UPMC
SORBONNE UNIVERSITÉS

THÈSE DE DOCTORAT DE
l'UNIVERSITÉ PIERRE ET MARIE CURIE

Spécialité

Informatique

École doctorale Informatique, Télécommunications et Électronique (Paris)

Présentée par

Xavier RENARD

Pour obtenir le grade de

DOCTEUR de l'UNIVERSITÉ PIERRE ET MARIE CURIE

Sujet de la thèse :

Time Series Representation for Classification: A Motif-Based Approach

soutenue le 15 septembre 2017

devant le jury composé de :

Mme. Ahlame Douzal	Rapportrice
M. Louis Wehenkel	Rapporteur
M. Patrick Gallinari	Examinateur
M. Pierre-François Marteau	Examinateur
M. Themis Palpanas	Examinateur
M. Marcin Detyniecki	Directeur de thèse
Mme. Maria Rifqi	Directrice de thèse
M. Gabriel Fricout	Encadrant industriel

Time Series Representation for Classification

A Motif-Based Approach

Abstract

Résumé

Acknowledgements

Contents

Chapter 1

Introduction

Time series data is everywhere: any measurement of a phenomenon over time produces time series. Every scientific discipline has many examples: physics and natural sciences provide large amounts of time series datasets with measurements of many parameters such as temperature, pressure, flow or concentration in meteorology, climatology, hydrology or earth science. In medicine, electrocardiogram (ECG) and electroencephalogram (EEG) are classical time series use-cases together with many other physiological parameters. In economics and in financial markets, stock market prices are notorious examples. In the industry, process monitoring produces massive amounts of sensor measurements. Time series represent our environment with large quantities of data that require the development of automatic techniques to analyze and extract the relevant information they contain.

Time series mining is the discipline dedicated to the development of such techniques, for the automatic discovery of meaningful knowledge from time series data. It provides techniques and algorithms to perform machine learning tasks on time series (classification, clustering, motif discovery, etc.). Time series mining exists as a specific field because time series have their own properties and challenges. In particular the meaningful information in the time series is encoded across time with trends, shapes or subsequences usually with distortions. Approaches have been developed to overcome these issues while the high computational complexity is handled: for instance, the core of most time series mining algorithms is composed of specific distance measures and time series representations.

This manuscript put an emphasis on the representation of the information contained in the time series. We have identified three main groups of time series representations: time-based representations (a raw time series is transformed into another time series that can be denoised, compressed and with the meaningful information highlighted), feature-based representations (a raw time series is transformed into a classical feature vector with features mainly derived from the feature analysis field, to characterize the structure of the time series for instance) and motif-based representations (meaningful subsequences

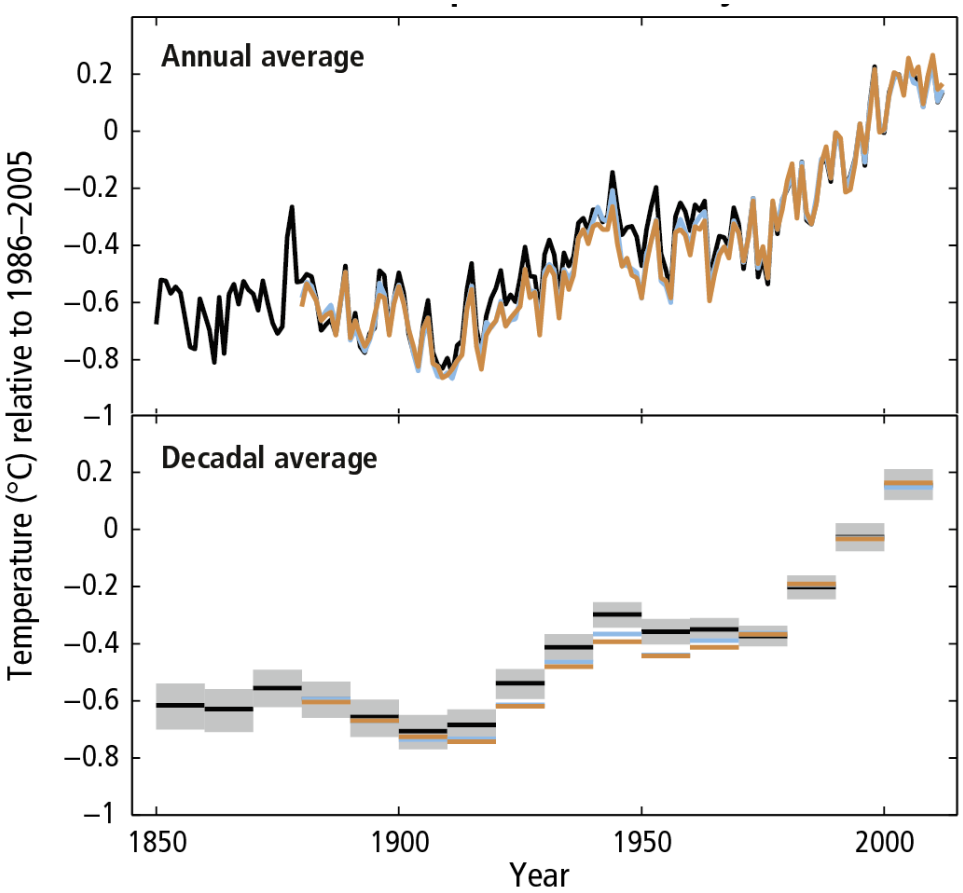


Figure 1.1: An example of time series: the observed globally averaged combined land and ocean surface temperature anomaly (from the 2014 Synthesis Report on climate change - Intergovernmental Panel on Climate Change)

are discovered from the time series: such subsequences can be recurrent, surprising or discriminant).

In this work, our objective is to develop a framework to learn subsequence-based representations from time series datasets to perform classification. One major drawback of existing classification strategies based on time series subsequences is their very high computational complexity. This complexity contributes to limit the expressiveness and the richness of the learned representation from the time series. This work aims to overcome the computational complexity issue to enable in practice the discovery of a rich subsequence-based time series representation for classification.

The theoretical developments of this work are benchmarked on industrial applications in the frame of a *CIFRE* agreement (industrial research agreement) with Arcelormittal, the worldwide market leader for steel production. Our proposition has been applied to the detection of defective products based on production line's sensor measurements.

This manuscript is divided into three parts. The first part presents an overview of the time series mining field, the issues to perform machine learning on time series data and usual solutions developed in the literature. The second part details our proposition to learn a subsequence-based representation from time series meaningful for time series classification. The third part is about the evaluation of our approach on the industrial applications.

First part: state of the art to learn from time series

The first part of this manuscript is an overview of the time series mining field, with its challenges and solutions.

In **Chapter 2** we detail the notation and the concepts used in the manuscript and we present the main time series mining tasks. Then, we expose the issues to train common machine learning algorithms on time series data and the common approaches to overcome them. Our work is focused on time series classification. We gather the time series classification approaches in two groups: time-based classification (based on distance measures) and feature-based approaches (based on suitable time series representations). Our work focuses on the second approach and more precisely on time series representations for classification: in **chapter 3,** we present an overview of the time series representation literature.

Second part: our proposition to learn a subsequence-based representation for time series classification

In the second part of this manuscript we introduce our proposition to learn subsequence-based representation for time series classification.

After an overview of the field's literature in **chapter 4**, we present our vision of the

problem in **chapter 5** to formalize a framework for the discovery of subsequence-based representation from time series to perform classification, which produces a classical feature vector suitable for machine learning algorithms. However, the computational complexity of the discovery on time series datasets prevents us to instantiate the framework on real world use-cases. In **chapter 6,** we demonstrate the possibility to reduce drastically this complexity thanks to the observation that many subsequences are redundant in a time series dataset. Finally, in **chapter 7**, we cast the discovery of the representation into a classical feature selection problem. Our proposition is evaluated through extensive experiments on more than one hundred datasets from the classical time series mining benchmark.

Third part: industrial applications

Our work is framed in a *CIFRE* agreement (industrial research agreement) with Arcelormittal. The evaluation of our proposition is presented in the third part of this work.

The industrial context, the use-cases and the datasets are presented in **chapter 8**. In **chapter 9,** our proposition is benchmarked on the industrial datasets. This chapter also illustrates the potential of subsequence-based representations, and our proposition in particular, to extract interpretable and relevant insights from the time series for process experts to perform further analysis.

We complete this manuscript by a conclusion and a discussion on the perspectives opened by this work.

Part I

Learning from Time Series

This part aims to provide the background for our proposition. Chapter 2 begins with a short introduction to the definitions and notations of the main concepts used in the work. Then, an overview of the time series mining field is provided, with a description of the main tasks of the domain. The application of common machine learning algorithms on time series is not straightforward: it raises several issues and challenges that will be discussed. In particular, we will see that a time series hardly fit in the classical static attribute-value model typical in machine learning [Kadous and Sammut, 2005]. Two typical ways to overcome these issues consist in the use of either adequate distance measures or the extraction or meaningful representations of the time series. Since our proposition is about the development of a representation based on subsequences, chapter 3 proposes an overview of the time series representations .

Chapter 2

Machine Learning on Time Series

In this chapter, we introduce the time series mining field. We begin with the definitions of the main concepts and notations used in this manuscript. Then, we propose an overview of the main time series mining tasks. Training machine learning models on time series raise issues. We will discuss them and describe two main strategies developed in the literature to overcome them: all time series mining approaches make use of time series representations and distance or similarity measures. Our work, and thus this manuscript, is mainly focused on time series classification.

2.1 Definitions & Notations

We introduce here the concept of time series and the notations used in this manuscript.
A time series is an ordered sequence of real variables, resulting from the observation of an underlying process from measurements usually made at uniformly spaced time instants according to a given sampling rate [Esling and Agon, 2012]. A time series can gather all the observations of a process, which can result in a very long sequence. A time series can also result of a stream and be semi-infinite.

Machine learning techniques typically consider feature vectors with independent or uncorrelated variables. This is not the case with time series where data are gathered sequentially in time with repeated observations of a system: successive measurements are considered correlated and the time-order is vital [Analyis et al., 2010]. The correlation across time of the measurements lead to specific issues and challenges in machine learning, in particular to represent the meaningful information, usually encoded in shapes or trends spanning over several points in time.

Formally, we have a dataset $\mathcal{D}$ composed of N time series T_n such that

$$\mathcal{D} = \{T_1, \ldots, T_n, \ldots, T_N\}$$

Figure 2.1: Example of medical time series: an electrocardiogram (ECG). The relevant information in an ECG is encoded through shapes, thus successive points are correlated (from *Wikimedia*)

A time series T_n has a length $L =\mid T_n \mid$ with $L_{min} \leq L \leq L_{max} \in \mathbb{N}^*$ where L_{min} is the smallest time series of $\mathcal{D}$ and L_{max} is the longest one. Then a time series T_n is noted:

$$T_n = [T_n(1), \ldots, T_n(i), \ldots, T_n(L)]$$

A time series is univariate if for each timestep i the time series value is a scalar, usually a real number such that:

$$T_n(i) = x \text{ such that } x \in \mathbb{R}, \forall i \in [1 \ldots L]$$

A time series is multivariate if for each timestep i the time series value is a vector of scalars, usually real numbers. The vector is of dimension M such that:

$$T_n(i) = [T_{n,1}(i), \ldots, T_{n,m}(i), \ldots, T_{n,M}(i)] \text{ such that } T_{n,m}(i) \in \mathbb{R}, \forall i \in [1, \ldots, L]$$

The variable m of a multivariate time series T_n is noted:

$$T_{n,m} = [T_{n,m}(1), \ldots, T_{n,m}(i), \ldots, T_{n,m}(L)]$$

A multivariate time series is described by a matrix $L \times M$ while a univariate time series is described by a vector of length L.

2.2 Overview of the time series mining field

The time series mining field can be seen as an instance of the data mining field applied to time series. It involves machine learning techniques to discover automatically meaningful knowledge from sets of time series. Some time series mining tasks, such as classification, perform predictions based on a representation of the time series. We propose in this section an overview of the different time series mining tasks.

2.2.1 Motif discovery

To introduce the motif discovery task we must define the concept of subsequence. A subsequence s is a contiguous set of points extracted from a time series $T_{n,m}$ starting at position i of length l such that:

$$s_{T_{n,m}}(i,l) = [T_{n,m}(i), \ldots, T_{n,m}(i+l-1)]$$

A motif is a subsequence with specific properties. Motif discovery is the process that returns a *motif* or a *set of motifs* from a time series or a set of time series. Several types of motifs exist.

Recurrent motif A recurrent motif is a subsequence that appears recurrently in a longer time series or in a set of time series without trivial matches (overlapping areas) [Lin et al., 2002]. Recurrent motifs can also be defined as high density in the space of subsequences [Minnen et al., 2007], their discovery then consist in locating such regions.

Infrequent or surprising motif & anomaly detection An infrequent or surprising motif is a subsequence that has never been seen or whose frequency of occurrence is significantly lower than other subsequences. Several concepts exist around this definition, such as the *time series discords* [Keogh et al., 2005, Keogh et al., 2007] defined as "subsequences maximally different to all the rest of the subsequences (...) most unusual subsequences within a time series" or defined as the detection of previously unknown patterns [Ratanamahatana et al., 2010]. This kind of motif is related with anomaly detection in time series, which aims to discover abnormal subsequences [Weiss, 2004, Leng, 2009, Fujimaki et al., 2009], defined as a motif whose frequency of occurrence differs substantially from expected, given previously seen data [Keogh et al., 2002].

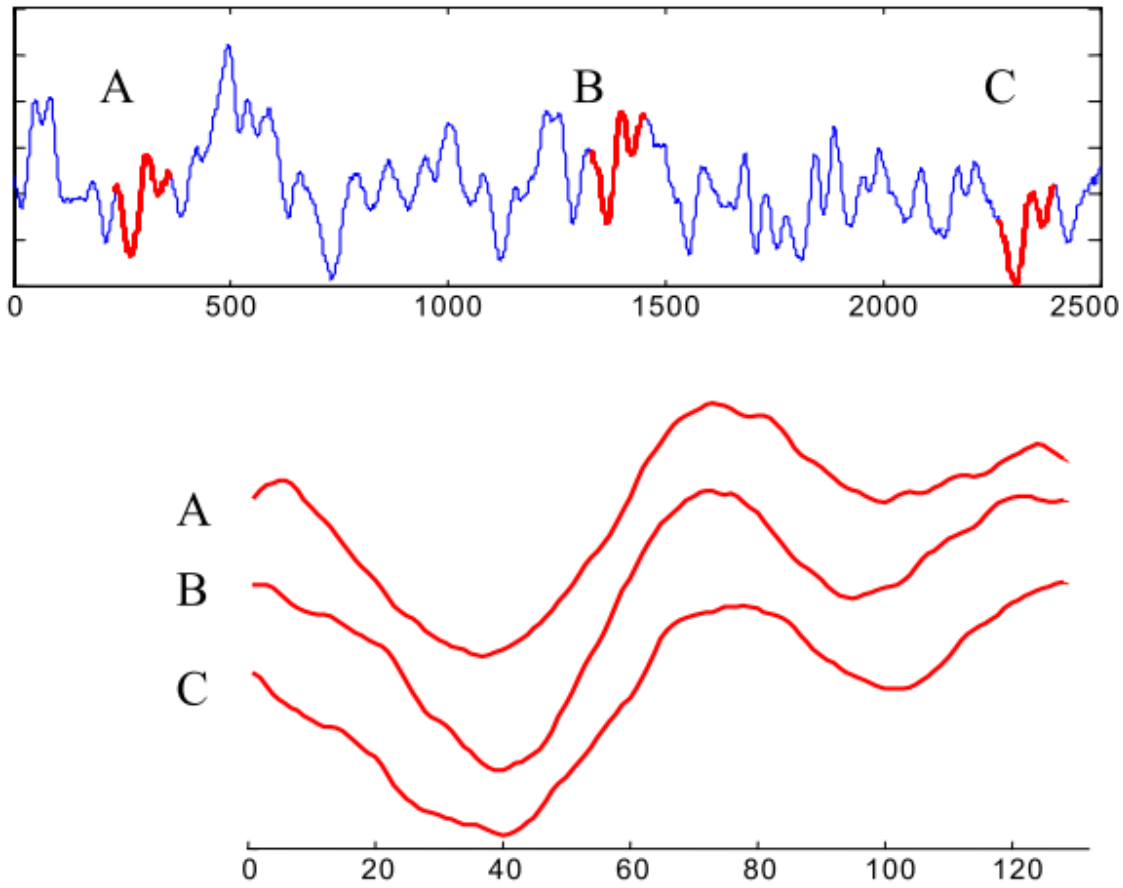


Figure 2.2: Recurrent motif: 3 similar subsequences can be identified in the time series (illustration from [Lin et al., 2002])

Task-specific relevant motif Motif discovery can be driven with a specific task in mind. For instance, it may be expected from discovered motifs to support a classification task. In this case, a motif or a set of motifs is said to be discriminant of class labels. The time series shapelet [Ye and Keogh, 2009] is one instance of this definition, among others [Zhang et al., 2009].

Many publications discuss the concept of time series motifs with often a proposition of algorithm. Such algorithms usually intend to solve the computational complexity of the motif discovery, since the naive discovery based on exhaustive subsequence enumeration is prohibitive [Keogh et al., 2002, Weiss, 2004, Mueen, 2013]. The precise definition of recurrent, surprising or relevant is not obvious, many articles propose a criterion or a heuristic to detect motif among subsequences according to their own definition: the definition depends on the task to solve [Ratanamahatana et al., 2010]. A deeper review on motif discovery is performed in section 3.4.

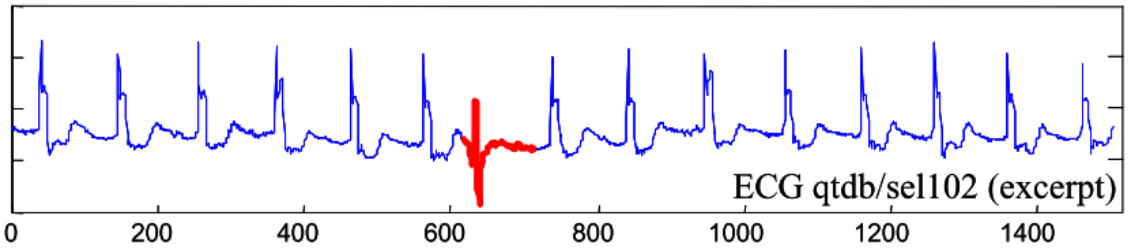


Figure 2.3: Abnormal, surprising motif (illustration from [Keogh et al., 2005])

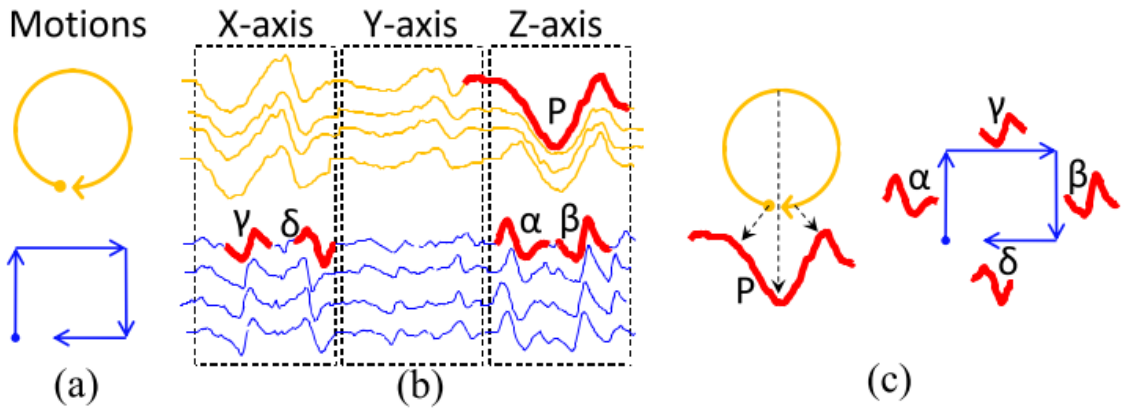


Figure 2.4: Examples of characteristic motifs to label motion types from an accelerometer sensor (illustration from [Mueen et al., 2011])

The motif is a central concept in time series mining since it allows the capture of temporal shapes.

2.2.2 Time series retrieval

Given a query time series T_n, time series retrieval aims to find the most similar time series in a dataset $\mathcal{D}$ based on their information content [Agrawal et al., 1993a, Faloutsos et al., 1994]. The querying can be performed on complete time series or on time series subsequences. In the latter case, every subsequence of the series that matches the query is returned. These two querying approaches are named respectively whole series matching and subsequence matching [Faloutsos et al., 1994, Keogh et al., 2001, Ratanamahatana et al., 2010]. Subsequence matching can be seen as a whole time series matching problem, when time series are divided into subsequences of arbitrary length segments or by motif extraction.

One major issue is the computational complexity of the retrieval. Many approaches have been proposed [Agrawal et al., 1993a, An et al., 2003, Faloutsos et al., 1997] based on a combination of time series representation (to reduce the dimensionality of the time series and accelerate the querying process) approximate similarity measure*s* (to quickly discard irrelevant candidates without false dismissal, cf. lower bounding principle) and efficient indexing.

Time series retrieval has caught most of the research attention in time series mining [Esling and Agon, 2012].

2.2.3 Clustering

Time series clustering aims to find groups of similar time series in a dataset. These groups are usually found by assembling similar time series in order to decrease the variance inside groups of similar time series and increase the variance between them. A survey on time series clustering can be found in [Liao, 2005]. The main issue of most clustering approaches is to determine the number of clusters. Since clusters are not predefined, a domain expert may often be required to interpret the obtained results [Ratanamahatana et al., 2010].

Clustering is divided into *whole* sequence clustering and *subsequence* clustering [Ratanamahatana et al., 2010]. In whole sequence clustering, the whole time series are grouped into clusters. When the raw time series are considered, specific distance measures can be used to handle distortions and length differences with classical clustering algorithms. In subsequence clustering, subsequences are extracted from time series to perform the clustering. Literature suggests that extracting subsequences must be handled with care to avoid meaningless results [Keogh and Lin, 2005]. In particular, subsequence clustering may not provide meaningful results if the clustering is performed on the entire set of subsequences.

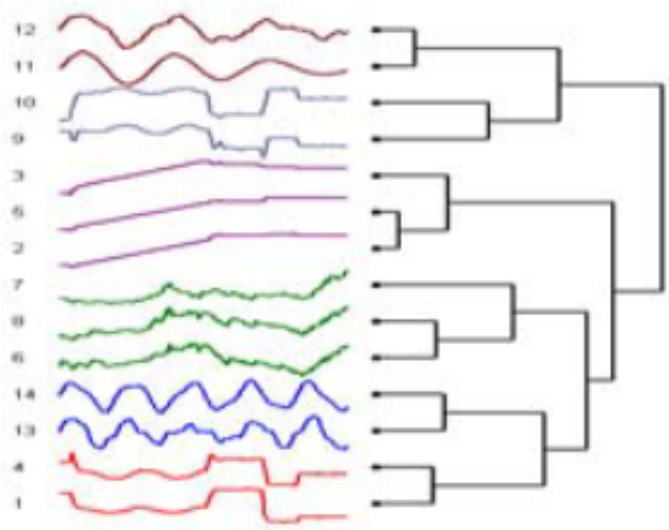


Figure 2.5: An example of time series clustering (illustration from [Wang et al., 2005])

Relevant subsequences should be extracted from the time series prior to clustering, for instance with a motif discovery algorithm [Chiu et al., 2003].

Time series clustering can be a stage for several other time series mining tasks, such as summarization or motifs discovery algorithm [Keogh et al., 2007].

