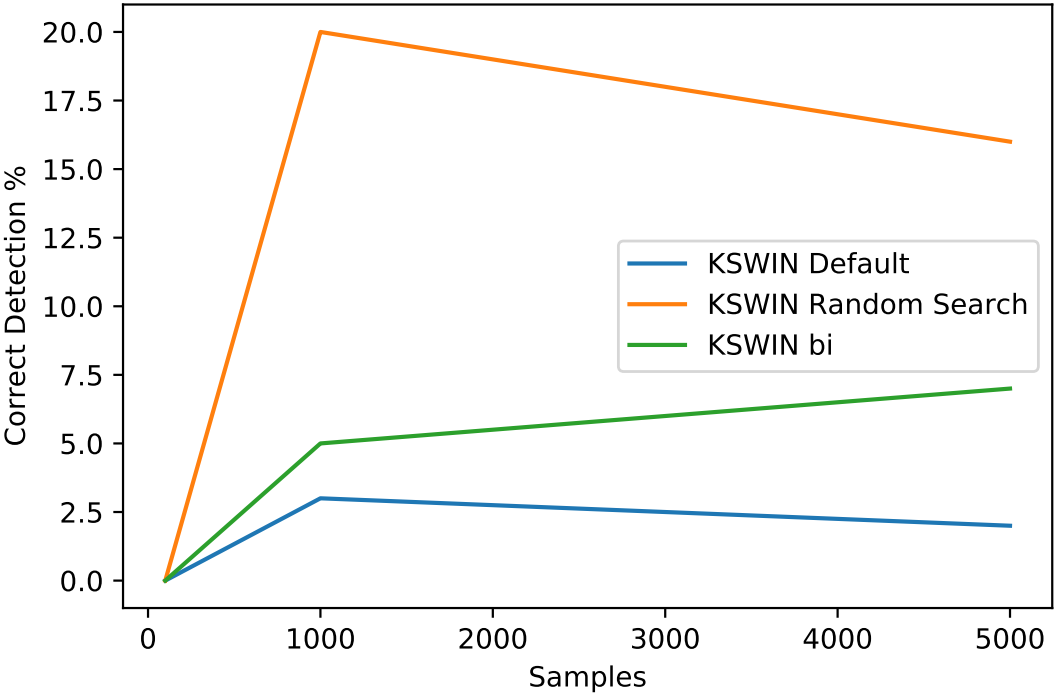


Figure 25 – Page-Hinkley with perturbations.

