Universidad del País Vasco

Euskal Herriko Unibertsitatea

Contributions to time series data mining towards the detection of outliers/anomalies

by

Ane Blázquez García

Supervised by Angel Conde and Usue Mori

June, 2022

To my family.

Acknowledgments

Completing this thesis has not been an easy path to follow. Frustration has prevailed at many moments, and discipline and hard work have been relevant throughout the entire process. Even so, I have also had many good moments full of joy, satisfaction and learning. And, most importantly, I have been fortunate to be surrounded by wonderful people who, in one way or another, have made this thesis possible.

First of all, I would like to express my sincere gratitude to my supervisors, Angel Conde, Usue Mori, and Jose Antonio Lozano, for their patience, guidance, and helpful advice. You all have been a constant support, and your insightful comments pushed me to refine my thinking and took my work to a higher level. Benetan, mila esker!

I would like to extend my sincere thanks to IKERLAN, the Research Centre where I have carried out this thesis, for supporting this dissertation. Special thanks to all my colleagues who have accompanied me in this process: thank you for the great working atmosphere you have created, for all the coffees, the cheering up in difficult moments, the chats at lunch... Above all, thank you for making this thesis fun as well.

I am also grateful to Robert Jenssen for giving me the opportunity to visit the UiT Machine Learning group in Tromsø. It has been a great pleasure to work with you. Thanks also to all the members of the group for being so welcoming, for all the interesting discussions held, and for all the unforgettable experience provided. Tusen takk!

I wish to extend my special thanks to all my friends for all the moments of joy and fun, and for helping me disconnect when I most needed it. You have been an essential support and it is clear that without you everything would have been much more difficult.

Finally, I want to express my deepest appreciation to my family, for giving me light in my darkest moments and for always being there. My best acknowledgment to my parents, Isa and Pruden, and brother, Hector: gracias por ese cariño y apoyo incondicional, por todos los valores inculcados, por aguantar mis repentinos cambios de humor, y por hacer que nunca pierda la esperanza.

All things are difficult before they are easy.
Thomas Fuller.

Abstract

Recent advances in technology have brought about major breakthroughs in data collection, enabling a large amount of data to be gathered over time. These data are often presented in the form of time series, where the observations have been recorded in an orderly fashion and are correlated in time. In recent years, a great interest has arisen in extracting meaningful and useful information from such data. The research area that focuses on this task is called time series data mining.

The time series data mining community has been devoted to solving different tasks, including the detection of outliers/anomalies. Outliers or anomalies are those observations that do not follow the expected behavior in a time series. These observations typically represent unwanted data or events of interest, and thus, detecting them is desirable because they may worsen the quality of the data or reflect interesting phenomena that the analyst intends to detect.

This thesis presents several contributions in the field of time series data mining, more specifically, on the detection of outliers or anomalies. Indeed, the contributions we present in this thesis are 1) a comprehensive review and taxonomy for the unsupervised outlier/anomaly detection techniques in time series data; 2) a novel self-supervised anomaly detection technique for whole univariate time series, aimed at detecting water leaks; and 3) a new technique for processing multivariate time series with missing values based on selective imputation, which can be applied in subsequent tasks such as outlier/anomaly detection.

Contents

List of Figures

List of Tables

1

Introduction

This thesis deals with various problems encountered in the field of time series data mining with a specific focus on the detection of outliers/anomalies. Specifically, it presents three contributions in this area. Before introducing such contributions, and to ease the understanding of the present dissertation, this chapter provides the necessary background. Firstly, an introduction to time series data mining is presented in Section 1.1, which gives some basic definitions used throughout the dissertation and a brief description of the most popular tasks in this area of study. Then, more details on the outlier/anomaly detection task are provided in Section 1.2, as it is the main focus of this thesis. Next, the objectives of the thesis are described in Section 1.3, and finally, the overview of the dissertation is outlined in Section 1.4.

1.1 Time series data mining

A time series is a set of time-ordered observations that are correlated in time. In practice, time series are generated in a variety of domains such as economy (e.g., quarterly unemployment rate [2] or tourism demand data [3]), astronomy (e.g., variable-star photometric data [4]), meteorology (e.g., sequential monthly data on surface temperature, humidity or pressure [5]), or health (e.g., heart rate signals [6]), to name a few.

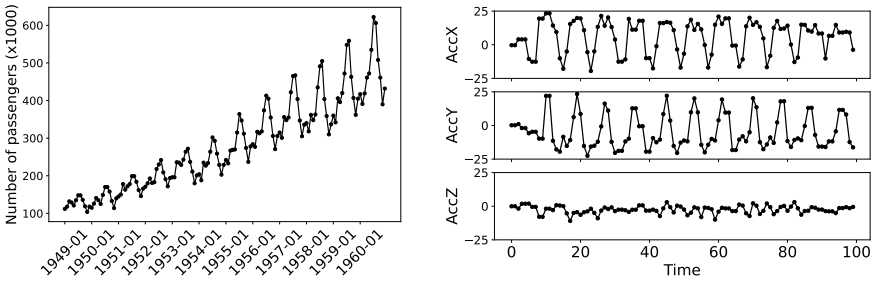
Formally, time series have been defined in different ways in the literature, and below, the definition that is used in this dissertation is presented.

Definition 1 (Time series). *A time series* $Y = \{\boldsymbol{y}_t\}_{t \in \mathcal{T}}$ *is defined as an ordered set of* L*-dimensional vectors,* $\boldsymbol{y}_t = (y_{1t}, ..., y_{Lt})$*, each of which is recorded at a specific time* $t \in \mathcal{T} \subseteq \mathbb{Z}^+$ *and consists of* $L \in \mathbb{N}$ *real-valued observations.*

More specifically, when a time series is an ordered set of real-valued observations (i.e., $L = 1$), then the time series is denominated *univariate*. In this case, y_{1t} (y_t for the sake of simplicity) is said to be the *point* or observation

collected at time $t \in \mathcal{T}$ and $S = \{y_p, y_{p+1}, ..., y_{p+n-1}\}$ the *subsequence* of length $n \leq |\mathcal{T}|$ starting at position p of the time series Y, where $p \in \mathcal{T}$ and $p \leq |\mathcal{T}| - n + 1$. It is assumed that each observation y_t is a realized value of a certain random variable $\mathcal{Y}_t$. An example of a univariate time series which is a classical example in the literature is shown in Fig. 1.1a. It is a univariate time series that collects the monthly number of international airline passengers from 1949 to 1960 [7].

Conversely, if at each time step a multi-dimensional vector is collected (i.e., $L > 1$), then the time series is called *multivariate*. As with univariate time series, $\boldsymbol{y}_t$ is said to be the (multivariate) point collected at time $t \in \mathcal{T}$ and $\boldsymbol{S} = \{\boldsymbol{y}_p, \boldsymbol{y}_{p+1}, ..., \boldsymbol{y}_{p+n-1}\}$ the (multivariate) subsequence of length $n \leq |\mathcal{T}|$ starting at time point $p \in \mathcal{T}$, where $p \leq |T| - n + 1$. Additionally, each multivariate observation $\boldsymbol{y}_t$ is a realized value of the multivariate random variable (random vector) $\boldsymbol{\mathcal{Y}}_t = (Y_{1t}, ..., Y_{Lt})$. Note that, for each dimension $j \in \{1, ..., L\}$, $Y_j = \{y_{jt}\}_{t \in \mathcal{T}}$ is a univariate time series. In this case, each observation y_{jt} may depend not only on its past values but also on the values of the other time-dependent variables. An example of a multivariate time series is illustrated in Fig. 1.1b, where the observations are obtained through a smart watch that collects 3D accelerometer data from a person that is running [8]. In particular, the data in this example is sampled at 10 Hz for a ten second period, thus generating a time series with 100 multivariate observations.



(a) Monthly totals of international airline passengers ($L = 1$).

(b) 3D accelerometer data of a person that is running ($L = 3$).

Fig. 1.1: Examples of univariate and multivariate time series data.

In either the univariate or the multivariate case, the most basic and common scenario in the literature assumes that the set $\mathcal{T}$ in Definition 1 is a finite and equally-spaced set of time points. Additionally, it is also typically assumed that all variables are observed at all times. For instance, the examples shown in Figure 1.1 represent this common scenario: in both cases, $\mathcal{T}$ is finite ($|\mathcal{T}| = 144$ in Fig. 1.1a, and $|\mathcal{T}| = 100$ in Fig. 1.1b), equally-spaced (the elapsed time between consecutive observations remains regular), and all the variables are always observed. Nevertheless, other scenarios have also been

considered in the literature and in real world problems. We will mention a few that will appear throughout the thesis.

In some scenarios, the variables of a time series may not be observed at all time points (i.e., some y_{jt} values might be missing). Missing data is a common drawback that arises in many real-world scenarios for many reasons. For instance, in control-based applications (e.g., traffic monitoring or industrial processes), missing values usually emerge due to failures in the control equipment or data collection mechanism, interruption of communication between the data collectors and the central management system (e.g., due to power outage), or failures in the hardware or software system [9].

In this case, when some of the time points in $\mathcal{T}$ do not have observations in all of its variables, the time series is said to be *partially-observed*. An example is shown in Fig. 1.2, where the missing values are highlighted with a red shading.

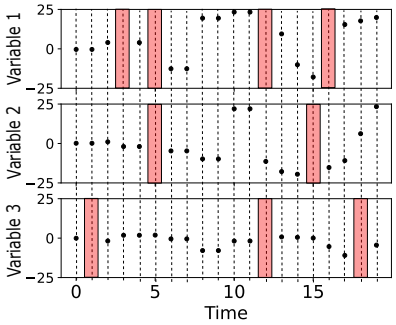
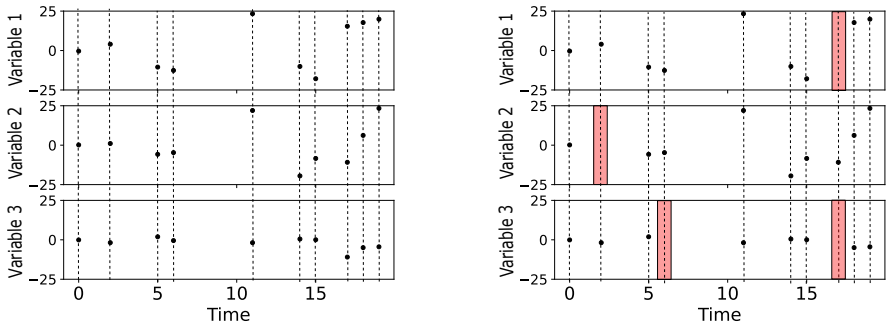


Fig. 1.2: Partially-observed multivariate time series.

In some other cases, the observations of a time series may be collected at unequally-spaced time points. For instance, in the healthcare domain, the data of a patient can only be collected at the time points when the patient visits the doctor [10], which usually occurs on an irregular basis.

When the elapsed time between consecutive time points in $\mathcal{T}$ is not constant, the time series is said to be *irregularly-sampled*. For instance, in Fig. 1.3a, the multivariate time series is irregularly-sampled since the set of time points $\mathcal{T} = \{0, 2, 5, 6, ...\}$ is unequally-spaced. Additionally, this time series is fully-observed because all the variables at time points $\mathcal{T}$ have been observed. However, the multivariate time series in Fig. 1.3b, which collects observations at the same time points $\mathcal{T}$, is irregularly-sampled and partially-observed.

Note that these two types of time series (partially-observed and irregularly-sampled) can have the same representation and can be treated equally, because the differences lie in the semantics or meaning of the missingness.



(a) Fully-observed. (b) Partially-observed.

Fig. 1.3: Irregularly-sampled multivariate time series.

In addition to these scenarios, data can arrive in a continuous manner. This means that the set $\mathcal{T}$ in Definition 1 can also be infinite. In this case, the time series is said to be a *streaming time series*.

In this thesis, and unless otherwise stated, we assume by default that the time series are fully-observed and that the set $\mathcal{T}$ is finite and equally-spaced. However, we will also refer to more complex configurations.

The definitions given so far are focused on a single time series, either univariate or multivariate, but often a set of time series is collected. This leads to the following definition:

Definition 2 (Time series dataset). *A time series dataset* $D = \{Y^1, ..., Y^N\}$ *is a collection of* N *univariate or multivariate time series that gather observations of the same variables.*

As an example, in the healthcare domain, it is common to obtain datasets where each time series gathers vital signs (e.g., heart rate or temperature) and laboratory values (e.g., glucose or platelets) of a given patient [11]. Some well-known public repositories that contain time series datasets are the UCI Machine Learning Repository [12], the UCR/UEA Time Series Archive [13, 8], and the Physionet repository [11].

Despite the different definitions and particularities, in all cases, the key characteristic of time series data is that the observations are ordered over time. In this sense, time series data mining attempts to exploit the temporal dependencies and extract meaningful knowledge from either a single time series or a time series dataset. For this purpose, the research community has focused on solving diverse tasks, some of which are introduced in the following section.

1.1.1 Core tasks of time series data mining

This section provides an overview of the most popular tasks addressed by the time series data mining research community [14]: prediction, classification, clustering, outlier/anomaly detection, segmentation, query by content, and motif discovery.

Prediction (forecasting)

Time series prediction or forecasting is one of the most popular tasks in the field of time series data mining. This task consists of predicting the future (unobserved) values of a given time series by explicitly modeling the time and variable dependencies [15]. An example of this task is shown in Fig. 1.4, where the predictions (the last 25 time points) are shown in orange.

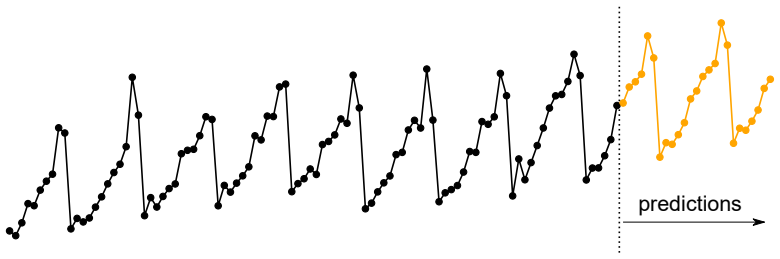


Fig. 1.4: Illustration of the time series prediction task.

Several time series prediction models have been proposed in the literature. Among the most basic and traditional methods we can find the AutoRegressive (AR) model, the Moving Average (MA) model, and the combination between both of them (e.g., ARIMA model) [16]. These techniques are aimed at univariate time series, but an extension to the multivariate context has also been proposed, for instance, the vector AR (VAR) model, the vector MA (VMA) model, and the vector ARIMA (VARIMA) model [17, 18]. In recent years, deep learning prediction algorithms such as Recurrent Neural Networks (RNN) [19] have gained popularity to predict future values in time series data.

Classification

Given a training time series dataset composed of time series and class label pairs, time series classification aims to learn a mapping function between time series and class labels [20, 21]. This function is denominated *classifier*. Once the classifier is learned, it can be used to predict the labels of new unlabeled time series. This task falls within the category of *supervised learning* [22], since the class labels are available and used in the learning process.

An example of this task is illustrated in Fig. 1.5 (adapted from [23]) in which the standard UCR Cylinder-Bell-Funnel (CBF) dataset [13] is used. It should be noted that, in this case, the three classes are mainly characterized by their shape, but the discriminative characteristic may be more complex and visually not apparent in other cases.

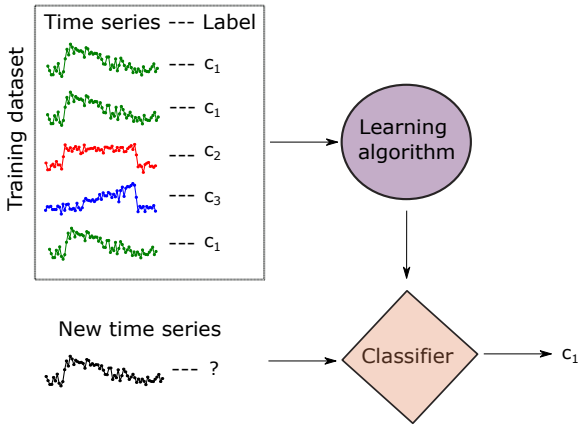


Fig. 1.5: Illustration of the time series classification task.

To address the task of time series classification, many different techniques have been proposed in the literature [20, 24]. Among the most traditional and basic techniques are those that attempt to directly adapt conventional techniques such as the k-Nearest Neighbour (k-NN) classifier [25] to the time series context. Since k-NN is a distance-based classifier, the Euclidean distance (ED) has been commonly used within this classifier [26]. However, the way in which the points in two time series are compared with ED is fixed (see Fig. 1.6a), making the ED very sensitive to noise and misalignments in time. Moreover, the ED cannot handle time series with different lengths. To overcome these limitations, and enable more flexible comparisons, elastic similarity measures have been proposed [27], including the Dynamic Time Warping (DTW) [28]. An example of the difference between these two distances is illustrated in Fig. 1.6. The combination of the k-NN classifier and the DTW distance is commonly used for the time series classification task [26].

Some other popular classification methods include Time Series Forest (TSF) [29], Fast Shapelets [30], Bag-of-SFA-Symbols (BOSS) [31], and Random Interval Spectral Ensemble (RISE) [32]. A detailed categorization and analysis of the different techniques that have been proposed in the literature can be found in [20], for univariate time series, and [21], for multivariate time series.

(a) Euclidean distance (ED). (b) Dynamic Time Warping (DTW).

Fig. 1.6: Differences between distances with fixed and flexible mapping.

Clustering

Given a time series dataset, clustering is the task of categorizing the time series into groups, called *clusters*, based on the similarity between them [33, 34]. In other words, the most similar time series are grouped into the same group, and the groups should be very dissimilar from each other. This task falls under the category of *unsupervised learning* [35] since only the input data is available (there is no output information), and the groups are not predefined. That is, the underlying structure of the data is extracted from the data itself. The intuition of clustering is depicted in Fig. 1.7, where the initial set of time series is grouped into three clusters.

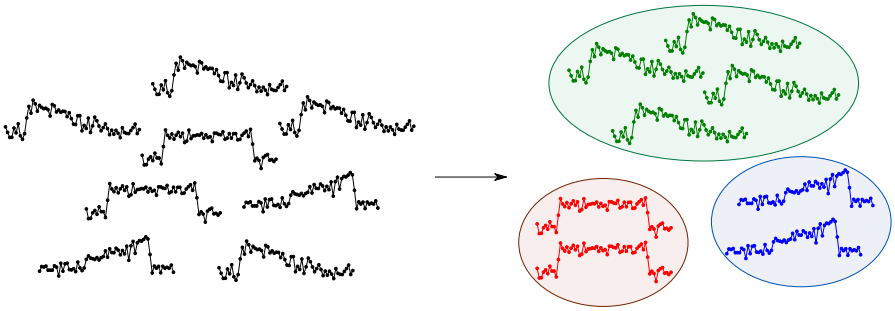


Fig. 1.7: Illustration of the time series clustering task.

Similar to the techniques employed in the time series classification task, many of the techniques for time series clustering focus on modifying existing clustering algorithms for non-temporal data so that they can be used with time series [33]. In this sense, techniques usually rely on traditional distance-based clustering methods and replace the distance or similarity measure used for non-temporal data with an appropriate one for time series such as DTW [36, 37, 38]. Another common option used in the literature is to transform the time series data into a feature vector of lower dimension so that conventional clustering algorithms can then be applied [33].

Outlier/anomaly detection

Given a time series, the outlier/anomaly detection task seeks to find the observations or subsequences that do not follow the expected behavior [39, 40]. As an example, the time series shown in Fig. 1.8, which monitors insect feeding using an Electrical Penetration Graph (EPG) apparatus [41, 42], contains an anomalous subsequence shown in orange.

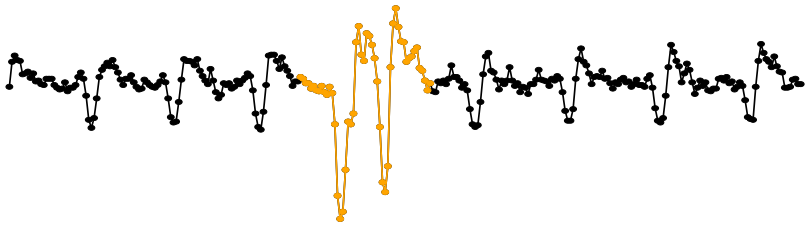


Fig. 1.8: Illustration of the outlier/anomaly detection task in a time series.

Analogously, given a time series dataset, the aim of this task may be to find the most unusual time series. For example, Fig. 1.9 illustrates this task in which the time series shown in orange colour would be the anomalous time series in the given time series dataset.

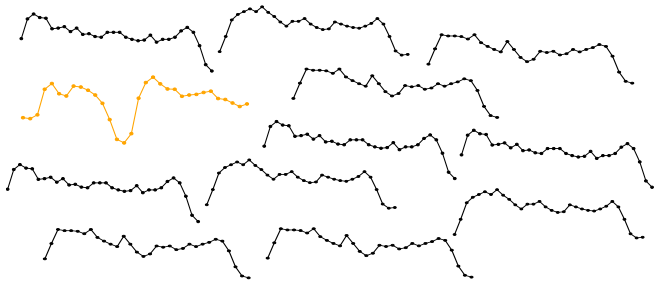


Fig. 1.9: Illustration of the outlier/anomaly detection task in a time series dataset.

It should be mentioned that the outlier detection problem has been assumed to be unsupervised by default [43, 39], but some techniques in the literature have also tackled this task in a supervised way. In particular, the supervised outlier detection problem is often approached as a binary classification problem, but with the limitation that the class labels are highly imbalanced, the anomaly class being in minority. However, addressing this

task from a supervised perspective is often challenging because of the difficulty involved in collecting labels (of high-quality) in many scenarios. In this context, the outlier/anomaly detection task has been addressed as a one-class classification problem [44, 45], where the classifier is learned based on a dataset labeled from only one class. In this case, the class used for learning is the normal class, while the samples that fall outside this class are considered to be anomalous. Less commonly, semi-supervised techniques have also been used [39, 46]. These techniques combine a small amount of labeled data with a large amount of unlabeled data during training.

As stated in the introduction, this dissertation is mainly focused on the outlier detection task but, in particular, we focus on the context of unsupervised learning. We will provide additional details about this task in Section 1.2.

Segmentation

Given a time series, the goal of segmentation is to reduce the dimensionality of the given time series and create an accurate approximation of it [47]. An example of this task is illustrated in Fig. 1.10, where the original time series represented in black is reduced to only seven points (shown by red crosses). This task is interesting because often the exact values of the time series are not relevant, but rather the trends, shapes, and patterns contained within the time series [48], which are best captured by the reduced representation. Also, segmentation helps to reduce the required memory storage, especially useful when the time series are very long.

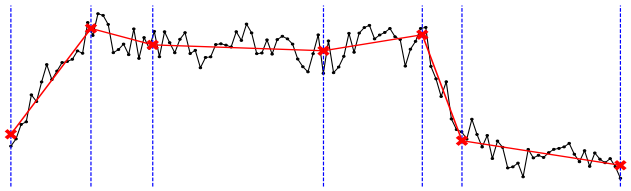


Fig. 1.10: Illustration of the time series segmentation task.

A widely known segmentation technique is the Piecewise Linear Approximation (PLA) [49], which divides a time series of length T into $K << T$ segments (not necessarily of the same length) and approximates each segment with a straight line. Another common option is to represent each segment by the average value of the set of data points within the corresponding segment. For example, the Piecewise Aggregate Approximation (PAA) [50] and the Adaptive Piecewise Constant Approximation (APCA) [51] techniques are within this group. Finally, some other more sophisticated techniques that con-

vert the time series into a symbolic form to reduce its dimensionality have also been proposed (e.g., the Symbolic Aggregate Approximation (SAX) [52]).

Query by content (indexing)

Given a query time series and a reference time series, the aim of this task is to identify the subsequences in the reference time series that are most similar to the query [53]. Similarly, this task can also be performed on a time series dataset, aiming to find the set of individual time series in the dataset that are most similar to the query. An illustration of the query by content task is shown in Fig. 1.11.

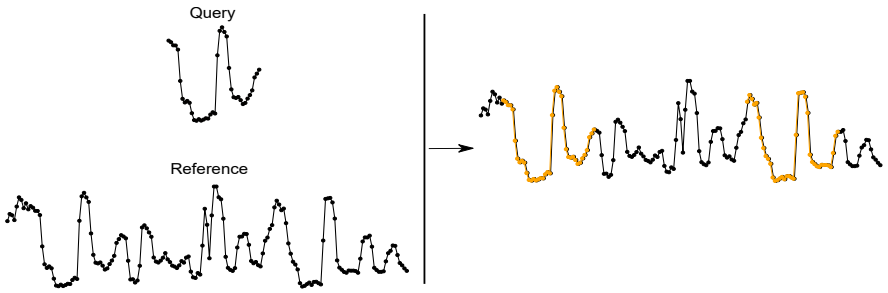


Fig. 1.11: Illustration of the query by content task.

This task has been traditionally solved by the ϵ-range query and the k-NN query approaches [14]. While the ϵ-range query returns the set of time series or subsequences that are within a distance ϵ of the query time series, the k-NN query returns the k closest time series to the query. In some contexts, comparing the query time series to all the sequences in the dataset (i.e., the brute force algorithm) can be computationally expensive, and thus, in these cases, techniques usually propose reducing the dimensionality of the time series and then indexing the transformed data [51, 53]. Indeed, in general, the techniques that address the query by content task rely on the following three components [54]: a representation of the time series, a distance measure between pairs of time series, and an efficient search mechanism to find the matches.

Motif discovery

The last popular time series data mining task that we will mention is motif discovery. Given a time series, this task aims to find subsequences that occur repeatedly in the original time series. For example, in Fig. 1.12, two repeated patterns (i.e., motifs) are highlighted. In particular, the purpose of motif discovery is usually to find the k-frequent motifs in the time series, that is, to identify the k most frequently recurring subsequences.

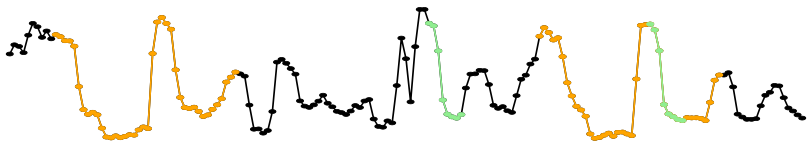


Fig. 1.12: Illustration of the motif discovery task.

This task requires to compare different subsequences by measuring the similarity between them. Most commonly, the Euclidean distance has been used [55].

Note that there may be several motifs within a time series, the motifs may be of different lengths, and also, they may overlap, as shown in Fig. 1.12. In this sense, the brute force algorithm is usually computationally expensive, and thus, many authors have focused on developing efficient motif discovery algorithms [55, 56, 57]. For example, among the most popular techniques is the Matrix Profile, which has been proposed to allow the efficient exact computation of the top-k motifs in a time series [58, 59].

1.2 Time series outlier/anomaly detection

Outlier or anomaly detection is a field that has been studied for many years due to its relevance in several application domains such as fraud detection [60], intrusion detection [61], or fault diagnosis [62]. A widely used definition for the concept *outlier* has been provided by [63]:

> "An observation which deviates so much from other observations as to arouse suspicions that it was generated by a different mechanism."

Therefore, outliers can be thought of as observations that do not follow the expected behavior.

Based on this intuition, the concept *outlier* has been denominated in several different ways in the literature, and, to this day, there is still no consensus on the terms used [64]; for example, outlier observations are often referred to as anomalies, discordant observations, discords, exceptions, aberrations, surprises, peculiarities or contaminants.

When dealing with time series data, the analysis of outliers examines anomalous behaviors across time [39]. Moreover, depending on the temporal context, outliers can be of two types. On the one hand, the outliers may behave unusually compared to all the values in the time series. This type of outliers are said to be *global* outliers. On the other hand, the outliers may behave unexpectedly compared to their neighboring points even though globally they are not rare observations. In this case, outliers are said to be *local* outliers. Note that all global outliers are also local, but not all local outliers

are global. More details and examples are shown in Chapter 2 (see Fig. 2.2 and Fig. 2.3).

Finally, it is worth mentioning that outliers have an underlying meaning. As shown in Fig. 1.13, the meaning of the outliers in time series can be categorized into two main groups, and the semantic distinction between them is mainly based on the interest of the analyst or the particular scenario considered. Outlier observations have been widely related to noise, erroneous, or unwanted data, which by themselves are not interesting to the analyst [65]. In these cases, outliers should be deleted or corrected to improve the data quality and generate a cleaner dataset that can be used by other data mining algorithms. For example, sensor transmission errors are eliminated to obtain more accurate predictions because the principal aim is to make predictions. Nevertheless, in recent years and, especially in the area of time series data, many researchers have aimed to detect and analyze unusual but interesting phenomena. In this case, outliers translate to significant information. Fraud detection is an example of this because the main objective is to detect and analyze the outlier itself as it may reflect unauthorized use of a credit card. Another example is an outlier in the industrial environment, where anomalous observations in a manufacturing machine can mean a failure in some component of the machine. Moreover, in public health data, outliers may represent symptoms of a new disease. These observations are often referred to as anomalies [65]. It should be mentioned that, in this thesis, the terms outlier and anomaly will be used interchangeably.

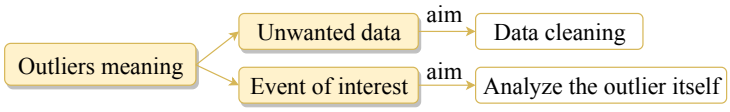


Fig. 1.13: Meaning of the outliers/anomalies in time series data depending on the aim of the analyst.

1.3 Objectives and challenges

Due to the interest of the research community in the field of time series data mining, the literature contains a large number of publications and solutions to different problems. In particular, and as mentioned in the previous sections, outlier detection is a topic that has gained much attention for the purpose of improving data quality or detecting interesting phenomena. Many techniques have been proposed to address the outlier detection problem in time series from an unsupervised perspective, but the existing techniques are not presented in a structured and comprehensive way in the literature. This limitation leads us to the first objective of the thesis:

Objective 1. *Provide a structured and comprehensive state-of-the-art on unsupervised outlier detection techniques in the context of time series data.*

In other words, in this dissertation, we provide a literature review on unsupervised outlier/anomaly detection techniques in time series, drawing a general idea of the current state-of-the-art methods and highlighting the relevant characteristics of each technique. Moreover, a taxonomy is presented based on the main aspects that characterize an outlier detection technique, namely, the input data type (univariate or multivariate time series), the outlier type (point, subsequence, or whole time series), and the nature of the method (univariate or multivariate). In addition, we also analyze some other more specific aspects, such as whether the techniques take into account the temporal correlation between the observations or not.

This detailed analysis has shown that some techniques ignore the temporal correlation between consecutive observations, but, in most cases, this should not be omitted since it is a relevant characteristic of time series. Also, the review has revealed that the detection of whole time series outliers is a task that has almost not been treated in the literature. However, this problem arises naturally in many application domains, such as water leak detection [66, 67].

Water distribution companies are usually interested in detecting the days in which a leak has occurred in order to repair it as soon as possible and, in this way, avoid further costs and damages. For this purpose, very simple methods based on thresholds are commonly applied to time series that represent water flow values over a day. However, these methods provide many false leak alarms. As such, the water companies have a great interest in reducing the number of false positives while maintaining a high leak detection rate. All this motivates the second objective of the thesis:

Objective 2. *Propose a novel whole time series anomaly detection technique for water leak detection.*

In particular, we solve the problem from an unsupervised perspective, that is, we do not use any leak/no leak labels in the learning process. To be more precise, we propose a method based on self-supervised learning, where we use classification techniques with labels that have been generated artificially and specifically for the problem at hand. Additionally, the method treats the data as time series, thus considering the temporal correlation between the measurements. The proposed technique is able to provide a low false positive rate, while maintaining a high leak detection rate.

So far, we have assumed that the time series are regularly-sampled and fully-observed (they have no missing values). Indeed, many traditional machine learning techniques for time series assume these conditions. However, for reasons such as failures in data collection mechanisms, time series often contain missing values and are thus incomplete. Since the presence of missing values hinders an advanced analysis of the time series (e.g., outlier/anomaly

detection), the treatment of missing values and their imputation is an important task to address. In particular, when the missing rate is very high, imputing all the missing values may not be the most appropriate solution since this would introduce many errors in the time series. Additionally, this can significantly affect the quality of the data and the results of downstream tasks. This issue has motivated our third contribution:

Objective 3. *Process multivariate time series with missing values by selective imputation.*

In particular, we propose a method that selectively imputes a multivariate time series dataset, avoiding to make unnecessary imputations. In other words, this method identifies a subset of missing points to impute in a multivariate time series dataset. This selection is based on both reducing the uncertainty of the imputations and representing the original time series as accurately as possible, and it will result in shorter and simpler time series. Moreover, the proposed method can be applied with any downstream task. In this thesis, we analyze its performance when addressing two popular time series data mining tasks: multivariate time series classification and whole time series anomaly detection.

1.4 Outline of the Dissertation

This dissertation is divided into three main parts. Part I collects the methodological contributions in the field of outlier/anomaly detection in time series data. In particular, Chapter 2 provides a literature review on outlier/anomaly detection techniques in time series data, and Chapter 3 presents a novel whole time series anomaly detection technique for water leak detection based on self-supervised learning. Then, Part II focuses on the contributions related to the treatment of time series with missing values and demonstrates its applicability in the field of outlier/anomaly detection. Specifically, Chapter 4 includes the details of the proposed methodology that selectively imputes a multivariate time series dataset with missing values. Finally, in Part III, the general conclusions drawn from this thesis are presented, as well as possible future lines of research. The main achievements are also mentioned in this last part.

Part I

Contributions to outlier/anomaly detection in time series data

2

A review on outlier/anomaly detection in time series data

This chapter deals with the first problem described in Section 1.3. As such, a structured and comprehensive state-of-the-art on unsupervised outlier detection techniques in the context of time series data is provided.

2.1 Introduction

As mentioned in the previous section, outlier detection has become a field of interest for many researchers and practitioners and is now one of the main tasks of time series data mining. In the first study on this topic, which was conducted by [68], two types of outliers in univariate time series were defined: type I, which affects a single observation; and type II, which affects both a particular observation and the subsequent observations. This work was first extended to four outlier types [69], and then to the case of multivariate time series [70]. Since then, many definitions of the term *outlier* and numerous detection methods have been proposed in the literature.

The purpose of this chapter is to present a structured and comprehensive state-of-the-art on outlier detection techniques in time series data and attempt to extract the essence of the concept *outlier*, focusing on the detection algorithms given by different authors. Despite the broad terminology that is used to refer to outliers, this review focuses on the identification of outliers in the unsupervised framework, regardless of the term used in the original papers.

Although a number of surveys on outlier detection methods have been presented in the literature [71, 72, 73, 74, 65, 75], very few focus on temporal data, including time series [76].

In this sense, the main contributions of this review are fourfold:

- We give a comprehensive review that focuses only on time series data. Thus, we provide an in-depth analysis, examining techniques that have hardly been explored in the literature so far. To the best of our knowledge,

none of the existing surveys focus exclusively on unsupervised outlier detection in time series, and so, they do not provide enough details of this specific problem and the methods published to solve it.
- We propose a novel taxonomy for outlier detection methods in time series data by extracting the most relevant characteristics of the existing methodologies. This taxonomy provides a global understanding of the outliers and their detection in time series, and helps to choose the type of technique that best adapts to a given problem. We also provide details about other features that characterize each technique. As far as we know, the existing surveys do not propose a complete taxonomy and do not extract the characteristics of each method.
- We provide the publicly available software related to the analyzed methods. This is an important point nowadays since it allows us to reproduce the methods. The existing related surveys do not present this information.
- We identify some future research directions on outlier detection in time series.

The rest of this chapter is organized as follows. In Section 2.1.1, the methodology followed in this study is described. In Section 2.2, a taxonomy for the classification of outlier detection techniques in time series data is proposed. Section 2.3, Section 2.4 and Section 2.5 present different techniques used for point, subsequence, and time series outlier detection, respectively. The techniques are classified according to the taxonomy proposed in Section 2.2, and the intuition of the concept *outlier* on which the methods are based is provided. In Section 2.6, the publicly available software for some of the considered outlier detection methods is presented. Finally, Section 2.7 contains the concluding remarks and outlines some areas for further research.