2.2.4 Temporal pattern mining - Rule discovery

Rule discovery aims to relate patterns in a time series to other patterns from the same time series series or others [Das et al., 1998]. One approach is based on the discretization of time series into sequences of symbols. Then association rule mining algorithms are used to discover relationships between symbols (ie. patterns). Such algorithms have been developed to mine association rules between sets of items in large databases [Agrawal et al., 1993b]. [Das et al., 1998], for instance, has extended it to discretized time series. Instead of association rules, decision tree can be learned from motifs extracted from the time series [Ohsaki and Sato, 2002].

2.2.5 Anomaly detection

Anomaly detection is defined as the problem of detecting "patterns in data that do not conform to expected behavior" [Kandhari, 2009]. In a previous paragraph, we mentioned the discovery of motifs on a definition based on abnormal subsequences. Motif is not the only way to label a time series as anomalous, other descriptors can be used. For instance, [Basu and Meckesheimer, 2007] propose an approach based on the difference of a point to the median of the other points in its neighborhood.

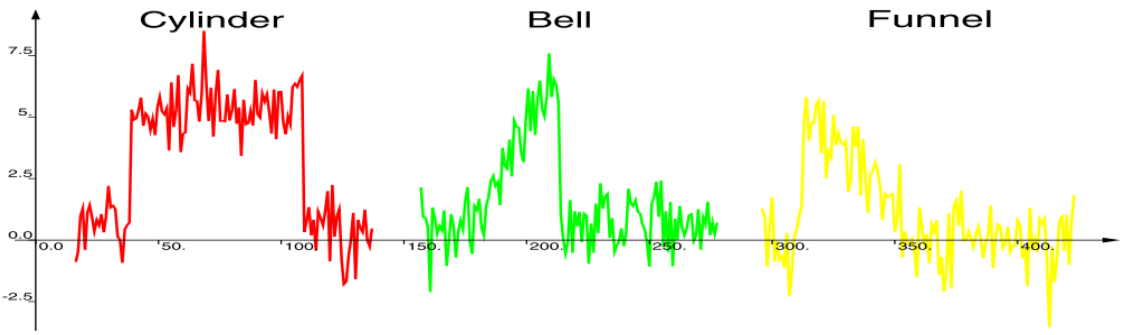


Figure 2.6: Example of time series classification: a dataset is composed by time series with cylinder, bell and funnel shapes. The task is to learn the information that discriminate these time series to label new time series (illustration from [Geurts, 2001])

2.2.6 Summarization

A time series is usually long with complex information, an automatic summary may be useful to get insights on the time series. Simple statistics (mean, standard deviation, etc.) are often insufficient or inaccurate to describe suitably a time series [Ratanamahatana et al., 2010]. A specific processing has to be used to summarize time series. For instance, anomaly detection and motif discovery can be used to get a summary of the time series content: anomalous, interesting or repeating motifs are reported. Summarization can also be seen as a special case of clustering, where time series are mapped to clusters, each cluster having a simple description or a prototype to provide a higher view on the series.

Related with the summarization task, visualization tools of time series have been developed in the literature (Time Searcher, Calendar-based visualization, Spiral, VizTree) [Ratanamahatana et al., 2010].

2.2.7 Classification

Time series data also supervised approaches, such as classification. Time series classification aims to predict a label given a time series in input of a model. The main task consists in the discovery of discriminative features from the time series to distinguish the classes from each other. Time series classification is used to perform for instance pattern recognition, spam filtering, medical diagnosis (classifying ECG, EEG, physiological measurements), anomaly or intrusion detection, transaction sequence data in a bank or detecting malfunction in industry applications [Ratanamahatana et al., 2010, Xing et al., 2010].

We can distinguish the *weak* classification from the *strong* classification.

Weak Classification One single class is associated with a time series. It is probably the most common classification scenario in the literature. An example is shown figure 2.6.

Strong Classification Several labels can be used to describe with more granularity a time series. An example is the record of a patient's health over time: various conditions may alternate from wealthy periods to illness along one time series describing a physiological parameter [Xing et al., 2010]. In this case a sequence of labels is associated with a time series instead of one single global label.

In the proposition of this work (part II) and in the industrial application (part III) we consider a *weak* supervised learning problem where each time series T_n is associated with exactly one class label $y(T_n) \in \mathcal{C}$ where $\mathcal{C} = \{c_1 \dots c_{|\mathcal{C}|}\}$ is a finite set of class labels and $Y = [y(T_1) \dots y(T_N)]$.

2.3 Relationships between time series mining, time series analysis and signal processing

Other scientific communities are involved in the development of techniques to process and analyze time series, for instance time series analysis and signal processing.

Time series analysis aims to develop techniques to analyze and model the temporal structure of time series to describe an underlying phenomena and possibly to forecast future values. The correlation of successive points in a time series is assumed. When it is possible to model this dependency (autocorrelation, trend, seasonality, etc.), it is possible to fit a model and forecast the next values of a series. Signal processing is a very broad field with many objectives: for instance to improve signal quality, to compress it for storage or transmission or detect a particular pattern.

Time series mining is at the intersection of these fields and machine learning: as we will see chapter 3, the application of machine learning techniques on time series data benefits from feature extraction and transformation techniques designed in other communities.

2.4 Time series mining raises specific issues

The tasks presented in the previous chapter are typical of the machine learning field, however time series have specificities that prevent us to apply common approaches developed in the general machine learning literature. The reasons for this are discussed below.

2.4.1 A time series is not a suitable feature vector for machine learning

In general, a raw time series cannot be considered as suitable to feed a machine learning algorithm for several reasons.

In statistical decision theory, a feature vector X is usually a vector of m real-valued random variables such as $X \in \mathbb{R}^{\mathrm{m}}$. In supervised learning, it exists an output variable Y, whose domain depends on the application (for instance a finite set of values $Y \in \mathcal{C}$ in classification or $Y \in \mathbb{R}$ in regression) such as X and Y are linked by an unknown joint distribution $Pr(X, Y)$ that is approximated with a function f such as $f(X) \to Y$. The function f is chosen according to hypothesis made on the data distribution and f is fitted in order to optimize a loss function $L(Y, f(X))$ to penalize prediction errors. The feature vector X is expected to be low-dimensional in order to avoid the *curse of dimensionality* phenomenon that affects performances of f because instances are located in a sparse feature-space [Hastie et al., 2009]. [Hegger et al., 1998] discusses the impact of high dimensionality to build a meaningful feature space from time series to perform time series analysis: with time series, the density of vectors is small and decreases exponentially with the dimension. To counter this effect an exponentially increasing number of instances in the dataset is necessary. Additionally, the relative position of a random variable in the feature vector X is not taken into account to fit f.

Time series data is by nature high dimensional, thousands points by instance or more is common. A major characteristic of time series is the correlation between successive points in time, with two immediate consequences. First, the preservation of the order of the time series points (ie. the random variables of X) is essential. Secondly, the intrinsic dimensionality of the relevant information contained in a time series is typically much lower than the whole time series dimensionality [Ratanamahatana et al., 2010], since they are correlated, many points are redundant.

To illustrate the specific issues while learning from time series, let's take a naive approach. We consider the whole time series as a feature vector in input of any classical machine learning algorithm (decision tree, SVM, neural network, etc.). Each point of the time series is a dimension of the feature vector as illustrated figure 2.7.

A given dimension of X is expected to be a random variable with the same distribution across instances of the dataset. It means that time series have to be aligned across the dataset. This strong assumption is hard to meet for several reasons, in particular because of the distortions a time series can suffer.

- Time series from various dataset's instances may not share the same length: the resulting feature vectors would not have the same number of dimensions.
  It is usually not possible to change the input size of a machine learning algorithm on-the-fly. Each time series that doesn't fit the input size would require an adjustment: either truncating if too long or adding meaningless points (eg. zero padding) if too small. The consequences would be a loss of information in the first case and the addition of noise in the latter.

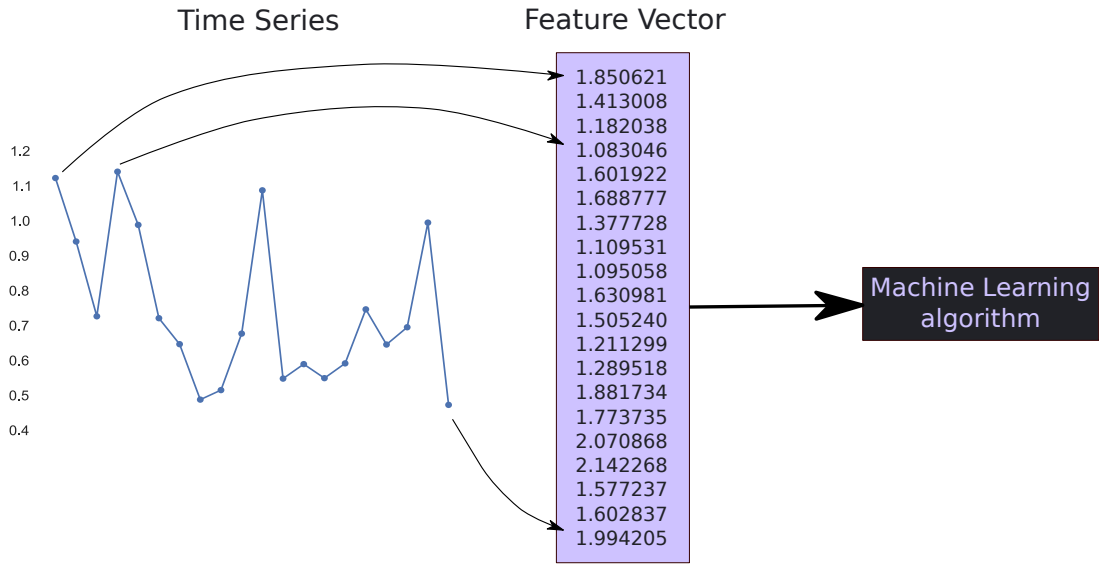


Figure 2.7: A time series is set up as input vector of a machine learning algorithm without pre-processing

- Time series measure the occurrence of a phenomenon: the recording must have been performed with a perfect timing or a posterior segmentation of the time series is required to isolate the phenomenon from raw measurements to obtain a feature vector X where each dimension samples the same phenomenon's stage across instances. For instance, if a set of time series store city temperatures, the first point of each time series must share the same time-stamp such as "January mean temperature" and so on. [Hu et al., 2013] highlights this issue: "literature usually assumes that defining the beginning and the ending points of an interesting pattern can be correctly identified while this is unjustified". Also, the phenomenon of interest may be a localized subsequence in a larger time series. This subsequence may be out of phase across instances and appears at random positions in the time series. Figure 2.8 illustrates this point. The red motif, shifted, would appear in different positions in the feature vector. Motif discovery is a complex task and dedicated approaches are needed with processing steps to form a proper feature vector.

- A given phenomenon may occur at various speeds, frequencies and it may have slight distortions such as local accelerations, deceleration or gaps. While the shape of the motif would be similar, the points would be warped in time and would appear in different positions in the feature space (see Figure 2.9). A time series can suffer several types of distortions discussed section 2.5.1.

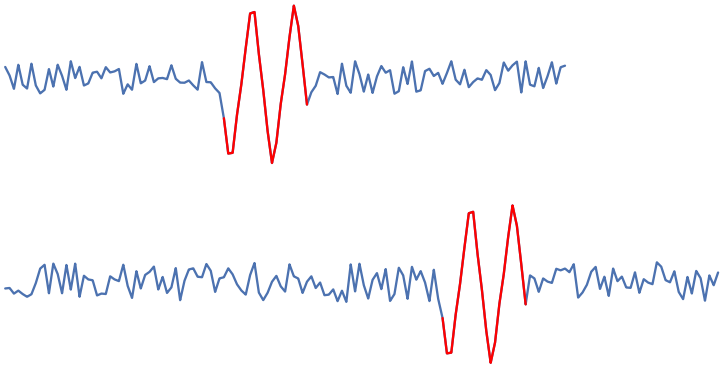


Figure 2.8: Two time series of various lengths recording a similar phenomenon described by the red motif

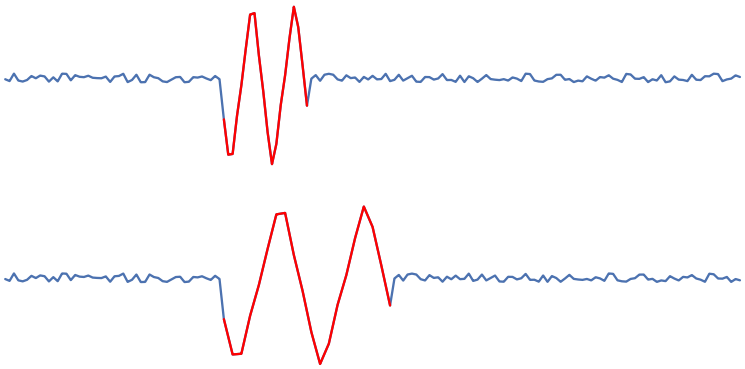


Figure 2.9: A similar motif of two time series but warped in time

- A time series can be multivariate: the previous issues are reinforced in this case.

Beside these issues and the curse of dimensionality, the high dimensionality of the time series is both a computing and a storage challenge. The information contained in a time series may be efficiently compressed using an adequate representation of the data. An adequate representation presents several advantages: beyond lower storage and computing requirements, a compact representation of the time series can ease the learning process by highlighting the relevant information and decreasing the dimensionality of the problem. The question of the representation of the time series information is discussed in detail chapter 3.

In the next section, we detail the main approaches developed in the literature to train machine learning algorithms from time series data to handle these issues.

2.5 Solutions to train machine learning algorithms on time series

Training predictors from time series often requires a quantification of the similarity between time series. A predictor will try to learn a mapping to label or cluster identically similar time series. The issue is on what grounds to decide that time series content is similar. The literature offers many propositions, but as mentioned in [DeBarr and Lin, 2007] there is not one single approach that performs best for all datasets. The main reason resides in the complexity and the heterogeneity of the information contained in time series.

To compare time series, literature usually makes use of two complementary concepts: *time series representation* and *distance measure*.

- A *time series representation* transforms a time series into another time series or a feature vector. The objectives are to highlight the relevant information contained in the original time series, to remove noise, to handle distortions and usually to reduce the dimensionality of the data to decrease the complexity of further computations. A representation exposes the features on which time series will be compared.

- A *distance measure* quantify the similarity between time series, either based on the raw time series or their representations. In the latter case, all the time series must obviously share the same representation procedure. Complex distance measures usually aims to handle time series distortions.

Based on these two concepts, two main strategies emerge from the literature to apply machine learning algorithms on time series data. We call them *time-based* and *feature-based* approaches.

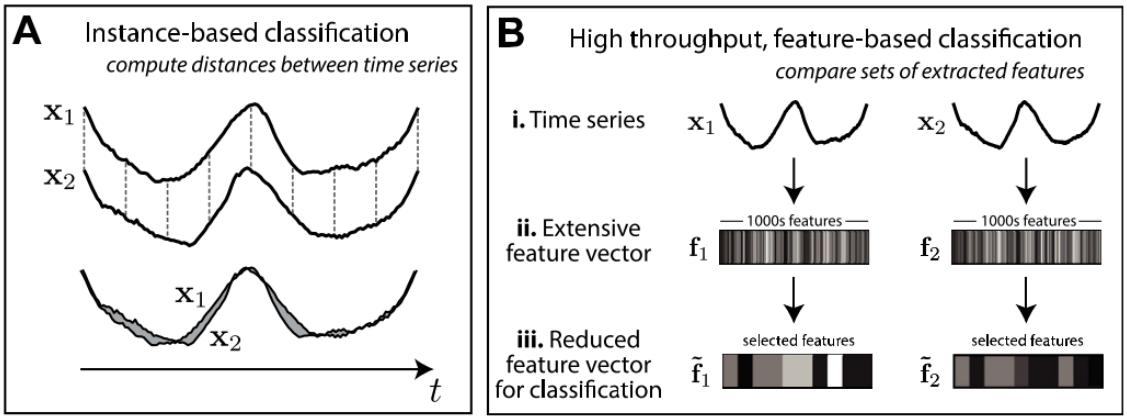


Figure 2.10: *Time-based* (also name *instance-based)* and *Feature-based* approaches. Time-based approaches compare the whole time series: suitable distance measures are crucial. Feature-based approaches extract representations of the information to generate features to feed typical machine learning algorithms (from [Fulcher and Jones, 2014])

Time-based approaches consider the whole time series and apply *distance measures* on the time series to quantify their similarity. A *time series representation* can be used, it typically produces another time series of lower dimension to decrease the computational requirements.

Feature-based approaches transform the time series into a vector of features. The resulting *time series representation* is not yet a time series, but a set of features manually or automatically designed to extract local or global characteristics of the time series.

Figure 2.10 illustrates these concepts. The following sections present an overview of the two approaches. Since our work is focused on time series classification, we mainly review *time-based* and *feature-based* classification approaches.

2.5.1 Time-based classification

Time-based classifiers are suitable when the time series are well segmented on the phenomenon to describe so that a matching, performed over the whole time series in the time domain using distance measures, is relevant. This approach has a lot to do with time series retrieval. It has been claimed that one instance of the *time-based* classification, the nearest neighbor ($k-NN$), is difficult to beat when it is associated with the right distance measure to handle the time series distortions [Batista et al., 2011]. However, this approach assumes that the whole time series are comparable and thus perfectly segmented around the phenomenon of interest. A $k-NN$ returns the k nearest time series in a dataset for an unobserved time series in input. Since the time series are associated with a label, the unobserved time series to label is associated with the majority class of its k neighbors. The $1-NN$ in association with the Euclidean distance [Keogh and Kasetty, 2002] has

been shown competitive in comparison with more complex distance measures (for eg. the Dynamic Time Warping -DTW) when the number of instances in the dataset is large [Ding et al., 2008].

Time-based classification requires from all the time series in the dataset to share the same length and the relevant patterns to be aligned. However, some distortions can be handled with suitable distance measures. In fact, most of the literature on *time-based* approaches focuses on distance measures to improve their capacity to handle misaligned time series and distortions using for instance *elastic* distance measures [Lines and Bagnall, 2015].

In the next paragraphs we propose a brief overview of the distortions a time series can suffer and the families of distance measures that have been developed to overcome them.

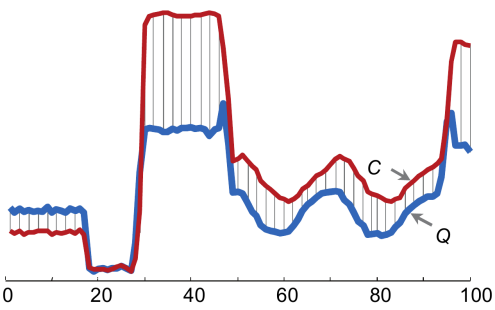
Time series distortions

Time series comes from real world measurements: similar time series will present distortions, in time or in amplitude, which will prevent us to identify them efficiently. In this section, we propose a brief overview of the distortions one can encounter with time series. The literature has many discussions on this topic, we rely on the overview by [Batista et al., 2011].

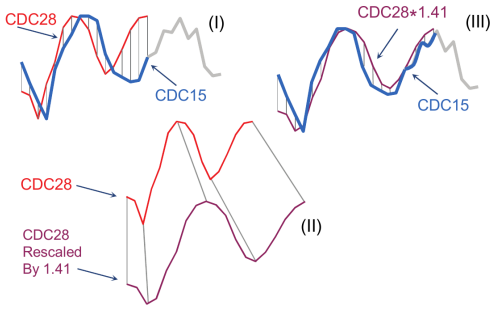
Difference of amplitude and offset Two time series can present similar shapes while their values are on different scales (see figure 2.11a). The phenomenon may not be measured in the exact same conditions (experimental conditions, measurement unit, etc.) or it may occurs with various amplitudes while the intrinsic shape is comparable. Some distance measures, such as the Euclidean distance, suffer from this kind of distortion and fails to identify similar time series with amplitude or offset differences. A normalization of each time series by their means and standard deviations (*z-normalization*) is suitable to address this distortion.

Time warping - Local scaling Two time series may present similar shapes but locally accelerated or decelerated (see figure 2.11c), this distortion is named *time warping*. One solution to address this distortion is the use of an elastic distance measure, for instance the *Dynamic Time Warping* (DTW).

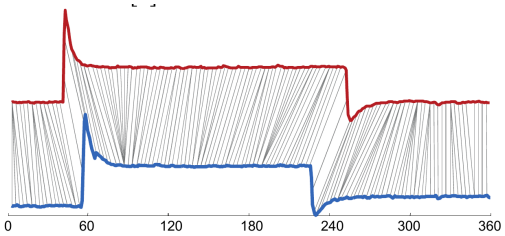
Uniform scaling The uniform scaling can be seen as a simple case of the time warping where the whole time series is uniformly warped in time (the time series is stretched or compressed, see figure 2.11b). For instance, several time series measure a similar phenomenon that occurs at various (but constant) speeds or measured with a distinct sampling rate.



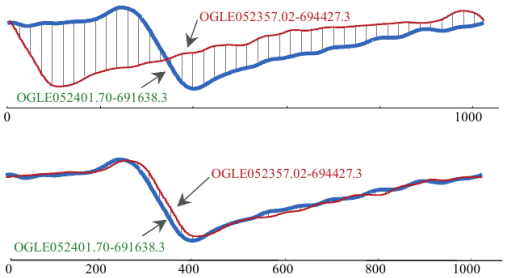
(a) Amplitude scaling



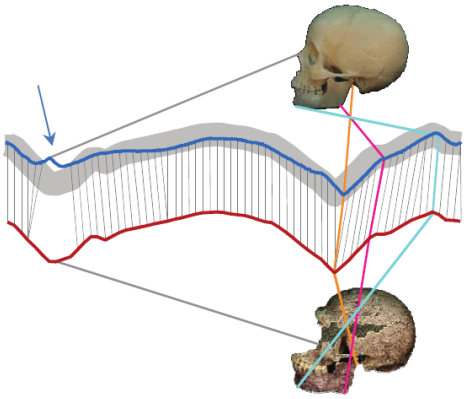
(b) Uniform scaling



(c) Time warping - Local invariance



(d) Phase invariance



(e) Occlusion invariance

Figure 2.11: Time series distortions (illustrations from [Batista et al., 2011])

The solution in this case is to find the factor to realign the time series [Keogh and Lin, 2005].

Phase invariance The phenomenon of interest may be randomly positioned in a time series: the time series is not segmented precisely around this phenomenon because the starting bounds are not known. A more complex situation is the occurrence of several motifs randomly positioned in the time series (see figure 2.11d).

There are two solutions for this issue. Either the whole time series are comparable and all the possible alignments must be tested, or the time series would benefit from a *representation* in another space than the original time domain to extract the relevant information. For instance periodicities are well represented with a Fourier transform and subsequences can be extracted with motif discovery algorithms.

Occlusion invariance [Batista et al., 2011] defines one last distortion, when part of the time series is missing (figure 2.11e). In this case, elastic distance measure is also suitable.

Quantification of time series similarity

Given two time series, the issue is how to measure their similarity having in mind the noise and the distortions that can affect them. It is not straightforward to define a distance measure and decide which invariances to distortions should be enabled. An additional invariance property usually requires additional computation complexity and a distortion can be problematic for a use case but not for another one: the distance measure selection is linked with hypothesis on the information contained in the time series. For instance, a motif dilated in time may represents the same information in one case while it may not in another one. The distance measure may somewhat be considered as an hyper-parameter of the machine learning algorithm it serves.

For time-based comparison of time series, distance measures can be grouped into lock-step distances where the i^{th} point of a time series is compared with the i^{th} of another time series, and *elastic* distances that allow comparison of one point of a time series to several points of another one in order to handle distortions in time [Wang et al., 2013].

Our objective in this section is not to be exhaustive on time series distance measures but rather to give some instances of the two families of distance measures: we advise the reader to refer to dedicated surveys such as [Ding et al., 2008, Wang et al., 2013].