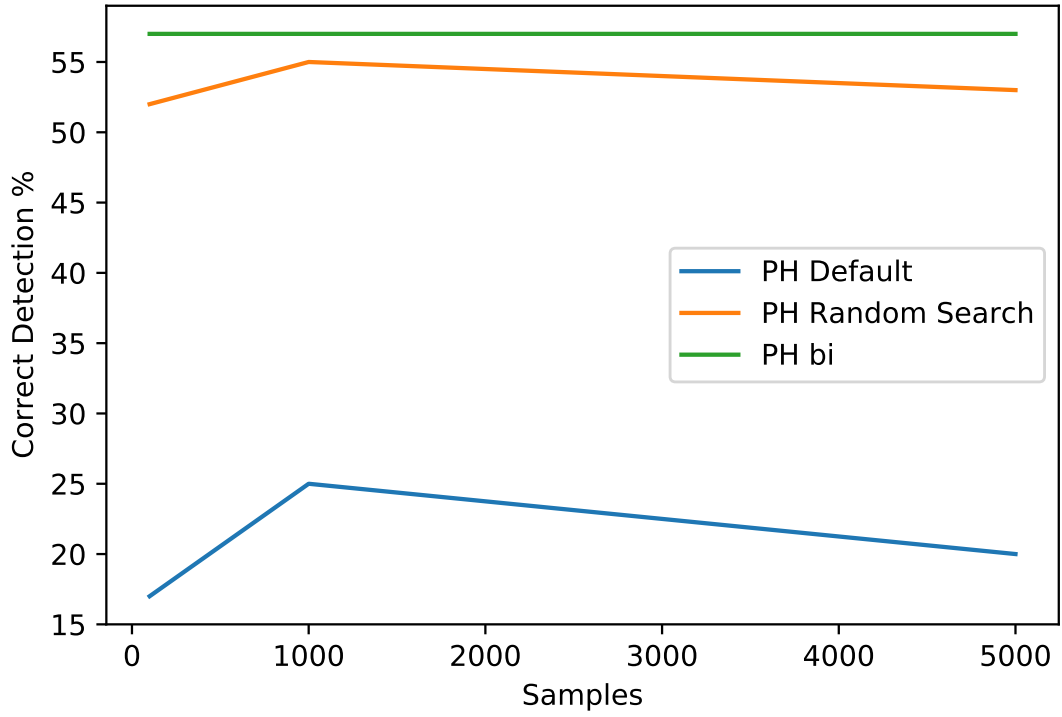
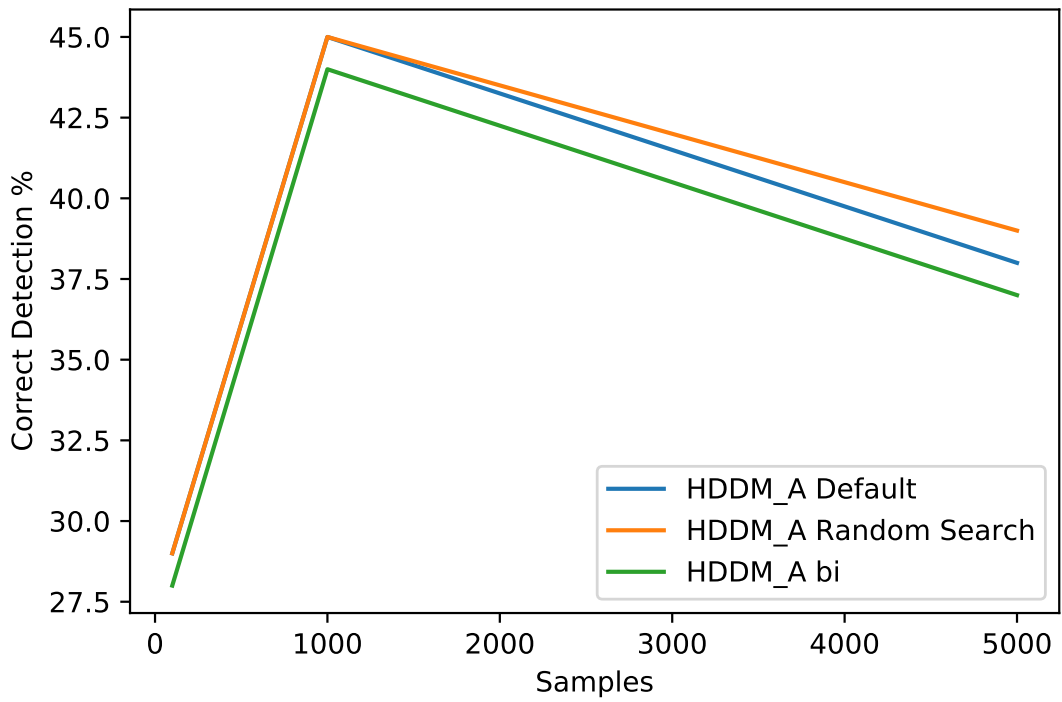
Figure 26 – HDDM_A with perturbations.