2.1.1 Methodology

This study has been oriented and organized with the intention to answer the following research questions:

(RQ1) What are the most important characteristics that define each outlier detection method? Based on this, how could the existing techniques be taxonomized?
(RQ2) How do existing techniques detect point, subsequence and time series outliers in time series data? Which are the main differences between these methods?
(RQ3) Are there publicly available software packages for outlier detection in time series? What type of methods do they implement?

The methodology used to provide answers to the proposed research questions is an ad-hoc methodology that consists of 3 modules: Database Selection, Survey Search, and Literature Search.

A. Database Selection

The databases of scientific research used to do the literature search in this review are the following well-known repositories: Google Scholar, IEEE Xplore, ACM Digital Library, DBLP, Scopus, and ScienceDirect.

B. Survey Search

Since this study is a literature review, we searched for related reviews that have been previously published. The keywords used are "outlier detection", "anomaly detection", "survey", "review", and "time series".

C. Literature Search

To search for articles that propose outlier detection methods in time series, we mainly used the following keywords between 2000 and 2019: "time series", "outlier detection", and "anomaly detection". Other useful keywords have been "univariate", "multivariate", "time series database", and "subsequence". We also analyzed additional papers that were referenced within papers already identified. Finally, we manually filtered the results by excluding irrelevant papers which have not been published in high-quality forums and which apply a previously developed method to a particular case rather than proposing a new methodology.

2.2 A taxonomy of outlier detection techniques in the time series context

Outlier detection techniques in time series data vary depending on the input data type, the outlier type, and the nature of the method. Therefore, a comprehensive taxonomy that encompasses these three aspects is proposed in this section. Fig. 2.1 depicts an overview of the resulting taxonomy, and each axis is described in detail below.

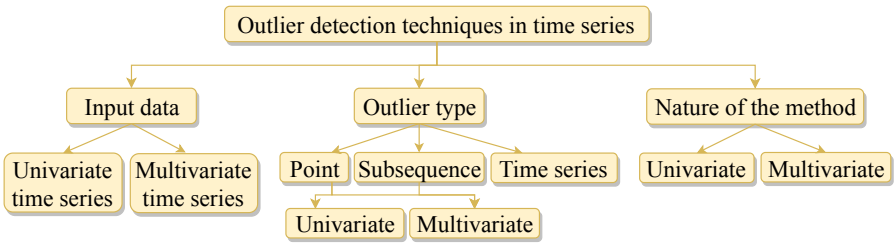


Fig. 2.1: Proposed taxonomy of outlier detection techniques in time series data.

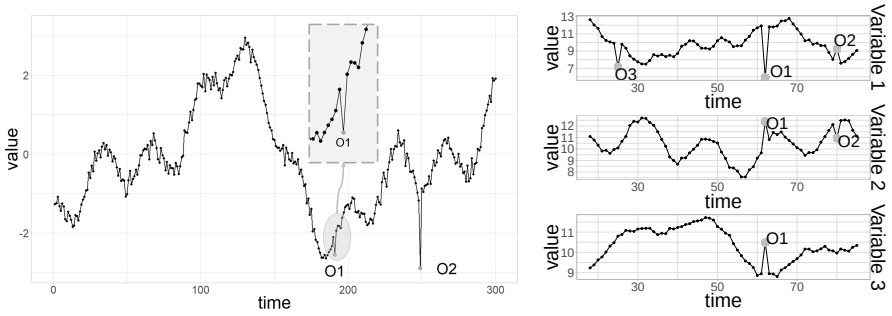
2.2.1 Input data

The first axis represents the type of input data that the detection method is able to deal with (i.e., a *univariate* or a *multivariate* time series). For the formal definition of these concepts, the reader should refer to the Definition 1 given in Section 1.1.

2.2.2 Outlier type

The second axis describes the outlier type that the method aims to detect (i.e., a point, a subsequence, or a time series).

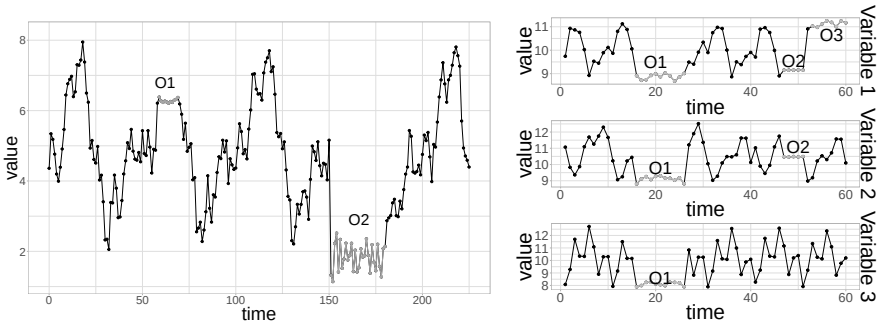
- *Point outliers.* A point outlier is a datum that behaves unusually in a specific time instant when compared either to the other values in the time series (global outlier) or to its neighboring points (local outlier). Point outliers can be univariate or multivariate depending on whether they affect one or more time-dependent variables, respectively. For example, Fig. 2.2a contains two univariate point outliers, O1 and O2, whereas the multivariate time series composed of three variables in Fig. 2.2b has both univariate (O3) and multivariate (O1 and O2) point outliers.



(a) Univariate time series. (b) Multivariate time series.

Fig. 2.2: Point outliers in time series data.

- *Subsequence outliers.* This term refers to consecutive points in time whose joint behavior is unusual, although each observation individually is not necessarily a point outlier. Subsequence outliers can also be global or local and can affect one (univariate subsequence outlier) or more (multivariate subsequence outlier) time-dependent variables. Fig. 2.3 provides an example of univariate (O1 and O2 in Fig. 2.3a, and O3 in Fig. 2.3b) and multivariate (O1 and O2 in Fig. 2.3b) subsequence outliers. Note that the latter do not necessarily affect all the variables (e.g., O2 in Fig. 2.3b).



(a) Univariate time series. (b) Multivariate time series.

Fig. 2.3: Subsequence outliers in time series data.

- *Outlier time series.* Entire or whole time series can also be outliers, but they can only be detected when the input data is a multivariate time series. Fig. 2.4 depicts an example of an outlier time series, *Variable 4*, in a multivariate time series composed of four variables. The behavior of *Variable 4* significantly differs from the rest.

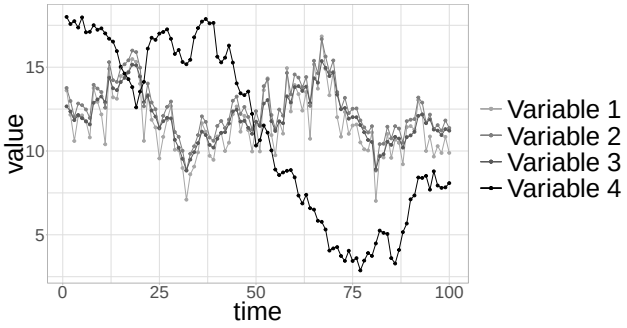


Fig. 2.4: Outlier time series (*Variable 4*) in a multivariate time series.

Observe that this axis is closely related to the input data type. If the method only allows univariate time series as input, then no multivariate point or subsequence outliers can be identified. In addition, outlier time series can only be found in multivariate time series. Finally, it should be noted that the outliers depend on the context. Thus, if the detection method uses the entire time series as contextual information, then the detected outliers are global. Otherwise, if the method only uses a segment of the series (a time window), then the detected outliers are local because they are outliers within their neighborhood. Global outliers are also local, but not all local outliers are

global. In other words, some local outliers may seem normal if all of the time series is observed but may be anomalous if we focus only on their neighborhood (e.g., O1 in Fig. 2.2a).

2.2.3 Nature of the method

The third axis analyzes the nature of the detection method employed (i.e., if the detection method is *univariate* or *multivariate*). A univariate detection method only considers a single time-dependent variable, whereas a multivariate detection method is able to simultaneously work with more than one time-dependent variable. Note that the detection method can be univariate, even if the input data is a multivariate time series, because an individual analysis can be performed on each time-dependent variable without considering the dependencies that may exist between the variables. In contrast, a multivariate technique cannot be used if the input data is a univariate time series. Thus, this axis will only be mentioned for multivariate time series data.

2.3 Point outliers

Point outlier detection is the most common outlier detection task in the area of time series. This section presents the techniques used to detect this type of outlier, in both univariate (Section 2.3.1) and multivariate (Section 2.3.2) time series data.

Specifically, as shown in Fig. 2.5, two key characteristics of these methods will be highlighted throughout their presentation. Concerning the first characteristic, or the treatment of temporality, some methods consider the inherent temporal order of the observations, while others completely ignore this information. The main difference between the methods that include temporal information and those that do not is that the latter methods produce the same results, even if they are applied to a shuffled version of the series. Within the methods that use temporality, a subgroup of methods use time windows. Consequently, the same results are obtained when shuffling the observations within the window, but not when shuffling the whole time series.

Fig. 2.5: Characteristics related to point outlier detection problems.

In relation to the second characteristic (see Fig. 2.5), some techniques are able to detect outliers in streaming time series by determining whether or not a new incoming datum is an outlier as soon as it arrives, without having

to wait for more data. Within this group, some methods use a fixed model throughout the stream evolution, whereas others update the models used for detection with the new information received—either by retraining the whole model or by learning in an incremental manner. We consider that a technique does not apply to a streaming time series (i.e., non-streaming) if it is unable to make a decision at the arrival of the new datum.

Most of the analyzed point outlier detection techniques can be applied in a streaming context and they take the temporality of the data into account, either by considering the full time series as an ordered sequence or with the use of time windows. Therefore, we will only make reference to this axis for methods that cannot be applied in a streaming environment or which completely ignore the temporal information in the data. Finally, even though many techniques can theoretically deal with streaming time series, very few are able to adapt incrementally to the evolution of the stream. Consequently, we will also highlight these techniques.

2.3.1 Univariate time series

The techniques that will be discussed in this section intend to detect point outliers in a univariate time series and are organized based on the diagram shown in Fig. 2.6. Since a single time-dependent variable is considered, recall that these outliers are univariate points and that only univariate detection techniques can be used for their detection.

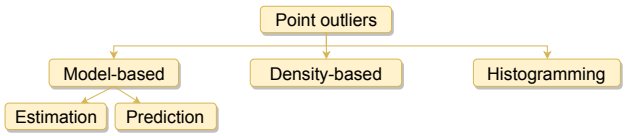


Fig. 2.6: Types of methods for detecting point outliers in univariate time series.

The most popular and intuitive definition for the concept of *point outlier* is a point that significantly deviates from its expected value. Therefore, given a univariate time series, a point at time t can be declared an outlier if the distance to its expected value is higher than a predefined threshold τ:

$$|y_t - \hat{y}_t| > \tau \tag{2.1}$$

where y_t is the observed data point, and $\hat{y}_t$ is its expected value. This problem is graphically depicted in Fig. 2.7, where the observed values within the shadowed area are at most at distance τ from their expected values.

The outlier detection methods based on the strategy described in equation (2.1) are denominated *model-based* methods in this dissertation and are

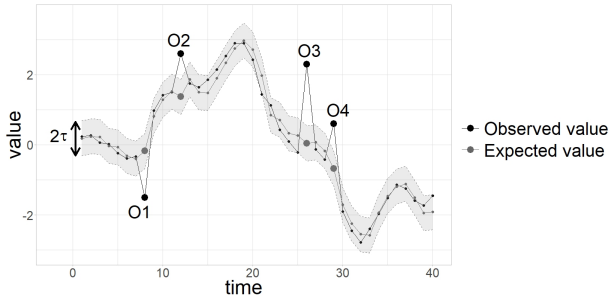


Fig. 2.7: Point outlier detection in univariate time series based on the comparison of expected and observed values.

the most common approaches in the literature. Even though each technique computes the expected value $\hat{y}_t$ and the threshold τ differently, they are all based on fitting a model (either explicitly or implicitly). As shown in Table 2.1, if $\hat{y}_t$ is obtained using previous and subsequent observations to y_t (past, current, and future data), then the technique is within the *estimation model-based* methods. In contrast, if $\hat{y}_t$ is obtained relying only on previous observations to y_t (past data), then the technique is within the *prediction model-based* methods. In practice, the main difference between using estimation or prediction methods is that techniques within this latter category can be used in streaming time series because they can determine whether or not a new datum is an outlier as soon as it arrives. In the case of estimation methods, this can only be done if, besides some points preceding y_t, only the current point y_t is used to compute the estimated value ($k_2 = 0$).

	Data used	$\longrightarrow$	**Expected value**	$\longrightarrow$	**Point outliers**
Estimation models	$\{y_{t-k_1}, ..., y_t, ..., y_{t+k_2}\}$	$\longrightarrow$	$\hat{y}_t$	$\longrightarrow$	$\lvert y_t - \hat{y}_t \rvert > \tau$
Prediction models	$\{y_{t-k}, ..., y_{t-1}\}$	$\longrightarrow$	$\hat{y}_t$		

Table 2.1: Data used in model-based techniques in univariate time series, for $k \geq 1$, and $k_1, k_2 \geq 0$ such that $k_1 + k_2 > 0$.

The most simple *estimation models* are based on constant or piecewise constant models, where basic statistics such as the median [77] or the Median Absolute Deviation (MAD) [78] are used to obtain $\hat{y}_t$. These statistics are calculated using the full series or by grouping the data in equal-length segments and cannot be applied in streaming when future data is needed ($k_2 > 0$). A more sophisticated approach is to utilize unequal-length segments obtained with some segmentation technique. [79] use the mean of each segment to determine the expected value of the points within that segment, and an adaptive

threshold $\tau_i = \alpha\sigma_i$, where α is a fixed value and σ_i the standard deviation of segment i.

Other estimation-based techniques intend to identify data points that are unlikely if a certain fitted model or distribution is assumed to have generated the data. For instance, some authors model the structure of the data using smoothing methods such as B-splines or kernels [80, 81, 82] or variants of the Exponentially Weighted Moving Average (EWMA) method [83], [84] and [85] use slope constraints, [78] assume that the data without outliers are approximately normal, and [86] use Gaussian Mixture Models (GMM).

Once a model or distribution is assumed and fitted, [80, 83] and [86] use equation (2.1) directly to decide whether a point is an outlier or not. More elaborately, [81] use extreme value theory to compute a confidence limit for the high values of the kernel smoothing residuals ($|y_t - \hat{y}_t|$) and flag point outliers as those exceeding that limit. Within the estimation-methods that use slope constraints, [84] establish a maximum and a minimum possible slope between consecutive values, whereas [85] model the change in slopes before and after time t, assuming that the slopes should not change significantly at a time point. The Extreme Studentized Deviate (ESD) test has also been used to make the decision [78, 82]: the null hypothesis considered is that there are no outliers, whereas the alternative is that there are up to k. Regardless of the temporal correlation, the algorithm computes k test statistics iteratively to detect k point outliers. At each iteration, it removes the most outlying observation (i.e., the furthest from the mean value). Finally, [82] also propose defining the threshold in equation (2.1) using Chebyshev's inequality or the quantiles of normal distribution, after dividing the residuals from kernel regression into homogeneous segments using change point analysis.

Some other univariate outlier detection methods analyze all of the residuals obtained from different models to identify the outliers. For example, [87] use the STL decomposition, and [88, 89, 90] use ARIMA models with exogenous inputs, linear regression or Artificial Neural Networks (ANNs). Although most of these models can also be used in prediction, in this case, the outliers are detected in the residual set using past and future data. Specifically, once the selected model is learned, hypothesis testing is applied over the residuals to detect the outliers. In [87], the ESD test is applied but using the median and MAD instead of the mean and standard deviation, for robustness. In [88, 89, 90], assuming that the underlying distribution of the residuals is known, the minimum and maximum values are examined simultaneously at each iteration of the algorithm. The hypothesis to be tested is whether an extremum is an outlier (alternative hypothesis) or not (null hypothesis). The detected outliers are corrected, and the process is repeated until no more outliers are detected.

In contrast to estimation models, techniques based on *prediction models* fit a model to the time series and obtain $\hat{y}_t$ using only past data; that is, without using the current point y_t or any posterior observations. Points that are very different from their predicted values are identified as outliers. Recall that all of the techniques within this category can deal with streaming time series.

Within the prediction-based methods, some use a fixed model and thus are not able to adapt to the changes that occur in the data over time. For example, the DeepAnT outlier detection approach presented by [91] applies a fixed Convolutional Neural Networks (CNNs) to predict values in the future. Other methods use an autoregressive model [92] or an ARIMA model [93], which obtain confidence intervals for the predictions instead of only point estimates. Consequently, these methods implicitly define the value of τ.

Other techniques adapt to the evolution of the time series by retraining the model. As the most basic approach, [77] describe a method that predicts the value $\hat{y}_t$ with the median of its past data. More elaborately, [94] fit an ARIMA model within a sliding window to compute the prediction interval, so the parameters are refitted each time that the window moves a step forward.

Extreme value theory has also been employed to detect point outliers in streaming univariate series, using past data and retraining. Given a fixed risk q, [61] use this theory to obtain a threshold value $z_{q,t}$ that adapts itself to the evolution of the data such that $P(\mathcal{Y}_t > z_{q,t}) < q$, for any $t \geq 0$, assuming that the extreme values follow a Generalized Pareto Distribution (GPD). Incoming data is used to both detect anomalies ($\mathcal{Y}_t > z_{q,t}$) and refine $z_{q,t}$. In particular, the authors propose two algorithms: SPOT, for data following any stationary distribution; and DSPOT, for data that can be subject to concept drift [95, 96].

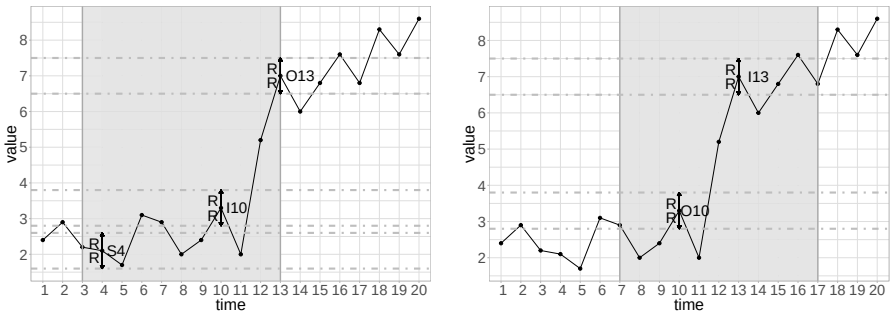
Some of these prediction-based methods retrain the underlying model periodically, or each time a new point arrives. Therefore, they can adapt to the evolution of the data. However, none of them applies incremental model learning approaches, where the model is not rebuilt from scratch each time but is updated incrementally using only the new information received. This avoids the cost associated with training the models more than once and permits a more gradual adaptation to changes that can occur in the data, which is of special interest in streaming contexts [97]. In this sense, and in contrast to the previous approaches, [98, 99] suggest modeling a univariate time series stream in an incremental fashion. This method uses Student-t processes to compute the prediction interval and updates the covariance matrix with the newly arrived data point. [100] use the Hierarchical Temporal Memory (HTM) network, which is also a prediction model-based technique that updates incrementally as new observations arrive.

The techniques mentioned so far are based on equation (2.1). However, not all the existing point outlier detection methods rely on that idea, such as the *density-based* methods, which belong to the second category depicted in Fig. 2.6. Techniques within this group consider that points with less than τ neighbors are outliers; i.e., when less than τ objects lie within distance R from those points. This could be denoted as

$$y_t \text{ is an outlier} \iff |\{y \in Y | d(y, y_t) \leq R\}| < \tau \tag{2.2}$$

where d is most commonly the Euclidean distance, y_t is the data point at time step t to be analyzed, Y is the set of data points (time series), and $R \in \mathbb{R}^+$.

The detection of density-based outliers has been widely handled in non-temporal data, but the concept of neighborhood is more complex in time series because the data are ordered. To take temporality into account, [101, 102] and [103] apply this method within a sliding window, which allows us to determine whether or not a new value of a streaming time series is an outlier upon arrival. An illustration of density-based outliers is provided in Fig. 2.8 at two different time steps with $R = 0.5$, $\tau = 3$, and a sliding window of length 11. When using sliding windows, a point can be an outlier for a window (e.g., O13 at $t = 13$) but not for another (e.g., I13 at $t = 17$). However, if a data point has at least τ succeeding neighbors within a window, then it cannot be an outlier for any future evolution (e.g., S4 at $t = 13$).



(a) $t = 13$ (b) $t = 17$

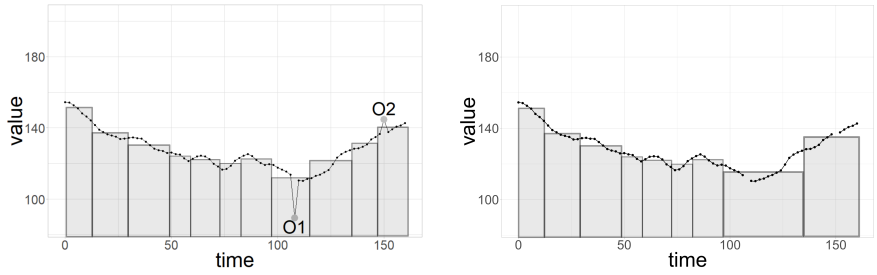
Fig. 2.8: Density-based outliers within a sliding window of length 11 at time step t.

Histogramming is the last group analyzed in this section. This type of method is based on detecting the points whose removal from the univariate time series results in a histogram representation with lower error than the original, even after the number of buckets has been reduced to account for the separate storage of these points (see Fig. 2.9). The histogram is built by computing the average of the values within each bucket, in which the order of the observations is preserved. Then, given a sequence of points Y and a number of buckets B, $D \subset Y$ is a deviant set if

$$E_Y(H_B^*) > E_{Y-D}(H_{B-|D|}^*) \tag{2.3}$$

where H_B^* is the optimal histogram (histogram with lowest approximation error) on Y with B buckets, $E_Y(\cdot)$ is the total error in the approximation and $H_{B-|D|}^*$ is the optimal histogram on $Y - D$ with $B - |D|$ buckets. [104] introduced the term *deviant* to refer to these point outliers. They proposed a dynamic programming mechanism to produce a histogram consisting of $B - |D|$ buckets and $|D|$ deviants, minimizing its total error. Some years later

[105] observed that for any bucket, the optimal set of $k = |D|$ deviants always consists of the l highest and remaining $k - l$ lowest values within the bucket, for some $l \leq k$. Moreover, they presented not only an optimal algorithm for non-streaming time series but also a closely approximate algorithm and a heuristic approach for the streaming case.



(a) Optimal histogram with eleven buckets.

(b) Optimal histogram with nine buckets and O1 and O2 removed.

Fig. 2.9: Example of a deviant set $D = \{O1, O2\}$ in a univariate time series.

To conclude, the techniques that detect point outliers in univariate time series can be grouped mainly in three categories based on the intuition they have of the concept of outlier, and basically what differentiates one technique from another are the characteristics gathered in Table 2.2. In particular, we consider that a method is iterative if it is repeatedly applied on the same set to find the outliers. Concerning the parametricity of the methods, we consider that parametric methods assume that the underlying distribution of the phenomenon under analysis belongs to a parametric family, and estimate its parameters (a fixed number) from the given data. Contrarily, in non-parametric techniques, the model or the family of distributions is not defined a priori but instead it is determined from the given data and has a flexible number of parameters. Also, we refer to semi-parametric methods as those that combine both parametric and non-parametric approaches.

In general, the authors use the term *outlier*, and the described techniques consider temporality and can be applied in a streaming context. Also, the few iterative methods are related to detecting unwanted data and improving the quality of the time series. Most of the model-based techniques are parametric or semi-parametric. Semi-parametric techniques usually assume a distribution over the residuals when defining a threshold value, even if the residuals are obtained with a non-parametric approach.

Finally, caution must be taken with the estimation methods that can theoretically be applied in a streaming time series; that is, those that do not use subsequent observations to the last arrived data point y_t ($k_2 = 0$ in Table 2.1). Although, it may in theory be possible to apply these techniques in streaming

Table 2.2: Summary of point outlier detection techniques in univariate time series.

Paper	Technique	Iterative	Temporality	Streaming	Meaning		Parametricity			Term
					I	U	P	SP	NP	
[104]	Histogram	✗	✓	✓	✓				✓	D
[105]	Histogram	✗	✓	✓*	✓				✓	D
[77]	Estimation	✗	W	✗		✓			✓	O
	Prediction	✗	W	✓		✓			✓	O
[101, 102]	Density	✗	W	✓*	✓				✓	O
[80]	Estimation	✗	✓	✓		✓		✓		O/C
[92]	Prediction	✗	✓	✓		✓		✓		A
[93]	Prediction	✗	✓	✓	✓	✓	✓			O
[83]	Estimation	✗	✓	✓*	✓		✓			A
[88]	Estimation	✓	✓	✓		✓	✓			O
[78]	Estimation[1]	✗	W	✗		✓		✓		O
	Estimation[2]	✓	✗	✓		✓	✓			O
[79]	Estimation	✗	✓	✓	✓			✓		O/A
[84]	Estimation	✗	✓	✓		✓			✓	Dirty
[85]	Estimation	✗	✓	✓		✓			✓	Dirty
[89]	Estimation	✓	✓	✓		✓	✓			O/A
[98, 99]	Prediction	✗	✓	✓*	✓				✓	O/A
[103]	Density	✗	✓	✓	✓				✓	A
[90]	Estimation	✓	✓	✓		✓	✓			O/A
[86]	Estimation	✗	✓	✓	✓		✓			O
[61]	Prediction	✗	✗	✓	✓				✓	O/A
[100]	Prediction	✗	✓	✓*	✓			✓		A
[87]	Estimation	✓	✗	✓	✗		✓			O/A
[94]	Prediction	✗	✓	✓		✓	✓			O
[81]	Estimation	✗	✓	✓	✓			✓		O
[82]	Estimation	✗	✓	✓	✓				✓	O
[91]	Prediction	✗	✓	✓	✓		✓			O/A

I: Event of interest; U: Unwanted data // W: Window // P: Parametric; SP: Semi-parametric; NP: Non-parametric // D: Deviant; O: Outlier; A: Anomaly; C: Corrupted.
[1] MAD; [2] Hypothesis testing; * Incremental updating.

contexts, these methods use the last observation received (y_t) and some other past information to calculate its expected value ($\hat{y}_t$) and they then decide whether or not it is an outlier. Consequently, they must perform some calculations after the new point has arrived. The cost of this computation depends on the complexity of each method (which has not been analyzed in this chapter or in every original works) but may not always guarantee a fast enough response. Thus, in practice, some of these techniques may not be applicable in streaming contexts and prediction methods are more recommendable for these situations.

2.3.2 Multivariate time series

The input time series is sometimes a multivariate time series with possibly correlated variables rather than a univariate time series. As opposed to the univariate time series case, the detection method used to identify point outliers in multivariate time series can deal not only with a single variable (Section 2.3.2.1) but also with more than one variable simultaneously (Section 2.3.2.2). Additionally, a point outlier in a multivariate time series can affect one (univariate point) or more than one (multivariate point, a vector at time t) variables (see Fig. 2.2b). As will be seen in the following sections, some multivariate techniques are able to detect univariate point outliers and (similarly) some univariate techniques can be used to detect multivariate point outliers. The characteristics mentioned in Fig. 2.5 will also be highlighted.

2.3.2.1 Univariate techniques

Given that a multivariate time series is composed of more than one time-dependent variable, a univariate analysis can be performed for each variable to detect univariate point outliers, without considering dependencies that may exist between the variables. Although the literature barely provides examples of this type of approach, in essence, all of the univariate techniques discussed in Section 2.3.1 could be applied to each time-dependent variable of the input multivariate time series. As one of the few examples, [106] propose using the Long Short-Term Memory (LSTM) prediction model-based method to predict spacecraft telemetry and find point outliers within each variable in a multivariate time series, following the idea of equation (2.1). The authors also present a dynamic thresholding approach in which some smoothed residuals of the model obtained from past data are used to determine the threshold at each time step.

Correlation dependencies between the variables are not considered when applying univariate techniques to each time-dependent variable, leading to a loss of information. To overcome this problem, and at the same time to leverage that univariate detection techniques are highly developed, some researchers apply a preprocessing method to the multivariate time series to find a new set of uncorrelated variables where univariate techniques can be applied. These methods are based on *dimensionality reduction* techniques, and as depicted in Fig. 2.10, the multivariate series is simplified into a representation of lower dimension before applying univariate detection techniques. Since the new series are combinations of the initial input variables, the identified outliers are multivariate; that is, they affect more than one variable.

Some of those dimensionality reduction techniques are based on finding the new set of uncorrelated variables by calculating linear combinations of the initial variables. For example, [107] propose an incremental Principal Component Analysis (PCA) algorithm to determine the new independent variables. In this case, the posterior univariate point outlier detection technique that is

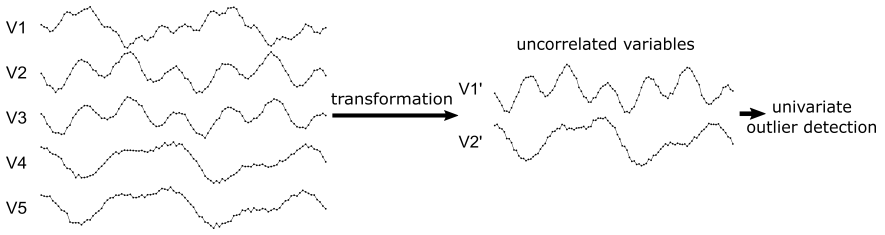


Fig. 2.10: Simplification of point outlier detection in multivariate time series.

applied is based on the autoregressive prediction model (AR). Alternatively, [108] suggest reducing the dimensionality with projection pursuit, which aims to find the best projections to identify outliers. The authors mathematically prove that the optimal directions are those that maximize or minimize the kurtosis coefficient of the projected time series. Univariate statistical tests [68, 109] are then applied iteratively to each projected univariate series for multivariate point outlier detection. Similarly, [110] propose using Independent Component Analysis (ICA) to obtain a set of unobservable independent nonGaussian variables. Outliers are identified in each new series independently if equation (2.1) is satisfied for $\hat{y}_{it} = \mu_i$ and $\tau_i = 4.47\sigma_i$, where μ_i is the mean and σ_i the standard deviation of the ith new variable.

Other techniques reduce the input multivariate time series into a single time-dependent variable rather than into a set of uncorrelated variables. [111] define the transformed univariate series using the cross-correlation function between adjacent vectors in time; that is, $\boldsymbol{y_{t-1}}$ and $\boldsymbol{y_t}$. Point outliers are iteratively identified in this new series as those that have a low correlation with their adjacent multivariate points. The threshold value τ is determined at each iteration by the multilevel Otsu's method. [112] also transform the multivariate time series into a univariate series using a transformation specifically designed for the application domain considered. Point outliers are identified using equation (2.1) and the 3-sigma rule.

The main characteristics of all of the techniques analyzed in this section are described in Table 2.3 in chronological order. Most of the methods reduce the dimensionality of the multivariate time series before applying a univariate detection technique. Particularly, note that the transformation methods proposed by [108] and [111] are specific for outlier detection and not general dimensionality reduction techniques. Also, most of the techniques are non-iterative and would be non-parametric if it were not for the thresholding approach which typically assumes normality. Finally, point outliers are events of interest for almost all the researchers and are mainly referred to as *outliers*.

As far as the characteristics mentioned in Fig. 2.5 are concerned, in dimensionality reduction techniques, the temporality depends on both the transformation and the outlier detection method applied. If at least one of these considers temporality, then so does the complete method because the same

Table 2.3: Summary of the univariate techniques used in multivariate time series for point outlier detection.

Paper	Technique	Iterative	Temporality	Meaning		Parametricity			Term
				I	U	P	SP	NP	
[107]	Dim. reduction	✗	✓	✓			✓		O
[108]	Dim. reduction	✓	✓		✓	✓			O
[110]	Dim. reduction	✗	✗	✓			✓		O
[112]	Dim. reduction	✗	✓	✓			✓		O/E
[111]	Dim. reduction	✓	✓	✓				✓	O
[106]	Prediction	✗	✓	✓				✓	A

I: Event of interest; U: Unwanted data // O: Outlier; A: Anomaly; E: Event // P: Parametric; SP: Semi-parametric; NP: Non-parametric

results are not obtained when the observations of the input multivariate time series are shuffled. In the case of PCA, projection pursuit, and ICA, these transformation methods do not consider temporality, so the temporality depends on the univariate detection method. Conversely, [112] and [111] use methods that include temporality in the transformation phase. In the other approach where the dimensionality is not reduced, the temporality directly depends on the applied detection method.

Finally, all of the methods that we have reviewed in this section can theoretically detect point outliers in a streaming context using sliding windows because none needs future data to provide an output. In this context, the prediction-based approach proposed by [106] would work the best because, in contrast to techniques based on dimensionality reduction, the newly arrived point to be analyzed is not used on the model construction. However, no incremental versions have been proposed.

2.3.2.2 Multivariate techniques

In contrast to the univariate techniques discussed previously, this section analyzes the multivariate methods that deal simultaneously with multiple time-dependent variables, without applying previous transformations. These approaches perform the detection directly using all the original input data variables and can be divided into three groups, as described in Fig. 2.11.

As in univariate time series, *model-based* techniques can also be used to detect point outliers in multivariate time series. Methods within this group are based on fitting a model that captures the dynamics of the series to obtain expected values in the original input time series. Then, for a predefined threshold τ, outliers are identified if:

$$||\boldsymbol{y}_t - \hat{\boldsymbol{y}}_t|| > \tau \tag{2.4}$$

where $\boldsymbol{y}_t$ is the actual L-dimensional data point, and $\hat{\boldsymbol{y}}_t$ its expected value. Note that this is a generalization of the definition given for the model-based

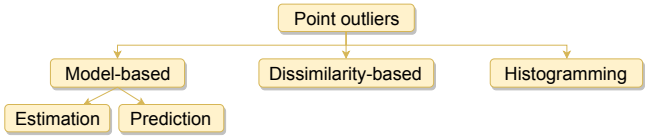
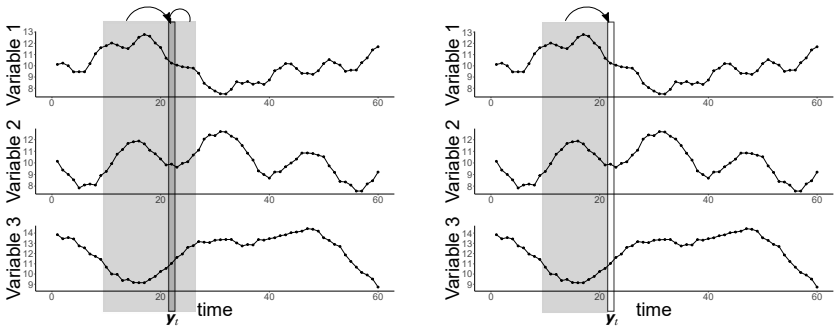


Fig. 2.11: Types of methods for detecting point outliers in multivariate time series.

techniques in univariate time series and that the intuition is repeated (refer to equation (2.1) and Table 2.1). In fact, $\hat{\boldsymbol{y}}_{\boldsymbol{t}}$ can be obtained using *estimation models*—which use past, current, and future values—or *prediction models*—which only use past values (see Fig. 2.12).



(a) Estimation models. (b) Prediction models.

Fig. 2.12: Example of the data used in model-based techniques.

Within the *estimation model-based* category, autoencoders are one of the most commonly used methods. Autoencoders are a type of neural network that learns only the most significant features of a training set used as the reference of normality. Since outliers often correspond to non-representative features, autoencoders fail to reconstruct them, providing large errors in equation (2.4). [113] use this method, where the input of the autoencoder is a single multivariate point of the time series. The temporal correlation between the observations within each variable is not considered in this approximation. Therefore, to account for temporal dependencies, [114] propose extracting features within overlapping sliding windows (e.g., statistical features) before applying the autoencoder. [115] suggest a more complex approach based on a Variational AutoEncoder (VAE) with a Gated Recurrent Unit (GRU). The input of the model is a sequence of observations containing $\boldsymbol{y}_t$ and l preceding observations to it. The output is the reconstructed $\boldsymbol{y}_t$ ($\hat{\boldsymbol{y}}_t$). Additionally, they apply extreme value theory [61] in the reference of normality to automatically select the threshold value.