Lock-step distances The most commonly used distance measure for time-based comparison of time series is the Euclidean distance. It is easy to implement, compute and interpret [Keogh and Kasetty, 2002]. The Euclidean distance has also been shown competitive with more complex distances, especially for large datasets [Ding et al., 2008].

Formally, given two time series T_1 and T_2 of length L the Euclidean distance is given by:

$$d(T_1, T_2)_{\mathcal{L}_2} = \sum_{i=1}^{L} \sqrt{(T_1(i) - T_2(i))^2}$$

The drawback of the Euclidean distance is its sensibility to the distortions mentioned previously. Moreover it is unable to deal with time series of various lengths, even one point of difference since the comparison is made point by point. Amplitude distortions can be handled with transformation of the time series, such as the z-normalization (handle difference of amplitude) or the Segmented Sum of Variations (SSV) [Lee et al., 2002]. Scale-invariant lock-step distances have also been proposed, such as the Minimal distance (Euclidean distance minus the time series' offsets) [Lee et al., 2002] or the Mahalanobis distance [Arathi and Govardhan, 2014]. The correlation coefficient has also been used to take advantage of its scale invariance [Mueen et al., 2015].

Time-warping distortions are more complex to handle. Elastic distances, presented in the next section, have been proposed to tackle this issue.

Elastic distances

Dynamic Time Warping The major elastic distance is the Dynamic Time Warping (DTW) [Berndt and Clifford, 1994]. Despite having been proposed decades ago, it has been shown competitive with more recent techniques [Wang et al., 2013]. The DTW is able to compare time series with reasonable length differences and time distortions. The DTW principle consists in finding the best possible alignment between two time series, called a warping path, with the following rules:

1. The path must starts (ends) at the respective firsts (lasts) points of the two time series (boundary condition).

2. Every points of both time series must be used, and it is not possible to let aside one point of a time series (the continuity is required).

3. It is not possible to choose an alignment that would go backward in time (monotonic condition).

4. The warping path cannot be too steep or too shallow, to avoid the matching of two subsequences too different in length (slope constraint condition).

The main issue with DTW is its scalability. Its computation complexity in $O(m^2)$ (with m the time series length) makes the DTW up to 3 orders of magnitude slower than the Euclidean distance. The high computation complexity results from the search for a good time series alignment. Propositions have been designed to speed up the DTW.

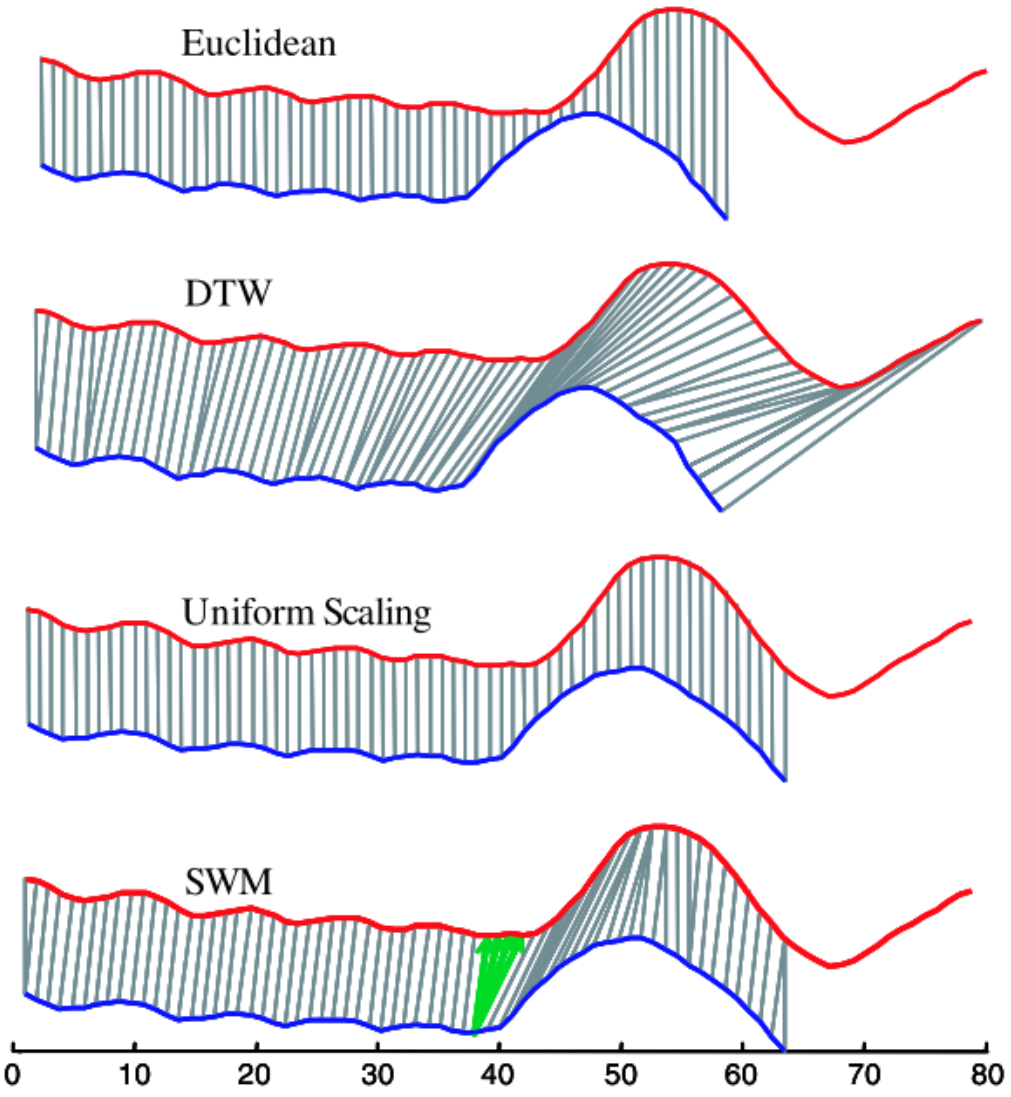


Figure 2.12: Comparison of flexible distance measures (including Dynamic Time Warping -DTW-) with the Euclidean distance (illustration from [Fu et al., 2008a])

Many distance calculations can be pruned to avoid obvious non-solutions for the warping path. For instance, the first point of one time series can be immediately discarded as a possible association with the last point of the other time series. Global constraints on the warping path have been proposed to bound the possible warping paths. Two common strategies are the Sakoe-Chiba band and the Itakura Parallelogram. The constraints (or envelop around the warping path) can also be learnt, it has even been shown as improving the accuracy [Ratanamahatana and Keogh, 2004]. Other speedup techniques involves the DTW discovery on a low resolution representation of the time series, such as the PAA (see chapter 3) [Keogh and Pazzani, 2000].

DTW purpose is mainly to handle localized time warping. Some works propose the combination of DTW with other techniques to handle more time series distortions such as the uniform scaling [Fu et al., 2008a] or the use of an ensemble of elastic distance measures [Lines and Bagnall, 2015].

Edit Distances The Edit Distances form another family of distance measures that are robust to temporal distortions. Edit distances have been developed in the bioinformatics and natural language processing fields to quantify similarity of two sequences of symbols. The idea is to evaluate the cost to transform one sequence to the other, by counting the minimum number of operations required [Moerchen, 2006]. The edition operations allowed are insertion, deletion and substitution of symbols, each operation has a specific cost associated. Since edit distances operate on symbolic sequences, the time series need to be transformed into a symbolic representation (see chapter 3). The most common edit distances for time series are the LCSS based on the Longest Common Subsequence principle, the Edit Distance on Real sequence (EDR) and the Edit Distance with Real Penalty.

2.5.2 Feature-based classification

The whole time series matching for classification may not be meaningful: it may be more relevant to match similar time series based on shared derived properties. In [Kadous and Sammut, 2005], authors highlight the difficulty to fit time series data into the classical "static attribute-value model common in machine learning" as discussed section 2.4.1, which apart from designing a custom learner, let the practitioner with the option to extract relevant features from the time series, at the price of a complex feature engineering stage. The temporal problem is transformed into a static problem with a static set of features [Fulcher and Jones, 2014]. For instance, similar time series may share the same global distribution of values well represented by its mean and standard deviation, or a global frequency content well represented by coefficients from a Fourier transform or more localized characteristics such as randomly localized subsequences well represented using

a representation based on motif discovery. A large set of features can be extracted from time series to describe and represent the information they contain. In [Fulcher and Jones, 2014], authors extract thousands of features developed in the time series analysis field.

After a feature extraction step, the *feature-based* approach relies on classical machine learning algorithms that take a typical constant feature vector in input (Random Forest, SVM, Neural Networks...) to learn a mapping f from a constant representation of the time series and to perform the prediction.

While in the *time-based* approach the focus is set on a suitable distance measures, in the *feature-based* approach the issue is to find a *representation* that gathers a suitable set of features to represent the relevant information contained in the time series while handling the distortions.

2.6 Conclusions

In this chapter, we presented an overview of the time series mining field and we discussed the reasons why time series is a complex datatype that present issues to apply common machine learning techniques on them. Time series have specific properties and they are affected by distortions in amplitude and time.

We presented two main approaches to overcome these issues, appropriate distance measures and time series representations. We also presented the typical ways to perform time series classification: we called the first approach time-based classification since the time series are compared as is using distance measures to overcome the distortions. We named the second approach feature-based classification: the principle is to discover and compute relevant representations of the time series information to train common classifiers.

In this work we focus on the discovery of meaningful time series representation to perform feature-based classification. The next chapter is dedicated to a review of the time series representations.

Chapter 3

Time Series Representations

As we have seen in the previous chapter, when it comes to learn from time series, we face four main issues. The first one is that time series is not a suitable "static attribute-value model" [Kadous and Sammut, 2005] that could enable to make a direct use of classical machine learning algorithms. The second issue is the distortions that affect time series. The third issue is the high dimensionality of the time series data that induces computational issues. The last issue is the heterogeneity of the information that can be stored in a time series. Learning a concept from a raw time series datasets is generally ineffective or even hopeless without a prior feature extraction step, designed with domain experts or with feature learning.

Two concepts allow us to handle these issues: the *distance measures* and the time series *representations*. Distance measures are mentioned in the previous chapter, here we focus on the representations. After a discussion on the time series *representation* principle, we propose an overview of the different types of time series *representations* with descriptions of some of their instances.

3.1 Concept of time series representation

The concept of time series representation is large and depends on the task to be solved and the approach used to do so. We define a time series representation Ψ as an operation that transforms a *raw* time series T_n into *another* time series $\hat{T}_n$ or a *scalar* $x_{\Psi(T_n)}$ such that $\hat{T}_n$ or $x_{\Psi(T_n)}$ summarizes a given feature in T_n.

$$\Psi(T_n) = \begin{cases} \hat{T}_n \\ or \\ x_{\Psi(T_n)} \end{cases}$$

It may seems counter-intuitive to collect precise values of measurements to replace

them by an approximate representation. However, with time series we are generally not interested in the exact measure of each data point. The relevant information of a time series often relies on trends, shapes, motifs and patterns [Agrawal et al., 1993a] that may be better captured and described in an appropriate high-level representation that would additionally remove implicitly the noise [Ratanamahatana et al., 2010].

The objectives pursued by a time series representation is a combination of the following ones:

- Reduction of the dimensionality of the *raw* time series, to speedup the computations or reduce the computational needs. Many time series representations have been proposed in time series *querying* to obtain a *transformed* time series $\Psi(T_n)$ of low dimensionality for a fast retrieval process.
- Reshape the time series to get it in a desired format, for instance a feature vector compatible with common machine learning algorithms. This feature vector should describe suitably the information contained in the time series.
- Summarization of the information contained in the time series.
- Accuracy of the representation to the raw time series, in particular for the *retrieval* task.
- Highlighting discriminative or characteristic information of the time series if the task is to perform classification, clustering or anomaly detection.
- Noise removal.
- Handling of time series distortions.

Many time series representations take origin or are related with techniques developed in the time series analysis and signal processing fields.

Several taxonomies exist to classify the time series representations, for instance the one proposed by [Esling and Agon, 2012], that groups time series representations into data adaptive and non-data adaptive representations.

We propose to sort the representations into 3 groups based on the type of information summarized and the format of the representation.

Time-based representations This group gathers representations that produce a time series from the whole raw time series, such as $\Psi(T_n) = \hat{T}_n$. The time-based representations are typically of lower dimensionality than T to speed up the computations.

Feature-based representations This group gathers representations that produce a scalar (or a fixed-length set of scalars) from a *raw* time series, such that $\Psi(T_n) = x_{\Psi(T_n)}$.

It is for instance the case of representations of time series by the description of the global distribution of the values, the global trends or the global frequency content.

Motif-based representations This group gathers representations that produce a time series from the whole *raw* time series, such as $\Psi(T_n) = \hat{T}_n$ but unlike time-based representations, the resulting time series is a subsequence, extracted from the raw time series based on desired properties, for instance recurring, abnormal, discriminative subsequences or sequences of subsequences.

Feature-based representations and time-based representations are often time-series-focused: the representation is computed independently by time series. Motif-based representations are more often dataset focused: motifs are discovered at dataset scale.

In the following sections, we illustrate the concept of representation with instances from each group. The reader must remember that the literature on time series representations is very large and this review doesn't intend to be exhaustive.

3.2 Time-based representations

Time-based representations represent time series by preserving the temporality or at least the sequentiality of the data, typically by computing an approximation of some feature at local scale. Many *time-based* representations exist, they are mainly used to perform fast time series retrieval, querying or indexing. They can also be used for whole time series classification (*time-based* classification, see section 2.5) to accelerate the classification and to be invariant to some distortions.

We can group the *time-based* representations according to the type of transformation applied to the data [Bettaiah and Ranganath, 2014].

- *Piecewise representations*: the time series is segmented then a local feature is computed for each segment (for eg. the mean, average rate of variations or a regression coefficient).

- *Symbolic representations*: the time series is segmented and then discretized. A dictionary of symbols is either learned or applied in order to convert the time series into a series of symbols.

- *Transform-based representation*: the time series is converted from the time domain into another domain (for eg. the time-frequency domain thanks to a wavelet transform).

We use this taxonomy in the next paragraph to introduce some *time-based* representations.

3.2.1 Piecewise Representations

Piecewise Representations (*PR*) aim to represent time series by reducing their dimensionality thanks to the segmentation of the time series followed by a representation of each segment by a value. The time series representations falling in this category can be sorted according to two axis:

- The segmentation process: there are *adaptive* and *non-adaptive* segmentation techniques.
- The representation of each segment: which feature or set of features is extracted to represent each segment. The number of possibilities for this axis is large. It is somehow shared with the *feature-based* representations, which operates at the global scale of the time series (see section 3.3).

We present here several illustrative *Piecewise Representations*, each of them with a specific pair of segmentation technique and segment's summarization technique. Representations are grouped according to their segmentation technique (adaptive or not). We begin with the simpler time series dimensionality reduction technique, a basic sub-sampling, to illustrate the challenges.

Sub-Sampling

The simpler *PR* is probably the basic sub-sampling of the time series. One value is conserved every h points of the time series. The sub-sampling method summarizes every segment of h points of the time series by the value of one single arbitrary point of the segment. The only constraint on h is to respect the *Nyquist* frequency theorem that states that no information is preserved over the *Nyquist* frequency after the sub-sampling. The *Nyquist* frequency is given by [Åström, 1969]:

$$f_c = \frac{1}{2h}$$

Sub-sampling is mainly used in signal processing when a continuous signal has to be converted into a discrete one. A signal can also be re-sampled when its acquisition frequency is higher than the one required by the *Nyquist* frequency theorem to represent the phenomenon by the time series.

A sub-sampled time series allows faster computations and less storage requirements. The method is easy to implement: the only parameter is h. However, the main drawback of sub-sampling is the distortion of the time series shape if the sampling rate is too low to depict the underlying phenomena [Fu, 2011]. When performing time series mining we usually ignore the information to be discovered, thus the *Nyquist* frequency is hard to guess to further determine the parameter h. Moreover, the discarded points are completely

ignored during the representation process, important patterns may be missed. To represent complex shapes, the required number of points with this approach may be high, with no or few improvements for computation and storage.

Non-adaptive segmentation

The Piecewise Aggregate Approximation (PAA) is maybe the most important *time-based* representation based on non-adaptive segmentation. It can be seen as an enhancement of the sub-sampling method. Instead of representing each segment of length h by its first value, the PAA summarizes the segment by its mean, expecting a higher fidelity in the representation of each segment [Keogh et al., 2001, Yi and Faloutsos, 2000].

A time series T_n of length L is segmented into N fixed-length segments. For each segment, the mean value is computed to represent the segment. T_n is then reduced to an approximate time series $\hat{T}_n = [\hat{T}_n(1), \ldots, \hat{T}_n(i), \ldots, \hat{T}_n(N)]$ of length N. PAA has one single parameter that is the length N of $\hat{T}_n$.

Each point of $\hat{T}_n$ is computed as:

$$\hat{T}_n(i) = \frac{N}{L} \sum_{j=\frac{L}{N}(i-1)+1}^{\frac{L}{N}i} \hat{T}_n(j)$$

The PAA has several drawbacks. The first one is the hypothesis made to represent the information. PAA assumes the information is suitably represented by the mean over one segment, which is not often the case. Variations of the PAA have been proposed to address this point such as [Quoc et al., 2008] that adds slope information to the mean for each segment, [Guo et al., 2010] adds variance to the mean. [Lee et al., 2002] proposes the *Segment Sum of Variation* representation (SSV) that computes the variation between two adjacent points, each segment is represented by the sum of variations within the segment. One advantage of the SSV is its invariance to amplitude and offset distortion. If two time series present different means, their shape can be compared directly with the SSV representation without any prior preprocessing (such as scaling or normalization of the data) or without any particular distance measure (such as the minimum distance, the Euclidean distance between two time series minus the mean distance between the two series). Two time series represented with SSV and with a similar shape but a distinct offset can be compared directly with Euclidean distance [Lee et al., 2002].

Another drawback is the resolution of the information: the segment size is fixed and one value summarizes the whole segment whatever the resolution of the shapes and trends in the time series. Th *Multi-Resolution PAA* (MPAA) has been proposed to compute several PAA representations of the same time series at various resolutions. At the first level, a typical PAA is computed on the *raw* time series. The process is recursively applied on the

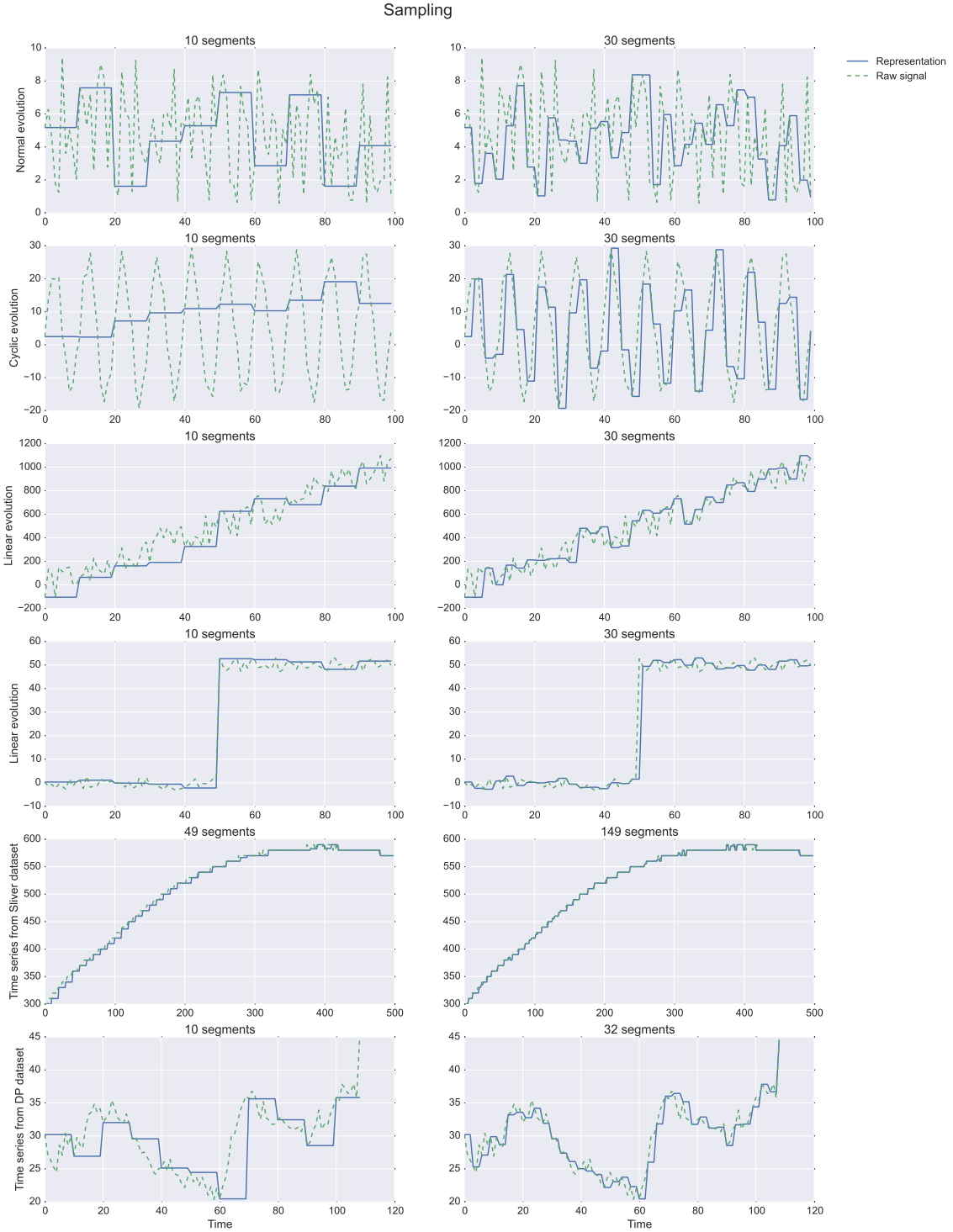


Figure 3.1: Several time series shapes and their associated representations using *sub-sampling* representation

result of the previous *PAA* resulting in a multi-resolution representation of the time series. *MPAA* representation has been proposed initially to perform a fast time series clustering with increasing finer representations [Lin et al., 2005].

The fixed-length segment issue has been addressed with another approaches called Adaptive Piecewise Constant Approximation.

Adaptive segmentation

The issue with representation*s* based on non-adaptive segmentation is that segments with constant values are represented with the same resolution than segments with many fluctuations. The *Adaptive Piecewise Constant Approximation* (*APCA*) representation has been proposed to improve the segmentation stage of the representation [Geurts, 2001, Chakrabarti et al., 2002]. Segment length is not fixed initially but instead each segment length is fitted to the shape of the series. Iteratively, the segment with the highest variance is identified over the time series and split into two smaller segments such as the division maximizes the variance reduction. Like the *PAA* each segment is represented by its mean. *APCA* requires a single parameter: the number of desired segments. The length of each segment is remembered by recording their right endpoint points.

As for non-adaptive representations, it is possible and desirable to represent the segment by other features than only the means. The number of possible features is large. We focus here on variations of the *Piecewise Linear Approximation.*

The *Piecewise Linear Approximation (PLA)* representation aims to represent a time series by successive straight lines to model the shape of the time series. Two main categories emerge for the definition of the straight lines: by interpolation [Keogh and Smyth, 1997] or by regression [Shatkay and Zdonik, 1996]. Interpolation simply links by a line the start and end points of each segment. Regression looks for the best fitting lines for every point of every segment (see figure 3.5). The segments are typically represented by the coefficients of the lines. The regression is more accurate but much more computationally demanding than the interpolation.