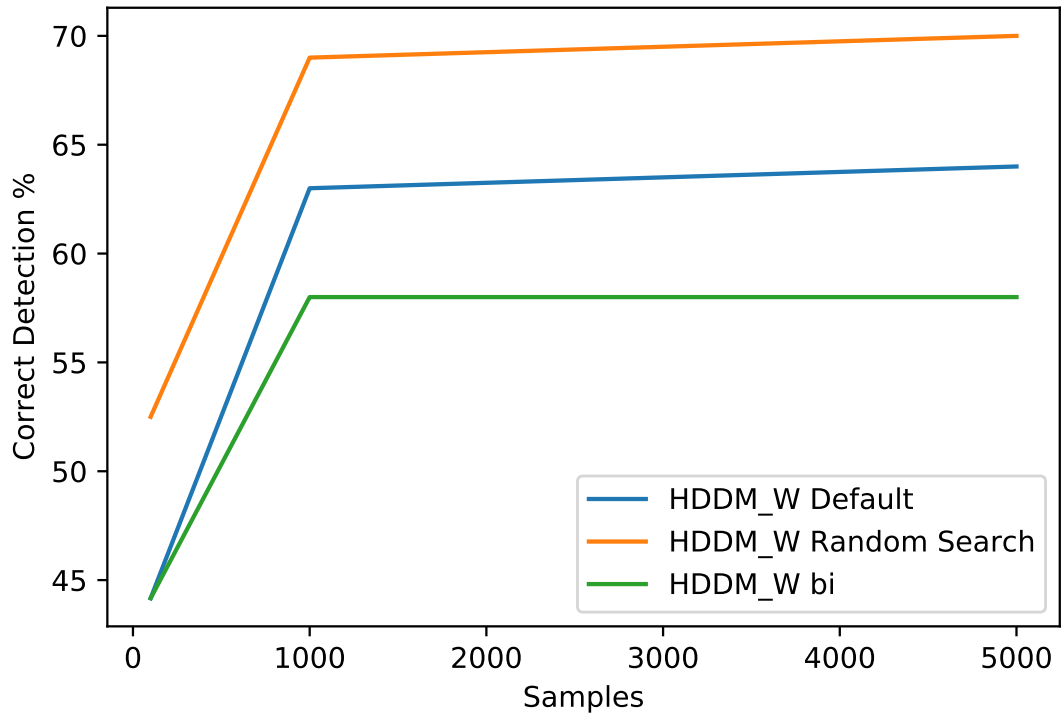
Figure 27 – HDDM_W with perturbations.

Figure 28 – KSWIN with perturbations.

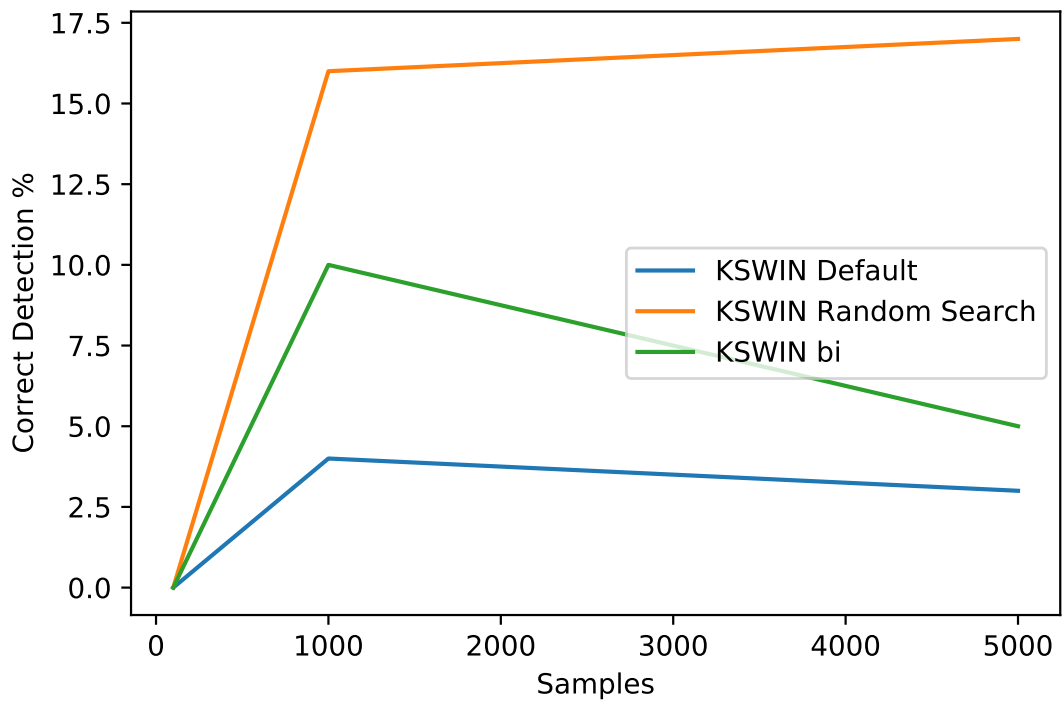


Figure 29 – Friedman’s statistical test on experiment J, size = 5000.