Apart from using autoencoders, $\hat{\boldsymbol{y}}_{\boldsymbol{t}}$ can also be derived from the general trend estimation of multiple co-evolving time series. [116, 117] use a non-parametric model that considers both the temporal correlation between the values within each variable and inter-series relatedness in a unified manner to estimate that trend. In this case, even though the trend of each variable is estimated using the multivariate time series, univariate point outliers are identified within each variable using equation (2.1) instead of L-dimensional data points.

Prediction model-based techniques can also be used to obtain $\hat{\boldsymbol{y}}_{\boldsymbol{t}}$ in equation (2.4) (see Fig. 2.12b). These techniques also fit a model to a multivariate time series, but the expected values are the predictions for the future made on the basis of past values. For example, the Contextual Hidden Markov Model (CHMM) incrementally captures both the temporal dependencies and the correlations between the variables in a multivariate time series [118]. The temporal dependence is modeled by a basic HMM, and the correlation between the variables is included into the model by adding an extra layer to the HMM network. The DeepAnt algorithm [91] mentioned in Section 2.3.1 is also capable of detecting point outliers in multivariate time series using the CNN prediction model. Once the model is learned, the next timestamp is predicted using a window of previous observations as input.

All of these estimation and prediction-based methods can theoretically be employed in a streaming context using sliding windows because no subsequent points to $\boldsymbol{y}_{\boldsymbol{t}}$ are needed. As with univariate time series, the estimation-based methods need to consider at least the newly arrived point $\boldsymbol{y}_{\boldsymbol{t}}$ ($k_2 = 0$ in Table 2.1), so prediction-based techniques are more appropriate for detecting outliers in a streaming fashion. Moreover, these model-based techniques all use a fixed model, and they do not adapt to changes over time, except for the proposal of [118], which is incrementally updated.

The *dissimilarity-based* methods will be discussed next. These techniques are based on computing the pairwise dissimilarity between multivariate points or their representations, without the need for fitting a model. Therefore, for a predefined threshold τ, $\boldsymbol{y}_{\boldsymbol{t}}$ is a point outlier if:

$$s(\boldsymbol{y}_{\boldsymbol{t}}, \hat{\boldsymbol{y}}_{\boldsymbol{t}}) > \tau \tag{2.5}$$

where $\boldsymbol{y}_{\boldsymbol{t}}$ is the actual L-dimensional point, $\hat{\boldsymbol{y}}_{\boldsymbol{t}}$ its expected value, and s measures the dissimilarity between two multivariate points. These methods do not usually use the raw data directly, but instead use different representation methods. For example, [119, 120] represent the data using a graph where nodes are the multivariate points of the series and the edges the similarity values between them computed with the Radial Basis Function (RBF). The idea is to apply a random walk model in the graph to detect the nodes that are dissimilar to the others (i.e., hardly accessible in the graph). By contrast, [121] propose recording the historical similarity and dissimilarity values between the variables in a vector. The aim is to analyze the dissimilarity of consecutive points over time and detect changes using $||.||_1$.

The last group depicted in Fig. 2.11 refers to the *histogramming* approach, where the term *deviant* has also been used in the context of multivariate series (see the definition given in (2.3)). [105] extend the technique for deviant detection explained in Section 2.3.1 to multivariate time series by treating the measurements collected at the same timestamp as a vector. The authors propose an algorithm for both streaming and non-streaming series to find approximate optimal deviants.

To conclude, the multivariate techniques that detect point outliers in multivariate time series can be grouped mainly in three categories based on the intuition of the concept of outlier that they employ. The main characteristics of these multivariate techniques are depicted in Table 2.4 in chronological order. Even if most of them find multivariate point outliers, some use the multivariate information to identify point outliers that only affect a single variable (i.e., univariate point outliers). All of the analyzed techniques are non-iterative, and outliers represent events of interest for the researchers. In addition, most of them obtain different results if they are applied to a shuffled version of the time series, i.e., most methods consider the temporal information. Although all of these methods can detect outliers in a streaming context, few are incremental or updated as new data arrives. Finally, most of the model-based techniques are parametric or semi-parametric, while all the techniques based on histograms or dissimilarity are non-parametric.

Table 2.4: Summary of the multivariate techniques used in multivariate time series for point outlier detection.

Paper	Technique	Point	Temporality	Incremental	Parametricity			Term
					P	SP	NP	
[105]	Histogramming	Multivariate	✓	✓			✓	D
[119, 120]	Estimation	Multivariate	✗	✗			✓	A
[121]	Dissimilarity	Univariate	✓	✓			✓	O
[113]	Estimation	Multivariate	✗	✗	✓			A
[118]	Prediction	Multivariate	✓	✓	✓			N/A'
[114]	Estimation	Multivariate	✓	✗	✓			O
[91]	Prediction	Multivariate	✓	✗	✓			A/O
[116, 117]	Estimation	Univariate	✓	✗			✓	O/N/E
[115]	Estimation	Multivariate	✓	✗		✓		A

D: Deviant; O: Outlier; A: Anomaly; A': Abnormality; E: Event; N: Novelty // P: Parametric; SP: Semi-parametric; NP: Non-parametric

2.4 Subsequence outliers

As shown in Fig. 2.1, subsequence outliers are the second type of outliers that can be detected in time series data. In this case, the aim is to identify a set of consecutive points that jointly behave unusually. To this end, subsequence outlier detection methods need to consider some key aspects, which are shown in Fig. 2.13, and which make the detection of subsequence outliers more challenging than point outlier detection.

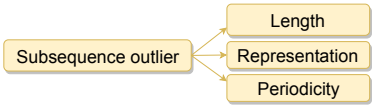


Fig. 2.13: Characteristics related to subsequence outlier detection problems.

To begin with, subsequences are composed of a set of points and not of a single point, so they have a certain length. Typically, methods consider fixed-length subsequences, although some techniques do allow us to detect subsequences of different lengths (variable-length) simultaneously (e.g., [122, 123])[1]. Methods based on fixed-length subsequences need the user to predefine the length, and they commonly obtain the subsequences using a sliding window over the time series. In contrast, methods that allow finding variable-length subsequences do not prespecify this length. A final aspect to consider regarding the length of the subsequence is that the number of subsequences that the method will consider and analyze depends on the chosen length (i.e., the shorter the length, the higher the number of subsequences).

The second characteristic that subsequence outlier detection methods need to consider is the representation of the data. Since the comparison between subsequences is more challenging and costly than the comparison between points, many techniques rely on a representation of the subsequences rather than using the original raw values. Discretization is a widely used representation method and can be obtained using approaches such as equal-frequency binning (e.g., [124]), equal-width binning (e.g., [125, 126, 60]), or SAX (e.g., [122, 123]). These discretization techniques can also be used as a starting point to obtain other representations, such as bitmaps (e.g., [127, 128]). A detailed overview of the existing research regarding outlier detection in discrete sequences can be found in [129], which highlights the applicability of those techniques to time series data. Additionally, apart from discretization, raw data has also been used directly to obtain representations based on dictionaries (e.g., [130]), exemplars (e.g., [131]), or connectivity values (e.g., [132]).

[1] Note that methods that find fixed-length subsequence outliers could also find subsequences of different lengths by applying the method repeatedly for each possible length.

Another issue that has been barely considered in the literature but which makes the detection of subsequence outliers more challenging is that they can be periodic. Periodic subsequence outliers are unusual subsequences that repeat over time [133, 60]. Unlike point outliers where periodicity is not relevant, periodic subsequence outliers are interesting in areas such as fraud detection to discover certain periodic anomalous transactions over time.

Finally, as opposed to point outlier detection, where some methods did not take into account the temporal dependencies, subsequences consider the temporality by default. In addition, when analyzing subsequence outliers in a streaming context, three cases can occur: i) a single data point arrives, and an answer (i.e., outlier or non-outlier) must be given for a subsequence containing this new point; ii) a subsequence arrives, and it needs to be marked as an outlier or non-outlier; and iii) a batch of data arrives and subsequence outliers need to be found within it. In either case, the literature provides methods that can give an answer in a streaming fashion using sliding windows. However, most of them keep the model fixed and do not adapt to changes in the streaming sequence. We will focus on these incremental techniques because they are more suitable for processing streaming time series.

2.4.1 Univariate time series

The detection methods used to detect univariate subsequence outliers in univariate time series will be analyzed in this section. We have grouped these techniques according to the different ideas or definitions on which they are based (see Fig. 2.14).

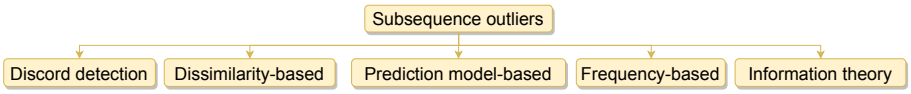


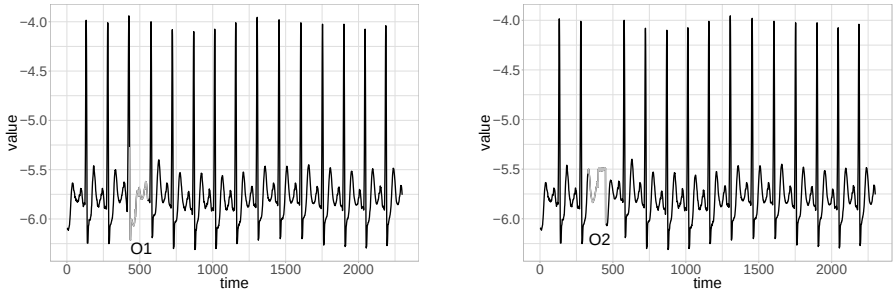
Fig. 2.14: Types of methods for detecting subsequence outliers in univariate time series.

The first and most straightforward idea consists of detecting the most unusual subsequence in a time series (denominated time series *discord*) [134, 135], by comparing each subsequence with the others; that is, S_D is a discord of time series Y if

$$\forall S \in A, \quad \min_{S'_D \in A, S_D \cap S'_D = \emptyset} (d(S_D, S'_D)) > \min_{S' \in A, S \cap S' = \emptyset} (d(S, S')) \tag{2.6}$$

where A is the set of all subsequences of Y extracted by a sliding window, S'_D is a subsequence in A that does not overlap with S_D (non-overlapping subsequences), S' in A does not overlap with S (non-overlapping subsequences), and d is the Euclidean distance between two subsequences of equal length.

Typically, discord discovery techniques require the user to specify the length of the discord. Two examples are given in Fig. 2.15, in which the most unusual subsequences (O1 and O2) are shown.



(a) (b)

Fig. 2.15: Discord examples using `jmotif` package [1].

The simplest way of finding discords is using brute-force, which requires a quadratic pairwise comparison. Even so, the process can be sped-up by reordering the search using heuristics and pruning off some fruitless calculations with the HOT-SAX algorithm [134, 136, 135], which is based on the SAX discrete representation.

Many variants of the HOT-SAX algorithm aim to reduce its time complexity by enhancing both the heuristic reordering and the pruning for discord detection, which may be based on the Haar wavelet transformation and augmented trie [137, 138], on bit representation clustering [139], on bounding boxes [140], on clustering categorical data [141], and on the iSAX representation [142]. Additionally, in [143], not only is the HOT-SAX algorithm applied in a streaming context using a sliding window but an incremental algorithm has also been suggested to simplify this computation. The method proposed by [141] is also based on this incremental algorithm.

Those discord discovery techniques require the user to prespecify the length of the discord, which in many cases may not be known. Consequently, [122] present two approaches to detect variable-length discords applying grammar-induction procedures in the time series discretized with SAX. Since symbols that are rarely used in grammar rules are non-repetitive and thus most likely to be unusual, subsequence outliers correspond to rare grammar rules, which naturally vary in length. Given that the lengths of the subsequences vary, the Euclidean distance between them is calculated by shrinking the longest subsequence with the Piecewise Aggregate Approximation (PAA) [144] to obtain subsequences of the same length.

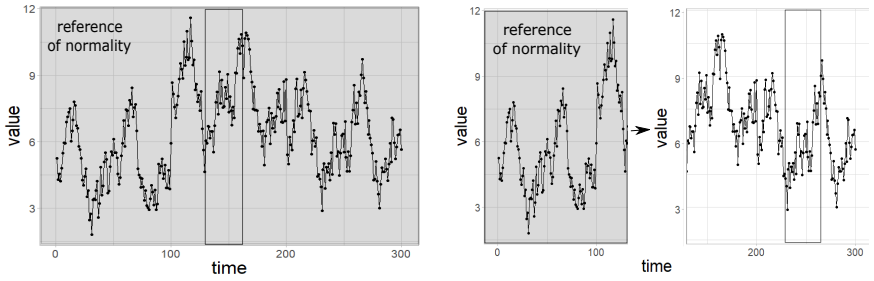
The abovementioned discord detection techniques are limited to finding the most unusual subsequence within a time series. However, since they do

not have a reference of normality or a threshold, they cannot specify whether it is indeed an outlier or not. Therefore, this decision must be made by the user. Conversely, the other categories in Fig. 2.14 consider a criterium of normality and contain specific rules to decide whether or not the identified subsequences are outliers.

For example, *dissimilarity-based* methods are based on the direct comparison of subsequences or their representations, using a reference of normality. In this category, the reference of normality, as well as the representations used to describe the subsequences, vary widely, in contrast to the dissimilarity-based techniques analyzed in Section 2.3.2.2. Then, for a predefined threshold τ, subsequence outliers are those that are dissimilar to normal subsequences; that is,

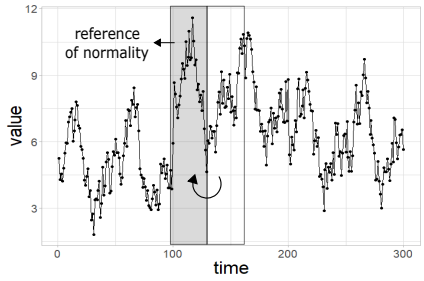
$$s(S, \hat{S}) > \tau \tag{2.7}$$

where S is the subsequence being analyzed or its representation, $\hat{S}$ is the expected value of S obtained based on the reference of normality, and s measures the dissimilarity between two subsequences. Typically, S is of fixed-length, non-periodic, and extracted via a sliding window. Some dissimilarity-based approaches are described below ordered based on the considered reference of normality (see Fig. 2.16), which will be used to obtain the expected value $\hat{S}$.



(a) Same time series.

(b) External time series.



(c) Previous subsequence.

Fig. 2.16: Reference of normality used by dissimilarity-based approaches.

A. Same time series

The most straightforward approach is to consider the same time series object of the analysis as the reference of normality (Fig. 2.16a). Clustering techniques commonly use this reference of normality to create clusters by grouping subsequences that are similar to each other and by separating dissimilar subsequences into different clusters (see Fig. 2.17). Then, $\hat{S}$ in equation (2.7) can be defined by the centroid or center of the cluster to which S belongs. For example, [125] and [126] cluster discretized subsequences of the same length and flag subsequences that are far from the nearest centroid ($\hat{S}$) as outliers. In this case, the distance used is the Euclidean norm of three specific distances between the discretized subsequences. Alternatively, [145] employ Fuzzy C-Means (FCM) clustering on the raw data, allowing each subsequence to belong to more than one cluster. In particular, the authors use the Euclidean distance in (2.7). Note that these two clustering approaches are not applied to streaming time series in the original papers but could be extended using stream clustering algorithms [146].

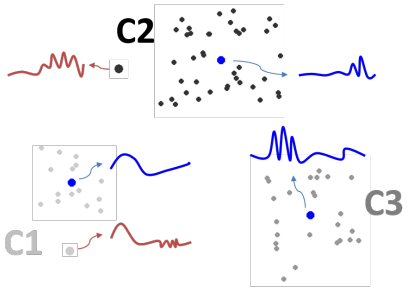


Fig. 2.17: Clustering of the subsequences in a univariate time series. Cluster centroids are highlighted, and $C1$ and $C2$ contain subsequence outliers.

As opposed to the clustering methods that detect fixed-length subsequence outliers, [123] propose dynamic clustering to identify variable-length outliers in a streaming context. The method works directly with raw data and assumes that the observations arrive in fixed-length batches (B_i is the i^{th} batch). Given U initial batches, grammar-induction is used to find a rule for dividing the incoming batches of data into M non-overlapping variable-length subsequences; that is, $B_i = S_{i1} \cup S_{i2} \cup ... \cup S_{iM}$ (all batches are split in the same points). Clustering is then applied to the set of subsequences $S_{1j}, S_{2j}, ..., S_{Uj}$ for each j separately. A new incoming subsequence is flagged as an outlier using dynamic clustering and observing when it is either far from its closest cluster centroid or belongs to a sparsely populated cluster.

There are other techniques besides clustering that also use the same time series as the reference of normality. For instance, [132] use the same time series

to represent each subsequence with a value that indicates how dissimilar it is with the rest. To this end, a Markov chain random walk model is applied in a graph where each node is a subsequence and each edge the pairwise similarity value between the nodes [147]. These pairwise similarity values are computed based on the Piecewise Aggregate Pattern Representation (PAPR), which is a matrix representation that captures the statistical information covered in the subsequence.

B. External time series

The methods that rely on an external time series as a reference of normality assume that it has been generated by the same underlying process but with no outliers. For example, [131] use an external time series to find a set of exemplars that summarizes all of its subsequences within it and which detects outliers as those that are far from their nearest exemplar ($\hat{S}$). In this case, subsequences are represented by a feature vector of two components that captures both the shape and the stochastic variations of the subsequences in a time series, using smoothing and some statistics (e.g., the mean and the standard deviation). The authors use a weighted Euclidean distance in equation (2.7), taking into account the components of both a subsequence and its nearest exemplar.

In this category, we have also included techniques that use past non-outlier subsequences as the reference of normality. Even if these subsequences belong to the same time series, this set is considered as an external set where outliers are not detected, unlike the previous category. For instance, [130] collect the most relevant subsequences within a series of past subsequences in a dictionary. Subsequence outliers are those that cannot be accurately approximated by a linear combination of some subsequences of the dictionary ($\hat{S}$), resulting in a large error in equation (2.7) using the Euclidean distance [148, 149].

C. Previous subsequence

Some techniques only use the previous adjacent non-overlapping window to the subsequence being analyzed as the reference of normality, which means that they have a much more local perspective than the others. [127] and [128] use this reference of normality to obtain $\hat{S}$. Specifically, the authors represent each subsequence by a bitmap, a matrix in which each cell represents the frequency of a local region within a subsequence. A subsequence whose local regions differ from the regions of its previous adjacent subsequence ($\hat{S}$) is flagged as an outlier, using the squared Euclidean distance between each pair of elements of the bitmaps. In this case, bitmaps are incrementally updated at each time step.

Returning to the general classification in Fig. 2.14, the third group of methods belongs to the *prediction model-based* category, which assumes that normality is reflected by a time series composed of past subsequences. These

methods intend to build a prediction model that captures the dynamics of the series using past data and thus make predictions of the future. Subsequences that are far from those predictions are flagged as outliers because, although they may resemble subsequences that have appeared previously, they do not follow the dynamics captured by the model:

$$\sum_{i=p}^{p+n-1} |y_i - \hat{y}_i| > \tau \tag{2.8}$$

where $S = y_p, ..., y_{p+n-1}$ is the subsequence being analyzed, $\hat{S}$ is its predicted value, and τ is the selected threshold. Predictions can be made in two different manners: point-wise or subsequence-wise. Models that use point-wise prediction make predictions for as many points as the length of the subsequence iteratively. With this in mind, any method within Section 2.3.1 could be used for this purpose. However, since each predicted value is used to predict the subsequent point, the errors accumulate as predictions are made farther into the future. In contrast, the subsequence-wise prediction calculates predictions for whole subsequences at once (not iteratively). Within this category, [91] use Convolutional Neural Networks (CNN) to predict whole subsequences and detect outliers using a model built with past subsequences.

The next subsequence outlier detection methods are the *frequency-based*, as shown in Fig. 2.14, which also use one of the reference of normality mentioned in Fig. 2.16. A subsequence S is an outlier if it does not appear as frequently as expected:

$$|f(S) - \hat{f}(S)| > \tau \tag{2.9}$$

where $f(S)$ is the frequency of occurrence of S, $\hat{f}(S)$ its expected frequency, and τ a predefined threshold. Due to the difficulty of finding two exact real-valued subsequences in a time series when counting the frequencies, these methods all work with a discretized time series.

A paradigmatic example can be found in [124]. Given an external univariate time series as reference, a subsequence extracted via a sliding window in a new univariate time series is an outlier relative to that reference if the frequency of its occurrence in the new series is very different to that expected. [60] also propose an algorithm based on the frequency of the subsequences but the aim is to detect periodic subsequence outliers together with their periodicity [150]. In this case, the reference of normality is the same time series. The intuition behind this method is that a periodic subsequence outlier repeats less frequently in comparison to the more frequent subsequences of same length.

As shown in Fig. 2.14, the last group of subsequence outlier detection methods correspond to *information theory* based techniques, which are closely related to the frequency-based methods. In particular, [133, 151] focus on detecting periodic subsequence outliers in discretized univariate time series using this theory. They assume that a subsequence that occurs frequently

is less surprising and thus carries less information than a rare subsequence. Therefore, the aim is to find infrequent but still repetitive subsequences with rare symbols, using the same time series as the reference of normality; that is, [133, 151] mark S as an outlier if

$$I(S) \times f(S) > \tau \tag{2.10}$$

where $I(S) \geq 0$ is the information carried by S and $f(S) \geq 1$ is the number of occurrences of S. $I(S)$ is computed taking into account the number of times the symbols within S are repeated through the series, so a discretized subsequence S that has symbols that do not commonly appear in the time series has a large $I(S)$. Conversely, if $f(S)$ is large (S occurs frequently), then the information $I(S)$ will be lower, closer to 0.

To conclude, the techniques that detect subsequence outliers in univariate time series can be divided into five categories according to the different ideas on which they are based. These techniques usually consider a reference of normality, which can be of three types, when detecting the outliers.

A summary of all of these methods is presented in Table 2.5. Most of them detect non-periodic outliers of fixed-length, assuming the length is known in advance, and use a discretized version of the time series to compare real-valued subsequences effectively or to speed up the search process of outliers, this one specifically in the discord discovery category. In contrast to the many methods that discover point outliers, the analyzed methods for subsequences are not iterative, and there are very few fully parametric methods. In addition, for all of the authors, the outlier represents an event of interest, and they often use the term *discords* to refer to the subsequence outliers.

2.4.2 Multivariate time series

This section presents some of the detection techniques that have been used in the literature to detect subsequence outliers in multivariate time series data. The nature of these methods can be either univariate (Section 2.4.2.1) or multivariate (Section 2.4.2.2), and they can detect subsequence outliers that affect either one variable (univariate outliers) or multiple variables, which are all aligned (multivariate outliers).

2.4.2.1 Univariate detection techniques

Each variable of the multivariate time series can contain subsequence outliers that are independent of other variables. The identification of those subsequences may be carried out by applying the univariate techniques discussed in Section 2.4.1 to each time-dependent variable. In particular, [152, 131] apply the exemplar-based method to each variable of the multivariate input time series, using an external time series as the reference of normality and without considering the correlation that may exist between variables. Recall that omitting this correlation can lead to a loss of information.

Table 2.5: Summary of the characteristics of subsequence outlier detection approaches in univariate time series.

Paper	Technique	Periodic outliers	Length		Discretization	Parametricity			Term
			F	V		P	SP	NP	
[133, 151]	Inf. theory	✓	✓		✓			✓	S
[124]	Frequency	✗	✓		✓	✓			S
[127]	Similarity	✗	✓		✓			✓	A
[128, 134, 136, 135]	Discord	✗	✓		✓			✓	D
[137]	Discord	✗	✓		✓			✓	D/A
[138]	Discord	✗	✓		✓			✓	D
[143]	Discord	✗	✓		✓			✓	D
[142]	Discord	✗	✓		✓			✓	D
[125, 126]	Dissimilarity	✗	✓		✓		✓		O/A
[139]	Discord	✗	✓		✓			✓	D
[145]	Dissimilarity	✗	✓		✗			✓	A
[140]	Discord	✗	✓		✗			✓	D/A
[60]	Frequency	✓	✓		✓			✓	O/S
[122]	Discord	✗		✓	✓			✓	D/A
[130, 149]	Dissimilarity	✗	✓		✗		✓		O/A
[131]	Dissimilarity	✗	✓		✗			✓	A
[132]	Dissimilarity	✗	✓		✗	✓			A
[141]	Discord	✗	✓		✓			✓	D/O/A
[91]	Model	✗	✓		✗	✓			D/O/A
[123]	Dissimilarity	✗		✓	✓		✓		A

F: Fixed; V: Variable // S: Surprise; D: Discord; O: Outlier; A: Anomaly // P: Paramteric; SP: Semi-parametric; NP: Non-parametric.

Intending to simplify the multivariate task but without completely ignoring the correlation between the variables, some methods reduce the dimensionality of the input series before applying a univariate detection technique. For example, [123] extend their method for univariate subsequences by applying clustering to a simplified series to detect variable-length subsequence outliers. The simplified series is obtained by computing the distance to some representative subsequences, which have been obtained by applying their univariate technique to each of the variables independently (see Section 2.4.1). Each new multivariate batch of data is then represented by a vector of distances, $(d_1, d_2, ..., d_l)$, where d_j represents the Euclidean distance between the j^{th} variable-length subsequence of the new batch and its corresponding representative subsequence. As with their univariate technique, the reference of normality that is considered by this method is the same time series.

The technique proposed by [153] is also based on reducing the dimensionality of the time series and allows us to detect variable-length discords, while

using the same time series as the reference of normality. This is based on the fact that the most unusual subsequences tend to have local regions with significantly different densities (points that are similar) in comparison to the other subsequences in the series. Each point in the new built univariate time series describes the density of a local region of the input multivariate time series obtained by a sliding window. This series is also used to obtain the variable-length subsequences. Discords are identified using the Euclidean and Bhattacharyya distances simultaneously.

To conclude, Table 2.6 provides a summary of all these univariate techniques together with their characteristics. Most of the discussed techniques reduce the dimensionality of the multivariate time series before employing a univariate detection technique to detect subsequence outliers. If the univariate technique is applied to each variable, then the detected subsequence outliers will affect a single variable. Otherwise, if the technique is employed in a reduced time series, then the outliers commonly affect multiple variables because the new series contains multivariate information. It should be noted that although [123] identify multivariate batch outliers, the variable-length subsequence outliers that are then identified within those batches affect a single variable. None of them detects periodic subsequence outliers nor are any of them fully parametric. Subsequence outliers are mainly referred to as *anomalies*.

Table 2.6: Summary of the univariate techniques used in multivariate time series for subsequence outlier detecion.

Paper	Technique	Length		Subsequence	Discretization	Parametricity		Term
		Fixed	Variable			P SP	NP	
[152]	Dissimilarity	✓		Univariate	✗		✓	A
[123]	Dim.red.		✓	Univariate	✓	✓		A
[153]	Dim.red.		✓	Multivariate	✗		✓	A/D

A: Anomaly; D: Discord // P: Parametric; SP: Semi-parametric; NP: Non-parametric

2.4.2.2 Multivariate detection techniques

The techniques for multivariate subsequence outlier detection that will be reviewed in this section deal with multiple time-dependent variables simultaneously and typically detect temporally aligned outliers that affect multiple variables. As shown in Fig. 2.18, these techniques are divided into two main groups. However, the philosophy behind some of them is repeated because they are an extension of simpler techniques introduced in previous sections.

The first type of method to be discussed is the *model-based* method. As mentioned in previous sections, the aim is to detect the subsequences that are far from their expected value, which can be obtained using an *estimation* or a

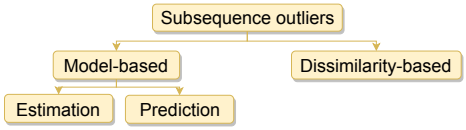


Fig. 2.18: Types of subsequence outlier detection methods in multivariate time series.

prediction model. A subsequence $S = \boldsymbol{y_p}, ..., \boldsymbol{y_{p+n-1}}$ of length n is an outlier for a predefined threshold τ if

$$\sum_{i=p}^{p+n-1} ||\boldsymbol{y_i} - \hat{\boldsymbol{y}}_{\boldsymbol{i}}|| > \tau. \tag{2.11}$$

Within the techniques based on *estimation models*, the approach proposed by [152] intends to find pairs of related variables using a set of nonlinear functions. These functions are learned using an external multivariate time series as the reference of normality (see Fig. 2.16). Then, a subsequence within a variable of a new time series is estimated using the learned functions and data from another variable of the new series (past and future observations with respect to $\boldsymbol{y_t}$). In contrast, the CNN method described by [91] is a *prediction model-based* technique, which is a direct extension of the method explained in Section 2.4.1 to detect time-aligned outliers that affect multiple variables.

The second group corresponds to the *dissimilarity-based* techniques, a generalization of equation (2.7) that finds unusual subsequences in a multivariate time series based on the pairwise dissimilarity calculation between subsequences or their representations. Unlike in the univariate subsequence outlier detection, this type of technique has barely been used in multivariate series. In the only example, [119, 120] extend their method for point outliers (Section 2.3.2.2) to obtain how dissimilar is a node representing a fixed-length subsequence with the others. This dissimilarity value is obtained by applying a random walk model in the graph, and the computation of the pairwise similarity between those nodes is also based on the RBF.

To summarize this section, the multivariate techniques that detect subsequence outliers in multivariate time series can be grouped into two different categories according to the intuition behind the methods. A summary of these techniques is given in Table 2.7. In all of the cases, the subsequence outliers represent an event of interest and are denominated as *anomalies*. In addition, the techniques find non-periodic outliers of fixed-length without using a discretization technique. Finally, the model-based techniques are parametric, while the only dissimilarity-based method is non-parametric.

Table 2.7: Summary of the multivariate techniques used in multivariate time series for subsequence outlier detecion.

Paper	Technique	Subsequence	Parametricity	Term
[119, 120]	Dissimilarity	Multivariate	Non-parametric	Anomaly
[152]	Estimation	Univariate	Parametric	Anomaly
[91]	Prediction	Multivariate	Parametric	Anomaly / Outlier

2.5 Outlier time series

The analyzed task has so far been to identify point and subsequence outliers within a time series, either univariate or multivariate. However, there are some cases where it may also be interesting to find entire unusual variables in a multivariate time series. This could lead, for example, to the identification of malicious users or mail servers behaving unusually. Recall that outlier time series can only be detected in multivariate time series and using a multivariate detection technique.

Some of the key aspects presented for the subsequence outliers (see Fig. 2.13) can also appear when attempting to detect outlier time series. First, each time-dependent variable in a multivariate time series can have a different length. Second, representation methods such as discretization can also be used to facilitate the comparisons between variables. In addition, all of the outlier time series detection methods can theoretically be applied in a streaming context using sliding windows. However, in contrast to subsequences, the property of temporality is not necessarily considered by these methods.

This section will examine the techniques that detect outlier time series, following the diagram given in Fig. 2.19.

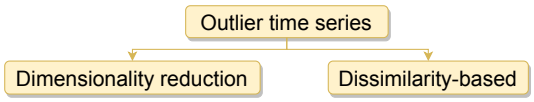


Fig. 2.19: Types of methods for detecting outlier time series in multivariate time series.

The first type of technique to be discussed is based on *dimensionality reduction*. As mentioned in the previous sections, the aim of these techniques is to reduce the dimensionality of the input multivariate time series into a set of uncorrelated variables. As shown in Fig. 2.20, [154] propose reducing the dimensionality by extracting some representative statistical features from each time series (e.g., mean, and first order of autocorrelation) and then applying PCA. Outlier time series are detected by their deviation from the highest density region in the PCA space, which is defined by the first two principal components. [155] use clustering in that PCA space to detect outlier time

series by measuring the deviation between the cluster centroids and the points that represent the time series.

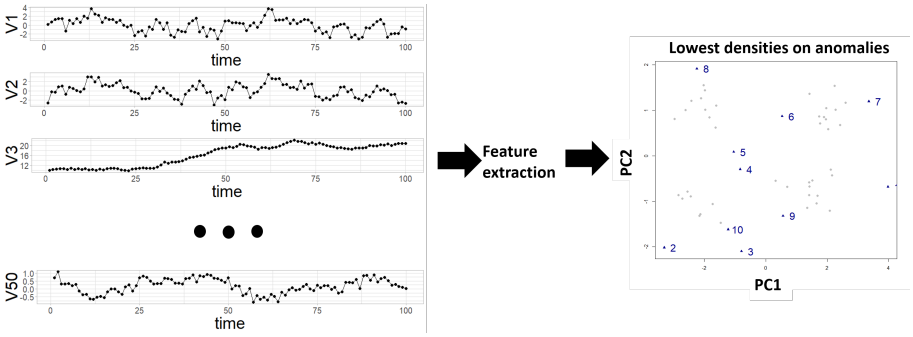


Fig. 2.20: Outlier time series detection in a multivariate time series composed of 50 variables using PCA in the extracted features with the `anomalous` package in `R`.

Instead of applying a detection technique in a lower space, original raw data can also be used directly. Indeed, the *dissimilarity-based* techniques in Fig. 2.19 directly analyze the pairwise dissimilarity between the time series. The most common approach within this category is clustering. The intuition is similar to that depicted in Fig. 2.17, but whole time series are considered instead of subsequences. In this case, the reference of normality is the same multivariate time series.

Within the dissimilarity-based techniques that use clustering, [156] propose applying Phased k-means in unsynchronized periodic multivariate time series to obtain a set of representative time series centroids. This technique is a modification of k-means so that the phase of each time-dependent variable to its closest centroid is adjusted prior to dissimilarity calculation at every iteration. The outliers are identified using equation (2.7) and cross-correlation as the similarity measure between time series. [157] also use clustering but, in this case, based on the Dynamic Time Warping (DTW) measure, which allows the series to be warped and shifted in time and have different lengths. The method optimizes an objective function by both minimizing the within-cluster distances using the DTW measure and maximizing the negative entropy of some weights that are assigned to each time series. Time series that increase the within-cluster distances to their closest cluster have smaller weights. Specifically, time series with small weights are considered to be outliers. A different approach that also relies on clustering time series has been proposed by [158]. This method uses agglomerative hierarchical clustering with single linkage and identifies time series outliers based on four criteria where basically the size of the cluster is considered. Clustering is conducted using a time series

similarity measure that has the same underlying idea as the Jaccard similarity coefficient.

Other dissimilarity-based techniques are based on shapelets, which are representative time series subsequences [159]. [160] use them to describe the shape of normal variables and detect outlier time-dependent variables in a multivariate time series. The shapelets are learned using an external time series as the reference of normality. The idea is to measure the dissimilarity that each learned shapelet has with a variable using the Euclidean distance. Subsequences of outlier variables are dissimilar to the learned shapelets.

To conclude, outlier time series can be grouped into two categories based on the intuition behind the detection method. A summary of these techniques is provided in Table 2.8. The works reviewed in this section are non-parametric and non-iterative techniques that intend to find events of interest by directly using raw data and considering temporality. Unlike the subsequences, the length of the series is always specified. The approach proposed by [157] is the only one that can deal with time series with variables of different lengths.

Table 2.8: Summary of the characteristics of outlier time series detection in multivariate time series.

Paper	**Technique**	**Variable-length**	**Term**
[156]	Dissimilarity-based	✗	Anomaly / Outlier
[154]	Dimensionality reduction	✗	Anomaly / Outlier / Unusual
[157]	Dissimilarity-based	✓	Outlier
[160]	Dissimilarity-based	✗	Anomaly
[158]	Dissimilarity-based	✗	Outlier

2.6 Publicly available software

In this section, the publicly available software for outlier detection in time series data according to the techniques mentioned in previous sections is provided. A summary of this software can be found in Table 2.9, in which the technical descriptions (*Related research* column) and the link to access the code (*Code* column) are presented. The table is organized based on the outlier type the technique detects.