The main challenge for these techniques is to discover the best start and end points of the segments. Several variations exist:

- The *Piecewise Linear Representation* (*PLR*) is a bottom-up algorithm. It begins by creating a fine approximation of the time series with $L/2$ segments to approximate a time series of length L. The *PLR* merges iteratively pairs of segments that minimize the representation error. The process stops when the number of segments required is reached [Keogh and Smyth, 1997].

- The *Perceptually Important Point* (*PIP*) representation is a top-down algorithm. It identifies iteratively the point that least fits in its segment. This point is used to

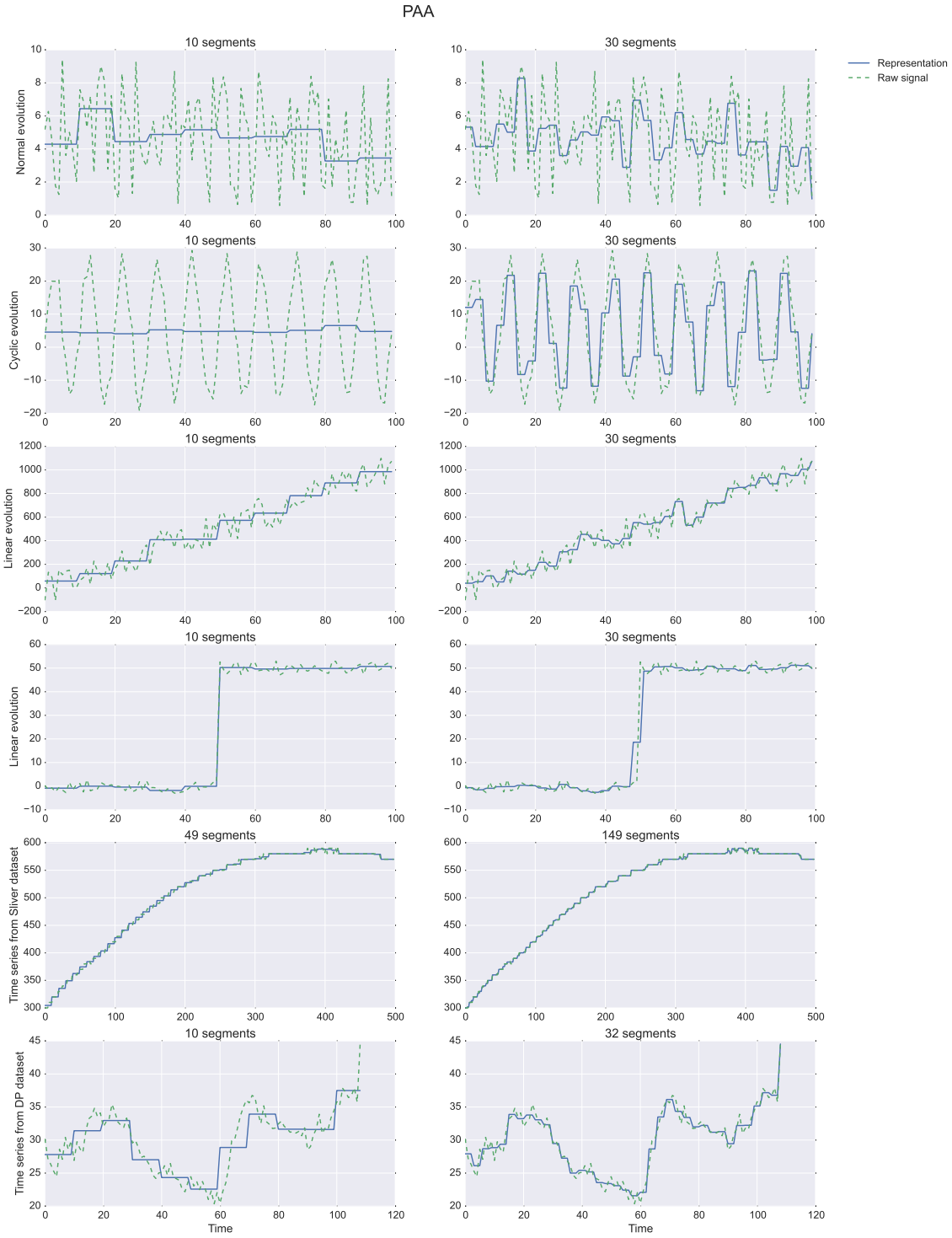


Figure 3.2: Time series and their associated representations using a *PAA* representation

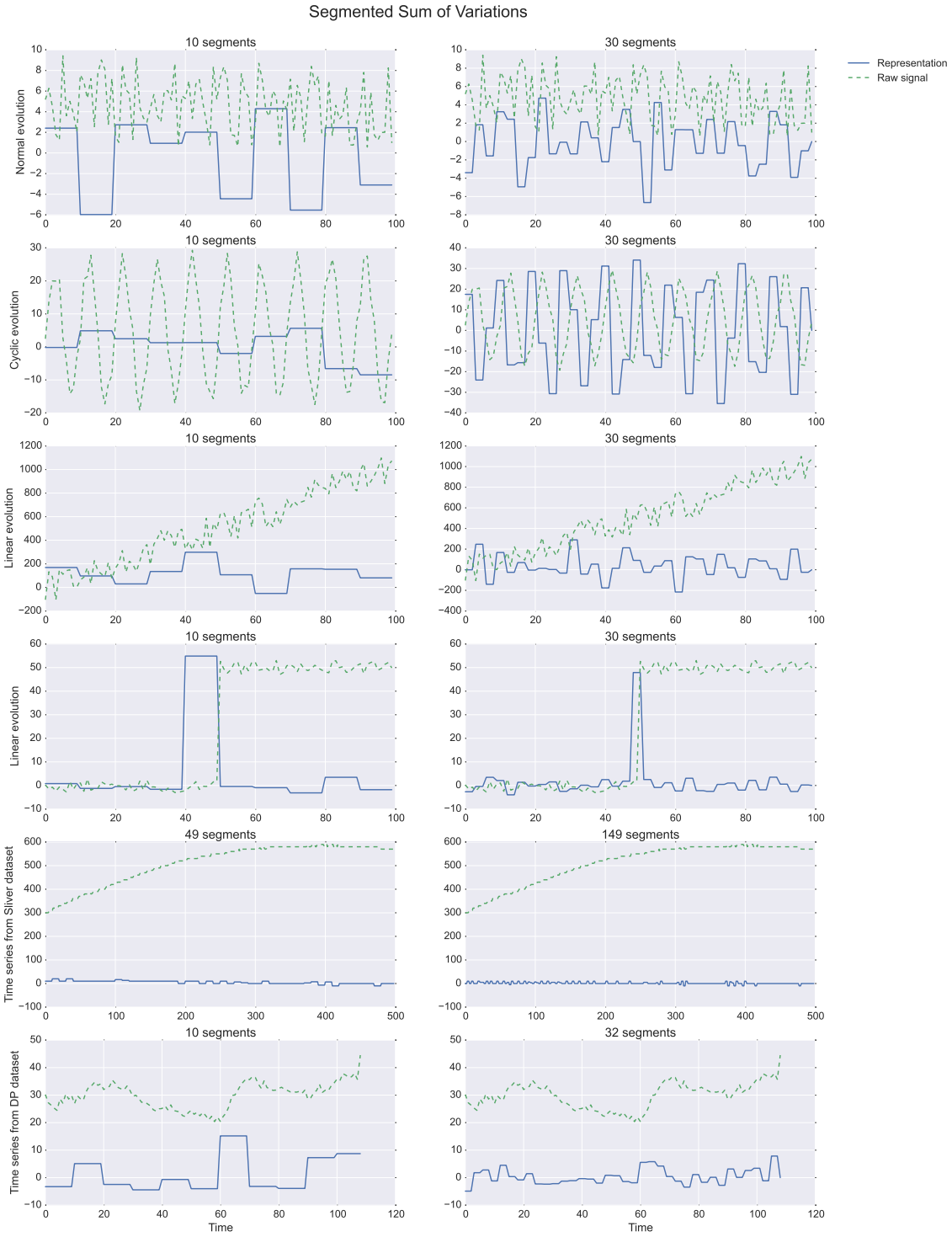


Figure 3.3: Time series and their associated representations using a *SSV* representation

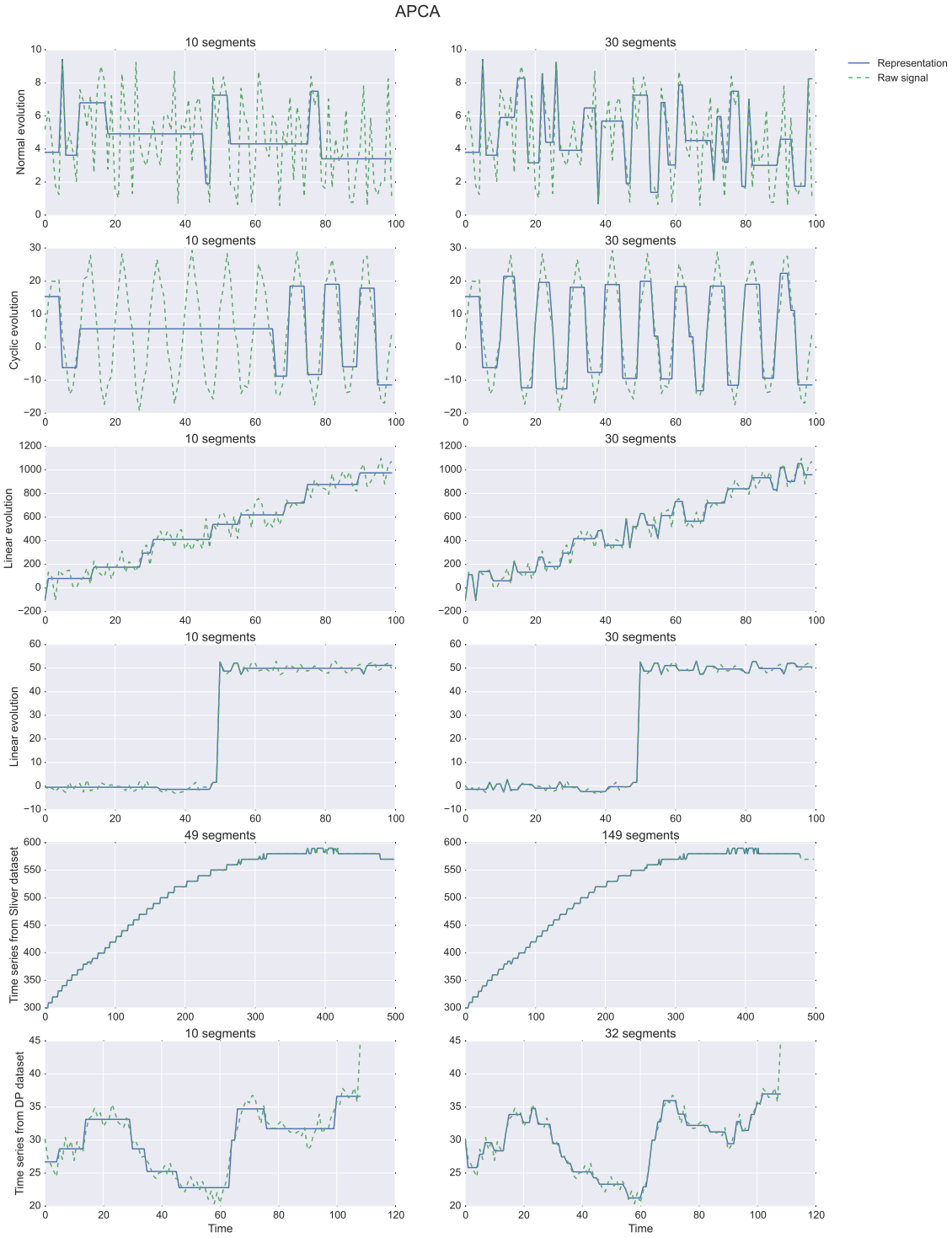


Figure 3.4: Time series and their associated representations using a*n APCA* representation

split the corresponding segment in two [Fu et al., 2008b].

- The *Important Points* representation identifies the most important local minima and maxima to segment a time series [Pratt and Fink, 2002]. This way, minor fluctuations are discarded and the shape of the time series is conserved. For the *Important Points* representation, an important minimum (respectively maximum) is the minimum (maximum) $T_n(i_{min})$ among $[T_n(i)...,T_n(j)]$ such as $\frac{T_n(i)}{T_n(i_{min})} \geq R$ and $\frac{T_n(j)}{T_n(i_{min})} \geq R$ where R is a parameter to control the compression ratio required.
- The Landmark model also intents to locate important points [Perng et al., 2000]. With this representation, an important point is a point such as its derivative is 0 (ie. local minimum or maximum). A first-order landmark is a point whose first derivative is 0. A second-order landmark is a point whose second derivative is 0 (ie. inflection point), and so on. Landmark points are then linked with segments. With more derivative orders the representation becomes more accurate (except for volatile time series where the highest orders are useless). To avoid landmark points to represent noise, a criterion called the Minimal Distance / Percentage Principle (MDPP) is used. Basically, to decide if a landmark point $T_n(i_{min})$ is relevant, its closeness to the following landmark point is evaluated. If it is too close, the point is removed. This criterion is setup through a parameter to fix the value below which the landmark should be removed.
- Other similar approaches exist such as [Bao and Yang, 2008] to identify "turning points" in the time series or [Man and Wong, 2001] to identify important peaks and troughs.

Many other *Piecewise Representations* exist, we refer the reader to dedicated surveys such as [Ding et al., 2008, Fu et al., 2008b, Esling and Agon, 2012, Bettaiah and Ranganath, 2014].

3.2.2 Symbolic representations

Symbolic representations aim to reduce further the dimensionality of the time series by discretization of the time series into a sequence of symbols. Symbolic representations can be seen as an overlay of the *Piecewise Representations* since the discretization into symbols is usually performed after the segmentation of the time series and the representation of each segment. To transform numerical values into symbols, a quantization is performed providing both an additional dimensionality reduction of the time series (less computational needs) at the price of an additional loss of precision on the shape and trend information over the numerical representations like the *Piecewise Representations* [Bettaiah and Ranganath, 2014]. One major advantage of symbolic representations is the possibility to use many

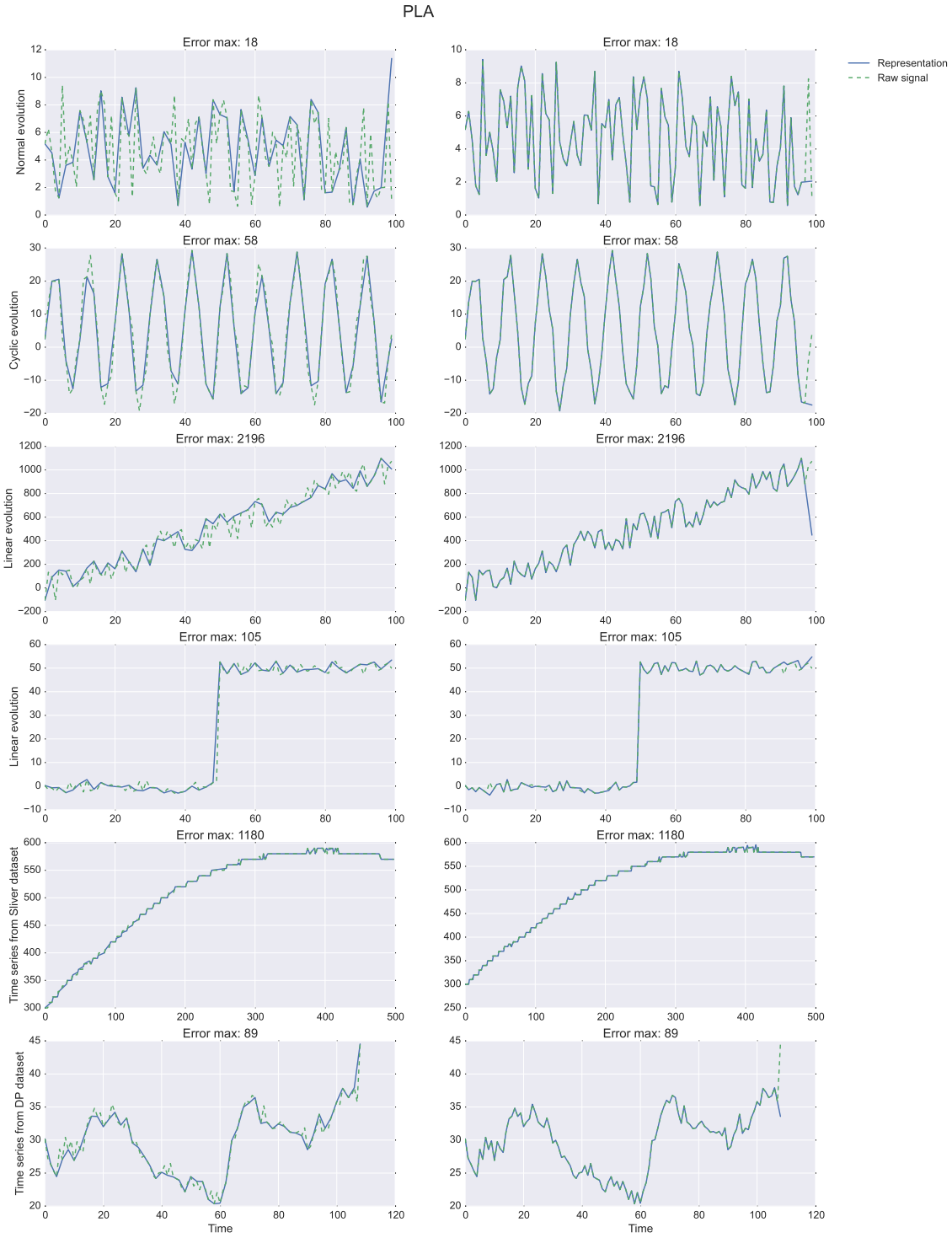


Figure 3.5: Time series and their associated representations using a *PLA* representation

existing algorithms to manipulate strings, in particular from text retrieval or bioinformatics fields [Lin et al., 2012].

Many symbolic representations have been proposed, we illustrate some of them starting by *SAX*, which is an emblematic symbolic representation of time series.

The *Symbolic Aggregate approXimation* (*SAX)* representation relies on the *PAA* to produce the symbolic sequence [Lin et al., 2002]. In order to transform a time series, *SAX* proceeds in three steps.

1. The time series is normalized to zero mean and unit standard deviation.
2. The time series is segmented using a *PAA* representation.
3. The *PAA* representation is finally quantized: a discretization step maps each real value into a symbol. Over the whole series, each symbol is given the same probability of appearance. This step is based on the hypothesis of a normal distribution of the values over the y-axis.

SAX converts a time series into a sequence of symbols of length N. The alphabet of symbols has a length $a > 2$, $a \in [1, \ldots, N]$. *SAX* parameters N and a are defined by the user. Each symbol is given the same probability of appearance along the time series (breakpoints are based on the assumption of a Gaussian distribution of the values).

SAX representation has been initially proposed to support an efficient algorithm for motif discovery [Lin et al., 2002]. *SAX* comes with a distance measure based on the Euclidean distance using a lookup table with distances between symbols, an indexation mechanism and an algorithm to efficiently localize recurrent patterns.

A drawback of *SAX* representation is the critical choice of the parameters (length of the windows, size of the alphabet). With parameter's values too small *SAX* may represent noise, with parameter's values too large the representation may miss the shape or trends of the time series.

Many variations of *SAX* have been proposed, which propose enhancements on the segmentation process, the features extracted from each segment and the discretization of these features to get symbols.

[Hugueney, 2006] changes the segmentation of the time series. Instead of a segmentation based on *PAA* with fixed-length segments, the segment length is adaptive similarly to an *APCA*. The discretization process is similar to the one of *SAX*.

[Lkhagva et al., 2006] proposes to modify the features extracted from each segment and add the minimal and the maximal values to the mean to form a symbol. The *Shape Description Alphabet* (*SDA*) computes the first derivative between pairs of adjacent points [André-Jonsson and Badal, 1997]. The derivative values are mapped to symbols thanks to the *Shape Description Alphabet*, as defined table 3.1, to describe the shape of the time

Symbol	Description	Low Value (for the derivative)	High value
a	Highly increasing	5	$+\infty$
u	Slightly increasing	2	4.99
s	Stable	-1.99	1.99
d	Slightly decreasing	-4.99	-2
e	Highly decreasing	$-\infty$	-5

Table 3.1: Definition of the Shape Description Alphabet (SDA)

series. *SDA* is associated with a signature generation technique in order to perform time series retrieval. With a similar idea to catch time series shapes, the *gradient alphabet* represents time series with symbols such as upward, flat and downward [Qu et al., 1998].

[Mörchen and Ultsch, 2005] proposes Persist, a representation with a discretization process that adapts to the data instead of assuming a Gaussian distribution of the values. With *Multiple Abstraction Level Mining* (*MALM*) representation, the regression coefficient, mean square error and other statistics are extracted from each segment of the time series at several resolutions [Li et al., 1998]. A clustering is performed on these features to get the dictionary of symbols and represent the time series in a sequence of symbols. With MALM a query can be performed at features level, symbol level and sequence level. The *Piecewise Vector Quantized Approximation* (*PVQA*) represents segments by a symbol from a dictionary, which is built from the data [Wang and Megalooikonomou, 2008]. Segments are clustered and each cluster centroid becomes a symbol. The number of symbols required leads the number of clusters for the clustering. The dictionary is learned from subsequences: each symbol corresponds to a prototype of shape. *Multi-resolution Vector Quantization Approximation* (*MVQA*) has been proposed to get a multiple resolution *PVQA* representation.

3.2.3 Transform-based representations

We mentioned in the introduction (chapter 1) the close relationship between some techniques from the time series mining and signal processing fields. Signal processing offers many tools to transform time series into novel time series where the information has been filtered, decomposed or the noise has been removed.

A first example is filters. A filter removes undesired component from the time series, for instance specific frequencies (low-pass, high-pass, band-pass). One application of filters is the removal of background noise. The result is another time series without the filtered component.

Among other signal processing techniques in the time domain, we can cite the time frequency analysis with short-time Fourier transformation and wavelets, in particular the Discrete Wavelet Transform (DWT). The idea of these techniques is to estimate the in-

stantaneous frequency of a time series. The short-time Fourier transformation is based on successive Fourier transforms applied on overlapping windows rolling on the time series, with a limited frequency resolution due to the window size. The wavelet transform is based on a mother wavelet with specific properties useful for signal processing [Nason and von Sachs, 1999, Li et al., 2002]. To perform the wavelet transform, the mother wavelet is convolved over the time series. The process is performed iteratively using dilated versions of the mother wavelet to analyze the signal at different scales. The wavelet transform allows to have a multi-resolution decomposition of the time series with respect to its frequency content (see figure 3.6). The relevance of the DWT has been demonstrated as a time series representation [Struzik and Siebes, 1999].

Another notable decomposition techniques is the Empirical Mode Decomposition (EMD) [Huang et al., 1998]. The EMD decomposes a time series into several Intrinsic Mode Functions (IMFs), also in the time domain, from IMFs with mostly high frequency components to IMFs with mostly low frequency components.

These techniques are useful to preprocess the raw time series, remove the noise, and gain access to the relevant shapes of the time series. These techniques are also useful to reduce the dimensionality of the time series: the global trend can be extracted from the raw data.

Beyond these approximate definitions, we invite the reader to refer to dedicated documentation on this topic, for instance [Feng et al., 2013] proposes a survey on time-frequency analysis of time series with feature extraction for machinery fault diagnosis.

3.3 Feature-based representations

Time-based representations preserve the temporality of the time series while reducing their dimensionality. On the contrary, feature-based representations produce static representations (ie. without temporality) by computing features to measure specific properties of the time series. This type of time series representation is particularly well suited to make use of classical machine learning algorithms, since the resulting representation is a classical feature vector.