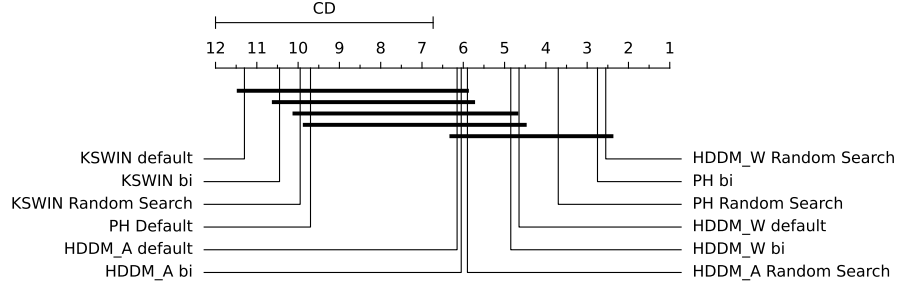
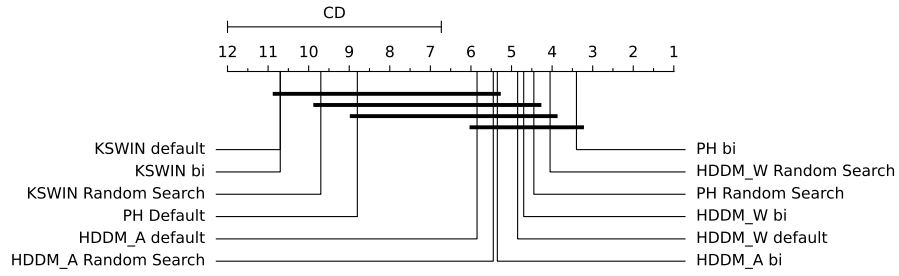


Figure 30 – Friedman’s statistical test on experiment O, size = 5000.



5.7.3 Statistical analysis

To examine the noticeable differences between DDs configurations, we utilized Friedman’s statistical test alongside the Nemenyi post-hoc test. Friedman’s test involves ranking the observations of each configuration in ascending order and calculating the sum of differences between these observed ranks and the expected ranks, which would be identical if all samples had the same median.

For this test, we focused on configurations J and O from Table 6. These configurations represent the most complex settings highlighted in our evaluations. This specific selection aimed to illustrate the test within the most complex scenario for drift detection in comparison to the other experiments.

From the results depicted in Fig. 29, it is evident that the KSWIN, HDDM_A, and HDDM_W approaches displayed no statistically significant differences between the default configuration and the tuned configurations. However, concerning the PH detector, both PH with BI and PH with RS configurations exhibited statistically significant differences compared to the default setting. A similar trend is observed in Fig. 30, where only the PH configuration showed substantial differences compared to the standard configuration.

5.8 Conclusions

This study presents critical findings about the impact of different hyperparameter setups on drift detection performance across various scenarios. The RS configuration stands out as the most effective in identifying drifts, yet it relies on user knowledge and labeled databases, limiting its use in real-world unsupervised settings.

Conversely, the BI tuning approach offers an unsupervised solution independent of specialized supervision or labeled datasets. It shows promise, especially in scenarios with high amplitudes and significant variations, consistently outperforming default configurations in multiple detectors. This highlights the importance of unsupervised tuning methods in practical applications for detecting drifts.

The current landscape of labeled datasets for concept drift is inadequate to meet the growing demands of developing adaptive ML models. Recognizing the urgency of this issue is imperative to undertake initiatives for generating artificial but controlled datasets such as CSG to simulate the complexities of concept drift.

6 FINAL CONCLUSION

In conclusion, this thesis underscores the necessary role of TS analysis in extracting valuable insights from data that evolve over time. Through diligent exploration and identification of previously unexplored areas within the literature, this work has facilitated the development of innovative techniques such as the CDS segmenter and the ADF-based segmentation. These tools significantly enhance data prediction accuracy and efficiency by refining the data segmentation process and effectively reducing the size of the datasets required for analysis. This not only improves the manageability of possible large data streams but also enhances the performance of predictive models.

Moreover, the study highlights the fundamental importance of tuning processes for DDs, with a special emphasis on unsupervised approaches. The introduction of unsupervised tuning methods marks a pivotal advancement in the field, offering a way to diminish the reliance on expert knowledge and supervised data, which are often limiting in dynamic and real-time applications. This approach not only broadens the applicability of DDs but also enhances their flexibility and utility in various operational environments.

Looking forward, the potential for applying the findings of this thesis in online scenarios presents a promising avenue for future work. Implementing the CDS segmenter and ADF-based optimizer in real-time systems could improve even more the process to respond to data drifts. Further research could explore integrating these tools into live systems, testing their effectiveness and robustness in different industries such as finance, telecommunications, and e-commerce, where data streams are particularly volatile and rich in information. Additionally, continuing to refine unsupervised tuning techniques will be crucial in ensuring that these systems remain accurate and relevant, irrespective of the evolving nature of the data they process.