2.7 Concluding remarks and future work

In this chapter, an organized overview of outlier detection techniques in time series data has been proposed. Moreover, a taxonomy that categorizes these

Table 2.9: Summary of the publicly available software in chronological order.

Name	Language	Related research	Code
Point outliers			
tsoutliers	R	[109]	https://cran.r-project.org/web/packages/tsoutliers
spirit	Matlab	[107]	http://www.cs.cmu.edu/afs/cs/project/spirit-1/www
STORM	Java	[101, 102]	https://github.com/Waikato/moa/tree/master/moa/src/main/java/moa/clusterers/outliers/Angiulli
SCREEN	Java	[84]	https://github.com/zaqthss/sigmod15-screen
EGADS	Java	[155]	https://github.com/yahoo/egads
SCR	Java	[85]	https://github.com/zaqthss/sigmod16-scr
libspot	C++	[61]	https://github.com/asiffer/libspot
AnomalyDetection	R	[87]	https://github.com/twitter/AnomalyDetection
Nupic	Python	[100]	https://github.com/numenta/nupic
telemanon	Python	[106]	https://github.com/khundman/telemanom
OmniAnomaly	Python	[115]	https://github.com/smallcowbaby/OmniAnomaly
OTSAD	R	[83, 103, 161]	https://cran.r-project.org/package=otsad
envoutliers	R	[81, 82]	https://cran.r-project.org/web/packages/envoutliers/index.html
Subsequence outliers			
tsbitmaps	Python	[127, 128]	https://github.com/binhmop/tsbitmaps
jmotif	R	[134, 136] [122, 162]	https://github.com/jMotif/jmotif-R
jmotif	Java	[134, 136] [122]	https://github.com/jMotif/SAX
saxpy	Python	[134, 136]	https://pypi.org/project/saxpy https://github.com/seninp/saxpy
EBAD	C	[131]	http://www.merl.com/research/license
GrammarViz	Java	[122, 162]	https://github.com/GrammarViz2/grammarviz2_src
Outlier time series			
anomalous	R	[154]	http://github.com/robjhyndman/anomalous-acm

methods depending on the input data type, the outlier type, and the nature of the detection method has been included. This section first discusses some general remarks about the analyzed techniques, and it then introduces the conclusions regarding the axes of the taxonomy.

As seen in previous sections, a broad terminology has been used to refer to the same concept: the outlier in unlabeled time series data. These terms are usually related to the objective of the detection such that *outlier* is mostly used when detecting unwanted data, whereas *anomaly* has been used when detecting events of interest. In most of the analyzed works, the concept outlier represents an event of interest; that is, the authors mainly focus on extracting the outlier information considered as useful data rather than on cleaning the useless or unwanted data to improve the data quality for further analysis. Thus, it might be interesting to extend this type of methods by developing

techniques that improve the data quality, mainly of multivariate time series data.

Despite the variety in terminology and purpose, all of the methods that we have reviewed are based on the same idea: detecting those parts in a time series (point, subsequence, or whole time series) that significantly differ from their expected value. Each author uses different means to obtain this expected value and compute how far it is from the actual observation to decide whether or not it is an outlier. Although some techniques obtain the expected value based on an external or reference set, caution must be taken because it can itself contain outliers and distort the detection. In fact, they are in the limit of the unsupervised framework because even if the time series has no labels, it is usually assumed that all the external or reference data are non-outliers.

Once the expected value is obtained, a threshold is often needed to decide whether or not we have found an outlier. Given that the results directly vary depending on that selection and few techniques give an automatic way to determine the threshold [61, 111, 106, 115], an interesting future line of research would be to deepen on the dynamic and adaptive selection of thresholds in both univariate and multivariate time series. Indeed, to the best of our knowledge, there are no methods that include this type of threshold in the subsequence and entire time series outlier analysis.

As a final general remark and before proceeding with the conclusions regarding the axes of the taxonomy, the time elapsed from one observation to the subsequent is also an important aspect to consider. The vast majority of methods assume that the time series are regularly-sampled. However, real life often does not provide this type of data, and converting it into such type is not always the best option. Therefore, outlier detection in an irregularly-sampled time series is an interesting future direction line.

Having provided some general conclusions, we will now focus on each of the axes. Starting from the first axis, the most remarkable conclusion is that even if most of the analysis has been performed on univariate input time series data, in recent years special emphasis has been placed on multivariate time series.

For the second axis, point outlier detection is the most researched problem due to its simplicity in comparison to the rest. Within this category, it is remarkable that some techniques do not consider the temporal information at all. This can lead to not detecting outliers that violate the temporal correlation of the series. Thus, possible future work may include temporal information to techniques that do not consider it (e.g., [119, 120, 61]). Additionally, even if theoretically many techniques can determine if a new data point is an outlier upon arrival, no study has been conducted to analyze whether these methods can really give an immediate response in real time or not. Consequently, an in-depth investigation could be done to analyze the computational cost of outlier detection techniques and to determine whether these methods can be practically used in real-time scenarios. There is also a chance for further

incremental algorithm developments to ensure quick responses that adapt to the evolution of the stream.

Subsequence outliers and outlier time series have been handled less frequently. Most of the subsequence outlier detection methods find non-periodic and fixed-length outliers. Indeed, there are no techniques that identify periodic subsequence outliers in multivariate time series. This can be a promising area for further investigation, especially in fields such as cybersecurity to detect periodic network attacks, or in credit-fraud detection to detect periodic thefts. Also, within techniques that detect subsequence outliers, care must be taken with the way in which clustering is performed because it has been shown that clustering all of the subsequences extracted from a time series by a sliding window produces meaningless results [163].

Not much work has been carried out on the detection of outlier time series. Other research directions in this line include using different distance measures and creating more effective methods for dealing with time-dependent variables of different lengths.

Within the dissimilarity-based techniques that detect either subsequence or whole time series outliers, the dissimilarity measure that is used influences the results obtained. The Euclidean distance accounts for the majority of all published work mentioned in this review due to its computation efficiency. However, other measures, such as DTW, could lead to an improvement in the detection of outliers because they include temporal information. An interesting research direction would be to observe how different dissimilarity measures influence the outlier detection algorithms in time series data to see if any of them improve the results of the Euclidean distance. In particular, meta-learning approaches such as that proposed by [38] could be used as they provide an automatic selection of the most suitable distance measure.

In addition to these types of outliers, there could be other types of unexplored outliers. For example, it may be interesting to detect outliers that propagate over time within different variables of a multivariate time series; that is, an outlier may begin in a unique variable and then propagate through other variables in later time steps. As far as we know, this problem has not been addressed yet in the literature, or at least it has not been done under the name of outlier/anomaly detection.

Finally, with regard to the nature of the detection method, it should be noted that some univariate techniques (e.g., the density-based) can easily be generalized to the multivariate case by considering the distance between vectors instead of points. However, complex correlations between the variables in a multivariate time series are not taken into account. This may lead us to not identify observations that look normal in each variable individually but which violate the correlation structure between variables. In addition, when applying a univariate technique separately to each variable in a multivariate time series, the correlation dependencies are ignored. This can be computationally expensive when the number of variables is large. Hence, an extension of the univariate detection techniques applied to multivariate time series should be

studied (e.g., [99, 106]) so that the correlations between variables representing complex system behaviors are included.

3

Water leak detection using self-supervised time series classification

As mentioned in Chapter 2, the detection of whole time series outliers/anomalies is a task that has almost not been treated in the literature. In this chapter, we address this problem for the purpose of identifying water leaks.

3.1 Introduction

Water leaks are of special interest for water distribution companies, not only because of the economic loss they entail but also because of the environmental consequences they generate. A water leak in a water distribution network consists of an accidental escape of water through a component of the network (e.g., a hole or a crack). Consequently, in order to continue to supply all customers with enough water, the flow of the water in the network needs to be increased until the leak is fixed.

An example of a water leak on a water distribution network in Yorkshire[1] is depicted in Fig. 3.1, where the shaded area indicates the day the leak was repaired. Indeed, a remarkable increase in the flow can be appreciated in the data between the 20^{th} and 23^{rd} of November. In particular, leaks are usually more noticeable during the night (from 1 a.m. to 5 a.m., highlighted in orange in Fig. 3.1), when customer demand is low [164].

Until recently, leakage management procedures within the water industry tended to be predominantly resource-intensive manual processes [165]. These methods are also known as hardware-based methods [66, 166] and are based on using specialized hardware equipment, such as leak noise correlators, leak noise loggers, and gas injection. Even though these methods are very accurate, they are very expensive and not practical on a day-to-day basis.

Technological advances in recent years have brought major breakthroughs in data collection, enabling a large amount of data to be gathered. This has led to the development of new automatic and effective data-driven leak detection

[1] https://datamillnorth.org/dataset/yorkshire-water-leakage-dma-15-minute-data

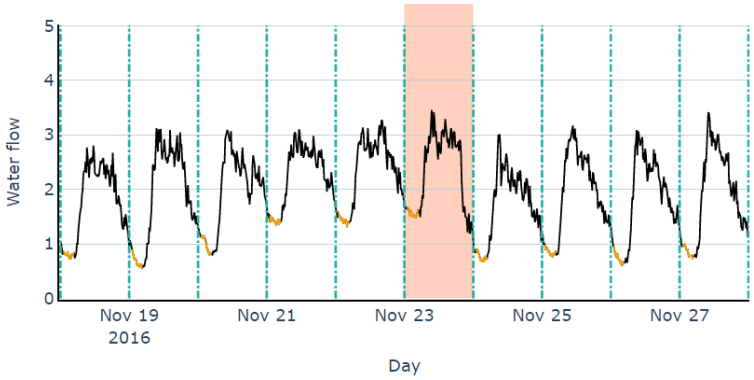


Fig. 3.1: Example of the increase that a leak has caused in the water flow.

techniques, which have been denominated software-based methods [66, 166, 167] and have become increasingly popular in recent years. These techniques are typically applied to hydrological data such as water flow or pressure data, although some methods use multiple variables simultaneously [168, 169, 170, 171, 172], or even include other types of data (e.g., data collected by acoustic emission sensors [67]).

Leak detection is not a trivial task, however, and software-based techniques also have their limitations. To begin with, these techniques rely on data, and so they will only be able to detect the leaks that are reflected in the data. Thus, they will usually not detect structural leaks or leaks that have existed from periods prior to the data collection. Typically, other alternatives, such as periodic inspections on the part of the company or those resulting from external customer calls, are needed to detect these leaks.

Additionally, when using hydrological data, leaks can often be mistaken for sporadic large consumption (e.g., filling pools in summer or occasional high consumption in industry) because both involve an unusual increase in the flow in the network. However, the high consumption of consumers tends to be sporadic, whereas leakage continues over time and causes a permanent increase in the flow until it is fixed.

In the literature, some of the software-based methods proposed for leak identification are based on supervised classification [169, 171, 67, 173], where the idea is to learn a classifier that discriminates between leakage and non-leakage periods of time.

The main drawback of these techniques is that leaks must be identified and labeled to form the training set. However, obtaining a sufficiently large number of leakage samples is complicated in real circumstances because leaks are typically not very common. Moreover, leak identification is normally carried out using reports from the water company. These reports do not always include all the existing leaks and are frequently diffuse and uncertain regarding the

recorded leaks. All this largely complicates the learning of reliable supervised classification models for leak detection.

As an alternative to the supervised classification techniques, some authors use unsupervised software-based methods, which do not need labels at training time. These methods usually assume that the training set is only composed of normal (no leakage) data.

The most widespread unsupervised method relies on the analysis of the measured Minimum Night Flow (MNF), which is the lowest flow supplied to an area during the night (e.g., 00:00–05:00 [165, 169] or 02:00–04:00 [164, 173]). A leak alarm is generated when the MNF exceeds a threshold typically set by water utilities. Although this method uses only nightly data when the customer demand is usually low, it analyzes single points, and so leaks can still be confused with occasional high night consumption or sensor failures. In this context, despite the simplicity and intuitiveness of this method, it provides many false leak alarms, which involves additional effort in the search for leaks that do not exist. Additionally, false alarms burden workers in water companies and there is a risk that workers will start ignoring alerts. Therefore, since leaks rarely occur in water distribution networks, it is desirable that the detection methods maintain a small number of false alarms even at the expense of reducing the number of detected leaks to within an acceptable range [167].

More sophisticated unsupervised methods to detect leaks rely on fitting a prediction model to normal data to predict values over time and identify the observations that significantly deviate from their predicted values. Even if the data have an evident temporal nature, most methods do not take this aspect into account or partially consider it using time windows [165, 168, 172, 174, 175, 176, 177, 178].

The prediction-based techniques proposed in the literature typically make one-step-ahead (point) predictions [170, 172, 175, 176, 177, 178, 179, 180] rather than predicting full time windows (e.g., one-day-ahead prediction) [165, 168, 174], and use threshold values to make comparisons and identify large deviations from normality. In particular, some of these methods build a model for each time step, thus generating a large number of models [170, 175, 179, 180]. Also, since point analysis produces false alarms more easily due to sudden high water usage or sensor faults, some prediction-based techniques analyze consecutive residuals [175, 176, 177, 180] to avoid confusing leakage with occasional high consumption, or apply a denoising approach before building the model to remove the noise caused by sensor faults [176, 177].

Less commonly, other unsupervised techniques which are not based on prediction models have also been proposed. Some of them identify leakage days by projecting the flow data into a space of lower dimension to detect the projections that lie outside control limits [181]. Others attempt to learn the normal behavior of flow subsequences using one-class classification [178]. Similarity-based techniques have also been used to identify flow points [182] or subsequences [183, 184, 185] with low similarity with respect to a reference of normality. These types of techniques do not usually consider the temporal

correlation of the observations beyond the inclusion of temporal features (e.g., the hour of the day) or the use of features extracted within time windows [178, 181, 183, 184]. Moreover, as with prediction-based techniques, some of these methods also build a model for every time step [182, 185], thus generating a large number of models.

In this chapter, we propose a novel water leak detection technique based on the self-supervised classification of time series, which, to the best of our knowledge, has not been used in this context before. Self-supervised learning refers to learning methods that exploit the structure of unlabeled data to provide appropriate supervision signals and thus define a new problem that can be solved from a supervised perspective. Although this general learning approach has been previously used in the literature, mainly in the field of computer vision, the methodology proposed in this chapter is the first to address the general problem of anomaly detection and, in particular, the problem of leak detection, based on the philosophy of self-supervision and using time series data.

The aim is to detect nights with water leakage by learning a model of the normal behavior using only flow data. This approach does not require a statistical or hydraulic model to be fitted to the data, nor does it require leakage labels in the training phase. Moreover, the proposed method considers the contextual information by analyzing full time series rather than point observations, and consequently, it does not require a model for every time step as other methods in the literature do. Additionally, the temporal nature of the data is taken into account by using specific classifiers for time series, instead of using time windows or temporal features, as most of the methods in the literature do. Consequently, the proposed method succeeds in detecting a high number of leaks while providing a low number of false alarms.

The rest of the chapter is organized as follows. In Section 2, the proposed methodology is described. Section 3 provides the experimental results. Finally, in Section 4, the conclusions and future research lines are outlined.

3.2 Methodology

The proposed method, *Self-Supervised Leak Detector (SSLD)*, aims to detect water leaks by learning the normal behavior of the water flow in a water distribution network. Self-supervised learning involves training a model that does not require external class labels and instead uses labels that have been assigned to artificially generated data. This is especially useful in problems where there are very few or no available labels, as is the case of water leak detection.

In this chapter, we use the self-supervised approach within the anomaly detection framework [64] (see Fig. 3.2). The idea is to define a self-labeled training dataset to learn a classifier that will be used to detect the anomalies. The self-labeled dataset is built by applying a set of different transformations

to the initial training instances, which are supposed to represent the normality, and by assigning a label, usually denominated *pseudo-label*, to the data instance obtained from each transformation. This self-labeled dataset is then used to learn a classifier that discriminates between the different transformations applied. Finally, the learned classifier will be used to determine whether new samples are anomalous based on a decision rule.

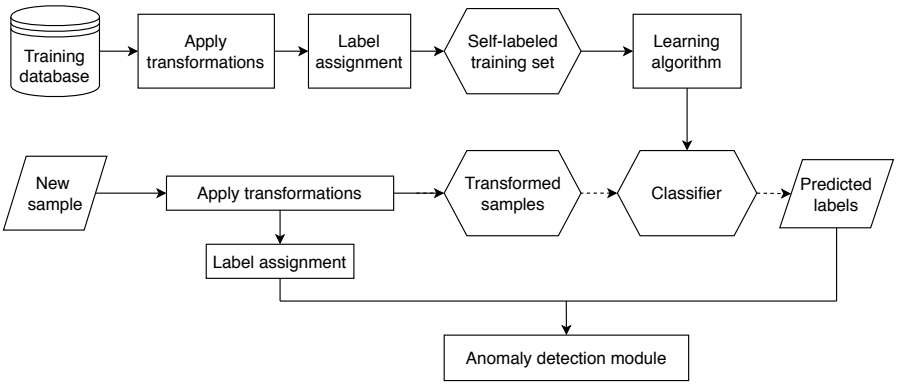


Fig. 3.2: Illustration of the self-supervised approach for anomaly detection.

The rationale behind this approach is that learning to distinguish between the different transformations applied to the data also helps to learn features that are likely to be unique to normal samples: since these features do not appear in the same way in anomalous samples, the classifier will fail to discriminate between the applied transformations. Indeed, the type of data at hand and the characteristics of the anomalies to be detected will be determinant when defining the transformations in this self-supervised framework. For example, in the context of images, self-supervision has recently been used to identify abnormal images [186, 187, 188] by applying transformations, such as geometrical transformations, to normal images.

3.2.1 Generation of the self-labeled dataset

In the first step of our self-supervised framework, a self-labeled training set is generated. To do this, for each univariate time series of nightly flow data $Y = \{y_t\}_{t=1}^{T}$ of length T in the initial training set, K time series are artificially generated, each one with an associated pseudo-label. These artificial time series are obtained by applying K different linear transformations (g_i where $i \in \{1, ..., K\}$), which are defined by:

$$\begin{aligned} g_i \colon \mathbb{R}^T &\to \mathbb{R}^T \\ Y &\mapsto (p_1 + ... + p_i)Y \end{aligned}$$

where $p_1 = 1$, $p_2, ..., p_K \in \mathbb{R}^+$, and $g_i(Y) = Y_i = \{(p_1 + ... + p_i)y_t\}_{t=1}^T$. Note that the first pseudo-label corresponds to the original non-transformed time series. An example of this procedure is shown in Fig. 3.3, where it can be seen that the different linear transformations increase the night water flow at different levels.

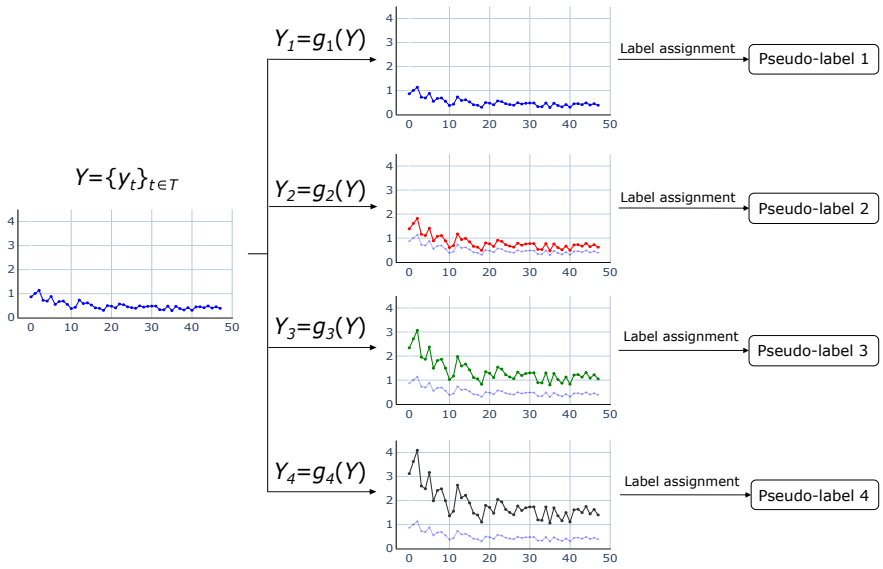


Fig. 3.3: Example of the generation of the self-labeled dataset, where $K = 4$, $p_1 = 1,\ p_2 = 0.6,\ p_3 = 1.1,\ p_4 = 0.9$.

The premise behind applying these linear transformations is that water leaks are characterized by a flow increase. Consequently, a night with a leak will resemble a normal night with an increase in flow. As such, since it is assumed that the classifier will be learned using only normal flow data, leak nights are expected to be assigned with pseudo-labels from 2 to K, depending on the severity of the leak, instead of being labeled with pseudo-label 1, which indicates that no transformation has been applied. Along the same line, classification errors will also be committed with the transformed time series obtained from a night with a leak because it is expected that the classifier will incorrectly assign higher pseudo-labels, which represent higher levels of flow.

Once the transformations have been defined, the training set that will be used to learn the self-supervised classifier has to be generated, as shown in Fig. 3.2. Let $D = \{Y^1, ..., Y^N\}$ be the initial training dataset consisting of N normal time series (with no leakage). Then, $\mathbb{T} = \{(\mathrm{Y}_1^1, \mathrm{c}_1^1 = 1), ..., (\mathrm{Y}_K^1, \mathrm{c}_K^1 = K), ..., (\mathrm{Y}_1^T, \mathrm{c}_1^T = 1), ..., (\mathrm{Y}_K^T, \mathrm{c}_K^T = K)\}$ is the generated set of $T \times K$ time

series with their associated class labels, where Y_j^i is the j^{th} transformed time series of Y^i and j is its associated label indicating the applied transformation (see Fig. 3.3). $\mathbb{T}$ is the training set that will be used to learn the classifier.

It should be noted that, in practice, it is difficult to ensure that the initial training set is only composed of normal flow data. In principle, we can discard identified leaks from this set if the information is available, but there may remain leaks that have gone unnoticed. However, leakage occurs very infrequently, so the training set will be predominated by non-leakage days.

3.2.2 Construction of the classifier

The aim of this step is to train a classifier $\mathcal{F}$ that learns to discriminate between the K linear transformations applied in the previous step. In other words, the purpose is to learn the mapping between each input time series and its corresponding label. In order to consider the temporal nature of the data at hand, and contrary to other leak detection methods in the literature, in this chapter we adopt a time series classification approach. This allows the contextual information to be considered rather than single observations, which is particularly helpful when detecting leaks because they remain over time.

Within the existing time series classifiers, we have chosen the Random Interval Spectral Ensemble (RISE) [32] due to the accuracy shown in other problems in the literature and also due to its robustness to noise, which makes it especially useful for the problem of leak detection in water distribution networks given that flow data are particularly noisy. RISE is an ensemble time series classification algorithm that consists of building a set of trees, each of which focuses on a randomly chosen time interval (subsequence) of the data. In particular, this method extracts spectral features over each random interval to learn the time series forest. It should be noted that our methodology is not limited to the RISE classifier, and thus, other classifiers could also be used.

3.2.3 Deployment of the model for anomaly (leak) detection

As shown in Fig. 3.2, to apply the learned classifier to a new time series Y^{new}, first, the K transformations have to be applied: $\{Y_1^{new} = g_1(Y^{new}), ..., Y_K^{new} = g_K(Y^{new})\}$. Then, the classifier is used to predict the label of each generated time series ($\{Y_1^{new} \xrightarrow{\mathcal{F}} \hat{c}_1^{new}, ..., Y_K^{new} \xrightarrow{\mathcal{F}} \hat{c}_K^{new}\}$).

Leaks are characterized by higher levels of flow, and so, it is expected that their pseudo-labels will not be correctly predicted by the self-supervised classifier. However, the severity and typology of the leak could have an influence on the output of the classifier. For example, small leaks could result in fewer misclassified pseudo-labels or different misclassification patterns in comparison to very severe leaks. Similarly, nights with no leaks could also present a few misclassified labels due to errors in the classifier or variations

in the normal patterns. In this context, a decision rule that will determine whether a night has suffered a leak must be defined based on the outputs of the self-supervised classifier in all its transformations.

In our method, the new time series Y^{new} is flagged as an anomaly if the classifier assigns pseudo-label K to at least two of the transformed time series of Y^{new}. That is,

$$|\{i \mid \hat{c}_i^{new} = K, i \in \{1, ..., K\}\}| \geq 2. \tag{3.1}$$

Note that the K^{th} transformation provides the highest increase in the flow data. Since leaks represent an increase in the flow, we expect the predictions of the labels to be displaced upwards, so the classifier should assign the largest pseudo-label to at least two of the generated time series.

3.3 Experimentation

The evaluation of the proposed method has been performed in both a private (scenario A) and a public (scenario B) dataset of different water distribution networks. The data used in the experiments are presented in Section 3.3.1, the implementation details and parameter selection of the proposed methodology in Section 3.3.2, the evaluation framework and metrics in Section 3.3.3, the methods used to compare the results of our method in Section 3.3.4, and finally the experimental results are outlined in Section 3.3.5.

3.3.1 Data

The available datasets in both scenarios consist of water flow measurements recorded over time at a certain granularity and divided into different zones of a particular locality. From all these data, the time series that will be used as input for the model consists of the flow measurements collected during the night period from 1 a.m. to 5 a.m., when water consumption is usually the lowest.

In addition to the flow measurements, both scenarios also contain information about the leaks, indicating the date on which a work report associated with the repair of each leak has been generated. Some of the recorded leaks refer to structural leaks or those that are not reflected in the data. Thus, from all the leaks, we only focus on those that correspond to the *detectable leaks* or, in other words, the leaks that generate a noticeable increase in the flow data. In particular, we will only consider those leaks that make the MNF higher than usual. In this context, and following the recommendations of the experts, the data used to calculate the MNF are collected between 2 a.m. and 5 a.m.

To sum up, each zone within each scenario has both an associated time series database of nightly flow data and the work reports of the leaks. Since

the flow behavior for each type of day is different, we divide the time series database into seven smaller databases, one for each day of the week. Furthermore, as shown in Fig. 3.4, each of these databases is divided into two new sets: the training dataset, which will be used to train the models, and the leakage dataset, which will be used to validate the models. To build the leakage dataset, the nights with a leakage record are separated from those with a non-leakage record. Since not all the leakage records represent detectable leaks, a predefined threshold value is used to select the nights on which the MNF exceeds it. These nights form the leakage dataset.

Regarding the training dataset, it should be composed only of normal days because our approach aims to learn the normal behavior of the water flow. However, even after separating the nights with a leakage record from the training dataset, there still might be unusual days that may misguide the classifier and should be discarded. Some of these unusual days are related to the leakage records: since a leakage report refers to the date when the leak is fixed, and not when it appeared, we also remove the three days prior to the leakage report, in accordance with expert guidance. Holidays might also show a different behavior, and thus, they have also been excluded from the training sets. The nights after removing both those unusual days and the nights with a leakage record form the final training dataset.

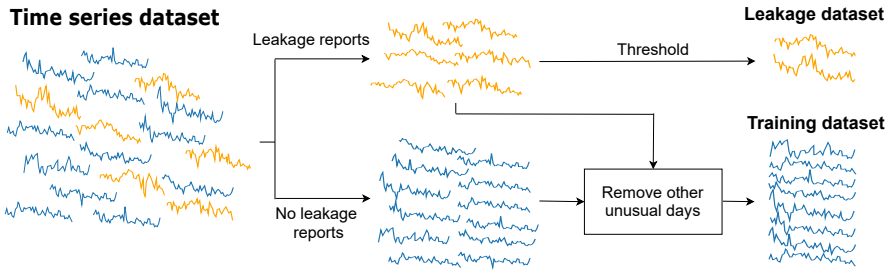


Fig. 3.4: The training and leakage datasets considered in the experimentation.

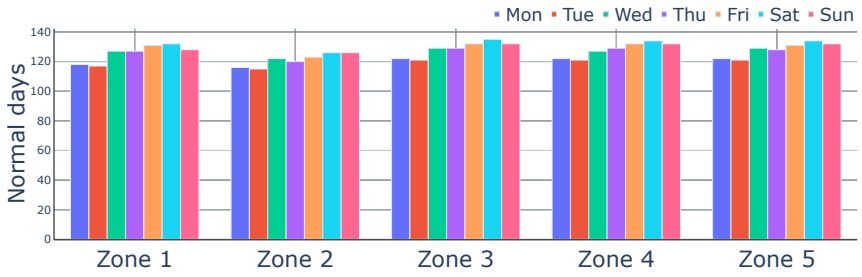
3.3.1.1 Scenario A

The dataset in this scenario consists of water flow measurements collected every 5 minutes over three years (2017–2019) in a private water distribution network located in Azkoitia, a town in the Basque Country. This network, which is managed by *Gipuzkoako Urak S.A.*[1], has five zones that measure the flow in different areas of the town, with at least one detectable leakage record in the study period. Each zone has very distinct flow patterns, and so we apply the proposed method to each zone individually.

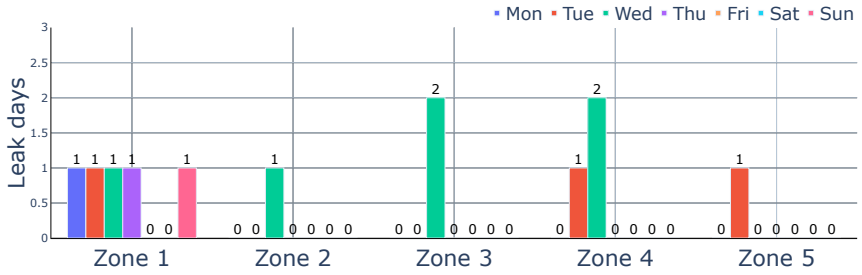
[1] https://www.gipuzkoakour.eus/

As additional useful information, this dataset contains information about the threshold value established by the water utility company, which is used to raise alarms in the MNF approach and is based on the annual flow values and other objectives set by the company (e.g., reducing the cost of the water loss every year). Recall that this threshold is also used to select the detectable leaks and thus define the leakage dataset.

Fig. 3.5a shows the number of days that are considered to be normal and thus belong to each training set. Similarly, Fig. 3.5b presents the number of reported leakage days used to test the models. Note the challenge of addressing this as a supervised problem due to the small number of detectable leakage records per training set.



(a) Number of normal days in each training set.



(b) Number of detectable leaks in each leakage dataset according to the provided threshold value.

Fig. 3.5: Number of samples in the training and leakage sets in scenario A.

3.3.1.2 Scenario B

Yorkshire Water is a water supply and treatment utility company in England that has released its data related to different domains, such as pollution, consumption, and leakage. In this chapter, we use the leakage dataset[1], which

contains the records of the water flow of more than 2000 Distribution Management Areas (DMAs) organized in 20 regions, with a 15-minute granularity for one year (April 2016–April 2017).

In contrast to scenario A, the water utility company does not explicitly provide any threshold for the analysis of the MNF. Thus, we define this threshold as a percentile value ($80, 85, 90$, and 95) of the MNF data of the training sets.

Due to the high number of DMAs, a single region (called E1 in the original database) has been chosen for the purpose of this chapter. Even if this region is composed of 117 DMAs, we have specifically selected the DMAs where:

a) there are no "invalid" flow records[1].
b) data are available for the whole year.
c) all flow records are non-negative.
d) there is at least one day with a detectable leakage record.

Note that the chosen threshold value will directly influence the number of detectable leaks and thus the size of the leakage dataset. In particular, the lower the threshold value, the greater the number of detectable leaks and the more DMAs that will be analyzed.

Taking this into account, Table 3.1 shows the final number of DMAs to be analyzed for the different threshold values. This table also describes the training and leakage datasets for each percentile and type of day, presenting the mean number of time series per DMA, together with the standard deviation. As an example, 24 DMAs have been analyzed with the 80^{th} percentile, and on average, these DMAs have 43.83 series in the training set used for Mondays and 0.67 series in the leakage set used for the same type of day. As the standard deviation is small, the actual number of elements in each DMA is close to these average values.

For all types of day, the number of nights in the training sets is quite uniform. Regarding the number of leaks, the DMAs contain very few leaks per type of day, and, in total, between 2 and 3 detectable leaks on average. Note that the percentiles 90 and 95 provide the same training datasets but different leakage datasets.

[1] The dataset contains an extra column called 'Flow Validity Code' that indicates if the record is valid ('V'), invalid ('I') or missing ('M').

Table 3.1: Description of the training and leakage datasets for each threshold value. The mean number of samples per DMA and day is shown, together with the standard deviation between parentheses.

Percentile	**Analyzed DMAs**	**Set**	**Mon**	**Tue**	**Wed**	**Thu**	**Fri**	**Sat**	**Sun**
80	24	Training	43.83 (2.43)	47.83 (2.37)	49.17 (1.79)	49.46 (1.98)	50.38 (2.14)	49.17 (1.99)	48.25 (1.89)
		Leakage	0.67 (0.92)	0.42 (0.72)	0.54 (0.72)	0.50 (0.78)	0.29 (0.55)	0.33 (0.48)	0.25 (0.53)
85	23	Training	43.78 (2.47)	47.70 (2.32)	49.09 (1.78)	49.35 (1.94)	50.30 (2.16)	49.13 (2.03)	48.17 (1.90)
		Leakage	0.61 (0.94)	0.35 (0.57)	0.57 (0.73)	0.52 (0.79)	0.30 (0.56)	0.35 (0.49)	0.22 (0.52)
90	22	Training	43.64 (2.42)	47.55 (2.26)	49.05 (1.81)	49.27 (1.96)	50.23 (2.18)	49.09 (2.07)	48.09 (1.90)
		Leakage	0.59 (0.91)	0.36 (0.58)	0.45 (0.51)	0.41 (0.73)	0.32 (0.57)	0.27 (0.46)	0.18 (0.39)
95	22	Training	43.64 (2.42)	47.55 (2.26)	49.05 (1.81)	49.27 (1.96)	50.23 (2.18)	49.09 (2.07)	48.09 (1.90)
		Leakage	0.55 (0.86)	0.27 (0.46)	0.41 (0.50)	0.41 (0.73)	0.23 (0.43)	0.27 (0.46)	0.18 (0.39)

3.3.2 Implementation details

The number of transformations that will be applied to build the self-labeled training sets has been set to $K = 4$ (four possible pseudo-labels), and the transformation parameters initially to $p_1 = 1$, and $p_2 = p_3 = p_4 = 0.7$. Note that very small transformation parameters would hinder the differentiation between these pseudo-labels, thus complicating the learning of the classifier, while very large parameters would make this difference too obvious, making it difficult to detect less severe leaks.

The parameters specified for the RISE classifier have been the number of trees, w, and the minimum interval length, l. While the number of trees $w \in \{10, 11, ..., 20\}$ has been chosen using a grid-search with 5-fold Cross-Validation (5-fold CV), the minimum interval length has been set to be $l = min(16, N/2)$, where n is the length of the time series, taking [189] as a reference.

Finally, as mentioned on more than one occasion, our method is a data-based one that can only detect leaks that generate a noticeable increase in the flow data (*detectable leaks*). As such, only the nights on which the MNF is higher than usual (i.e., MNF$> \tau$, where τ is a predefined threshold) will be introduced into our self-supervised classifier. In the deployment step, the nights which do not have an appreciable increase in the flow (MNF$\leq \tau$) will be directly labeled as "non-leak nights", without undergoing further analysis. In summary, our methodology can be seen as a 2-stage approach, where in the first stage, a night is predicted as a detectable leak or not using a threshold on the MNF, and in the second stage, only the detectable leaks are classified using SSLD to eliminate some of the false positives.

3.3.3 Evaluation framework and metrics

The evaluation of the proposed method is performed in terms of both the False Positive Rate (FPR), which estimates the false leak alarms generated in the training dataset, and the True Positive Rate (TPR), which measures the capacity to detect leaks in the leakage dataset (both datasets defined in Fig. 3.4).

As shown in Fig. 3.6, the number of false alarms or the FPR is estimated using a nested 5-fold CV in the full training set, in which the inner loop is used to tune the w parameter of the RISE classifier, and the outer loop is used to estimate the FPR. More specifically, the training set is split into 5 folds, and for each iteration $i \in \{1, ..., 5\}$, 4 of the folds are used to perform a grid search over the w parameter of the RISE classifier, using another 5-fold CV. The metric used in the inner loop is the classification accuracy, which computes the rate of correctly classified transformations. Once w has been selected, a model is trained using those 4 folds, and the resulting model is validated on the remaining fold to compute the FPR.