Many propositions have been made to extract global features from the time series. [Wang et al., 2005] proposes to perform time series clustering based on the extraction of such features (trend, seasonality, serial correlation, non-linearity, skewness, kurtosis, self-similarity, and chaos). A recent work [Fulcher and Jones, 2014] proposes to gather thousands time series analysis features used in various scientific fields. The extracted features summarize the structural information of the time series, for instance the global distribution of the data, the autocorrelation, the periodicity, the stationarity, information theory measures or parameters of fitted time series models for forecasting.

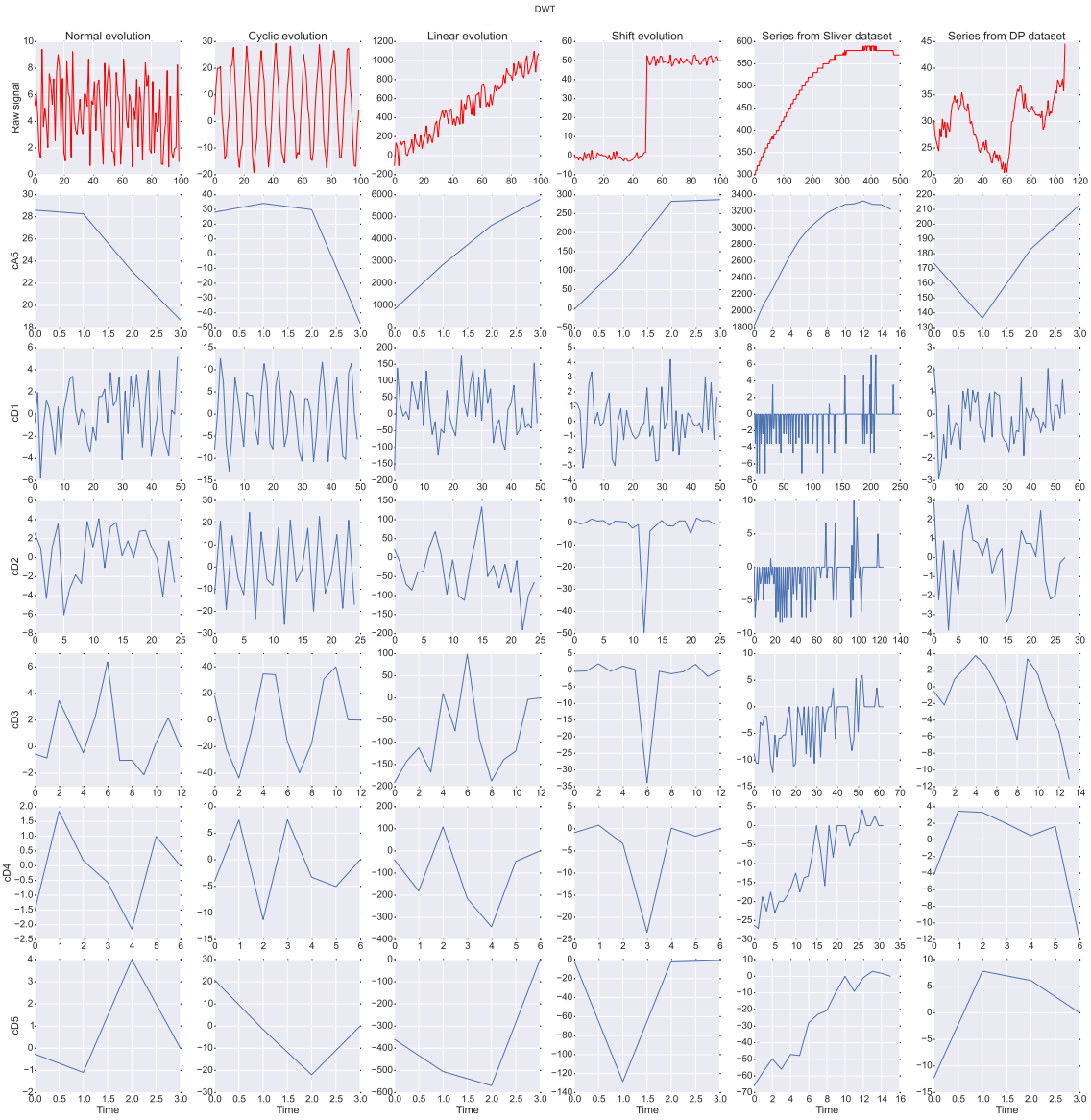


Figure 3.6: Time series and their associated representations using the *DWT*

The number of possible features is large. In this section, we review the overall principle and illustrate the principle with some examples. For an exhaustive list of features, we advise the reader to refer to [Fulcher and Jones, 2014].

3.3.1 Overall principle

Many techniques have been developed in the time series analysis field to analyze the structural behavior of time series or to fit forecasting models. A time series representation $\Psi(T_n)$ is extracted from a time series T_n based on the measurement on a time series of a specific property, which returns a scalar feature $x_{\Psi(T_n)}$ such that $x_{\Psi(T_n)} = \Psi(T_n)$.

The issue is to define which set of features is relevant to represent the time series of $\mathcal{D}$. One solution to perform time series classification is to extract a very large set of distinct features and then select the relevant ones for the classification task using feature selection [Fulcher and Jones, 2014].

In the next section, we illustrate briefly some features.

3.3.2 Brief overview of features from time series analysis

Distribution of the data & Basic statistics Global statistics of the time series data distribution are obvious features, such as the first four statistical moments (mean, standard deviation, skewness and kurtosis) on the raw time series or its first-order derivative [Nanopoulos et al., 2001].

Many other features on the data distribution of the time series values can be computed, for instance:

- Features from the position of the distribution: min, max, median, mode, etc.
- Features from the dispersion of the distribution: range, spread, coefficient of variation, quantiles and inter-quantile range, outliers, etc.
- Features from the shape of the distribution: variance and higher moments than the skewness and kurtosis previously mentioned.
- Features from fitted distributions: by extracting parameters of fitted parametric or non-parametric distributions, quantifying the goodness of fit of the distribution to actual time series values distribution, etc.
- Other features: length of the time series, number of zero-crossings of the time series, proportion of local maxima and minima, "burstiness" (bursty behavior), etc.
  [Deng et al., 2013] proposes the extraction of this kind of features (mean, standard deviation, slope) with the addition of features computed by time series interval.

Correlation & Periodicity Many features related to the autocorrelation and periodicity can be derived from the time series, for instance the autocorrelation with various lags (such that its change over time) or from Fourier transform coefficients (such that the periodogram).

Several approaches make use of frequency or time-frequency transformations to extract features, for instance: [Faloutsos et al., 1994] proposes to extract the 2 firsts coefficients of a Discrete Fourier Transform (DFT), [Morchen, 2003] proposes to use coefficient of a DFT and a wavelet decomposition but instead of keeping the first coefficients, they use as features the coefficients that maximize the energy preservation, [Caiado et al., 2006] proposes to use the normalized periodogram to perform time series classification.

Domain specific features The features can be setup for a particular application. For instance [Kadous and Sammut, 2005] associates global features extracted from the time series (min, max, mean, etc.) with the extraction of events, specifically designed by application (for example instance increasing, decreasing, flat, local min/max). Each event detected in a time series returns specific information (start position, duration, average gradient, etc.). Relevant events are discovered and used to create features to perform the classification.

The list of features from the time series analysis field is much larger: for instance, information theory based measures (to estimate the time series entropy), stationarity measures (test of the stationarity or non-stationarity behavior of the time series, occurrence of steps or local extrema in the time series, detection of variations in the statistics of local distribution of the time series, etc.), parameters of forecasting models (several features can be derived from such fitted models, such as AR or ARMA, to describe the behavior of a time series).

We refer the reader to [Fulcher and Jones, 2014] and the time series analysis literature for more details.

3.4 Motif-based representations

The motif discovery task has been presented in the introductory section on time series mining (section 2.2). We have seen that several objectives can be pursued with motif discovery. We may be interested by the discovery of frequent motifs or, on the contrary, we may be interested by the discovery of infrequent or anomalous motifs. We may be also interested by task-specific motifs, such as discriminant motifs supporting a classification task. Motifs may be discovered for knowledge discovery: the relevance of a motif to

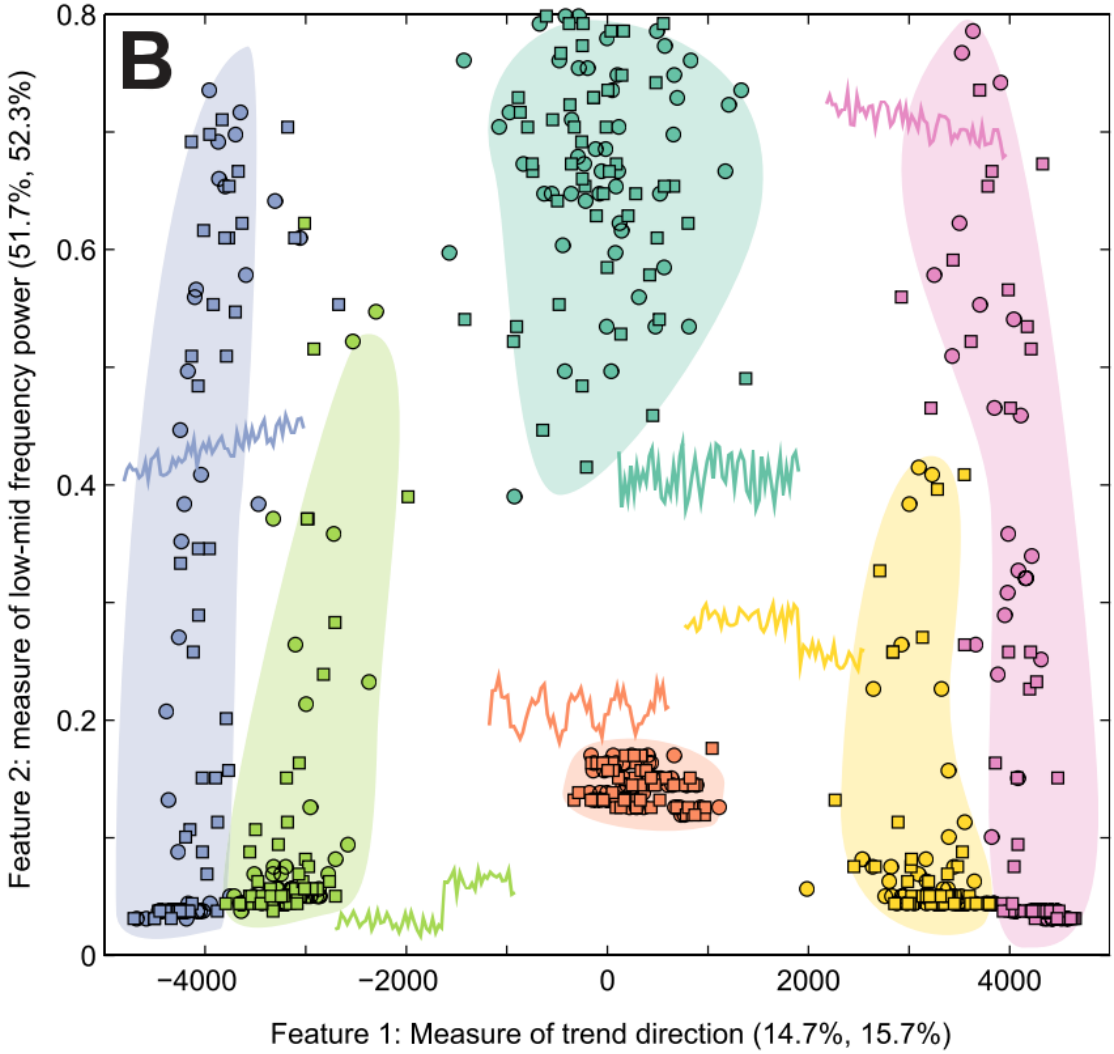


Figure 3.7: Examples of feature-based representations: the time series from a datasets a represented in a 2-dimensional feature space whose features discriminate effectively the different labels (illustration from [Fulcher and Jones, 2014])

summarize and visualize time series datasets has been shown. They may also be discovered to support another machine learning task such as clustering or classification [Lin et al., 2002][Mueen, 2013]. In fact, motif discovery is particularly relevant as a subroutine of other machine learning tasks: many contributions in the literature assume that time series are correctly segmented. It is assumed that the beginning and the ending points of interesting patterns can be correctly identified, which is unjustified [Hu et al., 2013]. The performance of time-based classification approaches (section 2.5.1) critically depends on this assumption.

The motif discovery task usually faces several significant challenges to design efficient discovery algorithms: in addition to the definition of the purpose of the motif and the format of the output, the distortions (section 2.5.1) and an important computational complexity make the task difficult.

In this section, we present an overview of the discovery of three main types of motifs: recurrent motif, infrequent or surprising motifs and discriminant motifs. After a definition of each type of motif, we outline the main approaches for their discovery.

3.4.1 Recurrent motif

A recurrent motif can be defined as a set of repeating similar subsequences in a time series or a set of time series. The definition is vague since many approaches have been proposed to discover recurrent motifs, most of them with a specific definition for the assessment of the recurrence of a motif. Beside the definition of a recurrent motif, the research to design effective recurrent motif algorithms has been driven by the objective to reduce the high computational complexity of the discovery. In fact, a naive approach based on the exhaustive enumeration of all the subsequences of a time series dataset is prohibitive.

A group of approaches defines a repeated subsequence as a subsequence with many matches in a time series datasets. A match is defined as a distance between two subsequences lower than an arbitrary threshold. Using this principle and to decrease the high computation complexity of the discovery, [Lin et al., 2002] proposes an approach based on a SAX representation of the time series and several speed-up techniques. These techniques exploit the symmetric property of the Euclidean distance to half the number of distance computations and the use of the triangular inequality to discard a subsequence that would not match another one (technique inspired by the Approximation Distance Map algorithm).

[Chiu et al., 2003] proposes another motif discovery algorithm also based on a SAX representation. The matching technique is based on Random Projection, inspired by *Projection*, a pattern discovery algorithm used in bio-sequences (symbolic series). Symbolized subsequences sharing the same length are hashed iteratively into buckets using two random index of the symbolized subsequences as a mask. When a collision occurs, a counter is incremented in a collision matrix. After several iterations (drawing of different indices),

the collision matrix is examined to look for subsequences with statistically high number of collisions. These subsequences are probable recurrent motifs and their effective recurrence is controlled in the raw time series.

[Xi et al., 2007] extends this principle by adding rotation invariance for the motifs to be discovered by adding shift SAX representations of each subsequence to the collision procedure.

[Yankov et al., 2007] proposes another motif discovery algorithm based on [Chiu et al., 2003] *(*SAX representation with Random Projection*)* on which they add the uniform scaling invariance.

In [Tanaka et al., 2005], the discovery of recurrent motif is extended to multivariate time series and varying length motif. A multivariate time series is transformed into a one-dimensional time series using a Principal Component Analysis (PCA). Then, each subsequence of the one-dimensional time series is discretized using SAX. Each symbolized subsequence is itself transformed into one symbol called *Behavioral Symbols*: at the end, the original time series is transformed into a sequence of *Behavioral Symbols.* Using the Minimum Description Length principle, each subsequence is assessed to discover the smallest motif that best summarizes the recurring behavior of the time series.

In [Minnen et al., 2007], the recurrent motif discovery is formalized as a problem of localization of high density areas in the space of all the time series subsequences. For each subsequence, the k-nearest subsequences are identified and the density around each subsequence is computed to get the most recurrent motifs. In [Castro and Azevedo, 2010], the iSAX multi-resolution time series representation is used to propose a multi-resolution motif discovery algorithm. In [Castro and Azevedo, 2012], the assessment of the recurrence of the motifs is based on statistical hypothesis test to compare the actual frequency to the expected frequency of a motif calculated using Markov Chain models. In [Mueen et al., 2009], an algorithm for the exact discovery of recurrent motifs is proposed. It makes use of reference points to compute lower-bounded distances with each subsequence: non-promising motif are pruned. Their algorithm is shown until 3 orders of magnitude faster than the naive discovery algorithm (exhaustive computation between every subsequences), but it keeps the same complexity of $O(|T|^2)$ in the worst cases. The Euclidean distance is used to compare motifs based on the discovery of [Ding et al., 2008], which states that Euclidean distance is competitive or superior to more complex distances with invariances on large datasets.

3.4.2 Surprising or anomalous motif

Surprising motif has been defined as a motif whose frequency of occurrence significantly differs from that expected by chance [Keogh et al., 2002].

[Keogh et al., 2007] introduces the time series *discords,* defined as subsequences that

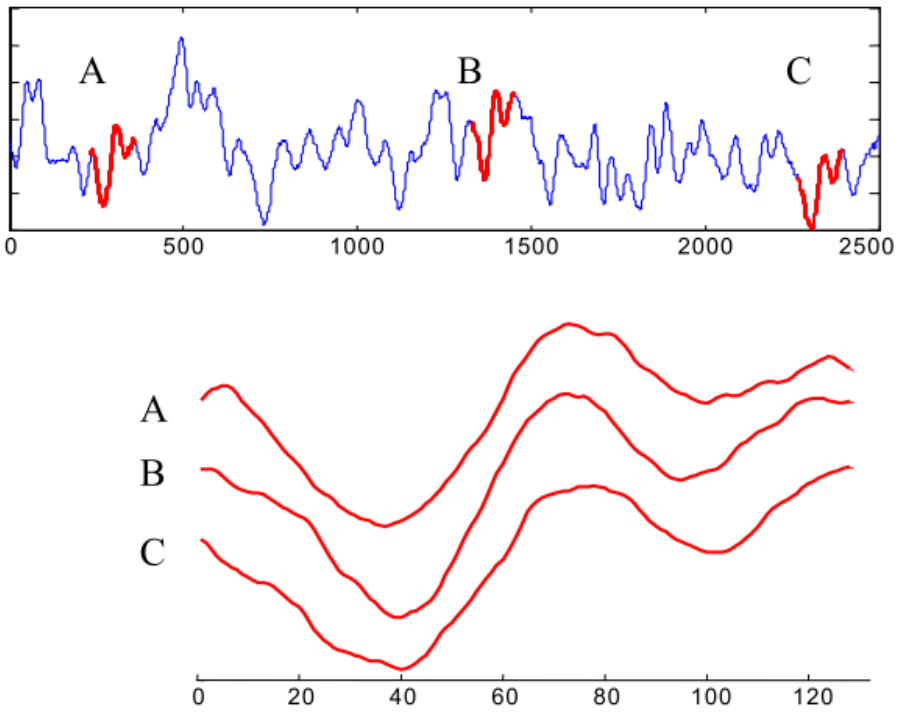


Figure 3.8: Recurrent motif: 3 similar subsequences can be identified in the time series (illustration from [Lin et al., 2002])

are maximally different to all the rest of the subsequences based on the distances of each subsequence to its nearest non-trivial matches. They propose HOT SAX to discover time series discords: subsequences with given length are extracted using a rolling window and discretized using SAX representation. The number of occurrence of each SAX word is conserved in a matrix and a tree with index information to retrieve them. Then, the SAX words with the minimal number of occurrences are evaluated to check their true distance to their nearest neighbors. Since these words have few occurrences, they are good candidates for being time series discords, which allows the other discord candidates to be quickly discarded during their evaluation.

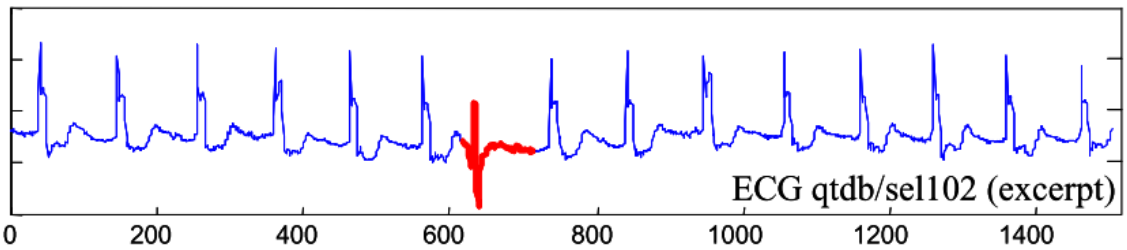


Figure 3.9: Abnormal, surprising motif (illustration from [Keogh et al., 2005])

3.4.3 Discriminant motif

Discriminant motifs can be roughly defined as motifs whose frequency of occurrence in the time series depends on the time series class label. A discriminant motif can be either characteristic of a class but not sufficient alone to discriminate instances from one particular class or strongly discriminant if the motif is able to discriminate instances from a particular class. Discriminant motifs are especially useful to perform time series classification.

Our proposition (part II) relies on this concept. The current leading approach is the time series shapelet introduced in [Ye and Keogh, 2009]. The shapelet concept is about the discovery of subsequences with high discriminatory power in a time series dataset. In the initial work, the shapelet were framed in a decision tree but since, it has been extended in many ways to make it more flexible and to decrease its computational complexity, its two main challenges. The premises of the shapelet principle can be seen in a previous work in [Geurts, 2001].

A detailed overview of this topic is performed further in chapter 4.

3.4.4 Set of motifs and Sequence-based representation

A subsequence alone may not be sufficient to form a motif, [Bagnall et al., 2014a] proposes approaches to discover set of subsequences to form a motif.

Sequence-based representation can be seen as a kind of motif-based representation. The overall principle is to discover sequences of disjoint events and possibly to learn the time intervals between them to form sequential motifs. An event may be a motif as described in the previous section. Then the focus is set on the discovery of relationships between them. There is a link between sequence-based approaches and sequential pattern mining [Agrawal et al., 1993b].

Most sequence-based approaches rely on their own time series discretization process based on specific event definitions. Once events are represented as symbols, temporal relations are sought between them and encoded in an appropriate way to perform the task at hand [Moskovitch and Shahar, 2015, Batal et al., 2016, Baydogan and Runger, 2015, Patel et al., 2008, Fradkin and Morchen, 2015, Moerchen, 2006]. When sequential motifs are discovered for their link with class labels, sequential motifs can be used to perform time series classification [Fradkin and Morchen, 2015].

3.5 Ensemble of representations

The time series representations mentioned in the previous sections are complementary: each of them capture a specific information. Working together they can help to learn better classifiers. In [Baydogan et al., 2013], a bag-of-features representation is created by

combining global properties of the time series with local patterns to form one feature vector and improve classification performances. The relevance of ensemble of time series representations has also been shown in a recent study [Lines and Bagnall, 2015]. They combine periodogram, shapelet, autocorrelation with a time-based comparison of the time series. This approach is currently the leading approach in terms of classification performances on the literature standard set of datasets (UCR [Chen et al., 2015]).

3.6 Conclusions

In this chapter we have reviewed the time series representations. First we described the objectives to generate such representations. We defined three main groups of time series representations (time-based representations, feature-based representations and motif-based representations). We then performed an overview of some instances for each group of representation.

Each time series representation has its own characteristics, for instance: the type of representation obtained (a new time series, a scalar, a vector of scalar), the type and the localization (local or global) of the extracted information from the time series, the computational complexity, etc.

To select a relevant time series representation for an arbitrary dataset is not easy and also depends of the task to solve. Ensemble of representations have been proposed recently to overcome the first issue (adaptation of a representation to a dataset) with success in classification [Lines and Bagnall, 2015]. For the adaptation of the representation to the machine learning task, we propose a processing pipeline (figure 3.10) to generate classification-compatible representations from time series.

If time-based classification is used, the raw time series can directly feed the classifier (usually a $k - NN$ possibly associated with elastic distance measure to handle distortions as we have seen chapter 2). Time series can also be preprocessed using time-based representations either to reduce the dimensionality of the time series, denoise it or highlight specific information.