WORKS PUBLISHED BY THE AUTHOR

1. E. J. Santana, R. P. Silva, B.B. Zarpelão, S. B. Junior, Photovoltaic Generation Forecast: Model Training and Adversarial Attack Aspects, Brazilian Conference on Intelligent Systems, 2020, 634-649.
2. E. J. Santana, R. P. Silva, B.B. Zarpelão, S. B. Junior, Detecting and Mitigating Adversarial Examples in Regression Tasks: A Photovoltaic Power Generation Forecasting Case Study, 2021, Information 12 (10), 394.
3. R. P. Silva, B. B. Zarpelão, A. Cano, S. B. Junior, Time series segmentation based on stationarity analysis to improve new samples prediction, Sensors, 2021, 21 (21), 7333.
4. R. P. Silva, S. B. Junior, B. B. Zarpelão and L. F. De Melo, Unsupervised Tuning for Drift Detectors Using Change Detector Segmentation, in IEEE Access, vol. 12, pp. 54256-54271, 2024, doi: 10.1109/ACCESS.2024.3388529.

REFERÊNCIAS

- [1] ROBERTS, R. A.; MULLIS, C. T. *Digital Signal Processing*. [S.l.]: Addison-Wesley Longman Publishing Co., Inc., 1987.
- [2] XU, Q.; WEN, Q.; SUN, L. Two-stage framework for seasonal time series forecasting. In: *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. [S.l.: s.n.], 2021. p. 3530–3534.
- [3] KUMAR, R. et al. Covid-19 outbreak: An epidemic analysis using time series prediction model. In: *2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*. [S.l.: s.n.], 2021. p. 1090–1094.
- [4] JIANG, Z. Z. F.; SHAO, X. Time series analysis of covid-19 infection curve: A change-point perspective. *Journal of Econometrics*, Elsevier, v. 232, n. 1, p. 1–17, 2023.
- [5] MAHALAKSHMI, G.; SRIDEVI, S.; RAJARAM, S. A survey on forecasting of time series data. In: *2016 International Conference on Computing Technologies and Intelligent Data Engineering (ICCTIDE'16)*. [S.l.: s.n.], 2016. p. 1–8.
- [6] SIAMI-NAMINI, S.; TAVAKOLI, N.; NAMIN, A. S. A comparison of arima and lstm in forecasting time series. In: *17th IEEE International Conference on Machine Learning and Applications (ICMLA)*. [S.l.: s.n.], 2018. p. 1394–1401.
- [7] SANTANA, E. J. et al. Photovoltaic generation forecast: Model training and adversarial attack aspects. *Cerri R., Prati R.C. (eds) Intelligent Systems. BRACIS 2020*, v. 12320, 2020.
- [8] JAMALI, S. et al. Detecting changes in vegetation trends using time series segmentation. *Remote Sensing of Environment*, v. 156, p. 182–195, 2015. ISSN 0034-4257. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S0034425714003538>>.
- [9] KEOGH, E. et al. Segmenting time series: A survey and novel approach. In: *Data mining in time series databases*. [S.l.]: World Scientific, 2004. p. 1–21.
- [10] LEE, W. et al. Time series segmentation through automatic feature learning. *CoRR*, abs/1801.05394, 2018. Disponível em: <<http://arxiv.org/abs/1801.05394>>.
- [11] CERAVOLO, P. et al. Evaluation goals for online process mining: a concept drift perspective. *IEEE Transactions on Services Computing*, p. 1–1, 2020.
- [12] VELOSO, B. et al. Hyperparameter self-tuning for data streams. *Information Fusion*, Elsevier, v. 76, p. 75–86, 2021.
- [13] MARTINS, V. E.; CANO, A.; JUNIOR, S. B. Meta-learning for dynamic tuning of active learning on stream classification. *Pattern Recognition*, Elsevier, v. 138, 2023.
- [14] YU, H. et al. Meta-add: A meta-learning based pre-trained model for concept drift active detection. *Information Sciences*, Elsevier, v. 608, p. 996–1009, 2022.