Similarly, to estimate the number of detected leaks or the TPR, first, the w parameter is tuned using a 5-fold CV in the training set, also based on the classification accuracy. Note that this accuracy is not calculated in terms of leak/no leaks, but in terms of the pseudo-labels created in the self-supervised context. Then, the final model is learned using all the training data and the selected best w parameter. This model is applied to the leakage set to compute the TPR.

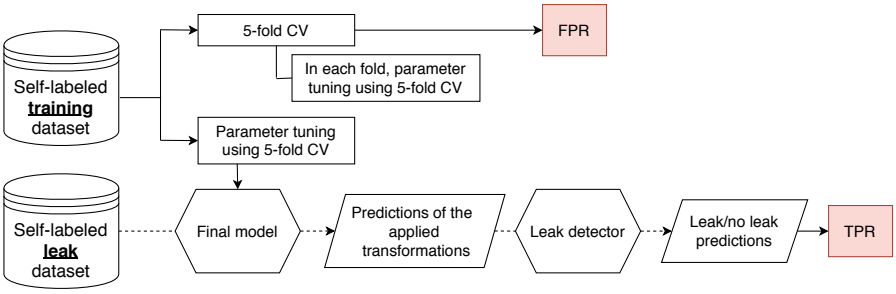


Fig. 3.6: Evaluation framework of the proposed methodology.

3.3.4 Comparison with other methods

The results of our method have been compared with two other unsupervised techniques. One of these techniques, which we will consider the baseline, is the traditional MNF method [164, 165, 169, 173]. Even though this method identifies all the detectable leaks by default, it usually provides a high number of false alarms. In particular, our method aims to reduce this quantity of alarms while maintaining a high number of detected leaks. Note that the FPR of the SSLD will never be greater than that provided by the MNF method because, as explained in Section 3.3.2, we will automatically label as non-leakage all the nights that have no noticeable increase in the flow (MNF$\leq \tau$).

We also compare our method with the ϵ-SVR prediction-based method [175] as prediction-based methods have been widely used to identify leaks in the literature. Unlike other prediction methods that do not provide the code and do not clearly state the implementation details, this method contains all the necessary specifications to reproduce it. The procedure followed and the parameter values established are outlined in [175]. We have applied this method to the nightly time series, as in our method, with an embedding dimension of one hour.

3.3.5 Experimental results

In the following sections, the results of the experiments performed in both scenario A (Section 3.3.5.1) and scenario B (Section 3.3.5.1) are presented. In

particular, we first provide a comparative analysis concerning the results of other methods and then a sensitivity analysis of the transformation parameters of the self-supervised framework.

3.3.5.1 Scenario A

The results for each zone and for the initially fixed transformation parameters ($p_1 = 1$, $p_2 = p_3 = p_4 = 0.7$) are summarized in Table 3.2. From the three methods that we have compared, MNF is the only method that can perceive all the detectable leaks. However, our proposed SSLD method provides significantly fewer false positives while still being able to identify almost all the leaks: taking into account all the zones, 91.67% of the leaks are detected. The reduction in the FPR is better appreciated in the *mean FPR* column in Table 3.2, which represents the average value of the FPR of all the day types for each zone. Also note that although the ϵ-SVR method obtains an even lower FPR than our SSLD method in most of the zones, it detects very few leaks. Moreover, in the only zone that it can detect all the leaks (Zone 5), it has a remarkably high FPR.

In conclusion, in finding a trade-off between the detected leaks and the number of false positives, our method is the most successful.

Table 3.2: Results in each zone of scenario A. The FPR values for each day and method are shown, along with the mean FPR and TPR values of all the models of each method.

Zone	**Method**	**FPR**							**Mean FPR**	**Mean TPR**
		Mon	**Tue**	**Wed**	**Thu**	**Fri**	**Sat**	**Sun**		
Zone 1	SSLD	0.1699	0.1283	0.1588	0.1422	0.1678	0.1808	0.2277	0.1679	**1.0000**
	MNF	0.3040	0.2915	0.3164	0.3644	0.2672	0.3088	0.3460	0.3141	**1.0000**
	ϵ-SVR	**0.0763**	**0.0950**	**0.1108**	**0.0788**	**0.0684**	**0.0527**	**0.0800**	**0.0803**	0.2000
Zone 2	SSLD	0.1717	0.2000	0.2135	0.2167	0.1722	0.2222	0.2065	0.2004	**1.0000**
	MNF	0.2841	0.2435	0.2878	0.3250	0.3074	0.4840	0.8889	0.4030	**1.0000**
	ϵ-SVR	**0.0087**	**0.0087**	**0.0167**	**0.0083**	**0.0167**	**0.0000**	**0.0000**	**0.0084**	0.0000
Zone 3	SSLD	0.2199	0.1560	0.1556	0.1614	0.2055	0.2444	0.2269	0.1957	0.5000
	MNF	0.3186	0.3220	0.3319	0.3079	0.3104	0.3926	0.3863	0.3385	**1.0000**
	ϵ-SVR	**0.0083**	**0.0080**	**0.0160**	**0.0000**	**0.0077**	**0.0074**	**0.0000**	**0.0068**	0.0000
Zone 4	SSLD	0.1301	**0.0910**	0.1399	0.1465	0.1220	0.1123	**0.1132**	0.1221	**1.0000**
	MNF	0.1551	0.1573	0.1867	0.2221	0.1758	0.1487	0.1516	0.1711	**1.0000**
	ϵ-SVR	**0.0897**	0.1153	**0.1022**	**0.0938**	**0.1148**	**0.0815**	0.1275	**0.1036**	0.6667
Zone 5	SSLD	**0.0910**	**0.1410**	**0.1774**	**0.1966**	**0.1755**	**0.1595**	**0.1132**	**0.1506**	**1.0000**
	MNF	0.1314	0.1823	0.2003	0.2651	0.2285	0.1749	0.1203	0.1861	**1.0000**
	ϵ-SVR	0.9596	0.9833	0.9920	0.9626	0.9692	0.9923	1.0000	0.9799	**1.0000**

Analysis of the transformation parameters

Once we have analyzed the ability of our method to detect leaks while maintaining low levels of false positives, in this section, we study the robust-

ness of our method to changes in the transformation parameters. In particular, while keeping the RISE parameters previously obtained, the aim is to analyze how the FPR and TPR change with respect to the parameters p_2, p_3, p_4 (recall that $p_1 = 1$). The transformation parameter space considered is $p_2, p_3, p_4 \in \{0.5, 0.7, 0.9, 1.1, 1.3, 1.5\}$. All possible combinations are considered, which total 216 combinations.

The FPR and TPR values obtained with different parameter combinations for each zone and model (type of day) are shown in Fig. 3.7. Due to the low number of leaks that the ϵ-SVR method detects, the results of this section are only compared with the MNF method.

As expected, our proposed method reduces the FPR of the MNF method, regardless of the value of the chosen transformation parameters (see Fig. 3.7). This reduction is particularly noticeable in zones and types of day where the MNF method provides many false alarms (e.g., weekend models of Zone 2), as it has more room for improvement.

Regarding the number of detected leaks, and only considering the types of days that have at least one leak, 52.96% of the parameter combinations are able to detect all of them in Zone 1, 79.63% in Zone 2, 0.46% in Zone 3, 85.42% in Zone 4, and 100% in Zone 5. The reason for obtaining worse results in Zones 1 and 3 is that both zones have small leaks that complicate their detection. If we analyze these zones in more detail, most parameter combinations in Zone 1 detect all the leaks on Mondays and Tuesdays (i.e., 81.02% on Mondays and 86.57% on Tuesdays), but the percentage is lower on the other types of days (30.09% on Wednesdays, 44.91% on Thursdays, and 22.22% on Sundays). In Zone 3, there are only two detectable leaks on Wednesdays, and although the method can hardly detect both of them, 24.07% of the combinations can detect at least one of the leaks.

In Fig. 3.8, the average performance of each parameter combination over all the models (day types) is studied. In particular, this figure shows the mean FPR obtained for different parameter combinations and highlights with a blue cross those that can detect all the leaks. Also, within the combinations that can detect all the leaks, the one that gives the lowest FPR is marked (e.g., $p_1 = 1$, $p_2 = 0.5$, $p_3 = 1.3$ and $p_4 = 0.7$ in Zone 1). As expected, all the transformation parameters reduce the mean FPR value in comparison to the MNF method, and even though some are not able to detect all the leaks, considering all zones together, 61.57% of combinations detect all of them.

To conclude, taking into account these results, our proposed method is robust to the transformation parameters, especially in terms of FPR.

Finally, the downward trend in some of the zones (Zone 1, Zone 2, and Zone 3) in Fig. 3.8 suggests that some parameters have more influence on the FPR value than others. In particular, the last parameter has a greater influence than the rest of parameters (i.e., the higher the p_4 is, the lower the mean FPR becomes). For example, the FPRs of the first 36 points, which correspond to $p_4 = 0.5$, are higher than the FPRs of the last 36 points, which correspond to $p_4 = 1.5$. Note that larger p_4 values make the difference between

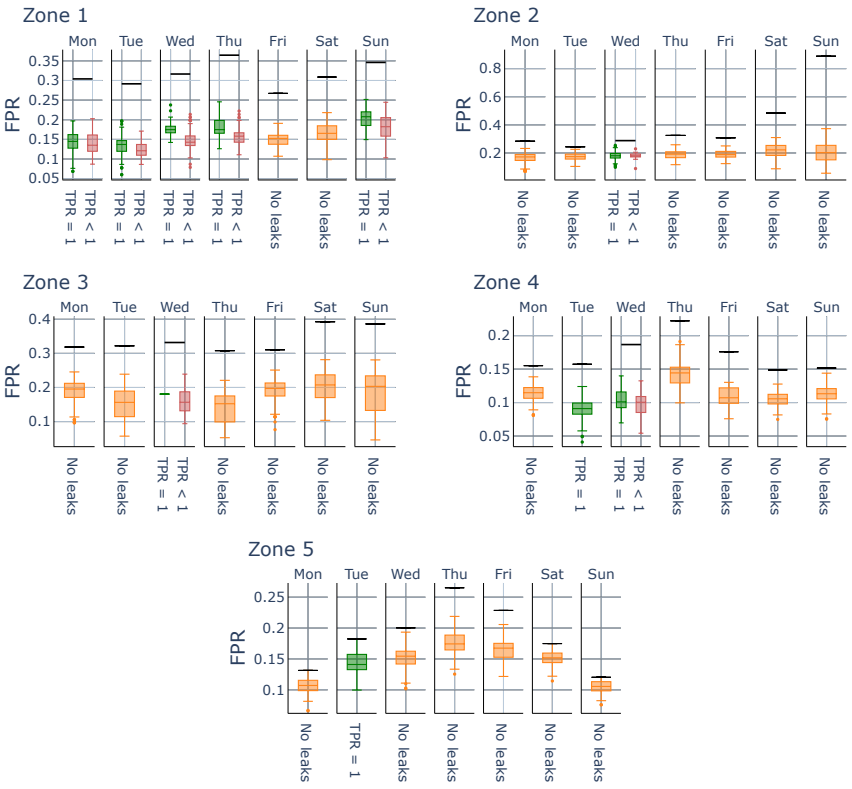


Fig. 3.7: FPR and TPR of the models obtained with all the different transformation parameters. *NO LEAKS* indicates that there are no leaks in the leakage dataset for the given day of the week, $TPR = 1$ indicates that all the leaks have been detected, and $TPR < 1$ that the method has not identified all the leaks. The baseline FPR of the MNF method is highlighted with a black horizontal line.

the magnitude of pseudo-label 3 and pseudo-label 4 higher (the labels used when identifying leaks), and thus, the classifier discriminates better between them, providing fewer false alarms.

3.3.5.2 Scenario B

The DMAs to be analyzed have been chosen according to the criteria established in Section 3.3.1.2, and the proposed SSLD method is applied to each of them individually. The results obtained for different threshold values are shown in Fig. 3.9, where the mean FPR values for each DMA are shown in the left column and the mean TPR values in the right column.

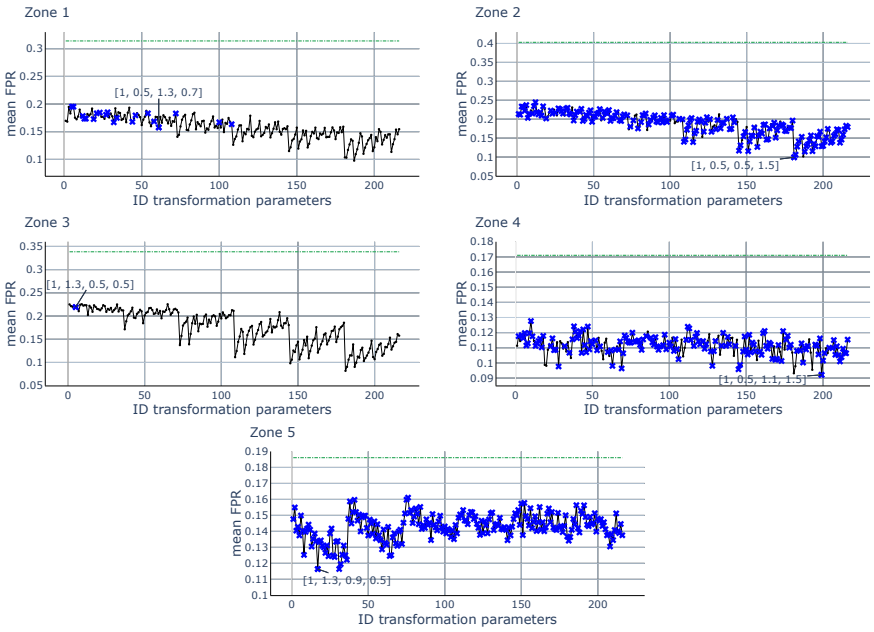


Fig. 3.8: Mean FPR and TPR according to different transformation parameters in each zone. The IDs assigned to the transformation parameters in this figure are ordered based on a triple loop of the parameters in ascending order. That is, the first six points in the graphs indicate the parameter combinations $[1, p_2, 0.5, 0.5]$, where p_2 traverses all the parameter space in ascending order, the next six points refer to $[1, p_2, 0.7, 0.5]$, and so on.

For any threshold obtained with the 80^{th}, 85^{th}, 90^{th} or 95^{th} percentiles, both the SSLD and the ϵ-SVR methods reduce the mean FPR of the MNF method. Note that the higher the percentile is, the lower the FPR of the MNF becomes, and therefore the lower the room for improvement is. As in scenario A, the ϵ-SVR method detects very few leaks, while our method able to detect most of them. In particular, the higher the percentile value is, the higher the mean TPR our method obtains. Consequently, the best threshold would be the one that provides the best detection rate which, in this case, is obtained using the 95^{th} percentile, because our method reduces the FPR of the MNF regardless of the chosen threshold. With this value and considering all the DMAs together, our SSLD method detects 82.35% of the leaks.

It should be noted that the training sets in the DMAs where our method reaches the lowest TPR have very variable flow values. Two examples are shown in Fig. 3.10. The red vertical lines indicate the reported leakage records, which, depending on the selected threshold, will be considered detectable or not. In these datasets, there is no clear pattern of normality. In fact, the training set contains several days that are assumed to be normal but have

Fig. 3.9: Mean FPR and TPR for each DMA and percentile value.

similar flow values to the leak days. This complicates the learning of the normality, and thus additional information would be needed to differentiate the leakage days from the normal days.

Analysis of the transformation parameters

As with scenario A, this section analyzes the influence of the parameters $p_2, p_3, p_4 \in \{0.5, 0.7, 0.9, 1.1, 1.3, 1.5\}$ (recall that $p_1 = 1$) over the FPR and the TPR, for the threshold value obtained with the 95^{th} percentile. This

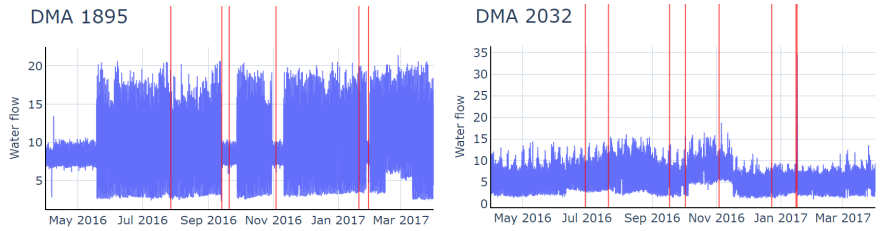


Fig. 3.10: Example of DMAs where the proposed method obtains a low TPR.

threshold has been chosen for simplicity as it provides the best performance regarding the detected number of leaks. Note that a total of 216 possible combinations are considered. Also, the results in this section are only compared to the MNF method because it provides better performance than the ϵ-SVR method.

Fig. 3.11 shows the mean FPR and TPR obtained for each DMA with all the different transformation parameters. Taking into account all the DMAs together, the mean FPR value is significantly lower for our method than that obtained by the MNF method for all the parameter combinations, as expected. The reduction in the FPR is better appreciated in the figure on the left in Fig. 3.11. Regarding the number of detected leaks and taking into account all the DMAs together, most combinations of parameters (80.25%) are able to detect all the leaks.

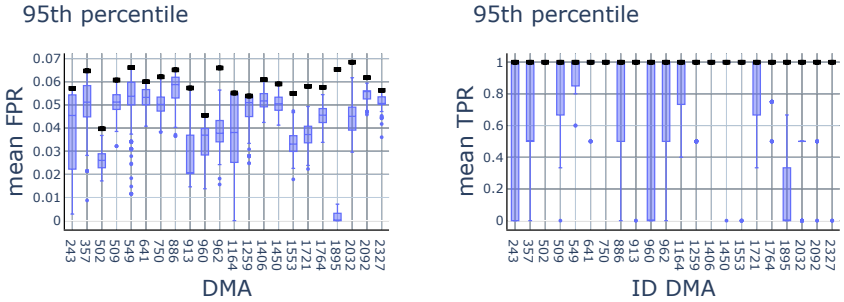


Fig. 3.11: The mean FPR and TPR for different transformation parameter combinations in each DMA. The values obtained with the baseline MNF method are highlighted with a horizontal black line.

In conclusion, our proposed method succeeds in significantly reducing the FPR in comparison to the baseline MNF method and also in detecting most of the detectable leaks.

3.4 Conclusions and Future Work

In this chapter, we have proposed a water leak detection method based on the self-supervised classification of nightly flow time series. The classifier we have chosen (RISE) is specific for time series, which allows considering the temporality of the data, and it is also robust to noise. In particular, our approach builds one model per day of the week, thus it has far fewer models than other methods that require a model for each time step. In addition, the proposed method is entirely data-driven and therefore does not require in-depth knowledge about the dynamics of the series.

The results obtained from the experiments show that, in comparison to other methods in the literature, the proposed SSLD method obtains the best trade-off between detecting the majority of the detectable leaks and providing a low FPR. Specifically, the results have been compared with the MNF, the ϵ-SVR, which, contrary to our method, either detect very few leaks or provide many false alarms.

Several combinations of transformation parameters have also been considered to define the self-labeled dataset that is used as input for the classifier. In all the considered scenarios, our method significantly reduces the FPR of the traditional MNF method, so the SSLD is robust to the choice of transformation parameters in terms of FPR. In particular, the higher the FPR of the MNF, the higher the reduction provided by our method. Although the transformation parameters are more sensitive regarding the TPR in some of the zones, they are in general able to detect most of the detectable leaks.

The main avenue that the results open for future research is related to the type of transformation applied to the data. The self-labeled dataset has been formed using linear transformations, but other types of transformations could also provide appropriate or even better results. An interesting future line of research would be to develop a theory about these transformations, taking into account both the type of available data in the training set and the type of anomalies to be detected. An additional interesting research line would be to test this approach on other types of problems different to leak detection with other types of data and anomalies.

Another promising future work would be to consider the correlation between different zones in the network and to address the self-supervised classification approach from a multivariate perspective with hierarchically structured time series. Although leaks are usually reflected in each zone, and it is generally sufficient to analyze each zone individually, additional information from the network could help to improve the results.

Finally, the proposed method deals with regularly sampled time series of the same length, so future research could also focus on improving this method in situations in which the time series are irregularly sampled or of variable lengths. This will allow series with missing values to be handled, which could appear in this type of problem where sensor failures may occur.

Part II

Contributions to time series with missing values

4

Selective imputation for multivariate time series with missing values

The methodological contributions presented in the previous part assume that the time series do not contain missing values and that they are regularly-sampled. However, as mentioned in Section 1.3, time series often contain missing values and are thus incomplete. In this chapter, we propose a new method that deals with this problem in multivariate time series datasets.

4.1 Introduction

The presence of missing values is known to hinder the analysis of time series data and complicate the downstream application of machine learning algorithms for tasks such as classification or anomaly detection [190]. Therefore, it is an important task to address the issue of missing values.

Techniques in the literature usually tackle this problem using imputation methods. In general, we can categorize the imputation methods for time series into: 1) agnostic methods, which are defined as pre-processing methods and are independent of the downstream machine learning task, 2) intrinsic methods, which are defined within the downstream machine learning algorithm that will be applied.

Among the agnostic imputation methods, basic imputation techniques such as forward filling [191, 192], zero imputation [191], or mean imputation [192] have been widely used. More advanced techniques such as Generative Adversarial Networks (GAN) [193, 194] have also been proposed in this category. The main advantage of these techniques is that they can be used in combination with any machine learning task (e.g., forecasting, classification, or clustering) as they do not depend on the task itself.

In contrast, the intrinsic methods for multivariate time series are usually defined for classification tasks and use the information of the labels of the time series to impute the missing values [190, 195, 196]. As an example, Gaussian Processes have been used together with deep learning methods to obtain the imputed values [197, 198, 199]. Note that the imputations provided by these

techniques are specific for the model and machine learning task (e.g., classifier) used.

In both the agnostic and intrinsic cases, the set of time points to impute needs to be determined beforehand. A naive solution is to impute all the missing values in all of the time points, assuming that the time series is regularly-sampled [195, 196, 200, 201, 202]. This solution is frequently adopted in the literature because many machine learning models require regularly-sampled time series without missing values (i.e., fully-observed time series) [14]. Indeed, it is common to consider that the time series has an hourly sampling [191, 192, 197, 203]. However, these methods tend to make too many imputations; as an extreme example, they carry out imputations even in the time points where there is no measurement in any of the variables. Imputing so many missing values can produce high errors and affect the results of downstream tasks, especially when the missing rate is high [43].

As such, more advanced techniques rely on imputing only the missing values in the time points where at least one of the variables has been observed [190, 204]. The resulting time series may have an irregular elapsed time between consecutive observations. Thus, for downstream tasks such as classification, techniques in this group require choosing algorithms that are capable of dealing with irregularly-sampled time series. Although these techniques need to impute fewer values than in the previous case, it is questionable whether imputing all those data points is necessary to adequately represent the time series.

In this chapter, we propose an agnostic method to selectively impute the missing values in a collection of multivariate time series, for the first time in the literature. In particular, the method selects the best subset of time points to impute based on the idea that selecting many time points can lead to a poor quality of the imputations, while selecting few time points can lead to a poor representation of the time series. We propose to address the selective imputation problem as a multi-objective optimization problem, and for this, we specifically exploit the beneficial properties of the Multi-task Gaussian Process (MGP). In this way, the proposed method allows to shorten and simplify the time series, besides reducing both the error introduced by the imputations and the cost in different aspects (e.g., computational cost or the cost associated with the data collection).

The rest of the chapter is organized as follows. Section 4.2 defines the context of the problem to be addressed and introduces the notation used throughout the chapter. Section 4.3 presents the details of the proposed methodology. Section 4.4 provides the conducted experiments and the corresponding results. Finally, the conclusions drawn and suggestions for future work are discussed in Section 4.5.

4.2 Problem setting and notation

Let $D = \{Y^1, ..., Y^N\}$ be a time series dataset composed of N multivariate time series. Each time series Y^i is formed by L variables and contains missing values[1]. Additionally, let $\Omega = \{t_1, t_2, ..., t_T\}$ be the set of time points with at least one observation in D. An illustration of the problem setting is shown in Fig. 4.1, where the actual observations of each time series are represented by black crosses. In this example, it can be seen that D has observations in a total of eight time points (i.e., $|\Omega| = T = 8$).

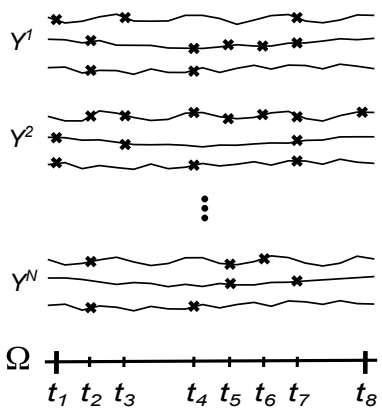


Fig. 4.1: Illustration of the problem setting.

In this context, the main focus of this chapter is to address the problem of imputing missing values of the multivariate time series in D. In particular, the objective of this chapter is twofold: 1) selecting the optimal subset of time points, which we denote as $\mathcal{P}^*$, where $\mathcal{P}^* \subseteq \Omega$, and 2) filling the missing information on those time points.

Once the selective imputation has been performed, the downstream task will be applied. Note that if the task is supervised (e.g., classification of time series), then we will additionally have a class label c^i associated with each time series Y^i.

For the sake of clarity, the notation used throughout this chapter is summarized in Table 4.1.

[1] Without loss of generality, in this chapter, we assume that the dataset has multiple time series ($N > 1$) and variables ($L > 1$), but the method is also applicable to a single time series ($N = 1$) and/or univariate time series ($L = 1$).

Table 4.1: Summary of the notation used.

$D \triangleq$	Time series dataset
$N \triangleq$	Number of multivariate time series in D
$Y^i \triangleq$	i^{th} multivariate time series in D
$L \triangleq$	Number of variables of the time series in D
$\Omega \triangleq$	Candidate set of time points
$T \triangleq$	Length of Ω
$\mathcal{P} \triangleq$	Subset of time points of Ω
$\mathcal{P}^c \triangleq$	Complementary set of $\mathcal{P}$ in Ω (i.e., $\mathcal{P}^c = \Omega \setminus \mathcal{P}$)
$\mathcal{P}^* \triangleq$	Optimal set of time points

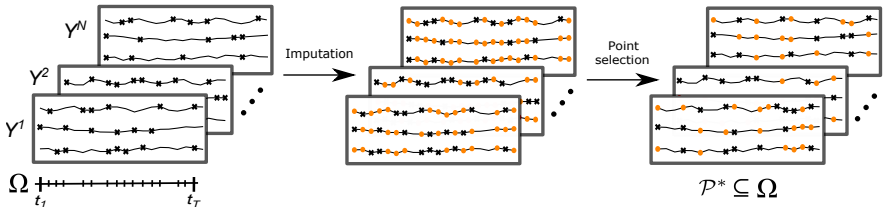


Fig. 4.2: Diagram of the proposed methodology. The estimated values are shown by orange points, while the actual observations by black crosses.

4.3 Methodology

The overall diagram of the proposed methodology is shown in Fig. 4.2. The first step consists of imputing all the missing values in the candidate set Ω (see Section 4.3.1). Then, the criterion to evaluate the different subsets of time points to impute is designed (see Section 4.3.2), and following this criterion, the optimal time points are identified (Section 4.3.3). Once the optimal subset of time points of a time series dataset has been selected, the time series are represented by those time points (see the last step in Fig. 4.2), and the downstream task would be performed using this reduced representation. The details of the methodology are explained below.

4.3.1 Imputation of the missing values

Since the time series in D contain missing values, the first step is to obtain imputations for all the time steps in Ω.

Gaussian Processes (GP) [205] have been widely used to model time series data due to their ability to naturally accommodate unequally-spaced (i.e., irregular) and uncertain observations. In a similar way, Multi-task Gaussian Processes (MGP) [206, 207] extend this capability to the multivariate time series context, allowing to consider the correlations between the variables. Both

GP and MGP are probabilistic in nature, and thus, they provide probabilistic predictions. So, these methods enable not only to impute missing values but also to provide the uncertainty of the imputations. Additionally, they are agnostic imputation techniques and are not tied to downstream tasks.

As such, to obtain the imputations of the multivariate time series in D, we leverage the multivariate and probabilistic nature of the MGP. These are key properties for later identifying the best subset of time points. See Appendix in Chapter 6 for more details on MGP.

In particular, an independent MGP will be fit to each of the multivariate time series in D. Given a multivariate time series Y^i, the corresponding model parameters will be learned using all its observed values. This can be seen as the first step of the pre-processing of the time series. Once the hyperparameters of the MGP model have been learned for each time series, we can obtain an estimated value together with its uncertainty for any missing time point in that time series.

4.3.2 Criteria for the time point selection

The next step consists of establishing a criterion to evaluate the quality of each subset of time points $\mathcal{P} \subseteq \Omega$. For this purpose, it should be taken into account that selecting a subset of time points $\mathcal{P}$ implies, on the one hand, having to impute the missing values in $\mathcal{P}$, and on the other hand, losing the actual observations that are not in this subset (i.e., observations in $\mathcal{P}^c$).

Therefore, the first criterion that we consider when evaluating a subset of time points $\mathcal{P}$ is the quality of the imputations of the missing values within $\mathcal{P}$. To this end, we quantify the uncertainty of the imputations such that low uncertainty represents high imputation quality (see Section 4.3.2.1). On the other hand, we propose a second criterion to assess the quality of the actual observations within $\mathcal{P}$ that is based on measuring the information that is lost by excluding some of the time points. In particular, the more information that is lost, the worse the set of time points $\mathcal{P}$ is. To measure this, we introduce a new concept denominated predictive capability of a set of time points (see Section 4.3.2.2).

The details of the two criteria are described in the following sections.

4.3.2.1 Quantification of the uncertainty

As the imputations have been made with a probabilistic model, the uncertainty for a subset of time points $\mathcal{P}$ can be quantified using the variances of the imputations. In particular, we quantify the uncertainty in $\mathcal{P}$ of a time series Y^i by computing the mean variance of the imputed values,

$$V_{\mathcal{P}}^i = \frac{1}{M_{\mathcal{P}}^i} \sum_{j=1}^{M_{\mathcal{P}}^i} \sigma_j^2 \tag{4.1}$$

where $M^i_{\mathcal{P}}$ is the number of missing values in time series Y^i and set $\mathcal{P}$, and σ^2_j is the variance of the j^{th} imputed value. Since we are using probabilistic models, σ^2_j is provided by the prediction of the MGP.

To illustrate the intuition behind this criterion, an example is shown in Fig. 4.3. The aim is to quantify the uncertainty of the imputed values that are represented with orange dots. Specifically, the selection of points $\mathcal{P}$ consists of four time points and contains $M^i_{\mathcal{P}} = 4$ missing values. Moreover, the uncertainty of their imputations is illustrated in blue by the confidence intervals of the predictions derived from the MGP. Based on this, in this example, the imputation of the missing value in the second variable has the poorest quality since it is the most uncertain.

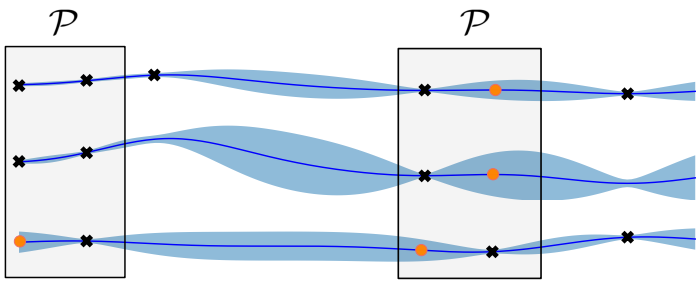


Fig. 4.3: Illustration of the uncertainty of the imputed missing values within the set of time points $\mathcal{P}$. The imputed values are shown with orange dots and the uncertainty with blue shading.

Finally, in a collection of N time series, the best point set $\mathcal{P}^*$ in terms of this first criterion is the set of points that has the smallest uncertainty:

$$\mathcal{P}^* = \underset{\mathcal{P}\subseteq\Omega}{\arg\min}\, f_1(\mathcal{P}) = \underset{\mathcal{P}\subseteq\Omega}{\arg\min}\, \frac{1}{N}\sum_{i=1}^{N} V^i_{\mathcal{P}} \tag{4.2}$$

where $f_1(\mathcal{P}) = \frac{1}{N}\sum_{i=1}^{N} V^i_{\mathcal{P}}$ measures the overall mean uncertainty of the time series dataset for point selection $\mathcal{P}$.

4.3.2.2 Quantification of the predictive capability

To measure the predictive capability of a set of time points $\mathcal{P}$, a new MGP model is learned using only the actual observations in $\mathcal{P}$. Then, we measure how well these points predict the observations that have not been included in $\mathcal{P}$ (see Fig. 4.4). The intuition is that if the points in $\mathcal{P}$ are able to predict the excluded observations accurately, then this exclusion is not causing a relevant loss of information.

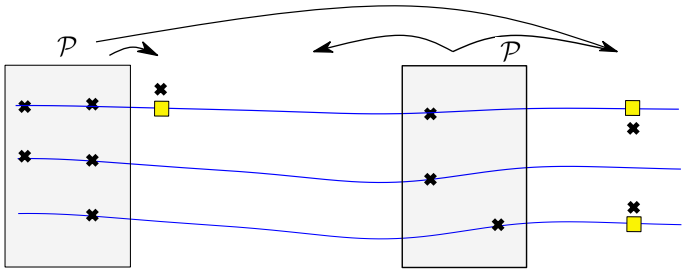


Fig. 4.4: Illustration of the predictions of the excluded observations obtained using the observations in $\mathcal{P}$. Actual observations are depicted by black crosses, and the predicted values of the excluded observations are shown by yellow squares.

To evaluate the predictive capability (PC), we propose to use the Root Mean Squared Error (RMSE) in the following way:

$$PC^i_{\mathcal{P}} = \begin{cases} \sqrt{\frac{1}{Q}\sum_{j=1}^{Q}(\hat{y}^i_{j,\mathcal{P}^c} - y^i_{j,\mathcal{P}^c})^2}, & \text{if } Q \geq 1 \\ 0, & \text{otherwise} \end{cases} \tag{4.3}$$

where Q is the total number of actual observations in $\mathcal{P}^c$ (note that $Q \geq 0$), $y^i_{j,\mathcal{P}^c}$ is the j^{th} actual observation outside $\mathcal{P}$ and time series Y^i, and $\hat{y}^i_{j,\mathcal{P}^c}$ is the respective predicted value that has been obtained using the observed values within set $\mathcal{P}$. As an example, in Fig. 4.4, there are $Q = 3$ actual observations within $|\mathcal{P}^c| = 2$ time points that have not been selected.

Finally, the best set of points $\mathcal{P}^*$ in a time series dataset consisting of N time series should obtain the maximum predictive capability globally, or, in other words, the minimum prediction error:

$$\mathcal{P}^* = \underset{\mathcal{P}\subseteq\Omega}{\arg\min}\, f_2(\mathcal{P}) = \underset{\mathcal{P}\subseteq\Omega}{\arg\min}\, \frac{1}{N}\sum_{i=1}^{N} PC^i_{\mathcal{P}} \tag{4.4}$$

where $f_2(\mathcal{P}) = \frac{1}{N}\sum_{i=1}^{N} PC^i_{\mathcal{P}}$ measures the overall mean predictive capability in the time series dataset for point selection $\mathcal{P}$.

4.3.3 Best sets of time points

The inclusion of many time points in $\mathcal{P}$ may involve having more missing values and a higher uncertainty of the imputations, but it also implies having a higher predictive capability since more actual observations are considered. On the contrary, the fewer points included in $\mathcal{P}$, the fewer missing values there will be, having a smaller uncertainty, but also worsening the predictive capability because many observations are excluded. In general, uncertainty and predictive capability are conflicting objectives.