If feature-based classification is used (with a standard static attribute-value classifier), a proper feature vector is expected: classical static features must be generated from the time series. In this case, the time series follows a more complex path in the pipeline. For this purpose, feature-based representations (mainly from the time series analysis field) and specific time series mining techniques (such as motif-based representations) are used. The time series can even be preprocessed with time-based representations (dimensionality reduction, denoising, specific information highlighting).

The next part of this manuscript describes our contribution. We have developed a motif-based time series representation to perform feature-based time series classification.

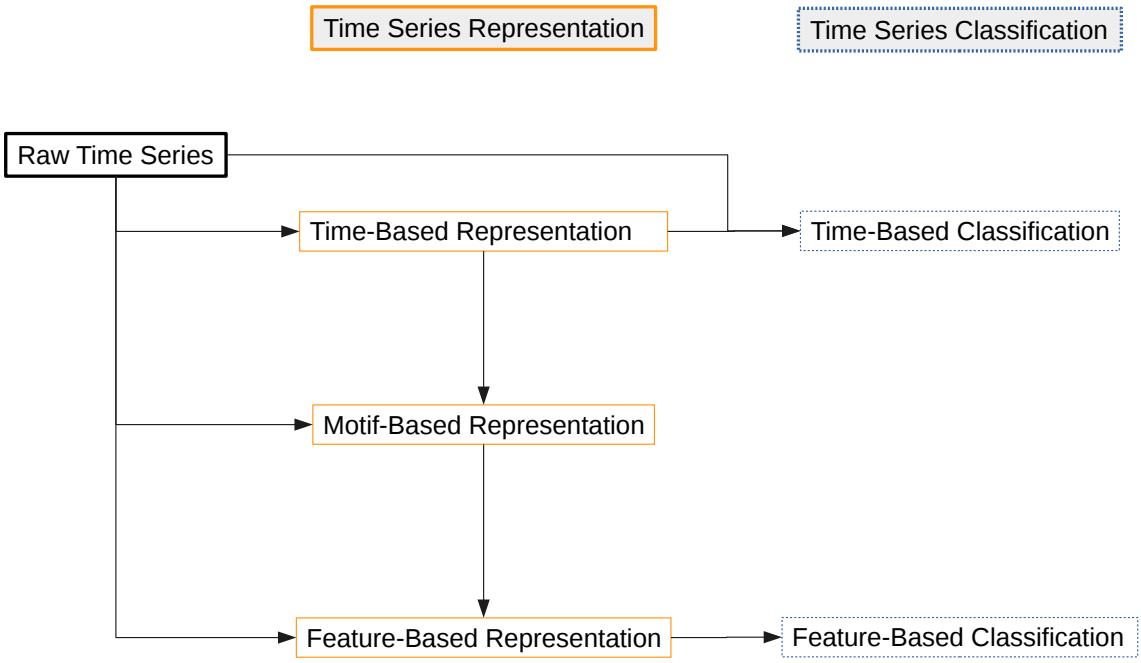


Figure 3.10: Time series processing pipeline to generate classification-compatible representations

Part II

Our Contribution: a Discriminant Motif-Based Representation

In part I, we have described the challenges and issues of time series mining, in particular when it comes to train a machine learning algorithm to perform classification. We have discussed the two techniques to overcome the issues at the core of most time series mining tasks: distance measures and time series representations. Chapter 3 was dedicated to an overview of the time series representations, which we grouped into three main families: time-based representations, feature-based representations and motif-based representations.

In this work, our concern is the development and the evaluation of a time series representation based on sets of subsequences meaningful to perform classification on time series of a dataset $\mathcal{D}$. This representation is discovered from $\mathcal{D}$ and supervised by the classification task at hand. This part described our proposition.

Chapter 4 is dedicated to a specific overview of the motif discovery task for time series classification, particularly the current leading concept, the time series shapelets. In chapter 5, we describe and formalize our vision of the motif discovery task for time series classification that results in a flexible framework that covers time series preprocessing, candidate motif generation and evaluation and the output to produce classical feature vectors. In chapter 6, we discuss the computational complexity of the motif discovery task and we propose a simple but effective solution experienced on one hundred datasets of the time series mining literature to reduce drastically the computational complexity of the discovery. In chapter 7, we rely on chapters 5 and 6 to instantiate the motif discovery framework to cast it into a classical feature selection problem. We demonstrate the effectiveness of the approach in terms of classification performance (similar to the best state of the art) for a fraction of the computational cost while preserving the interpretability abilities of motif-based representations. As the proposition is flexible, compatible with common machine learning algorithms and yet competitive, it opens several promising perspectives to improve its classification performances and its expressiveness.

Chapter 4

Motif Discovery for Classification

As we have seen in the previous chapters, the discovery of motifs in the time series is central in time series mining. The discussion section 3.4 has highlighted several possible purposes for discovered motifs: for instance clustering or summarization with recurrent motifs, anomaly detection with surprising motifs and classification with discriminant motifs. Many propositions in the literature come with their own definitions of recurrent, surprising, or discriminant. Each definition inspires a particular approach for the motif discovery, with specific assumptions. But they all share common issues, starting by the computational complexity of the discovery and the time series distortions.

This thesis is focused on the time series classification task. We propose to contribute by using discriminant motif discovery. The time series shapelet is the current leading principle for time series classification based on discriminant motifs. Other approaches exist: for instance [Geurts, 2001] propose a similar approach to the shapelet-tree (section 4.3). The computational complexity is handled by preprocessing the time series using the APCA representation and one single time series from each class is considered at each node of the decision tree. In [Zhang et al., 2009], the pattern extraction approach is similar to [Geurts, 2001] but a PAA representation is used instead of an APCA representation.

The time series shapelet principle inspires our work. Thus, in this chapter, we perform an overview of the works related with the shapelet. We begin by a general presentation of the shapelet principle. Then, we detail solutions proposed in the literature to overcome the computational complexity of the shapelet discovery and finally we describe various algorithms proposed around the shapelet principle for the shapelet discovery and to perform classification.

4.1 Time series shapelet principle

A time series shapelet is a discriminant motif discovery concept for time series classification that was first proposed in [Ye and Keogh, 2009]. The intuition behind the shapelet is to discover subsequences that have discriminatory power to support classification: the distribution of shapelet occurrences in the time series is not identical in the time series with respect to their class labels. In [Ye and Keogh, 2009], shapelets have been introduced informally as subsequences maximally representative of a class.

As we will see further in this section, many variants of the shapelet principle exist. However, we introduce the general principle of the shapelet discovery process with the following four steps:

Step 1: subsequence enumeration A subsequence is a sample of contiguous positions of length $l \leq L$ from time series $T_{n,m}$ of length L such that:

$$s_{T_{n,m}}(i,l) = [T_{n,m}(i), \ldots, T_{n,m}(i+l-1)]$$

All the time series subsequences are exhaustively enumerated from a dataset of labeled time series.

Step 2: distance calculation All the subsequences are evaluated to assess their fuzzy presence in all the time series of the dataset. The minimal Euclidean distance is used as a continuous proxy to indicate if at least one single occurrence of a subsequence is present in a time series. The minimum of the rolling distance is computed between each subsequence $s_{T_{n,m}}$ and all the time series $T_{n,m}$. Practically, the subsequence is shifted along the time series, point by point. The distance is computed between the subsequence and the corresponding points in the time series. The result is a time series with the distances between the subsequence and the time series for the successive positions. Then, the minimal distance value is retained: it is the distance of the best matching between the subsequence and the time series. Figure 4.1a illustrates this idea.

Given a subsequence, we obtain a scalar by time series that is the minimal distance, and thus we obtain a vector V of minimal distances between the subsequence and all the time series of $\mathcal{D}$.

Step 3: subsequence evaluation Then, the discriminatory power of a subsequence is calculated using its minimum distances with all the time series: we seek if the distributions of the distances depend on the time series classes, typically by using the information gain. For each subsequence, a threshold d is learned from the minimal distances in V such that we can divide the dataset D in two homogeneous subsets with respect to the labels: a subset $\mathcal{D}_1$ with time series closer to the subsequence and

a subset $\mathcal{D}_2$ with farther subsequences. If a subsequence has discriminatory power, $\mathcal{D}_1$, $\mathcal{D}_2$ or both should have more homogeneous label distributions. This is typically measured with the information gain. In this case the information gain evaluates if the entropy of the labels in $\mathcal{D}$ can be decreased by dividing the dataset in two subsets based on the minimal distances V to the subsequence and the threshold d.
The information gain in this case is:

$$IG = H(\mathcal{D}) - \left(\frac{|\mathcal{D}_1|.H(\mathcal{D}_1) + |\mathcal{D}_2|.H(\mathcal{D}_2)}{|\mathcal{D}_1| + |\mathcal{D}_2|} \right)$$

where $|.|$ is the cardinal and H is the entropy of a set as

$$H(\mathcal{D}) = -\sum_{c=1}^{\mathcal{C}} p_c.log(p_c)$$

with p_c the proportion of time series of class c in the set of time series $\mathcal{D}$.
Figure 4.1b illustrates the principle.

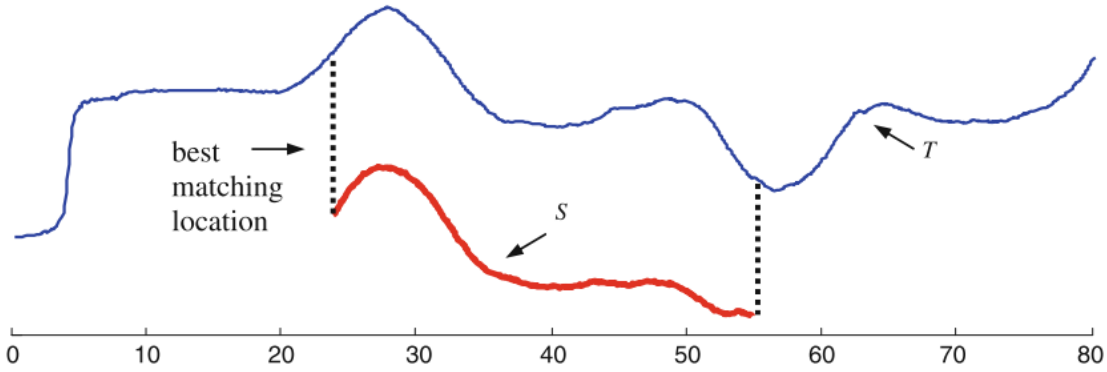
Step 4: shapelet selection The subsequences with the highest discriminatory power are conserved to perform the classification. While a subsequence is going through the discovery procedure (steps 1 to 3) it is usually called a *shapelet candidate*. Once it is selected for its high discriminatory power, it is called a *shapelet*.

The process to evaluate every subsequence of the dataset $\mathcal{D}$ is iterative.

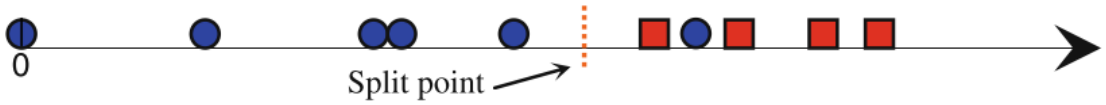
To classify a new time series, its minimal Euclidean distances to the shapelets are used. Several discovery and classification configurations have been proposed in the literature: some of them are discussed paragraph 4.3.

The time series shapelets have several advantages. The shapelets are particularly well suited when the discriminant information in the time series is related with localized subsequences. Time series classification approaches based on shapelets have shown competitive classification performances in recent literature benchmarks [Lines and Bagnall, 2015]. Once the shapelets have been discovered the classification of new time series is fast: the time required is linear with the length of the time series (to get the minimal distance). One other advantage of the shapelets is their interpretability. Since they are subsequences extracted from the time series, domain experts can easily interpret the underlying phenomena behind these subsequences and understand on what grounds the classification is performed.

However, the shapelets have several drawbacks. The main one is clearly the large computational complexity of the naive discovery of the shapelets in $O(L^4N^2)$, which is intractable for most datasets. The second drawback is the selection procedure: the information gain is applied independently by subsequence. Complementary or redundant subsequences are not taken into account. Relationships between subsequences may be required to classify some phenomena (a specific discussion is made further).



(a) Best matching position of a candidate shapelet and a time series. The Euclidean distance between the subsequence and the time series is returned



(b) Shapelet selection: blue discs and red squares are time series of distinct classes. The time series are sorted in ascending order according to their distance to a subsequence. The set of time series is split successively based on a threshold moving between the time series. The information gain is computed for each threshold. The threshold that gives the higher information gain is the split point used to perform the classification if the subsequence has the highest information gain

Figure 4.1: Illustration of the building of a *shapelet-tree* based on the shapelet discovery at each node (illustrations from [Ye and Keogh, 2011])

The time series shapelet principle has inspired many propositions. Many of them enhances the shapelets by addressing its issues. In the next paragraphs, we propose an overview of some of these propositions according to three axis: the improvement of the computation complexity of the discovery, the shapelet selection procedure and the algorithm for the classification based on shapelets.

4.2 Computational complexity of the shapelet discovery

The computational complexity of the discovery is the major drawback of the time series shapelet principle. Apart for the necessary time to get the shapelets, the high computational complexity also prevent us to extend the shapelet principle. Many propositions have been made to reduce the time needed for the discovery. We present some of them in this paragraph.

4.2.1 Early abandon & Pruning non-promising candidates

The original paper [Ye and Keogh, 2009] proposes to improve the computational complexity using two speedups. The first one is applied to the distance computation between a shapelet candidate and a time series. Since the minimum distance is sought, it is possible to stop

early a distance computation between the shapelet candidate and a subsequence from the time series once the distance exceeds the distance between the shapelet candidate and a previous subsequence of the time series. This principle called subsequence distance early abandon principle is illustrated in figure 4.2. The second speedup is applied at the shapelet candidate evaluation step. Instead of computing the information gain once the minimal distance has been computed for all the time series, the information gain can be computed after each time series, and an upper bound of the actual information gain can be calculated based on the optimistic assumption that the time series whose minimal distance have not been computed are perfectly separated according to their class labels for the current shapelet candidate. If the upper bound of the information gain is lower than the one of a previous shapelet candidate, then the shapelet candidate can be early abandoned.

4.2.2 Distance caching

The distance computations between all the subsequences for $\mathcal{D}$ induce many redundant calculation since many subsequences overlap. Subsequences overlap then it is possible to store the corresponding redundant calculations [Mueen et al., 2011] at the price of a space complexity in $O(NL^2)$. [Mueen et al., 2011] also proposes to get an upper bound for shapelet candidates to be evaluated, based triangular inequality and previously discovery shapelet candidate with high information gain. This trick allows to prune non-promising candidates with low upper bound information gain.

4.2.3 Discovery from a rough representation of the time series

Instead of tricks to early stop non-promising shapelet candidates, the fast-shapelets [Rakthanmanon and Keogh, 2013] proposes to perform a rough shapelet discovery on an approximate representation of the subsequences to select a small set of promising shapelet candidates among which the shapelet is discovered by performing the classical shapelet discovery on the raw time series for this small set of candidates. The *fast-shapelets* allows to accelerate the shapelet discovery by 1 to 3 orders of magnitude (depending on the datasets) while the classification performances are not significantly lower.

4.2.4 Alternative quality measures

The use of alternative quality measures for the shapelet discovery has been discussed in the literature. Instead of the Information Gain, [Lines et al., 2012] proposes the *F-stats* quality criterion and [Lines and Bagnall, 2012] proposes to use either the Kruskal-Wallis test or the Mood's Median test. They claim that while the classification performances are not significantly different, the time required for the shapelet discovery can be reduced by 18%

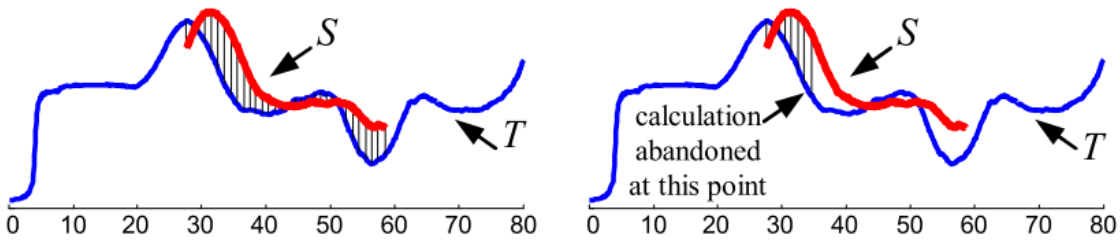


Figure 4.2: One speedup technique during the shapelet discovery, the *subsequence distance early abandon.* Once an ongoing subsequence distance computation exceeds a previous one, the distance computation is abandoned (illustrations from [Ye and Keogh, 2011])

in average in comparison with the information gain, because the number of computations required by these tests is lower.

4.2.5 Learning shapelet using gradient descent

While most of the shapelet discovery approaches are based on a search among a pool of shapelet candidates from subsequences from $\mathcal{D}$, [Grabocka et al., 2014] proposes to learn the best k shapelets using the minimization of a classification objective function that is a linear combination of the minimal distance between the shapelets and the time series of $\mathcal{D}$. The minimization is performed using stochastic gradient descent. Since the minimal distance between the shapelets and the time series is not derivable, they propose to estimate it using a soft minimum. Each iteration allows to fit the shapelet. One advantage of the approach is that the shapelets are fitted taking into account their interaction. However, the approach has some drawbacks: the number of parameters is much larger than a classical shapelet discovery based on shapelet candidates extracted from $\mathcal{D}$. Apart from specific parameters for the minimization (learning rate, number of iterations, regularization parameter, soft-minimum precision), the number of desired shapelets and their lengths have to be precised. Also, while for conventional shapelet discovery the number of desired shapelets doesn't impact the computational complexity, here the number of shapelets has to be fixed and directly impact the computational complexity. Finally, the convergence of the minimization also impacts the computational complexity through the number of iterations, while this parameter doesn't exist in conventional shapelet discovery.

4.2.6 Infrequent subsequences as shapelet candidates

In [He et al., 2012], the assumption is made that discriminative subsequences are infrequent in the whole set of subsequences that can be extracted from a dataset $\mathcal{D}$. The time series are discretized, and infrequent subsequences are discovered on the discretized time series. The infrequent subsequences become the shapelet candidates. The set of shapelet candidates should be much smaller than the exhaustive set of candidates and thus the discovery

should be faster. The shapelet discovery process (and the classification) is the same as the shapelet-tree, performed on the raw time series.While the discovery is said to be faster than the exhaustive search, there is no guarantee and the complexity of the discovery is the same as the one of the shapelet-tree in the worst cases. The experimentation have only been performed on 4 datasets with classification performances similar to the shapelet-tree.

4.2.7 Avoid the evaluation of similar candidates

In [Grabocka et al., 2015], the time series are transformed using *PAA*. Then to speedup the shapelet discovery, too similar shapelet candidates to previously tried candidates are pruned. The relevancy of a shapelet candidate is evaluated during the discovery as the other approaches, but instead of evaluating the candidate independently, it is evaluated together with previously accepted candidates using a multivariate forward selection [Guyon and Elisseeff, 2003] to take into account the shapelet relationships. The approach is said to be 3 to 4 orders of magnitude faster than the fast shapelets, the classification accuracy however doesn't reach the performances of the shapelet transform, considered as the reference while being particularly slow since the discovery requires an exhaustive enumeration of the subsequences.

4.3 Various shapelet-based algorithms for the discovery and classification stages

The original proposition for the shapelets was to embed them into a decision tree. The principle has been extended to improve the classification performances, for instance by learning more complex patterns. In this paragraph, we also present other configurations proposed for the discovery and the classification based on shapelets.

4.3.1 The original approach: the shapelet-tree

The first proposition, called shapelet-tree, embeds the shapelets in a decision tree [Ye and Keogh, 2009]. During the discovery phase a decision tree is expanded: each node of the tree is associated with one shapelet and a threshold on the minimal distance. A shapelet discovery is performed at each node of the tree on the subset of $\mathcal{D}$ that satisfies the path from the root to the current node.

During the classification, when a time series reaches a node, its minimal Euclidean distance to the shapelet is computed. Based on the threshold previously learned, a decision is taken, either to continue further the path or to label the time series. The shapelet-tree is illustrated figure 4.3.

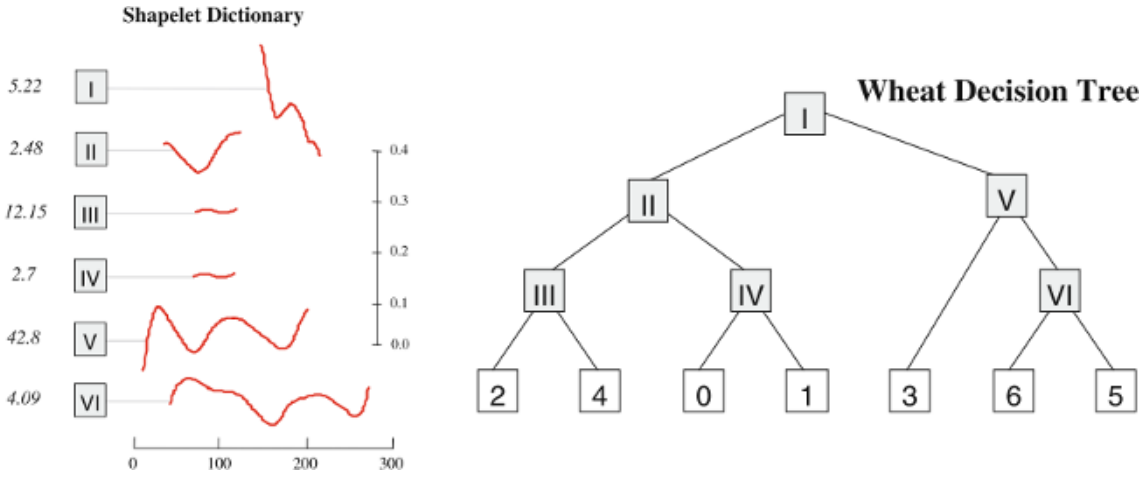


Figure 4.3: Illustration of the *shapelet-tree*. Shapelets (most discriminant motifs) are discovered at each node of the tree and support the classification of a new time series based on their similarity (illustrations from [Ye and Keogh, 2011])

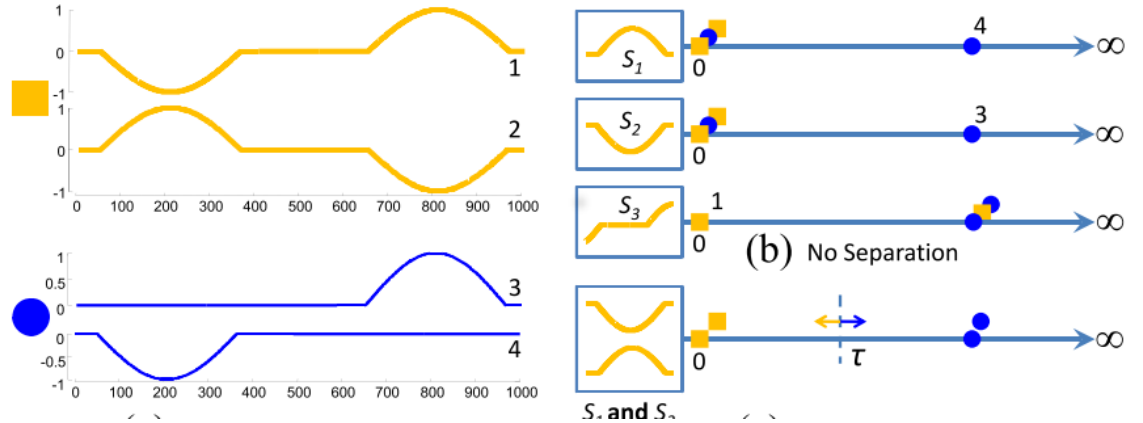


Figure 4.4: The *logical-shapelet* allows conjunction or disjunction of subsequences to improve the expressiveness of the shapelet principle (illustration from [Mueen et al., 2011])

The original shapelet-tree assumes that each shapelet alone is highly discriminant according to the information gain principle. With the shapelet-tree configuration, the shapelet's expressiveness is limited to a binary test of the shapelet's presence or absence in a time series, based on a single learned split point [Mueen et al., 2011]. Several works extend the shapelet-tree principle as detailed below.