- [15] JAISWAL, A. et al. A survey on contrastive self-supervised learning. *Technologies*, MDPI, v. 9, n. 1, p. 2, 2020.
- [16] MONTIEL, J. et al. Scikit-multiflow: A multi-output streaming framework. *Journal of Machine Learning Research*, v. 19, n. 72, p. 1–5, 2018. Disponível em: <<http://jmlr.org/papers/v19/18-251.html>>.
- [17] MONTIEL, J. et al. River: Machine learning for streaming data in python. *The Journal of Machine Learning Research*, JMLRORG, v. 22, n. 1, p. 4945–4952, 2021.
- [18] SILVA, R. et al. Time series segmentation based on stationarity analysis to improve new samples prediction. *Sensors*, v. 21, n. 11, p. 1, November 2021.
- [19] ZHOU, M. et al. Prediction model of coal consumption based on random forest variable selection and random-grid hyperparametric optimization algorithm. In: *Proceedings of the 2023 International Conference on Power System Technology (PowerCon)*. [S.l.: s.n.], 2023. p. 1–5.
- [20] BOX, G. E. P.; JENKINS, G. M. *Time Series Analysis: Forecasting and Control*. 3rd. ed. USA: Prentice Hall PTR, 1994. ISBN 0130607746.
- [21] PRADO, R.; WEST, M. *Time series modelling, inference and forecasting*. [S.l.]: Citeseer, 2010.
- [22] LI, Z.; LIU, Y.; TANAKA, G. Multi-reservoir echo state networks with hodrick–prescott filter for nonlinear time-series prediction. *Applied Soft Computing*, v. 135, p. 110021, 2023. ISSN 1568-4946. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S156849462300039X>>.
- [23] DUBEY, A. K. et al. Study and analysis of sarima and lstm in forecasting time series data. *Sustainable Energy Technologies and Assessments*, Elsevier, v. 47, p. 101474, 2021.
- [24] SCHÜLER, Y. S. On the cyclical properties of hamilton’s regression filter. *Available at SSRN 3559776*, 2021.
- [25] BERALDI, R. et al. A random walk based load balancing algorithm for fog computing. In: IEEE. *2020 Fifth international conference on fog and mobile edge computing (FMEC)*. [S.l.], 2020. p. 46–53.
- [26] SHUMWAY, R. H. et al. Arima models. *Time series analysis and its applications: with R examples*, Springer, p. 75–163, 2017.
- [27] ZHOU, T. et al. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In: PMLR. *International conference on machine learning*. [S.l.], 2022. p. 27268–27286.
- [28] LIU, Y. et al. Non-stationary transformers: Exploring the stationarity in time series forecasting. *Advances in Neural Information Processing Systems*, v. 35, p. 9881–9893, 2022.
- [29] LEON-GARCIA, A. *Probability and random processes for electrical engineering*. [S.l.]: Pearson Education, Upper Saddle River, New Jersey, 2008.