Thus, we formulate the problem of finding the best set of time points as a multi-objective optimization problem [208, 209] in terms of 1) uncertainty and 2) predictive capability:

$$\min_{\mathcal{P} \subseteq \Omega} \left(f_1(\mathcal{P}), f_2(\mathcal{P}) \right) \tag{4.5}$$

The objective of this optimization is to find a Pareto set similar to the one that can be seen in Fig. 4.5. As illustrated in this figure, all the solutions in the Pareto set contain non-dominated solutions (i.e., subsets of time points), that is solutions that cannot be improved in any of the objectives without worsening the other objective. Note that this set dominates all solutions within the shaded region.

In particular, the two extreme solutions of the Pareto set in our problem are highlighted by green crosses in Fig. 4.5. One of the extreme solutions corresponds to selecting all time points in Ω and is located at the bottom right in the figure (i.e., large f_1, and $f_2 = 0$). In this case, the prediction error is the minimum that can be obtained because no observations are excluded, while the uncertainty is very high since all missing values need to be imputed. Conversely, the other extreme solution, which is located on the top left of the figure (i.e., $f_1 = 0$, and large f_2), involves selecting a small set of time points in which no imputation has to be performed, and therefore, the uncertainty is 0 (i.e., the minimum that can be obtained). At the same time, this extreme set may contain very few time points and thus has the worst predictive capability because much information is lost and it is not able to reconstruct the excluded observations as well as other subsets in the Pareto.

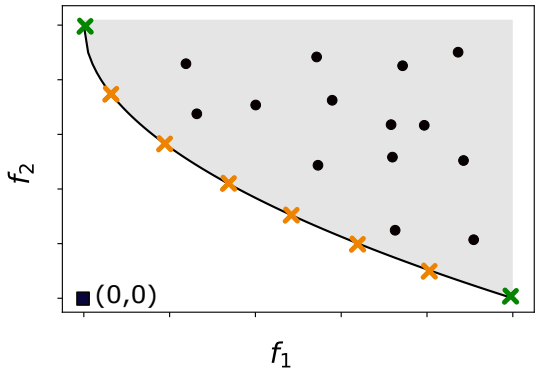


Fig. 4.5: Example of a Pareto set illustrated by crosses. The green crosses represent the extreme solutions in the Pareto.

Due to the large number of possible solutions (all possible subsets of Ω), we propose to use a meta-heuristic algorithm (e.g., NSGA-II [210]) to solve

this multi-objective optimization problem. It should be noted that this type of algorithm does not necessarily reach the optimum but usually provide suitable solutions [211]. Taking this into account, from this point on, we will refer to the sets in the Pareto as the optimal sets of time points but bear in mind that since we are using a heuristic, these solutions are an approximation of the Pareto.

4.4 Experiments

The experimentation is divided into three parts. The first part consists of analyzing the optimal sets of points $\mathcal{P}^*$ obtained by our method in synthetic datasets (see Section 4.4.1). In the second and third parts, we apply our selective imputation method and analyze its performance when we apply a downstream algorithm for classification (see Section 4.4.2) or anomaly detection (see Section 4.4.3).

In the three experiments, we assume that the missing behaviour in the dataset D is not random and that the time series share a common missing pattern. The reason for doing this is twofold. On the one hand, it will allow for a better interpretation and validation of the time point selection. On the other hand, in many time series datasets, missing data shares a common missing data pattern. For instance, in health data, patients admitted to the ICU that are progressing favorably and are not severely ill tend to receive less attention over time [212].

Parameter setting

The selected parameters for the MGP and the multi-objective optimization algorithm are common to both parts of the experimentation.

Concerning the MGP model, we use the `gpytorch` [213] library in Python and chose 100 iterations and a learning rate of 0.1. For the multi-objective optimization, we use the widely known NSGA-II algorithm [210], a multi-objective evolutionary algorithm that uses non-dominated sorting. This method has been selected based on its popularity due to its fast non-dominated sorting procedure and elitist approach. However, since it is only an element of the framework, it should be noted that the evolutionary algorithm could be modified by the user. In particular, we use the `pymoo` [214] library in Python to implement this algorithm. The specified parameters are the population size, which has been set to 20, and the number of generations, which has been set to 50. This selection has been made to limit the computational cost. Additionally, we have initialized the algorithm such that the initial population contains the individual $\mathcal{P} = \Omega$. The rest of the initial population is generated randomly.

4.4.1 Part I: Evaluation of the Pareto set in synthetic datasets

In this section, four different synthetic datasets are used to evaluate the performance of our approach in a controlled scenario. In all cases, the method is applied to a time series dataset of 100 bivariate time series.

4.4.1.1 Generation of the synthetic datasets

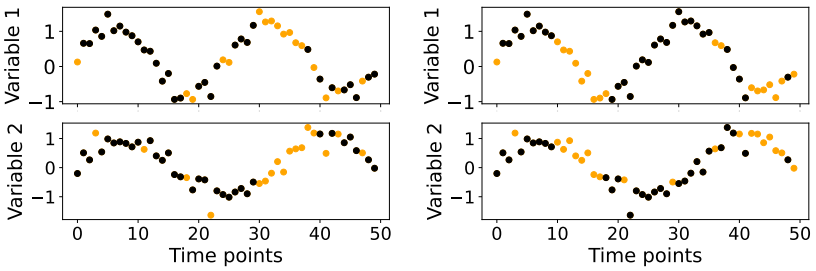
The four synthetic datasets can be divided into two groups. The first group consists of two datasets generated using sinusoidal functions such that

$$\begin{bmatrix} y_{1,t} \\ y_{2,t} \end{bmatrix} = \begin{bmatrix} \sin(4\pi t)/T \\ \sin(3\pi t)/T \end{bmatrix} + \begin{bmatrix} \xi_{1,t} \\ \xi_{2,t} \end{bmatrix}$$

where T is the length of the time series, $t \in \{0, ..., T\}$, and $[\xi_{1,t}, \xi_{2,t}]^T$ is the noise vector. For these experiments, we choose $T = 50$, and $\text{corr}(\xi_{1,t}, \xi_{2,t}) = 0.7$ to make $y_{1,t}$ and $y_{2,t}$ correlated. Moreover, for each $\xi_{i,t}$, given an interval $x^i = [x_1^i, x_2^i]$ with $x_1^i, x_2^i \sim N(0, 1)$ where the noise values will be, $\mathbb{E}(\xi_{i,t}) = \bar{x}^i$ and $\text{Var}(\xi_{i,t}) = (\sigma_{x^i}/3)^2$ where σ_{x^i} is the standard deviation of x^i.

Then, missing values are injected such that most of the missing values are within a certain time interval A: the probability that each observation $y_{j,t}$, where $t \in A$, is missing is 0.9 and 0.2 inside and outside A, respectively. In specific, the intervals chosen for conducting this experiment are $A_1 = [30, 40)$ and $A_2 = [10, 18) \cup [42, 48)$, each interval leading to a synthetic dataset in this group.

An example of a synthetic time series in this group is shown in Fig. 4.6, for both of the intervals being analyzed. Based on the underlying idea of our proposal, we expect the method to avoid selecting points in A (A_1 in the first dataset, and A_2 in the second dataset), since this interval will have many missing values and, thus, high uncertainty.



(a) $A_1 = [30, 40)$ (b) $A_2 = [10, 18) \cup [42, 48)$

Fig. 4.6: Example of a time series in the first group of the synthetic datasets. The missing observations are represented by orange dots.

The second group consists of two datasets generated based on a first-order Vector AutoRegressive (VAR) model such that

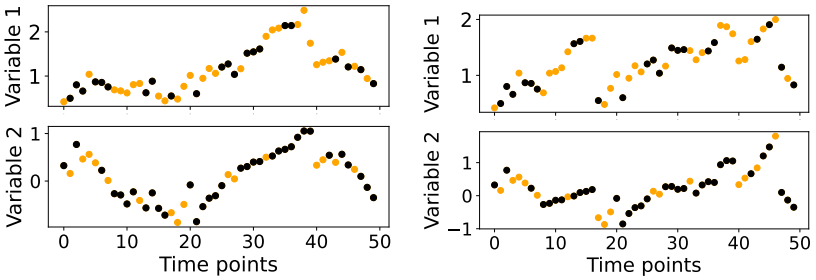
$$\begin{bmatrix} y_{1,t} \\ y_{2,t} \end{bmatrix} = \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} + \begin{bmatrix} \rho_1 & 0 \\ 0 & \rho_2 \end{bmatrix} \begin{bmatrix} y_{1,t-1} \\ y_{2,t-1} \end{bmatrix} + \begin{bmatrix} \xi_{1,t} \\ \xi_{2,t} \end{bmatrix}$$

where $t \in \{0, ..., T\}$. In particular, we choose $\alpha_0 = \alpha_1 = 0$, $\rho_1 = \rho_2 = 0.8$, and $T = 50$. Additionally, following [215], we choose the noise term such that $\text{corr}(\xi_{1,t}, \xi_{2,t}) = \rho(1 - \rho_1\rho_2)[(1 - \rho_1^2)(1 - \rho_2^2)]^{-1/2}$, where $\text{corr}(y_{1,t}, y_{2,t}) = \rho$ and $\rho = \rho_1 = \rho_2$. As in the previous group, given an interval $x^i = [x_1^i, x_2^i]$ with $x_1^i, x_2^i \sim N(0, 1)$ where the noise values will be, $\mathbb{E}(\xi_{i,t}) = \bar{x}^i$ and $\text{Var}(\xi_{i,t}) = (\sigma_{x^i}/3)^2$ where σ_{x^i} is the standard deviation of x^i. In this case, a particular time interval B is then replaced by a new, different process. This process consists of an increasing function such that for $t \in B$,

$$y_{i,t} = y_{i,t-1} + \epsilon_{i,t} \tag{4.6}$$

where $\epsilon_{i,t} \sim N(0, 0.2)$. Then, the missing values are injected uniformly throughout the time series with a probability of 0.4 of being missing.

As with the sinusoidal dataset, the intervals chosen for conducting the experiments are $B_1 = [30, 40)$ and $B_2 = [10, 18) \cup [42, 48)$ (see Fig. 4.7). Note that each of these intervals also leads to a synthetic dataset in this group. In this case, our hypothesis is that the method will tend to select the time points in B, since this interval cannot be inferred by the points outside the interval.



(a) $B_1 = [30, 40)$ (b) $B_2 = [10, 18) \cup [42, 48)$

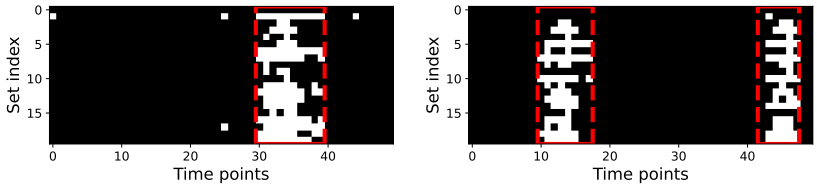
Fig. 4.7: Example of a time series in the second group of synthetic datasets. The missing observations are represented by orange dots.

4.4.1.2 Results

The evaluation of the optimal subsets of points obtained by our method is performed in two parts: the first part analyzes the Pareto set in a qualitative manner, and the second part evaluates this Pareto by comparing it with

randomly generated subsets of time points. In short, this section analyzes the results regarding the optimization part.

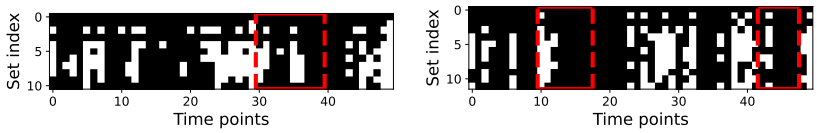
The selected subsets of time points for the two synthetic datasets in the first group are shown in Fig. 4.8. In particular, the black squares represent the time points that have been selected in each of the sets, and conversely, the white squares represent the time points that have not been selected. Also, the red lines highlight the intervals A_1 and A_2. As it can be seen in the figure, the most uncertain intervals (i.e., those with many missing values) are not selected: A_1 and A_2 contain most of the white squares.



(a) $A_1 = [30, 40)$ (b) $A_2 = [10, 18) \cup [42, 48)$

Fig. 4.8: Optimal sets in the first group of synthetic datasets.

For the second group of synthetic datasets, the optimal subsets of time points are shown in Fig. 4.9. Unlike the first dataset, the method tries to include the intervals B_1 and B_2 as they provide new information that the rest of the points do not contain. In this case, the Pareto set contains fewer optimal sets than in the first synthetic dataset.



(a) $B_1 = [30, 40)$ (b) $B_2 = [10, 18) \cup [42, 48)$

Fig. 4.9: Optimal sets in the second group of synthetic datasets.

As a second experiment, to demonstrate that the point sets in the Pareto are good in terms of uncertainty and predictive capability, each optimal set is compared to 20 randomly generated sets of the same size. For instance, if an optimal set contains 15 time points, this set is compared to 20 randomly generated sets, each consisting of 15 time points. This analysis will help checking if the solutions are good enough since the optimization method used is heuristic. That is, we will examine if the optimization part has been performed adequately.

Specifically, for each set in the Pareto, this comparison analyzes, on the one hand, how many random sets dominate the set being analyzed (the number of random sets located in region 1 in Fig. 4.10), and, on the other hand, the set in question how many random sets dominates (the number of random sets located in region 2 in Fig. 4.10). It is desirable to have few points in region 1 and most of them in region 2. In particular, we perform this comparison in the cases in which the optimal set is not of length T, because otherwise all the random sets would be the same and the analysis would be meaningless.

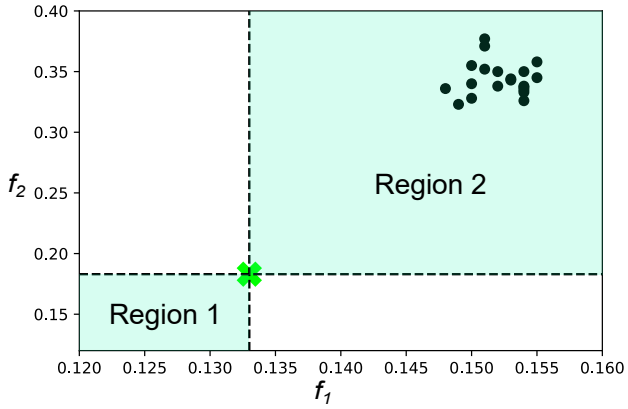


Fig. 4.10: An example of the comparison between a set in the Pareto depicted by a green cross, and 20 random sets of the same size illustrated by black dots.

For both groups of synthetic datasets, almost no random set dominates the corresponding optimal set (region 1 in Fig. 4.10). Particularly, in the first group, no optimal set is dominated by any random set, whereas, in the second group, there is only one random set that dominates the optimal set (on average, each set in the Pareto are dominated by 0.00% of random sets in scenario B_1, and 0.45% in scenario B_2). Conversely, when analyzing region 2, we find that the optimal sets dominate most of the random sets. In particular, the optimal sets dominate on average: in the first group, 95.26% and 96.58% of random sets in A_1 and A_2, respectively; in the second group, 55.50% and 71.36% of random sets in B_1 and B_2, respectively.

4.4.2 Part II: Application in classification tasks

In this section, the usefulness of the proposed method in the multivariate time series classification downstream task is shown. We would like to emphasize that this section does not aim to demonstrate that our method is the best

solution for the classification task but to illustrate how an appropriate selection of time points allows to not only reduce the uncertainty and imputation error, but also to improve the results of the classification task.

As a preliminary proof of this hypothesis, we show in Fig. 4.11 the evolution of the mean accuracy of five popular classifiers when we perform a backward analysis in the *Libras* dataset [8] by removing the globally most uncertain time point of the time series dataset at each iteration. To obtain these accuracy values, we first pre-process each time series and impute all its missing values using MGP. The purple line in Fig. 4.11 indicates the accuracy obtained in the downstream supervised classification task when we impute all the missing data points, which corresponds to the 0^{th} iteration. We show in the figure how by removing the most uncertain time points from the time series (up to almost half of the missing data points), we can obtain an improvement in the results of the classifier.

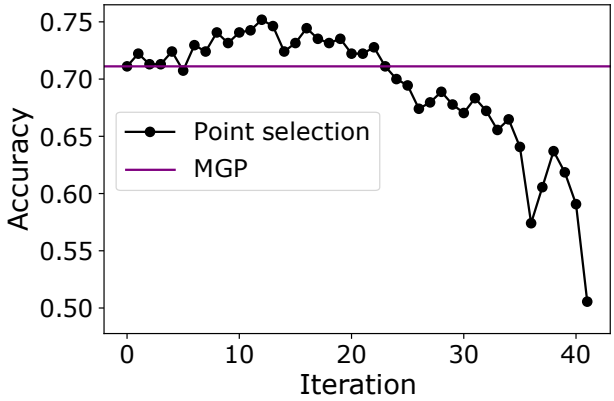


Fig. 4.11: Backward analysis in the *Libras* dataset with 85% of injected missing data.

Now that we have seen that the selection of time points to impute can be beneficial for downstream tasks such as multivariate time series classification, we will try to find the optimal set of time points in different datasets and analyze the results of this task when using the simplified dataset.

4.4.2.1 Datasets

The experiments are performed in different datasets and classification tasks from the UEA repository [8]. Additionally, the dataset from the Physionet Challenge [11] that aims to predict in-hospital mortality is also used. For the Physionet dataset, we follow previous works and use a subset of variables

[216]. In addition, for efficiency and simplification, we use a subset of samples, maintaining the mortality rate (14.29%).

The characteristics of the chosen datasets are summarized in Table 4.2, which shows the wide variety of the sets. In particular, for each case, 70% of the multivariate time series are used to identify the best sets of time points and learn the classifier, and 30% for evaluation. Additionally, it should be noted that all the datasets described in the table have originally a regular sampling. We denote these equally-spaced time points as $X = \{1, 2, ..., T\}$, where T is the length of the time series.

Table 4.2: Description of the datasets used in the multivariate time series classification task.

Dataset	Length	Dimensions	# of instances	Classes
Japanese Vowels	29	12	640	9
Racket Sports	30	6	303	4
Libras	45	2	360	15
Physionet	48	6	700	6
Finger Movements	50	28	416	2
Basic Motions	100	6	80	4
Epilepsy	206	3	275	4

While the Physionet dataset already contains missing values (it has an hourly sampling with missing values), the datasets from the UEA repository do not contain missing values. Thus, we inject the missing values in those datasets in such a way that the time series will contain more missing values at the end of the time series. In particular, if we denote mr_1 as the missing rate of the first half of the time series (i.e., $[1, ..., T/2]$), then $mr_1 \sim U(0.7, 0.8)$. In the same way, if we denote mr_2 as the missing rate of the second half of the time series (i.e., $[T/2 + 1, ..., T]$), then $mr_2 \sim U(0.9, 1)$.

The *Japanese Vowels* dataset from the UEA repository has time series of different lengths. In this case, a padding with missing values is made until the maximum length, which is 29, is reached. Then, the remaining missing values are injected to satisfy the missing rates described above.

The datasets used in this experimentation will be available in the GitHub repository[1] for further reproducibility.

4.4.2.2 Classifiers

Five traditional classifiers are used in the experimentation: Time Series Forest (TSF) [217], Mr-SEQL [218], 1-Nearest Neighbor using independent and dependant Dynamic Time Warping (DTW) distances [219], and RISE [32].

[1] https://github.com/ablazquezg/Selective-imputation

The score used throughout the experimentation is the mean accuracy of the five classifiers. For the classifiers that are designed to deal only with univariate time series (TSF, Mr-SEQL, and RISE), dimension concatenation is used [220]. The library used is `sktime` [220] in Python, and the hyperparameters of the classifiers are set to the default values.

4.4.2.3 Baseline methods

Since techniques in the literature usually impute all the missing values, the baseline methods will be naive methods that will impute all the values of all the (equally-spaced) time points (i.e., X). In particular, the baseline methods will impute the missing values with the widely used Forward Filling (FF, baseline 1) and also with the Multi-task Gaussian Process (MGP, baseline 2). These techniques have been chosen for their adaptability in the context of time series data. Once all the missing values have been imputed, all the time points will be used to learn the classifier.

4.4.2.4 Results

In this section, we analyze both the quality of the imputations by computing the imputation error and also the accuracy of the classifiers using the sets of time points selected by our method.

To begin with, the imputation error is calculated in the test set using RMSE and normalized data between 0 and 1. In particular, the imputation error has only been computed in those datasets that originally have no missing values. As shown in Table 4.3, the imputation error is always smaller using the probabilistic MGP method than the FF method. Moreover, there are always sets in the Pareto that manage to reduce this error by using less time points. In particular, this reduction becomes very significant for some datasets, such as the *Libras* dataset.

The results regarding the classification accuracy are shown in Table 4.4. On the one hand, we analyze the results obtained using the baseline methods (i.e., when all the missing points are imputed), and we find that, in general, the MGP imputation provides better accuracy than the FF imputation. On the other hand, if we compare the accuracy results of the sets in the Pareto with those obtained with the baseline methods, we conclude that the proposed methodology is always able to find sets of time points that improve the accuracy (see columns $\geq FF(\%)$ and $\geq MGP(\%)$ in Table 4.4).

Table 4.3: Results of the imputation errors. The two first columns report the average imputation error with the standard deviation between parentheses of the baselines over 5 different train/test partitions. The next three columns show some statistics of the imputation errors of the sets in the Pareto. The last two columns describe the percentage of the sets that achieve a lower imputation error than the baselines.

Dataset	**Baseline methods**		**Point selection**				
	FF	**MGP**	**min**	**mean**	**max**	**≤ FF (%)**	**≤MGP (%)**
Racket Sports	0.2869 (0.0065)	0.2362 (0.0014)	**0.2362** (0.0014)	0.2608 (0.0016)	0.2836 (0.0023)	96.00 (04.18)	5.00 (00.00)
Libras	0.2974 (0.0038)	0.2094 (0.0050)	**0.1054** (0.0059)	**0.1654** (0.0038)	**0.2094** (0.0050)	100.00 (00.00)	99.00 (02.24)
Finger Movements	0.2845 (0.0013)	0.2664 (0.0019)	**0.2318** (0.0042)	**0.2576** (0.0036)	0.2688 (0.0033)	100.00 (00.00)	84.11 (10.00)
Basic Motions	0.2772 (0.0075)	0.1967 (0.0025)	**0.1963** (0.0029)	0.1978 (0.0028)	0.2003 (0.0031)	100.00 (00.00)	26.00 (16.36)
Epilepsy	0.3067 (0.0017)	0.2246 (0.0011)	**0.2204** (0.0018)	**0.2230** (0.0013)	**0.2246** (0.0011)	100.00 (00.00)	99.00 (02.24)

Table 4.4: Accuracies in the classification task. The columns in the table follow the same rationale as Table 4.3.

Dataset	**Baseline methods**		**Point selection**				
	FF	**MGP**	**min**	**mean**	**max**	**≥ FF (%)**	**≥MGP (%)**
Japanese Vowels	**0.7613** (0.0128)	0.7502 (0.0133)	0.7098 (0.0154)	0.7343 (0.0121)	0.7542 (0.0120)	14.89 (21.02)	16.78 (18.91)
Racket Sports	**0.4954** (0.0198)	0.4440 (0.0221)	0.4193 (0.0307)	0.4514 (0.0232)	0.4792 (0.0235)	3.00 (04.47)	69.00 (18.51)
Libras	0.5111 (0.0276)	0.6963 (0.0216)	0.6822 (0.0249)	**0.7022** (0.0143)	**0.7315** (0.0132)	100.00 (00.00)	65.00 (25.74)
Physionet	0.8215 (0.0088)	0.8276 (0.0042)	0.8168 (0.0088)	0.8240 (0.0060)	**0.8309** (0.0051)	42.22 (46.80)	62.22 (51.88)
Finger Movements	0.5251 (0.0154)	0.5245 (0.0146)	0.4925 (0.0136)	0.5196 (0.0139)	**0.5450** (0.0096)	50.11 (34.00)	50.33 (29.68)
Basic Motions	0.7483 (0.0272)	0.7933 (0.0266)	0.7700 (0.0240)	**0.7955** (0.0243)	**0.8233** (0.0239)	83.00 (38.01)	72.00 (08.37)
Epilepsy	0.8106 (0.0037)	0.8607 (0.0098)	0.8429 (0.0055)	0.8583 (0.0082)	**0.8713** (0.0145)	100.00 (00.00)	50.00 (28.94)

Furthermore, the sets of points that fail to improve it still manage to obtain results similar to the baselines. Indeed, with a significantly less number of time points. For more details on the reduction of the time series, see Table 4.5. In general, in this table we can see that the sets in the Pareto reduce the time series by an average of 27.12% of the length per dataset.

It should also be noted that in those cases in which the accuracy results of our method are not as good as in the baseline methods, the imputation error is reduced. For example, the accuracy results obtained with the FF baseline method in the *Racket Sports* dataset are better than using our method. However, 96% of the sets in the Pareto obtain a lower imputation error than the FF baseline. In this particular case, the FF imputation may favour the specific supervised classification task.

Table 4.5: Length reduction using the sets in the Pareto. The columns describe 1) the dataset used, 2) the lengths of the sets that provide the maximum accuracy, 3) the average length of the sets in the Pareto, and 4) the percentage reduction of this average. The values shown are the mean values over the 5 partitions and the standard deviation between parenthesis.

Dataset	**Length max accuracy**	**Mean length**	**Mean reduction (%)**
Japanese Vowels	19.60 (4.98)	16.11 (1.30)	44.46
Racket Sports	17.20 (5.26)	21.62 (1.17)	27.93
Libras	26.20 (4.27)	32.70 (1.50)	27.33
Physionet	33.40 (5.94)	37.06 (2.26)	22.79
Finger Movements	36.80 (13.25)	32.72 (0.90)	34.56
Basic Motions	71.80 (16.39)	81.79 (3.38)	18.21
Epilepsy	178.60 (21.76)	176.02 (4.85)	14.55

4.4.3 Part III: Application in the anomaly detection task

In this section, the usefulness of the proposed methodology in the anomaly detection task is presented. As with the classification task, the aim of this section is not to demonstrate that our method is the best solution for anomaly detection but to illustrate that a proper selection of time points allows to improve the results of the anomaly detection task.

4.4.3.1 Dataset

The experiments are performed in a multivariate time series dataset from the MIT-BIH Arrythmia Database[1] [221]. In particular, we evaluate our proposed method on some pre-processed bivariate ECG time series from this dataset[2]

[1] https://physionet.org/content/mitdb/1.0.0/
[2] https://github.com/hi-bingo/BeatGAN

[222]. The time series in this repository have length 320. However, to decrease the computational burden, we reduce the time series to have length 80 by computing the mean value of non-overlapping windows of length 4. Additionally, since the datasets do not contain missing values, we follow the same procedure employed in the classification task to inject them. As such, $mr_1 \sim U(0.7, 0.8)$ and $mr_2 \sim U(0.9, 1)$, where mr_1 and mr_2 are the missing rates of the first and second half of the time series, respectively.

For training, we randomly select 495 bivariate time series from the set that contains normal samples (N_SAMPLES file), and 5 bivariate time series from an anomalous set (Q_SAMPLES file). In this way, the training set contains a 1% of anomalies. This set is used to both select the best subset of time points and train the anomaly detector (see Section 4.4.3.2).

Then, to evaluate our method in the anomaly detection task, a test dataset composed of 2000 normal and 64 abnormal time series will be used. The normal time series are randomly chosen from the normal dataset (N_SAMPLES). It should be noted that this set does not contain those samples included in the training phase, even though they have been extracted from the same file. Conversely, the anomalous time series are randomly chosen from a separate abnormal dataset (ABNORMAL_SAMPLES) located in the demo folder of the repository.

Both the training and the test datasets used in this experimentation will be available in the GitHub repository[1] for further reproducibility.

4.4.3.2 Anomaly detector

Due to the limited number of available techniques in the literature for detecting whole multivariate time series outliers, the anomaly detector used in this section is based on the intuition behind the discord discovery approach [134]. As such, given a reference time series dataset that will represent the normal behavior of the time series (the training dataset, in this case), a new multivariate time series will be identified as an anomaly if the distance to its nearest neighbor in the reference dataset is large enough:

$$\begin{cases} d(Y^{new}, Y^{new}_{NN}) > \tau \implies Y^{new} \text{ anomaly} \\ d(Y^{new}, Y^{new}_{NN}) \leq \tau \implies Y^{new} \text{ normal} \end{cases} \tag{4.7}$$

where Y^{new} is the new multivariate time series being tested, Y^{new}_{NN} is the nearest neighbor of Y^{new} in the reference/training dataset, d is the distance used to measure the similarity between the two time series, and τ is a predefined threshold value. In particular, to deal with multivariate time series, we use the dependant DTW distance.

[1] https://github.com/ablazquezg/Selective-imputation

4.4.3.3 Baseline methods

As with the classification task, the baseline methods impute all the missing values with both the FF and MGP methods first. Then, these baseline methods consist of applying the same above-mentioned anomaly detector, but they will consider all the (equally-spaced) time points rather than a subset of time points, as proposed by our methodology.

4.4.3.4 Results

Similar to the classification task, the results in the test dataset are analyzed with respect to three aspects: the imputation error, the anomaly detection, and length of the subsets of time points obtained.

To begin with, the results regarding the imputation error are shown in Table 4.6. As can be seen, the imputation error is always better when using MGP than FF. However, when using the MGP method to impute the missing values, the imputation error obtained by our method increases slightly compared to the MGP baseline (i.e., using all the time points). This can occur when the imputation of the missing values is simple (the actual observations are around the mean of the MGP model), but very uncertain. However, the difference is not large, and also, the imputation error obtained is still lower than using the traditional FF.

Table 4.6: Results of the imputation errors in the test dataset. The two first columns report the imputation error using the baselines. The next three columns show some statistics of the imputation errors of the sets in the Pareto. The last two columns describe the percentage of the sets that achieve a lower imputation error than the baselines.

Dataset	**Baseline methods**		**Point selection**				
	FF	**MGP**	**min**	**mean**	**max**	**≤ FF (%)**	**≤MGP (%)**
Test	0.1931	0.1273	0.1273	0.1409	0.1626	100.00	5.00

In addition, both the ability to identify the anomalies and the number of false alarms are measured. For this purpose, the True Positive Rate (TPR) and the False Positive Rate (FPR) are used: the TPR is obtained calculating the ratio of the correctly identified anomalies, whereas the FPR computes the ratio of anomalies that are incorrectly detected in the test set. Since the anomalies are identified based on the threshold τ in Eq. 4.7, different threshold values are used to obtain the Area Under the Curve (AUC). In particular, the threshold values used are $\tau \in \{0, 0.2, 0.4, ..., 4\}$.

The results of the anomaly detector in terms of AUC are presented in Fig. 4.12. As shown in this figure, all except one subset of time points in the Pareto

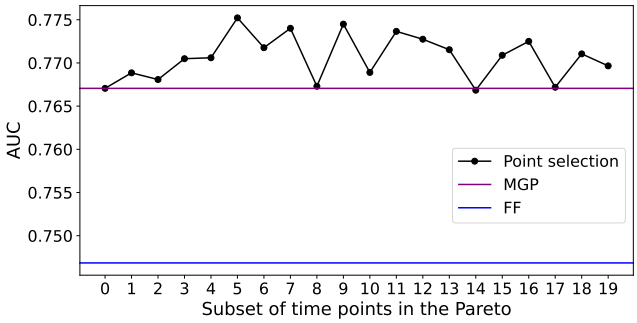


Fig. 4.12: AUC results using the subsets of time points obtained with our method. The horizontal lines represent the AUC values provided by the baseline methods.

achieve better results than the baselines. The difference between the subset that obtains a slightly lower AUC and the the MGP baseline is negligible.

An example of the relationship between the TPR and FPR values for different thresholds is shown in Fig. 4.13. This figure depicts the ROC curves of 1) the subset of time points in the Pareto that achieves the best AUC, and 2) the baseline methods. As can be seen, the imputation with MGP provides better AUC than FF, and in addition, this specific selection of time points slightly improves the AUC obtained with the MGP baseline.

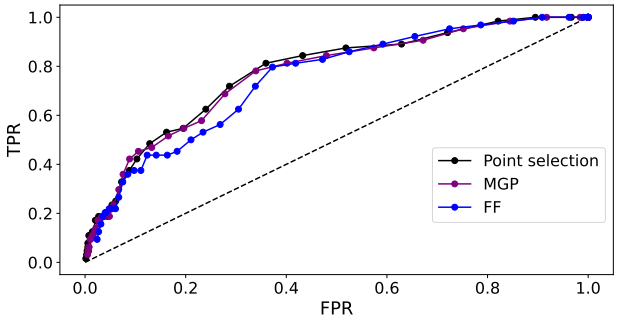


Fig. 4.13: Comparison of the ROC curves between the baseline methods and the subset of time points that obtains the highest AUC value.

Finally, it should be noted that the length of the subsets of time points is reduced 20.62% on average. In particular, the mean length of the subsets is

63.5, and the length of subset that provides the maximum AUC value is 59 (26.25% less than the original length, which is 80).

4.5 Conclusions

In conclusion, this chapter introduces a time point selection method to selectively impute the missing values in a multivariate time series dataset. This selection is based on the uncertainty of the imputed values and the predictive capability of the selected observations. In this way, the overall uncertainty of the dataset is reduced, and it allows to use simplified time series in downstream tasks such as multivariate time series classification and anomaly detection.

It must be noted that, with the aim of obtaining the optimal selection of time points, our method naturally tends to return time series with unequally-spaced time points. However, additional restrictions could be added to force the method to output time series with equally-spaced time points, if desired.

The imputation method used to fill in the missing values has been MGP, but other probabilistic models that provide the uncertainty of the imputed values could also be used. Since we use a probabilistic imputation method, an interesting future line of research could be to provide more sophisticated measures of uncertainty (e.g., using information theory). Moreover, in some contexts (e.g., when learning normality), it may be interesting to learn a single global imputation model on the whole dataset.

Reducing the set of time points does not only simplify the time series but can also help to improve the results of downstream tasks such as whole time series classification and anomaly detection. In fact, there are always sets in the Pareto that improve the results (accuracy in the classification task and the AUC in the anomaly detection task). Moreover, those that do not improve it remain with a similar performance, but using a shorter representation of the time series. However, the use of more sophisticated classifiers or anomaly detectors could help to improve the results. In this line, future research could focus solely on improving the results of the particular task.

As mentioned throughout the chapter, a significant advantage of the proposed method is that, in addition to the multivariate time series classification and anomaly detection tasks, this method can also be used in combination with other downstream machine learning tasks such as forecasting, or clustering. Thus, an interesting line for future work would be to test the applicability of the method in additional downstream tasks since the literature has mainly focused on the classification task.

Part III

General Conclusions and Future Work

5

General Conclusions and Future Work

To conclude the thesis, this last chapter introduces the main conclusions of the dissertation in Section 5.1, as well as some further research directions motivated by those contributions in Section 5.2. Finally, the main achievements of the thesis are summarized at the end of the chapter, in Section 5.3.

5.1 Conclusions

This thesis has presented several contributions to the field of time series data mining, mainly focused on the detection of outliers or anomalies. In particular, these contributions have been organized into two main parts. The first part has analyzed the detection of outliers or anomalies in time series, under the most typical and standard scenario: time series that are equally-spaced and fully-observed (with no missing values). Conversely, the second part has attempted to address this limitation and thus deal with the situations in which the mentioned conditions are not fulfilled. To this end, the main focus of the second part has been the pre-processing of time series with missing values.

Within the first block, the first contribution has been devoted to the analysis of the existing unsupervised outlier/anomaly detection techniques in time series. In particular, an organized overview of the state-of-the-art techniques has been proposed in Chapter 2. For this, a taxonomy based on the following three main axes has been provided: the input data type, the outlier type, and the nature of the detection method. This review of techniques has served not only to get a global idea of the concept of outlier but also to identify the existing gaps in the field of unsupervised outlier/anomaly detection in time series.

This review has shown that the techniques in the literature mostly focus on detecting anomalies in univariate time series, although there has been a special emphasis on multivariate time series in recent years. Moreover, point outlier detection has been the most researched problem, whereas subsequence outliers and outlier time series have been handled less frequently. Many of the

techniques for these latter types of outliers are dissimilarity-based techniques, although often the distance used is not specific for time series. In addition, the techniques in the literature are targeted at time series that have no missing values and are sampled on a regular basis. Lastly, this first contribution has also shown that many techniques do not consider the temporal correlation when detecting the outliers. It is important to keep in mind that this characteristic is usually relevant in the study of time series and should not, in general, be ignored.