4.3.2 Variants of the shapelet-tree

The logical-shapelets have been proposed to improve the expressiveness of the shapelet-tree [Mueen et al., 2011]. Instead of associating one single shapelet to each node, the idea is to associate several shapelets to each node by conjunction or disjunction of shapelets. The principle is shown figure 4.4.

In order to improve the classification performances and following the ensemble learning strategies, several works have proposed ensembles of shapelet-trees [Patri et al., 2014, Cetin et al., 2015, Karlsson et al., 2016].

4.3.3 Shapelet transform

One major drawback of the previous approaches is the lack of flexibility of the learning and classification stack. A specific learner is designed to perform the classification after having performed the shapelet discovery: we cannot take advantage of the performances of state-of-the-art classifiers and variations in the structure of the data (multivariate time series instead of univariate, addition of contextual data to the time series, etc.) would require the design of new classifiers and discovery procedure.

[Hills et al., 2014] introduces the shapelet transform, which is a first step towards the generalization of the shapelet principle. Instead of discovering iteratively the shapelets at each node of a shapelet-tree, the top-k shapelets in terms of information gain are simply learned from $\mathcal{D}$. Each shapelet must still be highly discriminant independently of the others, but state-of-the-art classifiers can be trained on a matrix of distances where each row is an instance of $\mathcal{D}$ and each column is a feature corresponding to the minimal Euclidean distance between each time series and each top-k shapelets.

This approach has the best classification performances for a shapelet-based approach on the UCR datasets (we use it as a reference for our experimentation). However the approach has two main drawbacks. First, the shapelet discovery is based on the exhaustive enumeration of the subsequences and thus is intractable on large datasets. Secondly, each shapelet is discovered independently from each other, making the assumption of a high discriminatory power for each subsequence meaningful for the classification that is untrue [Grabocka et al., 2014, Renard et al., 2016a].

4.3.4 Other distance measures

[Arathi and Govardhan, 2014] proposes to change the Euclidean distance, used to measure the similarity between a shapelet and the subsequences, by the Mahalanobis distance. The objective is to address the sensitivity of the Euclidean distance to the difference of scales.

4.3.5 Shapelet on multivariate time series

[Ghalwash et al., 2013] proposes a shapelet-based approach for multivariate classification in 3 steps. First, the approach works on each variable independently and perform the classical shapelet discovery based on minimal distance computation and search of a threshold to maximize the information gain associated with each shapelet candidate. Based on the distances between every shapelet candidate and the threshold, a binary matrix of absence

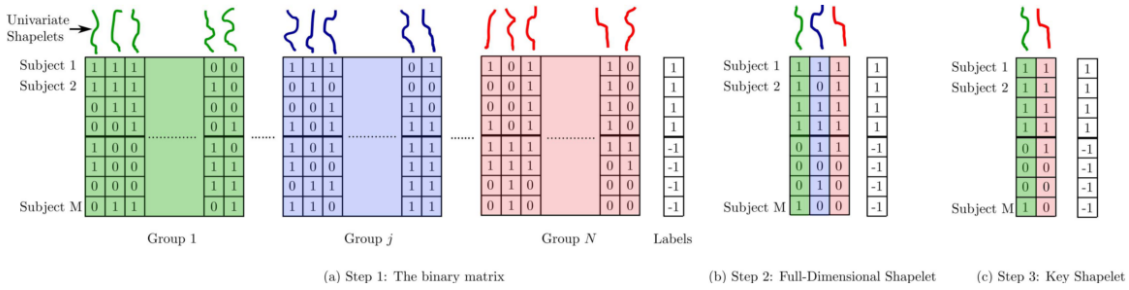


Figure 4.5: Illustration of the overall principle of the discovery of the *key-shapelet*. First a matrix of absence/presence of univariate shapelets in the time series is built. Then, univariate shapelets are assembled across the variables to maximize accuracy. Finally, only the typical univariate shapelets of one class are retained to form a *key-shapelet* (illustrations from [Ghalwash et al., 2013])

or presence of the candidates in the time series is built. In a second step, the univariate shapelet candidates are assembled into multivariate candidates (with one shapelet from each variable). In a third step, univariate shapelets irrelevant in the multivariate shapelet are removed to form a key-shapelet, which is used to perform the classification. Following this scheme, several key-shapelets are extracted. Steps 2 and 3 are based on an optimization procedure, to maximize accuracy for step 2 and for step 3 to maximize the typicality of the key-shapelet to one of the class and to maximize its sparsity. The overall procedure is displayed figure 4.5. The key-shapelets are used in [Ghalwash et al., 2013] to perform interpretable early time series classification.

Some shapelet-tree variants mentioned previously [Cetin et al., 2015, Karlsson et al., 2016] are designed to perform multivariate time series classification.

4.3.6 Early time series classification

[Xing et al., 2011] proposes to consider shapelets as features for early time series classification. The approach extracts all the shapelet candidates from $\mathcal{D}$ and for each of them learns distance threshold such that the candidate shapelet should be highly distinctive of one class. The shapelets retained as features are selected according to a criterion that combines the earliness of the shapelet in the time series and its support among the time series of $\mathcal{D}$.

[Ghalwash et al., 2014] proposes a shapelet-based early time series classification approach with an uncertainty estimate to decide if the prediction can be made or not. The approach is based on the EDSC (Early Distinctive Shapelet Classification) that discovers classically the shapelets. The shapelets are then ranked according to a score that characterizes their earliness and accuracy. The uncertainty score is based on the distance of a time series to a shapelet (the closer, the likely it belongs to its class) and the ability of a

given shapelet to accurately classify the time series. An extension is proposed to aggregate the uncertainties of all the discovered shapelets to decide which class is the most likely.

4.4 Conclusions

In this work, we focus on the discovery of time series motifs to perform time series classification. In this chapter, we have reviewed the main approaches to discover such motifs. The time series shapelet has emerged as a major approach. The shapelet faces three main problematics: the computational complexity of the shapelet discovery, the output produced by the shapelet to perform the classification (ie. development of a ad-hoc classifier or use of a standard classifier) and the expressiveness of the shapelet principle to catch the information contained in the time series. We performed a state-of-the-art in relation with these problematic.

The approaches proposed in the literature are not fully satisfactory according to some or all the three criteria mentioned previously. In the next chapter, we propose our own formalization of the discovery of discriminant set of motifs to perform time series classification. Based on this formalization, we develop our proposition evaluated further in this manuscript.

Chapter 5

Our Framework for the Discovery of a Discriminant Motif-Based Representation

In the previous chapter, we introduced the time series shapelet concept. In this chapter, we formalize our proposition for the discovery of a discriminant motif-based representation for time series classification. The idea is to discover in a dataset a set of motifs used to generate a classical feature vector to perform a feature-based classification of the time series. These motifs, and the resulting feature vector, form a representation of the information contained in the time series.

We propose a framework to learn such a representation. The framework relies on the concept of subsequence transformation: a subsequence enumerated from a dataset is used to transform a time series into a feature using a distance measure and an aggregation function. Many subsequence transformations (with different subsequences) are gathered to discover among them a discriminant subset of subsequence transformations. This discriminant subset of subsequence transformations is used to generate a feature space (the discriminant motif-based representation). The resulting feature space is used to train a feature-based classifier and perform classification of new time series (previously transformed using the subset of subsequence transformations). The overall principle is shown figure 5.1.

We begin this chapter by a review of the notations. Then we introduce the concept of subsequence transformation to generate the features of our representation. Then we discuss the discovery of the subset of subsequence transformations to obtain a motif-based representation of the time series.

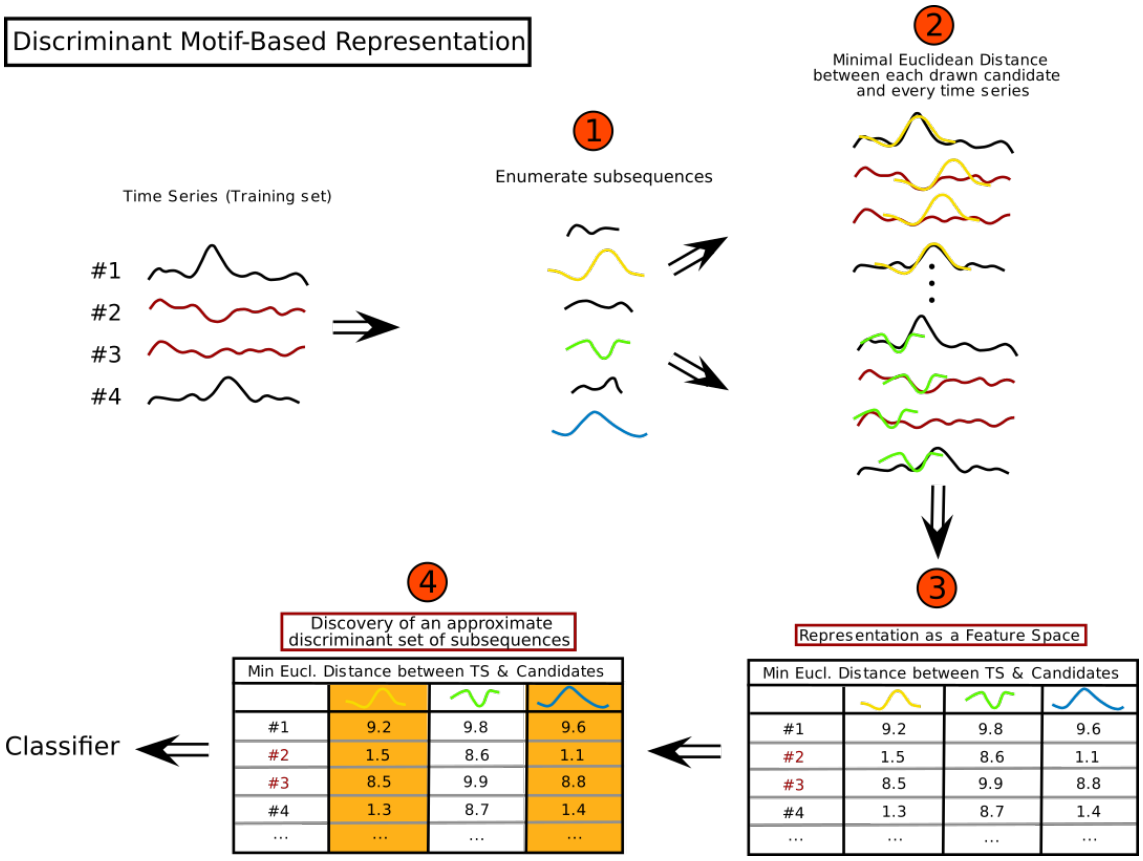


Figure 5.1: Discovery of a discriminant motif-based representation for time series classification. (1) Subsequences are enumerated from the dataset (2) The distances are computed between subsequences and time series (3) A feature space is generated from the distances (4) The Discriminant set of motif can be discovered from the feature space to perform classification

5.1 Notations

We review some of the notations introduced section 2.1. We have a training set $\mathcal{D}$:

$$\mathcal{D} = \{T_1, \ldots, T_n, \ldots, T_N\}$$

composed by $N \in [1, 2, \ldots, |\mathcal{D}|]$ time series T_n:

$$T_n = [T_n(1), \ldots, T_n(i), \ldots, T_n(|T_n|)]$$

Each time series has a length $L = |T_n|$ such that $L_{min} \leq L \leq L_{max} \in \mathbb{N}^*$ where L_{min} and L_{max} are respectively the smallest and the largest time series' lengths in $\mathcal{D}$. A time series can be univariate such that $T_n = T_{n,1}(i)$ or multivariate such that $T_n = T_{n,m}$ where $m \geq 1$ is the index of the variable.

Since our focus is the time series classification task, each time series T_n is associated with one single class label (weak classification) $y(T_n) \in \mathcal{C}$ where $\mathcal{C} = \{c_1, \ldots, c_{|\mathcal{C}|}\}$ is a finite set of class labels and $Y = [y(T_1), \ldots, y(T_N)]$ is the vector of labels of $\mathcal{D}$.

A time series subsequence s is extracted from a time series $T_{n,m}$ at a starting position i with a length l such that:

$$s_{T_{n,m}}(i, l) = [T_{n,m}(i), \ldots, T_{n,m}(i + l - 1)]$$

$\mathcal{S}$ is the set of all the subsequences s we can extract from $\mathcal{D}$ for all the possible subsequence's parameters: time series T_n, variable m, starting position i and length l of the subsequence. The size of $\mathcal{S}$ is:

$$|\mathcal{S}| = \frac{1}{2} \sum_{1 \leq n \leq |\mathcal{D}|} L_n(L_n + 1) \tag{5.1}$$

Then $|\mathcal{S}|$ is linearly dependent of the number of time series in $\mathcal{D}$ and quadratically dependent of the length L of the time series.

In this work, to generate $\mathcal{S}$, we only consider subsequences s with a maximum length L_{min} that is the length of the smallest time series of $\mathcal{D}$, *ie.* $|s| < L_{min}, \forall s \in \mathcal{S}$. The motivation of this constraint will become clear in the next section to perform subsequence transformation (see equation 5.2).

On this basis, the complexity to generate all the subsequences from $\mathcal{D}$ is:

$$O(\,|\mathcal{S}|\,) = O(\,|\mathcal{D}|\,.\,L_{min}^2\,)$$

We are interested in the subsequences because they can represent a shape or behavior

in the time series meaningful for our machine learning task.

5.2 Feature generation: the subsequence transformation principle

In chapter 4, we have seen that a time series representation can be obtained through subsequence transformation. The rough idea is to detect the presence or the absence of a subsequence s in a time series using a distance measure and an aggregation function. This results in a feature: each subsequence transformation applied on a time series produces a scalar that describes to which extent the subsequence is present in a time series. We use a large set of subsequence transformations on each time series to form a feature space.

We formalize below the subsequence transformation principle.

A set of subsequence transformations Ψ transforms a time series T_n using a set of subsequences $\mathcal{S}$ to produce a vector of scalars X_n^S such that:

$$X_n^S = \Psi(T_n, \mathcal{S})$$

Each scalar in X_n^S describes to which extent a subsequence of $\mathcal{S}$ is present in the time series T_n.

A set of subsequence transformations Ψ is composed of unitary transformations ψ for each subsequence $s_{T_{n,m}} \in \mathcal{S}$. Each of them returns a scalar such that:

$$X_n^{s_{T_{n,m}}} = \psi(T_{n,m}, s_{T_{n,m}})$$

A subsequence $s_{T_{n,m}}$ is extracted from the variable m of the time series $T_{n,m}$: the subsequence transformation ψ is only applied to the variable m of the time series $T_{n,m}$.

The subsequence transformation ψ is based on a distance measure d and an aggregation function a:

$$\psi_{a,d}(T_{n,m}, s_{T_{n,m}}) = a \circ d(T_n, s_{T_{n,m}})$$

d is a rolling distance on $T_{n,m}$ with $s_{T_{n,m}}$ the "window function". d produces a new time series that describes the distance between $s_{T_{n,m}}$ and T_n across time.

The aggregation function a is a "query" applied on the new time series produced by d: it describes with a scalar to which extent $s_{T_{n,m}}$ is present in $T_{n,m}$.

As example, we instantiate the subsequence transformation ψ with a classical configuration [Geurts, 2001, Ye and Keogh, 2009]. We define d as the rolling Euclidean distance and a as the minimum: in this case ψ will return the minimum of the rolling Euclidean

distance between $T_{n,m}$ and $s_{T_{n,m}}$. It will return the closest match between $T_{n,m}$ and $s_{T_{n,m}}$ according to the Euclidean distance. Formally we have:

$$\begin{aligned} \psi_{min,\mathcal{L}_2}(T_{n,m}, s_{T_{n,m}}) &= min \circ d_{\mathcal{L}_2}(T_{n,m}, s_{T_{n,m}}) \\ &= min \int_{i}^{i+l-1} \sqrt{(T_{n,m}(t) - s_{T_{n,m}}(t))^2}\, dt \ , \ \forall i \in [\, 1 \ldots |T_n| - l \,] \end{aligned} \tag{5.2}$$

The subsequence transformation ψ can be repeated for all the subsequences present in the dataset.

When we use a single pair of distance and aggregation functions (for instance rolling Euclidean distance and minimum), $X^{\mathcal{S}}$ is a $\mathbb{R}^{|\mathcal{D}|\times|\mathcal{S}|}$ matrix where the unitary subsequence transformations $\psi_{a,d}(T_{n,m}, s_{T_{n,m}})$ are the columns and the rows are the instances of $\mathcal{D}$ (*ie.* the time series $T_{n,m}$).

$X^{\mathcal{S}}$ is a feature space and a motif-based representation of the time series where each column $X_{s_{p,m}}$ is a feature and each line $x_{T_{n,m}}$ is an instance of the dataset $\mathcal{D}$, such that:

$$\mathbf{X}^{\mathcal{S}} = \begin{array}{c} \\ x_{T_{1,m}} \\ \\ x_{T_{n,m}} \\ \\ x_{T_{|\mathcal{D}|,m}} \end{array} \begin{array}{c} \begin{array}{ccccccc} X_{s_{1,m}} & & & X_{s_{p,m}} & & & X_{s_{|\mathcal{S}|,m}} \end{array} \\ \begin{pmatrix} \psi(T_{1,m}, s_{1,m}) & \cdots & \psi(T_{1,m}, s_{p,m}) & \cdots & \psi(T_{1,m}, s_{|\mathcal{S}|,m}) \\ \vdots & & \vdots & & \vdots \\ \psi(T_{n,m}, s_{1,m}) & \cdots & \psi(T_{n,m}, s_{p,m}) & \cdots & \psi(T_{n,m}, s_{|\mathcal{S}|,m}) \\ \vdots & & \vdots & & \vdots \\ \psi(T_{|\mathcal{D}|,m}, s_{1,m}) & \cdots & \psi(T_{|\mathcal{D}|,m}, s_{p,m}) & \cdots & \psi(T_{|\mathcal{D}|,m}, s_{|\mathcal{S}|,m}) \end{pmatrix} \end{array}$$

As discussed in the previous section, if we consider the exhaustive set $\mathcal{S}$ of subsequences generated from $\mathcal{D}$, the feature space $X^{\mathcal{S}}$ has a feature complexity in $O(\, |\mathcal{D}| \,.\, L^2_{min}\,)$.

Even by considering a simple setup (one single pair of distance and aggregation), the dimension of $X^{\mathcal{S}}$ becomes quickly very large with L. As we will see later, it is not only a space and memory issue: it is also a machine learning issue (curse of dimensionality) [Hughes, 1968, Hastie et al., 2009] and a major time complexity problem since each single $\psi_{a,d}$ has a non-negligible computational time.

Up to this point, the feature space $X^{\mathcal{S}}$ contains many irrelevant features: many subsequences in $\mathcal{S}$ are not meaningful. In the next sections, our concern is about the discovery of relevant subset of subsequences from $\mathcal{S}$ using X^S for the time series classification task.

5.3 Motif-based representation using discriminant set of subsequences

We make the assumption that an oracle knows the perfectly discriminant set of subsequences $\mathcal{Z} = \{z_1, \ldots, z_p, \ldots, z_{|Z|}\}$ with $p \in \mathbb{N}^*$ and $|z_p| \in [1; L_{min}]$ where z_p is discriminant of one class or a subset of classes of $\mathcal{C}$.

$\mathcal{Z}$ is perfectly discriminant in that it contains all the possible subsequences, which taken independently or not, are discriminant enough to solve the classification problem such that a function f is able to learn a mapping:

$$f(X_n^{\mathcal{Z}}) \mapsto y(T_n)$$

$$\iff f \circ \Psi(T_n, \mathcal{Z}) \mapsto y(T_n)$$

$X_n^{\mathcal{Z}}$ is the feature vector generated with the transformation of T_n using subsequences in $\mathcal{Z}$ *ie.* $X_n^{\mathcal{Z}} = \Psi(T_n, \mathcal{Z})$.

However, given a dataset $\mathcal{D}$ with class labels Y, the perfect discriminant set of subsequences $\mathcal{Z}$ is unknown.

Our objective is to discover from $\mathcal{D}$ an approximate discriminant set of subsequences $\hat{\mathcal{Z}} = \{\hat{z}_1 \ldots \hat{z}_p \ldots \hat{z}_{|\hat{\mathcal{Z}}|}\} \setminus \hat{\mathcal{Z}} \subset \mathcal{S}$ that produces a feature vector $X_n^{\hat{\mathcal{Z}}}$ using Ψ such that the classification performance defined by a measure PM obtained with $f(X_n^{\hat{\mathcal{Z}}})$ is as close as possible to the one obtained with $f(X_n^{\mathcal{Z}})$.

$$\hat{\mathcal{Z}} \subset \mathcal{S} \setminus PM(\, f(X_n^{\hat{\mathcal{Z}}})\,) \rightarrow PM(\, f(X_n^{\mathcal{Z}})\,)$$

The issue is how to discover an approximate discriminant set of subsequences $\hat{\mathcal{Z}}$.

The subsequence transformation Ψ of $\mathcal{D}$ using subsequences from $\mathcal{S}$ forms a feature space $X^{\mathcal{S}}$. Most of the subsequences $s_{T_{n,m}} \in \mathcal{S}$ used to compute features $\psi_{a,d}(T_n, s_{T_{n,m}})$ will not be part of $\hat{\mathcal{Z}}$ as they are likely to be meaningless for the classification task.

Given every subset of subsequences $\mathcal{S}' \subset \mathcal{S}$, the set $\hat{\mathcal{Z}}$ is obtained by minimizing:

$$\underset{\mathcal{S}' \subset \mathcal{S}}{argmin} \sum_{1 \leq n \leq |\mathcal{D}|} \left\| y(T_n) - f(X_n^{\mathcal{S}'}) \right\|_2^2 + \lambda \left\| \mathcal{S}' \right\| \tag{5.3}$$

Where$\|.\|$ is a generic norm and f is a function able to learn relationships between features $\psi_{a,d}(T_n, \hat{z}_p)$ of $X_n^{\mathcal{S}'}$ and class labels Y.

Equation 5.3 aims at discovering $\hat{\mathcal{Z}}$ by minimizing the classification error with a regularization applied on the number of subsequences in $\hat{\mathcal{Z}}$ to limit the computational complexity due to each unitary subsequence transformation ψ and improve the generalization.

As we will see chapter 7, the machine learning literature has solutions to solve this problem, in particular in the feature selection community.

5.4 Conclusions

In this chapter, we proposed a framework to discover discriminant motif-based representation. We defined three main concepts:

- The subsequence transformation $\psi_{a,d}(T_{n,m}, s_{T_{n,m}})$, which is the transformation of a time series based on three parameters: a subsequence $s_{T_{n,m}}$, a rolling distance measure $d(T_n, s_{T_{n,m}})$ and an aggregation function a. The idea behind this concept is to characterize to which extent a subsequence is present in a time series.

- The perfectly discriminant set of subsequences $\mathcal{Z}$ such that gathering each individual subsequence transformation $\psi_{a,d}(T_{n,m}, z_p) \,|\, z_p \in \mathcal{Z}$ generates a feature vector $X_n^{\mathcal{Z}} = \Psi(T_n, \mathcal{Z})$ for each time series T_n suitable to solve perfectly the classification problem $f(X_n^{\mathcal{Z}}) \mapsto y(T_n)$.