- [30] CHEUNG, Y.-W.; LAI, K. S. Lag order and critical values of the augmented dickey–fuller test. *Journal of Business & Economic Statistics*, Taylor & Francis Group, v. 13, n. 3, p. 277–280, 1995.
- [31] MACKINNON, J. G. Approximate asymptotic distribution functions for unit-root and cointegration tests. *Journal of Business & Economic Statistics*, Taylor & Francis, v. 12, n. 2, p. 167–176, 1994.
- [32] GHANI, N. L. A.; AZIZ, I. A.; MEHAT, M. Concept drift detection on unlabeled data streams: A systematic literature review. In: *2020 IEEE Conference on Big Data and Analytics (ICBDA)*. [S.l.: s.n.], 2020. p. 61–65.
- [33] LIU, T. et al. A stock price prediction method based on meta-learning and variational mode decomposition. *Knowledge-Based Systems*, Elsevier, v. 252, p. 109324, 2022.
- [34] CANO, A.; KRAWCZYK, B. Kappa Updated Ensemble for Drifting Data Stream Mining. *Machine Learning*, v. 109, n. 1, p. 175–218, 2020.
- [35] LARA-BENÍTEZ, P. et al. Temporal convolutional networks applied to energy-related time series forecasting. *Applied Sciences*, Multidisciplinary Digital Publishing Institute, v. 10, n. 7, p. 2322, 2020.
- [36] DONG, X. et al. A survey on ensemble learning. *Frontiers of Computer Science*, Springer, v. 14, p. 241–258, 2020.
- [37] HU, J.; SZYMCZAK, S. A review on longitudinal data analysis with random forest. *Briefings in Bioinformatics*, Oxford University Press, v. 24, n. 2, p. bbad002, 2023.
- [38] MANOHARAN, A. et al. Artificial neural networks, gradient boosting and support vector machines for electric vehicle battery state estimation: A review. *Journal of Energy Storage*, Elsevier, v. 55, p. 105384, 2022.
- [39] HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. *Neural Computation*, MIT Press, v. 9, n. 8, p. 1735–1780, 1997.
- [40] SURADHANIWAR, S. et al. Time series forecasting of univariate agrometeorological data: A comparative performance evaluation via one-step and multi-step ahead forecasting strategies. *Sensors*, v. 21, n. 7, 2021. ISSN 1424-8220. Disponível em: <<https://www.mdpi.com/1424-8220/21/7/2430>>.
- [41] POGHOSYAN, A. et al. An enterprise time series forecasting system for cloud applications using transfer learning. *Sensors*, v. 21, n. 5, 2021. ISSN 1424-8220. Disponível em: <<https://www.mdpi.com/1424-8220/21/5/1590>>.
- [42] BARZEGAR, V. et al. Multi-time resolution ensemble lstms for enhanced feature extraction in high-rate time series. *Sensors*, v. 21, n. 6, 2021. ISSN 1424-8220. Disponível em: <<https://www.mdpi.com/1424-8220/21/6/1954>>.
- [43] CARMONA-POYATO, A. et al. A new approach for optimal time-series segmentation. *Pattern Recognition Letters*, v. 135, p. 153–159, 2020. ISSN 0167-8655. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S0167865520301264>>.

- [44] BESSEC, M.; FOUQUAU, J.; MERITET, S. Forecasting electricity spot prices using time-series models with a double temporal segmentation. *Applied Economics*, v. 48, p. 1–18, 09 2015.
- [45] HOOI, B. et al. Beatlex: Summarizing and forecasting time series with patterns. In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. [S.l.: s.n.], 2017. p. 3–19.
- [46] BARROS, R. S. M.; SANTOS, S. G. T. C. A large-scale comparison of concept drift detectors. *Information Sciences*, v. 451–452, p. 348–370, 2018. ISSN 0020-0255.
- [47] SANTOS, S. G.; BARROS, R. S.; JÚNIOR, P. M. G. Optimizing the parameters of drift detection methods using a genetic algorithm. In: *Proceedings of the 2015 IEEE 27th International Conference on Tools with Artificial Intelligence (ICTAI)*. [S.l.]: IEEE, 2015. p. 1077–1084.
- [48] GEMAQUE, R. N. et al. An overview of unsupervised drift detection methods. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, Wiley Online Library, v. 10, n. 6, p. e1381, 2020.
- [49] AGRAHARI, S.; SINGH, A. K. Concept drift detection in data stream mining: A literature review. *Journal of King Saud University - Computer and Information Sciences*, v. 34, n. 10, Part B, p. 9523–9540, 2022. ISSN 1319-1578.
- [50] HU, H.; KANTARDZIC, M.; SETHI, T. S. No free lunch theorem for concept drift detection in streaming data classification: A review. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, Wiley Online Library, v. 10, n. 2, p. e1327, 2020.
- [51] BISCHL, B. et al. Hyperparameter optimization: Foundations, algorithms, best practices, and open challenges. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, Wiley Online Library, v. 13, n. 2, p. e1484, 2023.
- [52] BUSLIM, N. et al. Comparing bitcoin’s prediction model using gru, rnn, and lstm by hyperparameter optimization grid search and random search. In: *Proceedings of the 2021 9th International Conference on Cyber and IT Service Management (CITSM)*. [S.l.: s.n.], 2021. p. 1–6.
- [53] NUGROHO, A.; SUHARTANTO, H. Hyper-parameter tuning based on random search for densenet optimization. In: *Proceedings of the 2020 7th International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE)*. [S.l.: s.n.], 2020. p. 96–99.
- [54] GAHROOEI, M. R.; PAYNABAR, K. Change detection in a dynamic stream of attributed networks. *Journal of Quality Technology*, Taylor & Francis, v. 50, n. 4, p. 418–430, 2018.
- [55] GIL-ALANA, L. A. Long memory behaviour in the daily maximum and minimum temperatures in melbourne, australia. *Meteorological Applications*, Cambridge University Press, v. 11, n. 4, p. 319–328, 2004.
- [56] D.F., H. A. A. Monthly mean sunspot numbers. *Data. Springer Series in Statistics*, v. 35, n. 1, p. 213 – 216, 1985.