With the aim of filling in some of the gaps found in the literature of anomaly detection in time series, in Chapter 3, we have proposed a novel whole time series anomaly detection technique, applied to the water leak detection problem. In particular, the major challenges addressed have been the lack of (high-quality) ground-truth labels, and the temporal correlation. To address the lack of ground-truth labels, we have used self-supervised learning, a novel learning method that has not been applied to the context of anomaly detection in time series. To account for the temporal correlation between the observations, we have used a specific classifier for time series. This new contribution enables the detection of water leaks, providing fewer false positives than other traditional techniques.

Another challenge encountered in the literature has been that most of the anomaly detection methods assume that the time series are fully-observed, without missing values. Indeed, our first two contributions (i.e., the first part of the thesis) are based on this assumption. However, for different reasons such as failures in the data collection mechanism, time series often contain missing values. In the second part, we have considered this scenario, specifically with multivariate time series.

In this sense, in Chapter 4, we have proposed a selective imputation method that identifies a subset of time points with missing values to impute in a multivariate time series dataset. This selection, which will result in shorter and simpler time series, is based on both reducing the uncertainty of the imputations and representing the original time series as accurately as possible. In particular, the method uses multi-objective optimization techniques to select the optimal set of points, and, in this selection process, the beneficial properties of the Multi-task Gaussian Process (MGP) are leveraged. Furthermore, the usefulness of the method in downstream tasks, such as whole multivariate time series classification and anomaly detection, has been demonstrated.

In short, this thesis has addressed multiple challenges encountered in the field of time series data mining, with a focus on outlier/anomaly detection and missing data.

5.2 Future Works

The results and conclusions presented in this thesis have encouraged several promising research directions that deserve to be explored in the future. In

particular, we have identified two types of problems aligned with the two parts of this dissertation.

Regarding the lines of future research emerging from the first part, we highlight the following:

- **Analysis of unexplored types of outliers.** As mentioned in Chapter 2, the types of outliers can be classified into 1) point outliers, 2) subsequence outliers, and 3) whole time series outliers. However, there could also exist more complex outliers such as outliers that propagate over time and over the different variables of a multivariate time series. For example, in industrial scenarios, a failure of a machine may affect other related machines but at different time steps (i.e., failure propagation). Although the outliers in each variable would still be points and/or subsequences, when analyzing the whole multivariate time series, the outlier would no longer correspond to the definition of multivariate outlier point or subsequence analyzed in this thesis due to the time lag. This problem has been hardly analyzed in the literature [223] and there is still room for research, such as considering the cases in which multiple outliers/anomalies occur and propagate over time simultaneously.
- **Extension to streaming time series.** The set of time points $\mathcal{T}$ of a time series can also be infinite, as introduced in Chapter 2. In fact, in scenarios such as industrial process monitoring, it is very common to collect this type of (streaming) time series. In this context, a real-time identification of outliers/anomalies (as soon as they occur) is of great interest because a delay in their detection can lead to negative consequences (e.g., in economy or safety).

  In recent years, an effort has been made to develop techniques that are suitable for streaming time series [123, 224]. However, these techniques commonly focus on detecting point outliers in univariate time series, and, to the best of our knowledge, no work has been done on the detection of subsequence outliers in high-dimensional streaming multivariate time series data. As such, this might be a promising venue for research, especially focusing on the design of efficient online algorithms that are able to update as new observations arrive (without the need to repeatedly retrain models from scratch).

  Moreover, as mentioned in Chapter 4, most techniques assume that the time series are complete and regularly-sampled. In this sense, another interesting line of future work could be to develop online techniques for the detection of outliers/anomalies in time series with missing values or those with variables that have different sampling rates.
- **Generalization of the self-supervised dataset.** The solution presented in Chapter 3 is problem-specific and only deals with univariate time series. In this sense, it could be interesting to extend the generation of the self-supervised dataset to support multivariate time series, such that more variables related to water leaks can be considered.

Additionally, since the proposed method is problem-specific, it is not directly suitable for other data domains. As such, an interesting future line of research would also be to investigate the different transformations that can be applied to generate a self-supervised dataset in other fields. This will depend on the type of both the available data in the training set and the anomalies to be detected.
In this way, the proposed approach would be suitable for different settings (different types of problems, data and anomalies).

Regarding the limitations related to the second part of this dissertation, we propose the following research directions:

- **Treatment of the missing values.** The treatment of missing values can be done in two ways, as discussed in Chapter 4: by pre-processing the time series, or by performing the imputation and a specific data mining task simultaneously. Given that our approach is unsupervised and not exclusive to a specific data mining task, we have opted for the first option, where we use the MGP method to impute the missing values of each multivariate time series before applying the data mining task. This leads to two main possible lines of future work.
  On the one hand, it would be interesting to explore other more recent probabilistic imputation techniques that provide the uncertainty of the imputation for each time series independently (e.g., GP-VAE [225]) and compare the results with those obtained with MGP.
  On the other hand, it might also be promising to analyze a (single) global imputation model for the entire dataset of multivariate time series rather than imputing each of the time series independently. This may be of particular interest in contexts where the time series in the dataset are generated by the same underlying process (e.g., when learning normality).
- **Application in other downstream tasks.** In Chapter 4, the applicability of the proposed selective imputation method in two time series data mining tasks has been shown: multivariate time series classification and whole outlier/anomaly time series detection. In this sense, an interesting line for future work would be to analyze the usefulness of the method in additional downstream tasks such as clustering or forecasting.
- **Improvements in the time point selection algorithm.** Exploring and developing more sophisticated measures for time point selection could lead to an improvement in the results. As mentioned in Chapter 4, the uncertainty of a subset of time points in a time series is computed by the mean value of the variances (which are obtained by MGP) of the missing points. However, in this way, the uncertainty is not computed in a jointly manner for the points in the subset. Thus, future research could focus on elaborating more sophisticated measures that evaluate the overall uncertainty jointly (e.g., based on information theory).

Finally, a noteworthy line of research related to both parts, and, in general, in the field of the detection of outliers/anomalies in time series data would be the following:

- **Creation of benchmark datasets**. The evaluation of the outlier/anomaly detection methods (in general, any machine learning method) is a relevant step to conclude whether the proposed methods can be useful in practice or not. However, this evaluation is often complicated in scenarios where obtaining (high-quality) labels is challenging (e.g., in unsupervised scenarios).
  To overcome this challenge, researchers and practitioners have made an effort to create benchmark datasets aimed at the evaluation of time series anomaly detection methods. However, most of the commonly used datasets have flaws that make them unsuitable for evaluating the methods [42]. Recently, the UCR Time Series Anomaly Archive has been proposed[1], which tries to overcome the current benchmark's flaws [42]. Nevertheless, this archive is focused on univariate time series and subsequence outliers/anomalies. Thus, an interesting and useful line of research would be to generate a more general new dataset that allows for multivariate time series and other types of outliers.

5.3 Main Achievements

The research work conducted during this thesis has resulted in the following publications and stays:

5.3.1 Journal Papers

- **Blázquez-García, A.**, Conde, A., Mori, U., Lozano, J. A. (2021). A review on Outlier/Anomaly Detection in Time Series Data. *ACM Computing Surveys (CSUR), 54(3), 1-33.*
- **Blázquez-García, A.**, Conde, A., Mori, U., Lozano, J. A. (2021). Water leak detection using self-supervised time series classification. *Information Sciences, 574, 528-541.*
- **Blázquez-García, A.**, Wickstrom, K., Yu, S., Mikalsen, K.O., Boubekki, A., Conde, A., Mori, U., Jenssen, R., Lozano, J. A. (2022). Selective imputation for multivariate time series datasets with missing values. *IEEE Transactions on Knowledge and Data Engineering (TKDE).* Submitted.

[1] https://www.cs.ucr.edu/~eamonn/time_series_data_2018/UCR_TimeSeriesAnomalyDatasets2021.zip

5.3.2 Poster sessions

- **Blázquez-García, A.**, Conde, A., Mori, U., Lozano, J.A. (2019). A review on outlier detection in time series data. *2nd UPV/EHU Doctoral Conference*, Bizkaia Aretoa, Bilbao.
- **Blázquez-García, A.**, Conde, A., Mori, U., Lozano, J.A. (2019). A STUDY on outlier/anomaly detection in time series data. *3rd IK4-Ikerlan PhD. Student Meeting*, Orona Ideo, Hernani.
- **Blázquez-García, A.**, Conde, A., Mori, U., Lozano, J.A. (2020). Hierarchically structured time series outlier detection. *4th Ikerlan PhD. Student Meeting*, Online.
- **Blázquez-García, A.**, Conde, A., Mori, U., Lozano, J.A. (2021). On outlier/anomaly detection in time series data. *5th Ikerlan PhD. Student Meeting*, Online.

5.3.3 Short Visits

- 06 September-09 December 2021: UiT Machine Learning Group, Tromsø (Norway). Supervisor: Prof. Robert Jenssen.

Part IV

Appendixes

6

Multi-task Gaussian Process

Multi-task learning is a machine learning framework that aims to improve performance through the learning of multiple tasks at the same time, and sharing the information of each task [226]. Thus, Multi-task Gaussian Process (MGP) is an extension to Gaussian Processes (GPs) for handling multiple outputs at each time [206].

The objective of MGP is to model a set of processes $\{f_l(\boldsymbol{x})\}_{l=1}^{L}$, each one associated with a task, rather than a single process $f(\boldsymbol{x})$. When dealing with multivariate time series, the tasks refer to the dimensions of the time series (i.e., having L tasks means that the time series is L-dimensional). For convenience, we ignore the i^{th} superscript of the time series Y^i and use Y to refer to a time series in dataset $D = \{Y^1, ..., Y^N\}$. Additionally, we use $\tilde{T}$ to define the length of time series Y.

Given a set $X = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_{\tilde{T}}\}$ of $\tilde{T}$ indexes, the set of responses for L tasks is defined as the flattened vector $\boldsymbol{y} = [y_{11}, ..., y_{\tilde{T}1}, y_{12}, ..., y_{\tilde{T}2}, ..., y_{1L}, ..., y_{\tilde{T}L}]^T$, where y_{il} is the response for the l^{th} task on the i^{th} input $\boldsymbol{x}_i$. The observations are assumed to be noisy, and thus, each y_{il} is defined as

$$y_{il} = f_l(\boldsymbol{x}_i) + \epsilon_{il} \tag{6.1}$$

where $\epsilon_{il} \sim \mathcal{N}(0, \sigma_l^2)$. This can also be denoted as a $L \times \tilde{T}$ matrix:

$$Y = \begin{bmatrix} y_{11} & \cdots & y_{\tilde{T}1} \\ \vdots & \ddots & \vdots \\ y_{1L} & \cdots & y_{\tilde{T}L} \end{bmatrix} \tag{6.2}$$

Each l^{th} row represents the l^{th} dimension of time series Y, and the i^{th} column specifies the L-dimensional vector at time index $\boldsymbol{x}_i$.

When the time series being analyzed has missing values, only a subset of the values in Y are observed. Therefore, given a set of observations $\boldsymbol{y}_o \subseteq \boldsymbol{y}$, we can use a Multi-task Gaussian Process (MGP) to predict some of the unobserved values at some input locations for certain tasks (or variables). For

this, L different processes (latent functions) are modeled, $\{f_l\}_{l=1}^{L}$, assuming that each l dimension of the series is drawn from one of these f_l processes.

The most straightforward way to model the L processes is to assume that they are independent and thus use a GP for each. That is, each process is defined by a mean function, $\mu_l(\mathbf{x})$, and a covariance function, $k_l(\mathbf{x}, \mathbf{x'})$. For convenience, we assume the mean function to be is zero. Then, $f_l(\boldsymbol{x}) \sim GP(0, k_l(\boldsymbol{x}, \boldsymbol{x}'))$, and

$$\boldsymbol{y}_l = \begin{bmatrix} y_{1l} \\ \vdots \\ y_{\tilde{T}_l l} \end{bmatrix} \sim \mathcal{N}(\mathbf{0}, K_l + \sigma_l^2 I), \quad \text{where } l \in \{1, ..., L\}$$

where K_l is the covariance matrix associated with process f_l.

Additionally,

$$\begin{aligned} \begin{bmatrix} \boldsymbol{y}_1 \\ \vdots \\ \boldsymbol{y}_L \end{bmatrix} &\sim \mathcal{N}\left(\begin{bmatrix} \mathbf{0} \\ \vdots \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} K_1 & \cdots & \mathbf{0} \\ \vdots & \ddots & \vdots \\ \mathbf{0} & \cdots & K_L \end{bmatrix} + \begin{bmatrix} \sigma_1^2 I & \cdots & \mathbf{0} \\ \vdots & \ddots & \vdots \\ \mathbf{0} & \cdots & \sigma_L^2 I \end{bmatrix} \right) \\ &= \mathcal{N}(\mathbf{0}, K_{f,f} + \Sigma_L) \end{aligned} \tag{6.3}$$

where $K_{f,f}$ is the matrix containing the covariance matrices K_l in the diagonal, and Σ_L is the $L \times L$ diagonal matrix in which the $(l, l)^{th}$ element is σ_l^2.

This approach assumes that the processes are independent, and thus, the blocks outside the main diagonal of $K_{f,f}$ are zero. Contrarily, multi-task learning aims to exploit the dependencies between processes and define those terms outside the diagonal. In particular, the multi-task learning approach defines a covariance function that gives a positive semi-definite (PSD) covariance matrix $K_{f,f}$, also considering the dependencies between the processes.

Different models for defining the covariance function can be found in the literature. A widely used model is the Intrinsic Coregionalization Model (ICM) [227], which assumes that the $f_l(\boldsymbol{x})$ processes are defined by a linear combination of functions that have been sampled independently for the same GP, sharing the same covariance function $k(\boldsymbol{x}, \boldsymbol{x}')$. That is,

$$f_l(\boldsymbol{x}) = \sum_{i=1}^{R} a_d^i u^i(\boldsymbol{x}) \tag{6.4}$$

where $\{f_l(\boldsymbol{x})\}_{l=1}^{L}$ is the set of functions to be modeled, $a_d^i \in \mathbb{R}$ are the coefficients of the linear combination, and each $u^i(\boldsymbol{x})$ is sampled from $u(\boldsymbol{x}) \sim GP(0, k(\boldsymbol{x}, \boldsymbol{x}'))$. Then, the covariance function is defined as

$$\text{cov}(\boldsymbol{f}(\boldsymbol{x}), \boldsymbol{f}(\boldsymbol{x}')) = \boldsymbol{A}\boldsymbol{A}^T k(\boldsymbol{x}, \boldsymbol{x}') = \boldsymbol{B} k(\boldsymbol{x}, \boldsymbol{x}')$$

where $\boldsymbol{f}(\boldsymbol{x}) = [f_1(\boldsymbol{x}), ..., f_L(\boldsymbol{x})]^T$, $\boldsymbol{A} = [\boldsymbol{a}^1 \quad \boldsymbol{a}^2 \quad ... \quad \boldsymbol{a}^R]$, $\boldsymbol{B} \in \mathbb{R}^{L \times L}$, and k is a covariance function over inputs. The main idea is to place independent GP priors over the processes, with a shared correlation function k over time.

Following this ICM model, [206] define the covariance function of the MGP as:

$$\text{cov}(f_{l_1}(\boldsymbol{x}), f_{l_2}(\boldsymbol{x}')) = K^f_{l_1,l_2} k(\boldsymbol{x}, \boldsymbol{x}')$$

where $y_{il} \sim N(f_l(\boldsymbol{x}_i), \sigma_l^2)$, $K^f \in \mathbb{R}^{L \times L}$ is a PSD matrix that specifies the inter-task similarities, and $K^f_{l_1,l_2}$ is the $(l_1, l_2)^{th}$ element of matrix K^f. That is,

$$\begin{aligned} \begin{bmatrix} \boldsymbol{y}_1 \\ \vdots \\ \boldsymbol{y}_L \end{bmatrix} &\sim \mathcal{N}\left(\begin{bmatrix} \boldsymbol{0} \\ \vdots \\ \boldsymbol{0} \end{bmatrix}, \begin{bmatrix} K^f_{11}K & \cdots & K^f_{1L}K \\ \vdots & \ddots & \vdots \\ K^f_{L1}K & \cdots & K^f_{LL}K \end{bmatrix} + \begin{bmatrix} \sigma_1^2 I & \cdots & \boldsymbol{0} \\ \vdots & \ddots & \vdots \\ \boldsymbol{0} & \cdots & \sigma_L^2 I \end{bmatrix} \right) \\ &= \mathcal{N}(\boldsymbol{0}, K^f \otimes K + \Sigma_L) \end{aligned} \tag{6.5}$$

where $K_{f,f} = K^f \otimes K$.

Given the training index set X and the output observations $\boldsymbol{y}$, the posterior distribution of $\boldsymbol{f}(\boldsymbol{x}_*) = \{f_1(\boldsymbol{x}_*), ..., f_L(\boldsymbol{x}_*)\}$ at test point $\boldsymbol{x}_*$ is given by

$$\boldsymbol{f}(\boldsymbol{x}_*)|X, \boldsymbol{y}, \boldsymbol{x}_* \sim \mathcal{N}(\bar{\boldsymbol{f}}(\boldsymbol{x}_*), \Sigma_*) \tag{6.6}$$

where the mean and variance predictions are respectively given by

$$\begin{aligned} \bar{\boldsymbol{f}}(\boldsymbol{x}_*) =& (K^f \otimes K(\boldsymbol{x}_*, X))^T \Sigma^{-1} \boldsymbol{y} \\ \Sigma_* = \text{Var}(\boldsymbol{x}_*) =& (K^f \otimes K(\boldsymbol{x}_*, \boldsymbol{x}_*)) - \\ & (K^f \otimes K(\boldsymbol{x}_*, X)) \Sigma^{-1} (K^f \otimes K(X, \boldsymbol{x}_*)) \end{aligned} \tag{6.7}$$

where $\otimes$ denotes the Kronecker product, $\Sigma = K^f \otimes K(X, X) + \Sigma_L \otimes I$ is a $L\tilde{T} \times L\tilde{T}$, K^f is the matrix that specifies the inter-task similarities, $K(X, X)$ is the matrix of covariances between all pairs of training points, Σ_L is the $L \times L$ diagonal matrix in which the $(l, l)^{th}$ element is σ_l^2, and $K(\boldsymbol{x}_*, X)$ is the vector of covariances between the test point $\boldsymbol{x}_*$ and the training points.

Since only a subset of values $\boldsymbol{y}_o \subseteq \boldsymbol{y}$ has been observed, the covariance matrix Σ only needs to be computed at the observed values. That is, if the observed values $\boldsymbol{y}_o$ correspond to the values in the indexes I_o of the vector $\boldsymbol{y}$, then, from the matrix $(K^f \otimes K(\boldsymbol{x}_*, X))^T \Sigma^{-1}$ only the columns in those indices I_o are needed. This means that the covariance matrix Σ and its inverse only needs to be computed at the observed values. Additionally, from the matrix $(K^f \otimes K(\boldsymbol{x}_*, X))^T$, only the columns associated with the dimensions and time indexes with observations need to be computed (i.e., the columns in the I_o indexes).

Learning Hyperparameters

The parameters to be learned are $\boldsymbol{\theta} = (K^f, \{\sigma_l^2\}_{l=1}^L, \boldsymbol{\eta})$, where $\boldsymbol{\eta}$ are the parameters of the $k(\boldsymbol{x}, \boldsymbol{x}')$ kernel function. The aim is to learn the parameters $\boldsymbol{\theta}$ that maximize the marginal likelihood $p(\boldsymbol{y}_o|X, \boldsymbol{\theta})$. This can be done using 1) gradient-based methods, where the Cholesky decomposition can be used to guarantee the positive-semidefiniteness of K^f (i.e., $K^f = LL^T$, where L is lower triangular), or 2) the EM algorithm.

Taking into account the fact that $\boldsymbol{y}|X \sim N(\boldsymbol{0}, \Sigma)$, the log marginal likelihood to be maximized is defined by:

$$\begin{aligned} \mathcal{L} &= \log\ p(\boldsymbol{y}_o|X, \boldsymbol{\theta}) \\ &= -\frac{1}{2}\log\ \det \Sigma_o - \frac{1}{2}\boldsymbol{y}_o^T \Sigma_o^{-1} \boldsymbol{y}_o - \frac{n_o}{2}\log 2\pi \end{aligned} \tag{6.8}$$

where Σ_o is the covariance matrix at the observed values, and n_o is the length of vector $\boldsymbol{y}_o$.

References

1. P. Senin, "jmotif: Time Series Analysis Toolkit Based on Symbolic Aggregate Dicretization, i.e. SAX." https://cran.r-project.org/package=jmotif, 2018.
2. A. L. Montgomery, V. Zarnowitz, R. S. Tsay, and G. C. Tiao, "Forecasting the U.S. Unemployment Rate," *Journal of the American Statistical Association*, vol. 93, no. 442, pp. 478 —- 493, 1998.
3. H. Song and G. Li, "Tourism demand modelling and forecasting – A review of recent research," *Tourism Management*, vol. 29, no. 2, pp. 203–220, 2008.
4. J. W. Richards, D. L. Starr, N. R. Butler, J. S. Bloom, J. M. Brewer, A. Crellin-Quick, J. Higgins, R. Kennedy, and M. Rischard, "On machine-learned classification of variable stars with sparse and noisy time-series data," *Astrophysical Journal*, vol. 733, no. 1, 2011.
5. M. Maturilli, A. Herber, and G. König-Langlo, "Climatology and time series of surface meteorology in Ny-Ålesund, Svalbard," *Earth System Science Data*, vol. 5, no. 1, pp. 155–163, 2013.
6. A. Kampouraki, G. Manis, and C. Nikou, "Heartbeat Time Series Classification With Support Vector Machines," *IEEE Transactions on Information Technology in Biomedicine*, vol. 13, no. 4, pp. 512 —- 518, 2009.
7. G. E. P. Box and G. M. Jenkins, *Time series analysis: forecasting and control*. Holden-Day, 1976.
8. A. Bagnall, H. Dau, J. Lines, M. Flynn, J. Large, A. Bostrom, P. Southam, and E. Keogh, "The UEA multivariate time series classification archive, 2018," in *arXiv*, pp. 1 – 36, 2018.
9. P. J. García-Laencina, J. L. Sancho-Gómez, and A. R. Figueiras-Vidal, "Pattern classification with missing data: a review," *Neural Computing and Applications*, vol. 19, no. 2, pp. 263 – 282, 2010.
10. A. Sharafoddini, J. A. Dubin, D. M. Maslove, and J. Lee, "A New Insight Into Missing Data in Intensive Care Unit Patient Profiles: Observational Study," *JMIR medical informatics*, vol. 7, no. 1, pp. 1–19, 2019.
11. A. Goldberger, L. Amaral, L. Glass, J. Hausdorff, P. Ivanov, R. Mark, J. Mietus, G. Moody, C. Peng, and H. Stanley, "PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals.," *Circulation [Online].*, vol. 101, no. 23, pp. e215–e220, 2000.
12. D. Dua and C. Graff, "UCI Machine Learning Repository." http://archive.ics.uci.edu/ml, 2019.

13. H. A. Dau, A. Bagnall, K. Kamgar, C. C. M. Yeh, Y. Zhu, S. Gharghabi, C. A. Ratanamahatana, and E. Keogh, "The UCR time series archive," *IEEE/CAA Journal of Automatica Sinica*, vol. 6, no. 6, pp. 1293–1305, 2019.
14. P. Esling and C. Agon, "Time-series data mining," *ACM Computing Surveys (CSUR)*, vol. 45, no. 1, pp. 1–34, 2012.
15. J. G. De Gooijer and R. J. Hyndman, "25 Years of Time Series Forecasting," *International Journal of Forecasting*, vol. 22, no. 3, pp. 443–473, 2006.
16. J. Contreras, R. Espínola, S. Member, and F. J. Nogales, "ARIMA Models to Predict Next-Day Electricity Prices," *IEEE Transactions on Power Systems*, vol. 18, no. 3, pp. 1014–1020, 2003.
17. Y. Kamarianakis and P. Prastacos, "Forecasting Traffic Flow Conditions in an Urban Network: Comparison of Multivariate and Univariate Approaches," *Transportation Research Record*, no. 1857, pp. 74–84, 2003.
18. S. S. Jones, R. S. Evans, T. L. Allen, A. Thomas, P. J. Haug, S. J. Welch, and G. L. Snow, "A multivariate time series approach to modeling and forecasting demand in the emergency department," *Journal of Biomedical Informatics*, vol. 42, no. 1, pp. 123–139, 2009.
19. H. Hewamalage, C. Bergmeir, and K. Bandara, "Recurrent Neural Networks for Time Series Forecasting: Current status and future directions," *International Journal of Forecasting*, vol. 37, no. 1, pp. 388–427, 2021.
20. A. Bagnall, J. Lines, A. Bostrom, J. Large, and E. Keogh, "The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances," *Data Minining and Knowledge Discovery*, vol. 31, no. 3, pp. 606–660, 2017.
21. A. P. Ruiz, M. Flynn, J. Large, M. Middlehurst, and A. Bagnall, *The great multivariate time series classification bake off: a review and experimental evaluation of recent algorithmic advances*. Springer US, 2020.
22. S. B. Kotsiantis, "Supervised Machine Learning: A Review of Classification Techniques," *Informatica*, vol. 31, pp. 249–268, 2007.
23. U. Mori, *Contributions to time series data mining departing from the problem of road travel time modeling*. PhD thesis, University of the Basque Country, 2015.
24. A. Abanda, U. Mori, and J. A. Lozano, "A review on distance based time series classification," *Data Min. Knowl. Discov.*, vol. 33, no. 2, pp. 378–412, 2019.
25. P. Cunningham and S. J. Delany, "K-Nearest Neighbour Classifiers-A Tutorial," *ACM Computing Surveys (CSUR)*, vol. 54, no. 6, 2021.
26. W. A. Chaovalitwongse, Y. J. Fan, and R. C. Sachdeo, "On the time series K-nearest neighbor classification of abnormal brain activity," *IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans*, vol. 37, no. 6, pp. 1005–1016, 2007.
27. J. Lines and A. Bagnall, "Time series classification with ensembles of elastic distance measures," *Data Mining and Knowledge Discovery*, vol. 29, no. 3, pp. 565–592, 2015.
28. D. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series," *Workshop on Knowledge Knowledge Discovery in Databases*, vol. 398, pp. 359–370, 1994.
29. H. Deng, G. Runger, E. Tuv, and M. Vladimir, "A time series forest for classification and feature extraction," *Information Sciences*, vol. 239, pp. 142–153, 2013.

30. T. Rakthanmanon and E. Keogh, "Fast shapelets: A scalable algorithm for discovering time series shapelets," *Proceedings of the 2013 SIAM International Conference on Data Mining, SDM 2013*, pp. 668–676, 2013.
31. P. Schäfer, "The BOSS is concerned with time series classification in the presence of noise," *Data Mining and Knowledge Discovery*, vol. 29, no. 6, pp. 1505–1530, 2015.
32. J. Lines, S. Taylor, and A. Bagnall, "Time series classification with HIVE-COTE: The hierarchical vote collective of transformation-based ensembles," *ACM Transactions on Knowledge Discovery from Data*, vol. 12, no. 5, pp. 1–35, 2018.
33. T. Warren Liao, "Clustering of time series data - A survey," *Pattern Recognition*, vol. 38, no. 11, pp. 1857–1874, 2005.
34. S. Aghabozorgi, A. Seyed Shirkhorshidi, and T. Ying Wah, "Time-series clustering - A decade review," *Information Systems*, vol. 53, pp. 16–38, 2015.
35. A. Kassambara, *Practical Guide To Cluster Analysis in R: Unsupervised Machine Learning*. Sthda, 1 ed., 2015.
36. J. Paparrizos and L. Gravano, "k-Shape : Efficient and Accurate Clustering of Time Series," in *Proceedings of the 2015 ACM SIGMOD international conference on management of data*, pp. 1855–1870, 2015.
37. H. Izakian, W. Pedrycz, and I. Jamal, "Fuzzy clustering of time series data using dynamic time warping distance," *Engineering Applications of Artificial Intelligence*, vol. 39, pp. 235–244, 2015.
38. U. Mori, A. Mendiburu, and J. A. Lozano, "Similarity Measure Selection for Clustering Time Series Databases," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 1, pp. 181–195, 2016.
39. M. Gupta, J. Gao, C. Aggarwal, and J. Han, *Outlier Detection for Temporal Data*. Morgan & Claypool Publishers, 2014.
40. A. Blázquez-García, A. Conde, U. Mori, and J. Lozano, "A review on outlier/anomaly detection in time series data," *ACM Computing Surveys (CSUR)*, vol. 54, no. 3, pp. 1–33, 2021.
41. D. S. Willett, J. George, N. S. Willett, L. L. Stelinski, and S. L. Lapointe, "Machine Learning for Characterization of Insect Vector Feeding," *PLOS Computational Biology*, vol. 12, nov 2016.
42. R. Wu and E. Keogh, "Current Time Series Anomaly Detection Benchmarks are Flawed and are Creating the Illusion of Progress," *IEEE Transactions on Knowledge and Data Engineering*, 2021.
43. C. Aggarwal, *Data Mining*. Springer International Publishing, 1 ed., 2015.
44. S. S. Khan and M. G. Madden, "One-class classification: Taxonomy of study and review of techniques," *Knowledge Engineering Review*, vol. 29, no. 3, pp. 345–374, 2014.
45. S. Mauceri, J. Sweeney, and J. McDermott, "Dissimilarity-based representations for one-class classification on time series," *Pattern Recognition*, vol. 100, p. 107122, 2020.
46. V. Vercruyssen, W. Meert, G. Verbruggen, K. Maes, R. Baumer, and J. Davis, "Semi-Supervised Anomaly Detection with an Application to Water Analytics," in *2018 IEEE International Conference on Data Mining (ICDM)*, pp. 527–536, 2018.
47. E. Keogh, S. Chu, D. Hart, and M. Pazzani, "Segmenting Time Series: A Survey and Novel Approach," *Data mining in time series databases*, pp. 1–21, 2004.

48. C. A. Ratanamahatana, J. Lin, D. Gunopulos, E. Keogh, M. Vlachos, and G. Das, "Mining Time Series Data," in *Data Mining and Knowledge Discovery Handbook* (O. Maimon and L. Rokach, eds.), pp. 1049–1077, Boston, MA: Springer US, 2nd ed., 2010.
49. E. Keogh, S. Chu, D. Hart, and M. Pazzani, "An online algorithm for segmenting time series," in *Proceedings - IEEE International Conference on Data Mining, ICDM*, pp. 289–296, 2001.
50. E. J. Keogh and M. J. Pazzani, "Scaling up dynamic time warping for datamining applications," *Proceeding of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 285–289, 2000.
51. E. Keogh, K. Chakrabarti, S. Mehrotra, and M. Pazzani, "Locally adaptive dimensionality reduction for indexing large time series databases," *SIGMOD Record (ACM Special Interest Group on Management of Data)*, vol. 30, no. 2, pp. 151–162, 2001.
52. J. Lin, E. Keogh, S. Lonardi, and B. Chiu, "A Symbolic Representation of Time Series, with Implications for Streaming Algorithms," in *Proceedings of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery (DMKD '03)*, (San Diego, CA, USA), pp. 2–11, ACM, 2003.
53. Y.-W. Huang and P. S. Yu, "Adaptive query processing for time-series data," in *Proceedings of the fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-99)*, pp. 282–286, 1999.
54. E. Keogh and P. Smyth, "A probabilistic approach to fast pattern matching in time series databases," *Proceedings of the 3rd International Conference of Knowledge Discovery and Data Mining*, no. 1994, pp. 52–57, 1997.
55. A. Mueen, E. Keogh, Q. Zhu, S. Cash, and B. Westover, "Exact discovery of time series motifs," *Society for Industrial and Applied Mathematics - 9th SIAM International Conference on Data Mining 2009, Proceedings in Applied Mathematics*, vol. 1, pp. 469–480, 2009.
56. B. Chiu, E. Keogh, and S. Lonardi, "Probabilistic discovery of time series motifs," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 493–498, 2003.
57. J. Lin, E. Keogh, S. Lonardi, and P. Patel, "Finding motifs in time series," in *Proc. of the 2nd Workshop on Temporal Data Mining*, pp. 53–68, 2002.
58. C. Yeh, Y. Zhu, L. Ulanova, N. Begum, Y. Ding, H. Dau, D. Furtado Silva, A. Mueen, and E. Keogh, "Matrix Profile I: All Pairs Similarity Joins for Time Series: A Unifying View That Includes Motifs, Discords and Shapelets," in *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pp. 1317–1322, 2017.
59. Y. Zhu, Z. Zimmerman, N. S. Senobari, C. C. M. Yeh, G. Funning, A. Mueen, P. Brisk, and E. Keogh, "Matrix profile II: Exploiting a novel algorithm and GPUs to break the one hundred million barrier for time series motifs and joins," in *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pp. 739–748, Institute of Electrical and Electronics Engineers Inc., jan 2017.
60. F. Rasheed and R. Alhajj, "A framework for periodic outlier pattern detection in time-series sequences," *IEEE Transactions on Cybernetics*, vol. 44, no. 5, pp. 569–582, 2014.
61. A. Siffer, P. A. Fouque, A. Termier, and C. Largouët, "Anomaly Detection in Streams with Extreme Value Theory," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '17)*, (Halifax, NS, Canada), pp. 1067–1075, ACM, 2017.

62. A. B. Sharma, L. Golubchik, and R. Govindan, "Sensor faults," *ACM Trans. Sens. Netw.*, vol. 6, no. 3, pp. 1–39, 2010.
63. D. M. Hawkins, *Identification of outliers*. New York: Springer Netherlands, 1980.
64. A. Carreño, I. Inza, and J. A. Lozano, "Analyzing rare event, anomaly, novelty and outlier detection terms under the supervised classification framework," *Artificial Intelligence Review*, pp. 1–20, 2019.
65. C. Aggarwal, *Outlier Analysis*. Springer, 2 ed., 2016.
66. R. Li, H. Huang, K. Xin, and T. Tao, "A review of methods for burst/leakage detection and location in water distribution systems," *Water Science and Technology: Water Supply*, vol. 15, no. 3, pp. 429–441, 2015.
67. J. Kang, Y. J. Park, J. Lee, S. H. Wang, and D. S. Eom, "Novel leakage detection by ensemble CNN-SVM and graph-based localization in water distribution systems," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 5, pp. 4279–4289, 2018.
68. A. J. Fox, "Outliers in Time Series," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 34, no. 3, pp. 350–363, 1972.
69. R. S. Tsay, "Outliers, Level Shifts, and Variance Changes in Time Series," *Journal of Forecasting*, vol. 7, pp. 1–20, 1988.
70. R. S. Tsay, D. Peña, and A. E. Pankratz, "Outliers in multivariate time series," *Biometrika*, vol. 87, no. 4, pp. 789–804, 2000.
71. V. Hodge and J. Austin, "A Survey of Outlier Detection Methodologies," *Artificial Intelligence Review*, vol. 22, no. 2, pp. 85–126, 2004.
72. V. Chandola, A. Banerjee, and V. Kumar, "Outlier Detection: A Survey," 2007.
73. V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Computing Surveys (CSUR)*, vol. 41, no. 3, pp. 1–72, 2009.
74. H. Aguinis, R. K. Gottfredson, and H. Joo, "Best-Practice Recommendations for Defining, Identifying, and Handling Outliers," *Organizational Research Methods*, vol. 16, no. 2, pp. 270–301, 2013.
75. X. Xu, H. Liu, and M. Yao, "Recent Progress of Anomaly Detection," *Complexity*, vol. 2019, pp. 1–11, 2019.
76. M. Gupta, J. Gao, C. Aggarwal, and J. Han, "Outlier Detection for Temporal Data: A Survey," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 9, pp. 2250–2267, 2014.
77. S. Basu and M. Meckesheimer, "Automatic outlier detection for time series: An application to sensor data," *Knowl. Inf. Syst.*, vol. 11, no. 2, pp. 137–154, 2007.
78. S. Mehrang, E. Helander, M. Pavel, A. Chieh, and I. Korhonen, "Outlier detection in weight time series of connected scales," in *Proceedings of the 2015 IEEE International Conference on Bioinformatics and Biomedicine (BIBM '15)*, (Washington, DC, USA), pp. 1489–1496, IEEE, 2015.
79. M. C. Dani, F. X. Jollois, M. Nadif, and C. Freixo, "Adaptive Threshold for Anomaly Detection Using Time Series Segmentation," in *Proceedings of the 22nd International Conference on Neural Information Processing (ICONIP '15)*, (Istanbul, Turkey), pp. 82–89, Springer, Cham, 2015.
80. J. Chen, W. Li, A. Lau, J. Cao, and K. Wang, "Automated load curve data cleansing in power systems," *IEEE Trans. Smart Grid*, vol. 1, no. 2, pp. 213–221, 2010.

81. J. Holešovský, M. Čampulová, and J. Michálek, "Semiparametric outlier detection in nonstationary times series: Case study for atmospheric pollution in Brno, Czech Republic," *Atmospheric Pollut. Res.*, vol. 9, no. 1, pp. 27–36, 2018.
82. M. Čampulová, J. Michálek, P. Mikuška, and D. Bokal, "Nonparametric algorithm for identification of outliers in environmental data," *J. Chemom.*, vol. 32, no. 5, pp. 1–17, 2018.
83. K. M. Carter and W. W. Streilein, "Probabilistic reasoning for streaming anomaly detection," in *IEEE Statistical Signal Processing Workshop (SSP)*, vol. 1, (Ann Arbor, MI, USA), pp. 1–4, IEEE, 2012.
84. S. Song, A. Zhang, J. Wang, and P. S. Yu, "SCREEN: Stream Data Cleaning under Speed Constraints," in *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, (Melbourne, Victoria, Australia), pp. 827–841, ACM, 2015.
85. A. Zhang, S. Song, and J. Wang, "Sequential Data Cleaning: A Statistical Approach," in *Proceedings of the 2016 International Conference on Management of Data (SIGMOD '16)*, (San Francisco, CA, USA), pp. 909–924, ACM, 2016.
86. A. Reddy, M. Ordway-West, M. Lee, M. Dugan, J. Whitney, R. Kahana, B. Ford, J. Muedsam, A. Henslee, and M. Rao, "Using Gaussian Mixture Models to Detect Outliers in Seasonal Univariate Network Traffic," in *2017 IEEE Security and Privacy Workshops (SPW)*, (San Jose, CA, USA), pp. 229–234, IEEE, 2017.
87. J. Hochenbaum, O. S. Vallis, and A. Kejariwal, "Automatic Anomaly Detection in the Cloud Via Statistical Learning," *arXiv preprint arXiv:1704.07706*, 2017.
88. H. N. Akouemo and R. J. Povinelli, "Time series outlier detection and imputation," *2014 IEEE PES General Meeting — Conference Exposition*, pp. 1–5, 2014.
89. H. N. Akouemo and R. J. Povinelli, "Probabilistic anomaly detection in natural gas time series data," *International Journal of Forecasting*, vol. 32, no. 3, pp. 948–956, 2016.
90. H. N. Akouemo and R. J. Povinelli, "Data Improving in Time Series Using ARX and ANN Models," *IEEE Trans. Power Syst.*, vol. 32, no. 5, pp. 3352–3359, 2017.
91. M. Munir, S. A. Siddiqui, A. Dengel, and S. Ahmed, "DeepAnT: A Deep Learning Approach for Unsupervised Anomaly Detection in Time Series," *IEEE Access*, vol. 7, pp. 1991–2005, 2019.
92. D. J. Hill and B. S. Minsker, "Anomaly detection in streaming environmental sensor data: A data-driven modeling approach," *Environmental Modelling and Software*, vol. 25, no. 9, pp. 1014–1022, 2010.
93. Y. Zhang, N. A. S. Hamm, N. Meratnia, A. Stein, M. van de Voort, and P. J. M. Havinga, "Statistics-based outlier detection for wireless sensor networks," *International Journal of Geographical Information Science*, vol. 26, no. 8, pp. 1373–1392, 2012.
94. Y. Zhou, R. Qin, H. Xu, S. Sadiq, and Y. Yu, "A Data Quality Control Method for Seafloor Observatories: The Application of Observed Time Series Data in the East China Sea," *Sensors*, vol. 18, no. 8, p. 2628, 2018.
95. A. Tsymbal, "The problem of concept drift: definitions and related work," tech. rep., Department of Computer Science, Trinity College: Dublin, 2004.
96. J. Gama, I. Zliobaite, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A Survey on Concept Drift Adaptation," *ACM Computing Surveys (CSUR)*, vol. 46, no. 4, pp. 1–35, 2014.

97. V. Losing, B. Hammer, and H. Wersing, "Incremental on-line learning: A review and comparison of state of the art algorithms," *Neurocomputing*, vol. 275, pp. 1261–1274, 2018.
98. Z. Xu, K. Kersting, and L. von Ritter, "Adaptive Streaming Anomaly Analysis," in *Proceedings of NIPS 2016 Workshop on Artificial Intelligence for Data Science*, (Barcelona, Spain), 2016.
99. Z. Xu, K. Kersting, and L. von Ritter, "Stochastic Online Anomaly Analysis for Streaming Time Series," in *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI '17)*, (Melbourne, Australia), pp. 3189–3195, International Joint Conferences on Artificial Intelligence, 2017.
100. S. Ahmad, A. Lavin, S. Purdy, and Z. Agha, "Unsupervised real-time anomaly detection for streaming data," *Neurocomputing*, vol. 262, pp. 134–147, 2017.
101. F. Angiulli and F. Fassetti, "Detecting Distance-Based Outliers in Streams of Data," in *Proceedings of the 16th ACM Conference on Information and Knowledge Management (CIKM '07)*, (Lisbon, Portugal), pp. 811–820, ACM, 2007.
102. F. Angiulli and F. Fassetti, "Distance-based outlier queries in data streams: The novel task and algorithms," *Data Mining and Knowledge Discovery*, vol. 20, no. 2, pp. 290–324, 2010.
103. V. Ishimtsev, A. Bernstein, E. Burnaev, and I. Nazarov, "Conformal k-NN Anomaly Detector for Univariate Data Streams," in *Proceedings of the Sixth Workshop on Conformal and Probabilistic Prediction and Applications*, vol. 60, (Stockholm, Sweden), pp. 213–227, PMLR, 2017.
104. H. Jagadish, N. Koudas, and S. Muthukrishnan, "Mining Deviants in a Time Series Database," in *Proceedings of the 25th International Conference on Very Large Data Bases*, (Edinburgh, Scotland, UK), pp. 102–113, Morgan Kaufmann Publishers Inc., 1999.
105. S. Muthukrishnan, R. Shah, and J. S. Vitter, "Mining Deviants in Time Series Data Streams," in *Proceedings of the 16th International Conference on Scientific and Statistical Database Management*, (Santorini Island, Greece), pp. 41–50, IEEE, 2004.
106. K. Hundman, V. Constantinou, C. Laporte, I. Colwell, and T. Soderstrom, "Detecting Spacecraft Anomalies Using LSTMs and Nonparametric Dynamic Thresholding," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, (London, UK), pp. 387–395, ACM, 2018.
107. S. Papadimitriou, J. Sun, and C. Faloutsos, "Streaming Pattern Discovery in Multiple Time-Series," in *Proceedings of the 31st International Conference on Very Large Data Bases (VLDB 2005)*, (Trondheim, Norway), pp. 697–708, ACM, 2005.
108. P. Galeano, D. Peña, and R. S. Tsay, "Outlier detection in multivariate time series by projection pursuit," *Journal of the American Statistical Association*, vol. 101, no. 474, pp. 654–669, 2006.
109. C. Chen and L. M. Liu, "Joint Estimation of Model Parameters and Outlier Effects in Time Series," *Journal of the American Statistical Association*, vol. 88, no. 421, pp. 284–297, 1993.
110. R. Baragona and F. Battaglia, "Outliers detection in multivariate time series by independent component analysis," *Neural Computation*, vol. 19, no. 7, pp. 1962–1984, 2007.

111. H. Lu, Y. Liu, Z. Fei, and C. Guan, "An outlier detection algorithm based on cross-correlation analysis for time series dataset," *IEEE Access*, vol. 6, pp. 53593–53610, 2018.
112. M. S. Shahriar, D. Smith, A. Rahman, M. Freeman, J. Hills, R. Rawnsley, D. Henry, and G. Bishop-Hurley, "Detecting heat events in dairy cows using accelerometers and unsupervised learning," *Comput. Electron. Agric.*, vol. 128, pp. 20–26, 2016.
113. M. Sakurada and T. Yairi, "Anomaly Detection Using Autoencoders with Nonlinear Dimensionality Reduction," in *Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis*, (Gold Coast, Australia), pp. 4–12, ACM, 2014.
114. T. Kieu, B. Yang, and C. S. Jensen, "Outlier detection for multidimensional time series using deep neural networks," in *Proceedings of the 19th IEEE International Conference on Mobile Data Management*, (Aalborg, Denmark), pp. 125–134, IEEE, 2018.
115. Y. Su, Y. Zhao, C. Niu, R. Liu, W. Sun, and D. Pei, "Robust Anomaly Detection for Multivariate Time Series through Stochastic Recurrent Neural Network," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '19)*, pp. 2828–2837, 2019.
116. Y. Zhou, H. Zou, R. Arghandeh, W. Gu, and C. J. Spanos, "Non-parametric Outliers Detection in Multiple Time Series A Case Study: Power Grid Data Analysis," in *32nd AAAI Conference on Artificial Intelligence*, (New Orleans, Louisiana, USA), pp. 4605–4612, AAAI Press, 2018.
117. Y. Zhou, R. Arghandeh, H. Zou, and C. J. Spanos, "Nonparametric Event Detection in Multiple Time Series for Power Distribution Networks," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 2, pp. 1619–1628, 2019.
118. Y. Zhou, R. Arghandeh, and C. J. Spanos, "Online learning of Contextual Hidden Markov Models for temporal-spatial data analysis," in *2016 IEEE 55th Conference on Decision and Control (CDC)*, (Las Vegas, NV, USA), pp. 6335–6341, IEEE, 2016.
119. H. Cheng, P. N. Tan, C. Potter, and S. Klooster, "A robust graph-based algorithm for detection and characterization of anomalies in noisy multivariate time series," in *Workshop Proceedings of the 8th IEEE International Conference on Data Mining (ICDM '08)*, (Pisa, Italy), pp. 349–358, IEEE, 2008.
120. H. Cheng, P. N. Tan, C. Potter, and S. Klooster, "Detection and Characterization of Anomalies in Multivariate Time Series," in *Proceedings of the 2009 SIAM International Conference on Data Mining*, (Sparks, Nevada, USA), pp. 413–424, SIAM, 2009.
121. X. Li, Z. Li, J. Han, and J. G. Lee, "Temporal outlier detection in vehicle traffic data," in *2009 IEEE 25th International Conference on Data Engineering*, (Shanghai, China), pp. 1319–1322, IEEE, 2009.
122. P. Senin, J. Lin, X. Wang, T. Oates, S. Gandhi, A. P. Boedihardjo, C. Chen, and S. Frankenstein, "Time series anomaly discovery with grammar-based compression," in *Proceedings of 18th International Conference on Extending Database Technology (EDBT 2015)*, (Brussels, Belgium), pp. 481–492, OpenProceedings.org, 2015.
123. X. Wang, J. Lin, N. Patel, and M. Braun, "Exact variable-length anomaly detection algorithm for univariate and multivariate time series," *Data Minining and Knowledge Discovery*, vol. 32, no. 6, pp. 1806–1844, 2018.

124. E. Keogh, S. Lonardi, and B. Y. C. Chiu, "Finding surprising patterns in a time series database in linear time and space," in *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '02)*, (Edmonton, Alberta, Canada), pp. 550–556, ACM, 2002.
125. C. Chen and D. J. Cook, "Energy Outlier Detection in Smart Environments," in *Workshops at the 25th AAAI Conference on Artificial Intelligence*, (San Francisco, California, USA), pp. 9–14, AAAI Press, 2011.
126. C. Chen, D. J. Cook, and A. S. Crandall, "The user side of sustainability: Modeling behavior and energy usage in the home," *Pervasive and Mobile Computing*, vol. 9, no. 1, pp. 161–175, 2013.
127. L. Wei, N. Kumar, V. Lolla, E. Keogh, S. Lonardi, and C. Ann, "Assumption-free anomaly detection in time series," in *Proceedings of the 17th International Conference on Scientific and Statistical Database Management (SSDBM 2005)*, (Santa Barbara, CA, USA), pp. 237–240, Lawrence Berkeley Laboratory, 2005.
128. N. Kumar, N. Lolla, E. Keogh, S. Lonardi, and C. A. Ratanamahatana, "Time-series Bitmaps: A Practical Visualization Tool for working with Large Time Series Databases," in *Proceedings of the 5th SIAM International Conference on Data Mining*, (Newport Beach, CA, USA), pp. 531–535, SIAM, 2005.
129. V. Chandola, A. Banerjee, and V. Kumar, "Anomaly Detection for Discrete Sequences: A Survey," *IEEE Transactions on Knowledge and Data Engineering*, vol. 24, no. 5, pp. 823 – 839, 2012.
130. D. Carrera, B. Rossi, D. Zambon, P. Fragneto, and G. Boracchi, "ECG monitoring in wearable devices by sparse models," in *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, (Riva del Garda, Italy), pp. 145–160, Springer, 2016.
131. M. Jones, D. Nikovski, M. Imamura, and T. Hirata, "Exemplar learning for extremely efficient anomaly detection in real-valued time series," *Data Min. Knowl. Discov.*, vol. 30, no. 6, pp. 1427–1454, 2016.
132. H. Ren, M. Liu, Z. Li, and W. Pedrycz, "A Piecewise Aggregate pattern representation approach for anomaly detection in time series," *Knowledge-Based Systems*, vol. 135, pp. 29–39, 2017.
133. J. Yang, W. Wang, and P. S. Yu, "InfoMiner: Mining Surprising Periodic Patterns," in *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '01)*, (San Francisco, CA, USA), pp. 395–400, ACM, 2001.
134. E. Keogh, J. Lin, and A. Fu, "HOT SAX: Efficiently Finding the Most Unusual Time Series Subsequence," in *Proceedings of the 5th IEEE International Conference on Data Mining (ICDM '05)*, (Houston, TX, USA), pp. 226–233, IEEE, 2005.
135. J. Lin, E. Keogh, A. Fu, and H. Van Herle, "Approximations to Magic: Finding Unusual Medical Time Series," in *Proceedings of the 18th IEEE Symposium on Computer-Based Medical Systems*, (Dublin, Ireland), pp. 329–334, IEEE, 2005.
136. E. Keogh, J. Lin, S. H. Lee, and H. van Herle, "Finding the most unusual time series subsequence: Algorithms and applications," *Knowledge and Information Systems*, vol. 11, no. 1, pp. 1–27, 2007.
137. A. W. C. Fu, O. T. W. Leung, E. Keogh, and J. Lin, "Finding Time Series Discords Based on Haar Transform," in *Proceedings of the 2nd International Conference on Advanced Data Mining and Applications (ADMA '06)*, (Xi'an, China), pp. 31–41, Springer, Berlin, Heidelberg, 2006.

138. Y. Bu, T. W. Leung, A. W. C. Fu, E. Keogh, J. Pei, and S. Meshkin, "Wat: Finding top-k discords in time series database," in *Proceedings of the 7th SIAM International Conference on Data Mining*, (Minneapolis, Minnesota, USA), pp. 449–454, SIAM, 2007.
139. G. Li, O. Bräysy, L. Jiang, Z. Wu, and Y. Wang, "Finding time series discord based on bit representation clustering," *Knowledge-Based Systems*, vol. 54, pp. 243–254, 2013.
140. H. Sanchez and B. Bustos, "Anomaly Detection in Streaming Time Series Based on Bounding Boxes," in *Proceedings of the 7th International Conference on Similarity Search and Applications (SISAP '14)*, (Los Cabos, Mexico), pp. 201–213, Springer, Cham, 2014.
141. P. M. Chau, B. M. Duc, and D. T. Anh, "Discord Discovery in Streaming Time Series based on an Improved HOT SAX Algorithm," in *Proceedings of the 9th International Symposium on Information and Communication Technology*, (Danang City, Vietnam), pp. 24–30, ACM, 2018.
142. H. T. Q. Buu and D. T. Anh, "Time series discord discovery based on iSAX symbolic representation," in *Proceedings of the 3rd International Conference on Knowledge and Systems Engineering*, (Hanoi, Vietnam), pp. 11–18, IEEE, 2011.
143. Y. Liu, X. Chen, F. Wang, and J. Yin, "Efficient detection of discords for time series stream," in *Proceedings of the Joint International Conferences on Advances in Data and Web Management*, (Suzhou, China), pp. 629–634, Springer, Berlin, Heidelberg, 2009.
144. E. Keogh, K. Chakrabarti, M. Pazzani, and S. Mehrotra, "Dimensionality Reduction for Fast Similarity Search in Large Time Series Databases," *Knowledge and Information Systems*, vol. 3, no. 3, pp. 263–286, 2001.
145. H. Izakian and W. Pedrycz, "Anomaly detection in time series data using a fuzzy c-means clustering," in *Proceedings of the 2013 Joint IFSA World Congress and NAFIPS Annual Meeting*, (Edmonton, Alberta, Canada), pp. 1513–1518, IEEE, 2013.
146. J. A. Silva, E. R. Faria, R. C. Barros, E. R. Hruschka, A. C. De Carvalho, and J. Gama, "Data stream clustering: A survey," *ACM Computing Surveys (CSUR)*, vol. 46, no. 1, pp. 1–37, 2013.
147. H. Moonesinghe and P. N. Tan, "Outlier Detection Using Random Walks," in *Proceedings of the 18th IEEE International Conference on Tools with Artificial Intelligence (ICTAI '06)*, (Arlington, VA, USA), pp. 532–539, IEEE, 2006.
148. M. Longoni, D. Carrera, B. Rossi, P. Fragneto, M. Pessione, and G. Boracchi, "A wearable device for online and long-term ECG monitoring," in *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI'18)*, (Stockholm, Sweden), pp. 5838–5840, International Joint Conferences on Artificial Intelligence, 2018.
149. D. Carrera, B. Rossi, P. Fragneto, and G. Boracchi, "Online anomaly detection for long-term ECG monitoring using wearable devices," *Pattern Recognition*, vol. 88, pp. 482–492, 2019.
150. F. Rasheed, M. Alshalalfa, and R. Alhajj, "Efficient periodicity mining in time series databases using suffix trees," *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, no. 1, pp. 79–94, 2011.
151. J. Yang, W. Wang, and P. S. Yu, "Mining surprising periodic patterns," *Data Mining and Knowledge Discovery*, vol. 9, pp. 189–216, 2004.

152. M. Jones, D. Nikovski, M. Imamura, and T. Hirata, "Anomaly Detection in Real-Valued Multidimensional Time Series," in *International Conference on Bigdata/Socialcom/Cybersecurity*, (Stanford, CA, USA), Citeseer, 2014.
153. M. Hu, X. Feng, Z. Ji, K. Yan, and S. Zhou, "A novel computational approach for discord search with local recurrence rates in multivariate time series," *Information Sciences*, vol. 477, pp. 220–233, 2019.
154. R. J. Hyndman, E. Wang, and N. Laptev, "Large-Scale Unusual Time Series Detection," in *Proceedings of the 15th IEEE International Conference on Data Mining Workshop (ICDMW '15)*, (Atlantic City, NJ, USA), pp. 1616–1619, IEEE, 2015.
155. N. Laptev, S. Amizadeh, and I. Flint, "Generic and Scalable Framework for Automated Time-series Anomaly Detection," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '15)*, (Sydney, NSW, Australia), pp. 1939–1947, ACM, 2015.
156. U. Rebbapragada, P. Protopapas, C. E. Brodley, and C. Alcock, "Finding anomalous periodic time series: An application to catalogs of periodic variable stars," *Machine Learning*, vol. 74, no. 3, pp. 281–313, 2009.
157. S. E. Benkabou, K. Benabdeslem, and B. Canitia, "Unsupervised outlier detection for time series by entropy and dynamic time warping," *Knowledge and Information Systems*, vol. 54, no. 2, pp. 463–486, 2018.
158. J. Lara, D. Lizcano, V. Rampérez, and J. Soriano, "A method for outlier detection based on cluster analysis and visual expert criteria," *Expert Systems*, pp. 1–23, 2019.
159. L. Ye and E. Keogh, "Time series shapelets: a new primitive for data mining," in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '09)*, (Paris, France), pp. 947–956, ACM, 2009.
160. L. Beggel, B. X. Kausler, M. Schiegg, M. Pfeiffer, and B. Bischl, "Time series anomaly detection based on shapelet learning," *Computational Statistics*, vol. 34, no. 3, pp. 945–976, 2019.
161. A. Iturria, J. Carrasco, S. Charramendieta, A. Conde, and F. Herrera, "otsad: A package for online time-series anomaly detectors," *Neurocomputing*, vol. 374, pp. 49–53, 2020.
162. P. Senin, J. Lin, X. Wang, T. Oates, S. Gandhi, A. P. Boedihardjo, C. Chen, and S. Frankenstein, "GrammarViz 3.0: Interactive Discovery of Variable-Length Time Series Patterns," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 12, no. 1, pp. 1–28, 2018.
163. E. Keogh and J. Lin, "Clustering of time-series subsequences is meaningless: Implications for previous and future research," *Knowledge and Information Systems*, vol. 8, no. 2, pp. 154–177, 2005.
164. E. Farah and I. Shahrour, "Leakage Detection Using Smart Water System: Combination of Water Balance and Automated Minimum Night Flow," *Water Resources Management*, vol. 31, no. 15, pp. 4821–4833, 2017.
165. S. R. Mounce, A. J. Day, A. S. Wood, A. Khan, P. D. Widdop, and J. Machell, "A neural network approach to burst detection," *Water Science and Technology*, vol. 45, no. 4-5, pp. 237–246, 2002.
166. P. S. Murvay and I. Silea, "A survey on gas leak detection and localization techniques," *Journal of Loss Prevention in the Process Industries*, vol. 25, no. 6, pp. 966–973, 2012.

167. Y. Wu and S. Liu, "A review of data-driven approaches for burst detection in water distribution systems," *Urban Water Journal*, vol. 14, no. 9, pp. 972–983, 2017.
168. S. R. Mounce, A. Khan, A. S. Wood, A. J. Day, P. D. Widdop, and J. Machell, "Sensor-fusion of hydraulic data for burst detection and location in a treated water distribution system," *Information Fusion*, vol. 4, no. 3, pp. 217–229, 2003.
169. S. R. Mounce and J. Machell, "Burst detection using hydraulic data from water distribution systems with artificial neural networks," *Urban Water Journal*, vol. 3, no. 1, pp. 21–31, 2006.
170. G. Ye and R. A. Fenner, "Kalman filtering of hydraulic measurements for burst detection in water distribution systems," *Journal of Pipeline Systems Engineering and Practice*, vol. 2, no. 1, pp. 14–22, 2011.
171. M. Zadkarami, M. Shahbazian, and K. Salahshoor, "Pipeline leakage detection and isolation: An integrated approach of statistical and wavelet feature extraction with multi-layer perceptron neural network (MLPNN)," *Journal of Loss Prevention in the Process Industries*, vol. 43, pp. 479–487, 2016.
172. D. Vries, B. van den Akker, E. Vonk, W. de Jong, and J. van Summeren, "Application of machine learning techniques to predict anomalies in water supply networks," *Water Science and Technology: Water Supply*, vol. 16, no. 6, pp. 1528–1535, 2016.
173. P. Huang, N. Zhu, D. Hou, J. Chen, Y. Xiao, J. Yu, G. Zhang, and H. Zhang, "Real-time burst detection in District Metering Areas in water distribution system based on patterns of water demand with supervised learning," *Water*, vol. 10, no. 12, p. 1765, 2018.
174. S. R. Mounce, J. B. Boxall, and J. Machell, "Development and verification of an online artificial intelligence system for detection of bursts and other abnormal flows," *Journal of Water Resources Planning and Management*, vol. 136, no. 3, pp. 309–318, 2010.
175. S. R. Mounce, R. B. Mounce, and J. B. Boxall, "Novelty detection for time series data analysis in water distribution systems using support vector machines," *Journal of Hydroinformatics*, vol. 13, no. 4, pp. 672–686, 2011.
176. M. Romano, Z. Kapelan, and D. A. Savić, "Real-time leak detection in water distribution systems," in *Water Distribution Systems Analysis*, pp. 1074–1082, 2010.
177. M. Romano, Z. Kapelan, and D. A. Savić, "Automated detection of pipe bursts and other events in water distribution systems," *Journal of Water Resources Planning and Management*, vol. 140, no. 4, pp. 457–467, 2014.
178. M. Fagiani, S. Squartini, L. Gabrielli, M. Severini, and F. Piazza, "A statistical framework for automatic leakage detection in smart water and gas grids," *Energies*, vol. 9, no. 9, p. 665, 2016.
179. G. Ye and R. A. Fenner, "Weighted least squares with expectation-maximization algorithm for burst detection in U.K. water distribution systems," *Journal of Water Resources Planning and Management*, vol. 140, no. 4, pp. 417–424, 2014.
180. X. Wang, G. Guo, S. Liu, Y. Wu, X. Xu, and K. Smith, "Burst Detection in District Metering Areas Using Deep Learning Method," *Journal of Water Resources Planning and Management*, vol. 146, no. 6, pp. 1–12, 2020.

181. C. V. Palau, F. J. Arregui, and M. Carlos, "Burst detection in water networks using principal component analysis," *Journal of Water Resources Planning and Management*, vol. 138, no. 1, pp. 47–54, 2012.
182. Y. Wu, S. Liu, X. Wu, Y. Liu, and Y. Guan, "Burst detection in district metering areas using a data driven clustering algorithm," *Water Research*, vol. 100, pp. 28–37, 2016.
183. K. Aksela, M. Aksela, and R. Vahala, "Leakage detection in a real distribution network using a SOM," *Urban Water Journal*, vol. 6, no. 4, pp. 279–289, 2009.
184. S. Patabendige, R. Cardell-Oliver, R. Wang, and W. Liu, "Detection and interpretation of anomalous water use for non-residential customers," *Environmental Modelling and Software*, vol. 100, pp. 291–301, 2018.
185. Y. Wu and S. Liu, "Burst Detection by Analyzing Shape Similarity of Time Series Subsequences in District Metering Areas," *Journal of Water Resources Planning and Management*, vol. 146, no. 1, pp. 1–12, 2020.
186. I. Golan and R. El-Yaniv, "Deep anomaly detection using geometric transformations," in *Advances in Neural Information Processing Systems*, vol. 31, pp. 9758–9769, 2018.
187. S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," in *International Conference on Learning Representations*, pp. 1–16, 2018.
188. S. Wang, Y. Zeng, X. Liu, E. Zhu, J. Yin, C. Xu, and M. Kloft, "Effective End-to-end Unsupervised Outlier Detection via Inlier Priority of Discriminative Network," in *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 5962–5975, 2019.
189. A. Bagnall, M. Flynn, J. Large, J. Lines, and M. Middlehurst, "A tale of two toolkits, report the third: On the usage and performance of HIVE-COTE v1.0." arXiv preprint arXiv:2004.06069, 2020.
190. Z. Che, S. Purushotham, K. Cho, D. Sontag, and Y. Liu, "Recurrent Neural Networks for Multivariate Time Series with Missing Values," *Scientific Reports*, vol. 8, no. 1, pp. 1–12, 2018.
191. Z. Lipton, D. Kale, and R. Wetzel, "Modeling Missing Data in Clinical Time Series with RNNs," *Proceedings of Machine Learning for Healthcare*, vol. 58, no. 4, pp. 725–737, 2016.
192. H. Harutyunyan, H. Khachatrian, D. Kale, G. Ver Steeg, and A. Galstyan, "Multitask learning and benchmarking with clinical time series data," *Scientific Data*, vol. 6, no. 1, pp. 1–18, 2019.
193. Y. Luo, "Multivariate Time Series Imputation with Generative Adversarial Networks," in *Advances in Neural Information Processing Systems (NeurIPS'18)*, pp. 1—-12, 2018.
194. Y. Luo, Y. Zhang, X. Cai, and X. Yuan, "E2GaN: End-to-end generative adversarial network for multivariate time series imputation," in *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI'19)*, pp. 3094–3100, 2019.
195. S. Shukla and B. Marlin, "Interpolation-prediction networks for irregularly sampled time series," in *The Seventh International Conference on Learning Representations (ICLR'19)*, pp. 1–14, 2019.
196. S. Shukla and B. Marlin, "Multi-Time Attention Networks for Irregularly Sampled Time Series," in *The Ninth International Conference on Learning Representations (ICLR'21)*, pp. 1–15, 2021.

197. J. Futoma, S. Hariharan, and K. Heller, "Learning to Detect Sepsis with a Multitask Gaussian Process RNN Classifier," in *Thirty-fourth International Conference on Machine Learning (ICML'17)*, pp. 1174–1182, 2017.
198. J. Futoma, *Gaussian Process-Based Models for Clinical Time Series in Healthcare*. PhD thesis, Duke University, 2018.
199. S. Li and B. Marlin, "A scalable end-to-end Gaussian process adapter for irregularly sampled time series classification," in *Thirtieth Conference on Neural Information Processing Systems (NIPS'16)*, pp. 1—-11, 2016.
200. S. Li and B. Marlin, "Classification of Sparse and Irregularly Sampled Time Series with Mixtures of Expected Gaussian Kernels and Random Features," in *31st Conference on Uncertainty in Artificial Intelligence (UAI'15)*, pp. 1—-10, 2015.
201. Z. Liu and M. Hauskrecht, "Learning adaptive forecasting models from irregularly sampled multivariate clinical data," in *30th AAAI Conference on Artificial Intelligence (AAAI'16)*, pp. 1273—-1279, 2016.
202. F. Bianchi, L. Livi, K. Mikalsen, M. Kampffmeyer, and R. Jenssen, "Learning representations of multivariate time series with missing data," *Pattern Recognition*, vol. 96, p. 106973, 2019.
203. B. Marlin, D. Kale, R. Khemani, and R. Wetzel, "Unsupervised pattern discovery in electronic health care data using probabilistic clustering models," in *Proceedings of the 2nd ACM SIGHIT International Health Informatics Symposium (IHI'12)*, pp. 389–398, 2012.
204. I. Baytas, C. Xiao, X. Zhang, F. Wang, A. Jain, and J. Zhou, "Patient subtyping via time-aware LSTM networks," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'17)*, pp. 65–74, 2017.
205. C. Rasmussen and C. Williams, *Gaussian Processes for Machine Learning*. MA: MIT Press, 2006.
206. E. Bonilla, K. Chai, and C. Williams, "Multi-task Gaussian Process prediction," in *Proceedings of the 23rd Annual Conference on Neural Information Processing Systems (NIPS'09)*, pp. 1–8, 2009.
207. I. Urteaga, T. Bertin, T. Hardy, D. Albers, and N. Elhadad, "Multi-task Gaussian processes and dilated convolutional networks for reconstruction of reproductive hormonal dynamics," in *Machine Learning for Healthcare Conference (PMLR)*, pp. 60–90, 2019.
208. A. Freitas, "A Critical Review of Multi-Objective Optimization in Data Mining: a position paper," *ACM SIGKDD Explorations Newsletter*, vol. 6, no. 2, pp. 77–86, 2004.
209. A. Konak, D. Coit, and A. Smith, "Multi-objective optimization using genetic algorithms: A tutorial," *Reliability engineering & system safety*, vol. 91, no. 9, pp. 992–1007, 2006.
210. K. Deb, A. Member, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multi-objective genetic algorithm: NSGA-II," *IEEE transactions on evolutionary computation*, vol. 6, no. 2, pp. 182–197, 2002.
211. D. Jones, S. Mirrazavi, and M. Tamiz, "Multi-objective meta-heuristics: An overview of the current state-of-the-art," *European Journal of Operational*, vol. 137, pp. 1–9, 2002.
212. B. Afessa, M. Keegan, O. Gajic, R. Hubmayr, and S. Peters, "The influence of missing components of the Acute Physiology Score of APACHE III on the

measurement of ICU performance," *Intensive Care Medicine*, vol. 31, no. 11, pp. 1537–1543, 2005.
213. J. Gardner, G. Pleiss, D. Bindel, K. Weinberger, and A. Wilson, "Gpytorch: Blackbox matrix-matrix Gaussian process inference with GPU acceleration," in *32nd Conference on Neural Information Processing Systems (NeurIPS'18)*, pp. 1–21, 2018.
214. J. Blank and K. Deb, "Pymoo: Multi-Objective Optimization in Python," *IEEE Access*, vol. 8, pp. 89497–89509, 2020.
215. K. Mikalsen, C. Soguero-Ruiz, and R. Jenssen, "A Kernel to Exploit Informative Missingness in Multivariate Time Series from EHRs," *Explainable AI in Healthcare and Medicine*, pp. 23–36, 2021.
216. S. Bhattacharya, V. Rajan, and H. Shrivastava, "ICU mortality prediction: A classification algorithm for imbalanced datasets," in *31st AAAI Conference on Artificial Intelligence (AAAI'17)*, pp. 1288–1294, 2017.
217. H. Deng, G. Runger, E. Tuv, and M. Vladimir, "A time series forest for classification and feature extraction," *Information Sciences*, vol. 239, pp. 142–153, 2013.
218. T. Le Nguyen, S. Gsponer, I. Ilie, M. O'Reilly, and G. Ifrim, "Interpretable time series classification using linear models and multi-resolution multi-domain symbolic representations," *Data Mining and Knowledge Discovery*, vol. 33, no. 4, pp. 1183–1222, 2019.
219. M. Shokoohi-Yekta, B. Hu, H. Jin, J. Wang, and E. Keogh, "Generalizing DTW to the multi-dimensional case requires an adaptive approach," *Data Mining and Knowledge Discovery*, vol. 31, no. 1, pp. 1–31, 2017.
220. M. Löning, A. Bagnall, S. Ganesh, V. Kazakov, J. Lines, and F. Király, "sktime: A Unified Interface for Machine Learning with Time Series," in *arXiv*, pp. 1–9, 2019.
221. G. B. Moody and R. G. Mark, "The impact of the MIT-BIH arrhythmia database," *IEEE Engineering in Medicine and Biology Magazine*, vol. 20, no. 3, pp. 45–50, 2001.
222. B. Zhou, S. Liu, B. Hooi, X. Cheng, and J. Ye, "BeatGAN: Anomalous Rhythm Detection using Adversarially Generated Time Series," in *International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 4433–4439, 2019.
223. C. Zhang, D. Song, Y. Chen, X. Feng, C. Lumezanu, W. Cheng, J. Ni, B. Zong, H. Chen, and N. V. Chawla, "A Deep Neural Network for Unsupervised Anomaly Detection and Diagnosis in Multivariate Time Series Data," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 1409–1416, 2019.
224. P. Boniol, J. Paparrizos, T. Palpanas, and M. J. Franklin, "SAND: Streaming subsequence anomaly detection," *Proceedings of the VLDB Endowment*, vol. 14, no. 10, pp. 1717–1729, 2021.
225. V. Fortuin, D. Baranchuk, G. Rätsch, and S. Mandt, "GP-VAE: Deep Probabilistic Time Series Imputation," in *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS'20)*, pp. 1651–1661, 2020.
226. K. Hayashi, T. Takenouchi, R. Tomioka, and H. Kashima, "Self-measuring similarity for multi-task Gaussian process," in *Proceedings of ICML Workshop on Unsupervised and Transfer Learning (JMLR: Workshop and Conference Proceedings)*, pp. 145–153, 2012.

227. H. Wackernagel, "Multivariate Geostatistics: An Introduction with Applications," *International Journal of Rock Mechanics and Mining Sciences and Geomechanics Abstracts*, vol. 8, no. 33, p. 363A, 1996.