- [57] BIFET, A.; GAVALDA, R. Learning from time-changing data with adaptive windowing. In: *SIAM International Conference on Data Mining*. [S.l.: s.n.], 2007. p. 443–448.
- [58] PAGE, E. S. Continuous inspection schemes. *Biometrika*, JSTOR, v. 41, n. 1/2, p. 100–115, 1954.
- [59] BOX, G. E. P.; JENKINS, G. *Time Series Analysis, Forecasting and Control*. USA: Holden-Day, Inc., 1990. ISBN 0816211043.
- [60] HO, S.; XIE, M. The use of ARIMA models for reliability forecasting and analysis. *Computers & Industrial Engineering*, v. 35, n. 1, p. 213 – 216, 1998. ISSN 0360-8352.
- [61] PENA, E. H. et al. Anomaly detection using digital signature of network segment with adaptive arima model and paraconsistent logic. In: *IEEE Symposium on Computers and Communications (ISCC)*. [S.l.: s.n.], 2014. p. 1–6.
- [62] AKHTER, M. N. et al. Review on forecasting of photovoltaic power generation based on machine learning and metaheuristic techniques. *IET Renewable Power Generation*, IET, v. 13, n. 7, p. 1009–1023, 2019.
- [63] CERQUEIRA, V.; TORGO, L.; SOARES, C. Machine learning vs statistical methods for time series forecasting: Size matters. *arXiv preprint arXiv:1909.13316*, 2019.
- [64] BAI, S.; KOLTER, J. Z.; KOLTUN, V. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.
- [65] LEA, C. et al. Temporal convolutional networks: A unified approach to action segmentation. In: *ECCV Workshops - Computer Vision*. [S.l.]: Springer, 2016. (Lecture Notes in Computer Science), p. 47–54.
- [66] LIU, J.; XU, Y. T-friedman test: a new statistical test for multiple comparison with an adjustable conservativeness measure. *International Journal of Computational Intelligence Systems*, Springer, v. 15, n. 1, p. 29, 2022.
- [67] MAKOWSKI, D. et al. Methods and algorithms for correlation analysis in r. *Journal of Open Source Software*, v. 5, n. 51, p. 2306, 2020.
- [68] MAHDI, O. A. et al. Fast reaction to sudden concept drift in the absence of class labels. *Applied Sciences*, MDPI, v. 10, n. 2, p. 606, 2020.
- [69] SARIEDDEEN, H.; ALOUINI, M.-S.; AL-NAFFOURI, T. Y. An overview of signal processing techniques for terahertz communications. *Proceedings of the IEEE*, IEEE, v. 109, n. 10, p. 1628–1665, 2021.
- [70] BERGSTRA, J.; BENGIO, Y. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, v. 13, n. 2, 2012.
- [71] GAMA, J. et al. Learning with drift detection. In: BAZZAN, A. L. C.; LABIDI, S. (Ed.). *Advances in Artificial Intelligence – SBIA 2004*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004. p. 286–295. ISBN 978-3-540-28645-5.

- [72] FRÍAS-BLANCO, I. et al. Online and non-parametric drift detection methods based on hoeffding's bounds. *IEEE Transactions on Knowledge and Data Engineering*, v. 27, n. 3, p. 810–823, 2015.
- [73] BIFET, A. et al. Moa: Massive online analysis. *Journal of Machine Learning Research*, JMLR.org, v. 11, p. 1601–1604, August 2010. ISSN 1532-4435.
- [74] RAAB MORITZ HEUSINGER, F.-M. S. C. Reactive soft prototype computing for concept drift streams. *Neurocomputing*, v. 416, p. 340–351, 2020. ISSN 0925-2312.
- [75] SANTOS, R. S. B. S. G.; JR, P. M. G. A differential evolution based method for tuning concept drift detectors in data streams. *Information Sciences*, Elsevier, v. 485, p. 376–393, 2019.
- [76] HEYDARIAN, T. E. D. M.; SAMAVI, R. Mlcm: Multi-label confusion matrix. *IEEE Access*, IEEE, v. 10, p. 19083–19095, 2022.