- The approximate discriminant set of subsequences $\hat{\mathcal{Z}}$: the perfect discriminant set of subsequences $\mathcal{Z}$ is unknown and an approximate set $\hat{\mathcal{Z}}$ must be discovered from the dataset $\mathcal{D}$ such that the classification performances of $f(X_n^{\hat{\mathcal{Z}}})$ are as close as possible to the classification performances obtained with $f(X_n^{\mathcal{Z}})$.

In the next two chapters, we discuss the instantiation of this framework to discover discriminant sets of subsequences $\hat{\mathcal{Z}}$ in datasets $\mathcal{D}$. The next chapter discusses the scalability of the discovery.

Chapter 6

Scalable Discovery of Discriminant Motifs

In the previous chapter, we formalized the discovery of a discriminant motif-based representation using the concept of subsequence transformation $\psi_{a,d}(T_n, s_{T_{n,m}})$ of a time series T_n by a subsequence $s_{T_{n,m}}$ to generate a feature vector relevant for time series classification. The exhaustive enumeration of all the subsequences in $\mathcal{D}$ produces a very large set $\mathcal{S}$, whose size augments linearly with $|\mathcal{D}|$ and quadratically with L_{min}. It leads to intractable computations to generate the feature vector $X^{\mathcal{S}}$ and to perform calculations on it.

In this chapter, we address this computational complexity issue. Based on the observation that most subsequences in a time series dataset $\mathcal{D}$ are redundant, we show experimentally that we can reduce by several orders of magnitude the time required for the discovery of the approximate discriminant set of subsequences $\hat{\mathcal{Z}}$ in comparison to the exhaustive discovery. This result is achieved with a random sampling among the exhaustive set of subsequences without significantly deteriorating the classification performances. We also demonstrate the scalability of the approach, since the number of subsequences to draw is not related to the number of time series in the dataset $\mathcal{D}$.

6.1 An intractable exhaustive discovery among $\mathcal{S}$

As observed chapter 5 the complexity of the exhaustive enumeration of all the subsequences from a dataset of time series $\mathcal{D}$ grows in $O(\,|\mathcal{D}|\,.\,L_{min}^2\,)$. Figure 6.1 illustrates the evolution of the exhaustive number of subsequences $|\mathcal{S}|$ as a function of L_{min} and $|\mathcal{D}|$. $|\mathcal{S}|$ is the number of features $\psi_{a,d}(T_{n,m}, s_{T_{n,m}})$ in $X_n^{\mathcal{S}}$. The number of subsequences becomes quickly extremely large as L_{min} and $|\mathcal{D}|$ increase. For instance, with a dataset of moderate size composed of 1000 time series of 1500 points, the exhaustive number of subsequences is beyond 10^9.

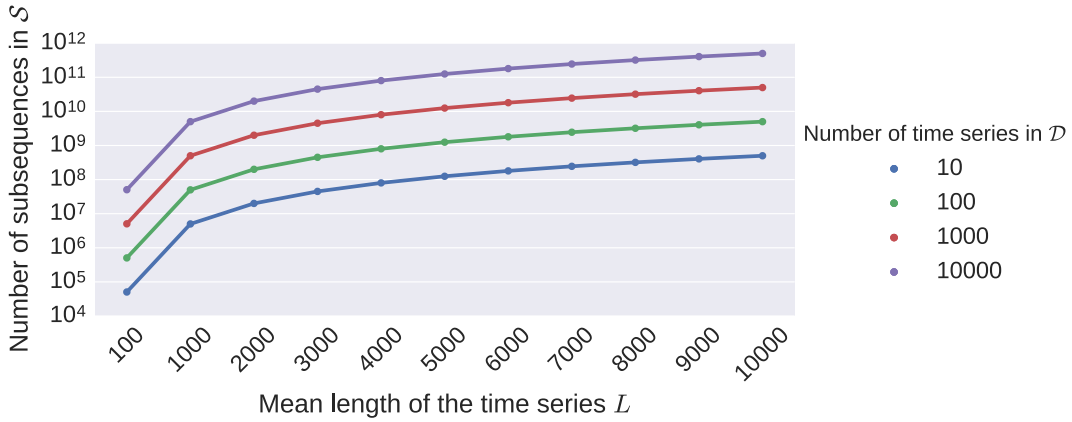


Figure 6.1: The exhaustive number of subsequences in a dataset $\mathcal{D}$ grows quadratically with the length of the time series and becomes quickly intractable

Such a dimensionality generates at least two issues.

The first issue is related to the building of the feature vector $X^{\mathcal{S}}$. The most demanding step in terms of computations is the calculation of each $\psi_{a,d}(T_{n,m}, s_{T_{n,m}})$ and more precisely the distance part $d(T_{n,m}, s_{T_{n,m}})$, which requires numerous calls to a distance function to compare subsequences in an iterative fashion [Renard et al., 2015]. For the overall procedure the distance calculations are performed between all the subsequences and the time series of $\mathcal{D}$. To get a better insight, we can estimate the number of distance computations involved in the exhaustive building of $X^{\mathcal{S}}$ while varying the two critical parameters: the minimum time series length L_{min} and the number of time series in $\mathcal{D}$. We have seen equation 5.1 that the number of subsequences in $\mathcal{S}$ is:

$$|\mathcal{S}| = \frac{1}{2} \sum_{1 \leq n \leq |\mathcal{D}|} L_n(L_n + 1)$$

If we set $L_n = L_{min}$ as discussed in the previous chapter:

$$|\mathcal{S}| = \frac{|\mathcal{D}| \,.\, L_{min}(L_{min} + 1)}{2}$$

The number of distance computations between each pair of subsequences $d(T_n, s_{T_{n,m}}) = d(s_{T_{n,m}}, T_n)$ is a $2-$combination with repetitions:

$$\Gamma^2_{|\mathcal{S}|} = \left(\frac{|\mathcal{S}| + 2 - 1}{2} \right) = \frac{(|\mathcal{S}| + 1)!}{2!\,(|\mathcal{S}| - 1)!} = \frac{(|\mathcal{S}| + 1)\,|\mathcal{S}|!}{2\,|\mathcal{S}|!\,\frac{1}{|\mathcal{S}|}} = \frac{|\mathcal{S}|^2 + |\mathcal{S}|}{2}$$

The number of distance calculations is quadratic with the number of subsequences in the set. As mentioned previously, a dataset can easily reach $|\mathcal{S}| = 10^9$ subsequences that would require $\Gamma^2_{10^9} = \frac{10^{18}+10^9}{2} \sim 10^{18}$ distance computations to perform.

In the literature, as discussed chapter 4, some approaches have been proposed to handle the computational complexity issue: many of them are based on pruning of non-promising subsequences and dimensionality reduction of the time series. However, the number of distance computations remains very large or intractable while the algorithm becomes more complicated.

The second issue is related to the selection of the approximate discriminant set of subsequences. Since the number of features in the feature vector $X^{\mathcal{S}}$ is related to $|\mathcal{S}|$, the number of distance measures d and the number of aggregation functions a used, the number of columns in $X^{\mathcal{S}}$ is at least $|\mathcal{S}|$, which can easily reach billions as we have seen. This is clearly an important constraint that limits our algorithmic possibilities to learn effectively $\hat{\mathcal{Z}}$ from $\mathcal{S}$: from a computational complexity point of view and also from a machine learning point of view (curse of dimensionality [Hughes, 1968, Hastie et al., 2009]).

6.2 Subsequence redundancy in $\mathcal{S}$

At this point, we can question the relevancy to enumerate and evaluate all the features $\psi_{a,d}(T_{n,m}, s_{T_{n,m}})$ that we can derive from all the subsequences in $\mathcal{S}$.

To illustrate the idea, let's take one random time series dataset from the literature. We extract two similar subsequences from two distinct time series (see figure 6.2). We also extract the next 20 subsequences of the same length in both cases. The shapes and thus the information are clearly the same for all the subsequences that have been extracted. The process could have been repeated for other positions, varying subsequence lengths and across all the time series of the dataset.

To assess more quantitatively the redundancy, we repeat a similar process for one single time series (T_1) of the same dataset. We extract 4 subsequences and we compute the distances $d_{\mathcal{L}_2}(T_1, s_{T_1})$ for each of them. The subsequences and the distances over the time series T_1 are shown figure 6.3. At the neighborhood of an extracted subsequence, the distance clearly tends to 0, meaning the subsequences around are strongly similar. By repeating the process with varying lengths and across time series, the number of subsequences with strongly similar shapes, and then redundant, is large.

The redundancy around an extracted subsequence is a well-known phenomenon in the literature and sometimes named *trivial matches* [Lin et al., 2002].

6.3 A random sub-sampling of $\mathcal{S}$ is a solution

Because of the inherent variability and heterogeneity of time series data, it is not an easy task to find an actuator or a heuristic to reduce the computational complexity of the discovery by reducing the subsequence redundancy.

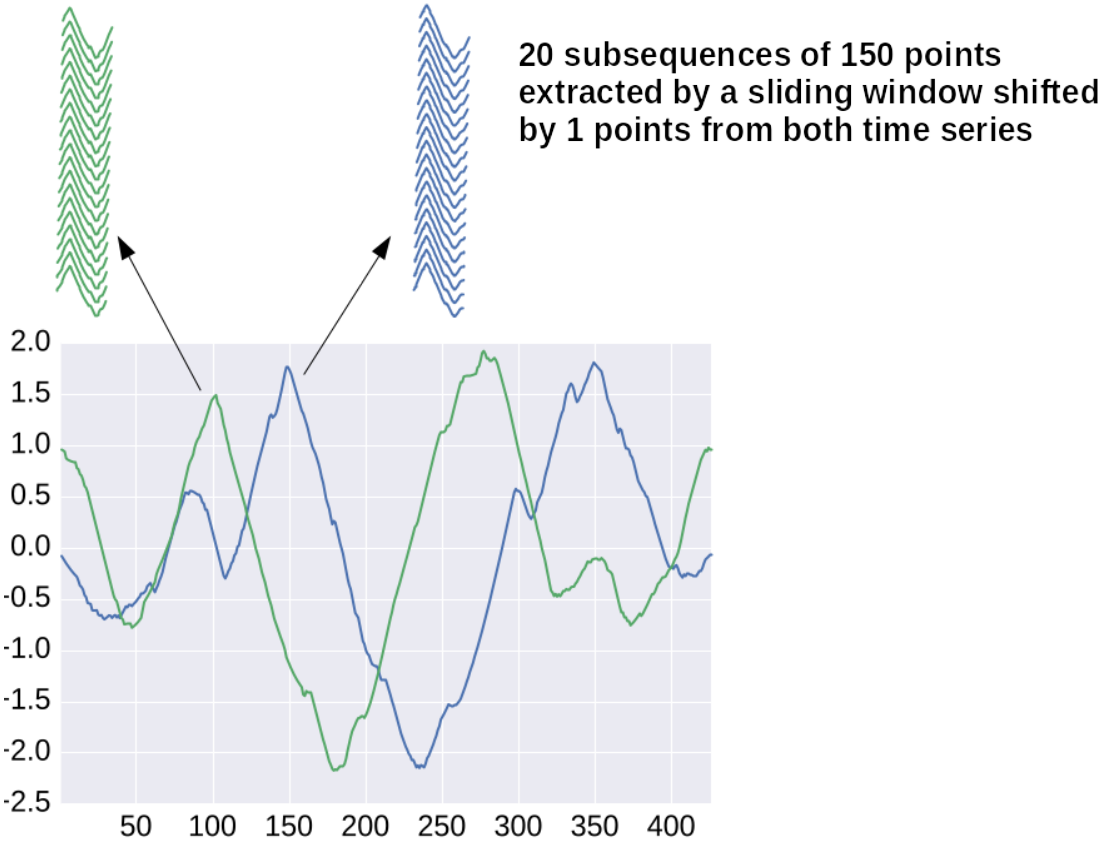
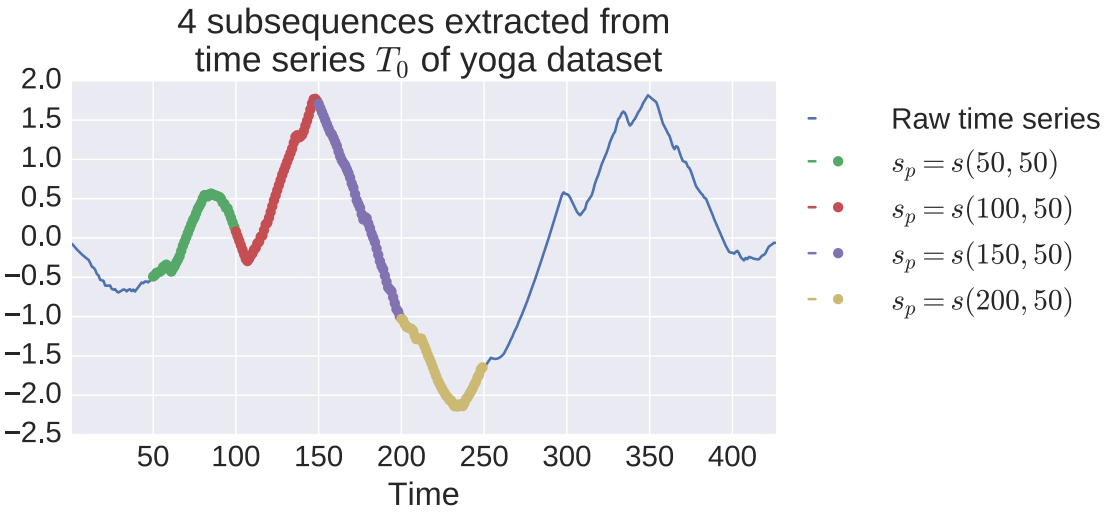
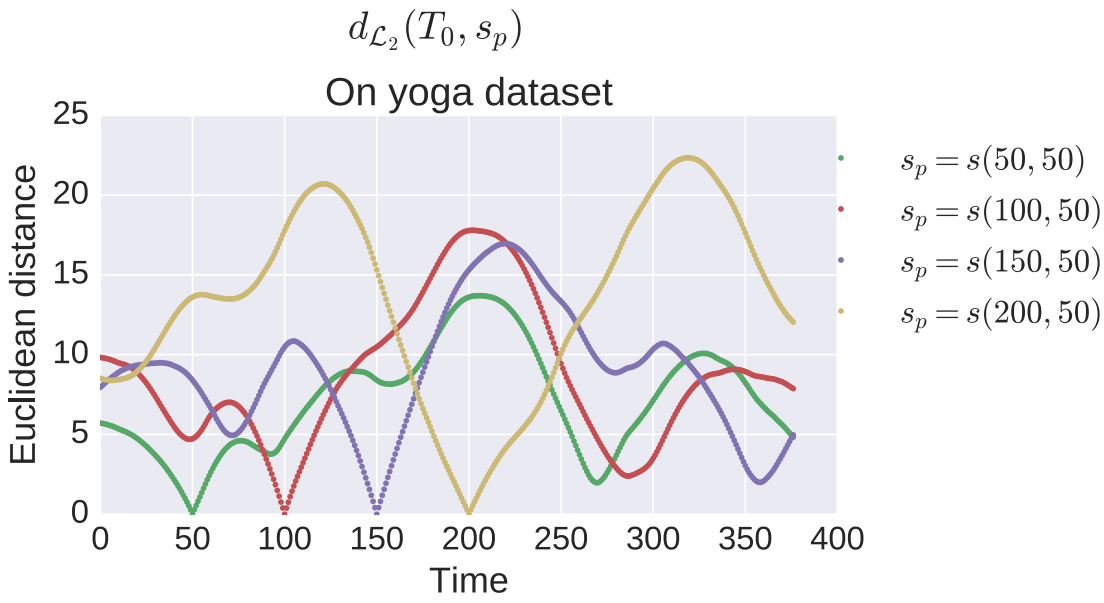


Figure 6.2: Two similar subsequences extracted from two distinct time series, and the 20 following subsequences of same length: their shapes and thus the information are mostly similar. The time series are extracted from the *yoga* dataset of the *UCR* repository



(a) 4 subsequences of length 50 are extracted from a raw time series at 4 distinct positions



(b) The rolling Euclidean distance $d_{\mathcal{L}_2}(T_1, s_{T_1})$ is computed between each subsequence and the raw time series

Figure 6.3: To illustrate subsequence redundancy intra-time series, we extract 4 distinct subsequences from one single time series and compute its Euclidean distance with the subsequences of the time series it comes from. The distance between each time series and its neighborhood is always close to 0

Oftentimes we have no prior knowledge on how the meaningful information appears in the time series like what kind of subsequences could be part of $\hat{\mathcal{Z}}$: it is hard to guess the range of their parameters (time series of origin, variable, position and length). Then, for the discovery of $\hat{\mathcal{Z}}$ it is undesirable to set up assumptions *a priori* to lead the discovery process in order to accelerate it.

Instead of an exhaustive enumeration and evaluation of all the subsequences, we propose to perform a random sub-sampling of $\mathcal{S}$, in order to generate a small but diverse fraction of the whole set of subsequences while covering a wide variety of shapes. This is performed by picking without replacement in a random order a small number of subsequences to be evaluated. Each subsequence is given the same probability to be drawn since we don't want to make any assumption on $\hat{\mathcal{Z}}$: the random sampling follows a uniform distribution.

The result is an approximate set of subsequences $\hat{\mathcal{S}}$ from $\mathcal{S}$, which is an approximate distribution of the shapes contained in $\mathcal{D}$.

The approach has one single parameter that is the size of the subset to be drawn among the exhaustive set of subsequences $\mathcal{S}$. A naive approach may be to select a percentage of $|\mathcal{S}|$. With this strategy, $|\hat{\mathcal{S}}|$ would grow with $\mathcal{S}$ and $\mathcal{D}$. As we will see in the next section it is an unnecessary drawback.

6.4 Discussion on $|\hat{\mathcal{S}}|$ the number of subsequences to draw

As we have seen previously, the exhaustive number of subsequences enumerated from $\mathcal{D}$ depends on L and $|\mathcal{D}|$. In this section, we demonstrate that the number of subsequences $|\hat{\mathcal{S}}|$ to draw from $\mathcal{S}$ to determine $\hat{\mathcal{Z}}$ is independent of $|\mathcal{D}|$. This allows a considerable gain for the training phase, in particular for datasets with many time series. While $|\hat{\mathcal{S}}|$ is complex to set theoretically, we will show during the experimentation on many datasets that it can be set in the few thousands to achieve state of the art performances.

The following lemma is demonstrated:

Lemma The probability of drawing a subsequence $\hat{z}_p \in \hat{\mathcal{Z}}$ is independent of the number of time series $|\mathcal{D}|$ in $\mathcal{D}$.

Proof The objective is to establish a lower bound on the probability to draw a subsequence $z_1 \in Z = \{z_1\}$ that appears exactly one time in all the time series of $\mathcal{D}$ of class $y_1 \in Y$ and only y_1. $\mathcal{S}$ is the set of all the subsequences that we can enumerate from $\mathcal{D}$. The probability to draw a subsequence $s \in \mathcal{S}$ that is z_1 is:

$$P(s = z_1) = \frac{|\mathcal{S}_{z_1}|}{|\mathcal{S}|} \tag{6.1}$$

Where $\mathcal{S}_{z_1} \subseteq \mathcal{S}$ is the set of all the subsequences $s \in \mathcal{S}$ such as $s = z_1$. If a unique subsequence satisfies this condition by time series (most pessimistic assumption), then $|\mathcal{S}_{z_1}| = N_{y_1}$ with $N_{y_1} \setminus 0 < N_{y_1} \leq N$ is the number of time series of class y_1 in $\mathcal{D}$.

$$|\mathcal{S}| = \frac{1}{2} \sum_{i=1}^{N} L_i(L_i + 1) \tag{6.2}$$

With $L_{min} \leq L_i \leq L_{max}$:

$$\frac{N * L_{min}(L_{min} + 1)}{2} \leq |\mathcal{S}| \leq \frac{N * L_{max}(L_{max} + 1)}{2} \tag{6.3}$$

$$\frac{2 * N_{y_1}}{N * L_{min}(L_{min} + 1)} \geq P(s = z_1) \geq \frac{2 * N_{y_1}}{N * L_{max}(L_{max} + 1)} \tag{6.4}$$

$$\boxed{\frac{2 * F_{y_1}}{L_{min}(L_{min} + 1)} \geq P(s = z_1) \geq \frac{2 * F_{y_1}}{L_{max}(L_{max} + 1)}} \tag{6.5}$$

$F_{y_1} = \frac{N_{y_1}}{N}$ is the proportion of time series of class y_1 in $\mathcal{D}$, that is specific of the use case and remains constant independently of N.

We now consider the case where several subsequences $z_i \in \mathcal{Z}$ are sought, each of them being discriminant or characteristic of a class or a set of classes. The probability to draw them all is:

$$P(\mathcal{Z}) = \prod_{k=1}^{|\mathcal{Z}|} P(s = z_k) \tag{6.6}$$

$$\boxed{\prod_{k=1}^{|\mathcal{Z}|} \frac{2 * F_{z_k}}{L_{min}(L_{min} + 1)} \geq P(\mathcal{Z}) \geq \prod_{k=1}^{|\mathcal{Z}|} \frac{2 * F_{z_k}}{L_{max}(L_{max} + 1)}} \tag{6.7}$$

Where F_{z_k} is the proportion of time series for which z_k is discriminant.

Hence a lower bound on the probability to discover Z independent of the number of time series in D exists. The assumption that at most one subsequence is discriminant by time series is pessimistic. Relevant subsequences may be encountered in smaller or longer enumerated subsequences from D, possibly affected by noise or time warping, while still being discriminant enough.

6.5 Experimentation: impact of random subsampling on classification performances

Having discussed the theoretical subsequence redundancy in a time series dataset, we want now to assess the actual relevance of the approach on a large set of literature datasets. The purpose of the experimentation is to evaluate the impact of the generation of a very small set of shapelet candidates with a random picking of subsequences among S.

We use the framework described chapter 5 and the standard evaluation of the shapelet candidates using the information gain. 2000 subsequences are randomly drawn from the time series datasets. The subsequences are used to perform Ψ subsequence transformation on the time series of each dataset $\mathcal{D}$ from the UCR repository, the usual literature benchmark for time series classification. The information gain is computed for these 2000 subsequences and the best ones are conserved to train a SVM classifier, in the same way the shapelet transformation do. We compare the classification accuracies with the current leading shapelet approach, the shapelet ensemble [Bagnall et al., 2014b]. The results of this experimentation is included in an extensive experimentation presented chapter 7. For more details on the setup of the experimentation, we invite the reader to refer to chapter 7, section 7.2. Also, the raw results and a statistical assessment of the classification performances are available chapter 7, section 7.2.4. We present here the important details to expose the relevance of the random picking of the subsequences.

The comparison of the classification accuracies is shown figure 6.4. While the shapelet ensemble has an advantage over the randomized shapelet discovery in terms of classification accuracy, the results are rather close for a small fraction of the computational complexity of the shapelet ensemble. Our approach evaluates only 2000 subsequences while the shapelet ensemble evaluates up to 10^9 subsequences on the largest dataset. Figure 6.6 shows the exhaustive number of subsequences by datasets versus the constant number of subsequences drawn with our approach. On all the datasets, the number of subsequences and thus the number of computations is drastically reduced. On the largest datasets, the difference is considerable. The largest dataset of the UCR repository, Star Light Curves, the exhaustive number of subsequence is close to 525.000.000. The shapelet ensemble has 98% while the randomized approach has 96% with 2000 subsequences.

One drawback of the randomization is the variability of the results. We report the standard deviation of the randomized shapelet discovery figure 6.5. The standard deviation is very low: the standard deviation on most datasets is lower than 2%, few of them present more than 5% of standard deviation with a maximum of 8%.

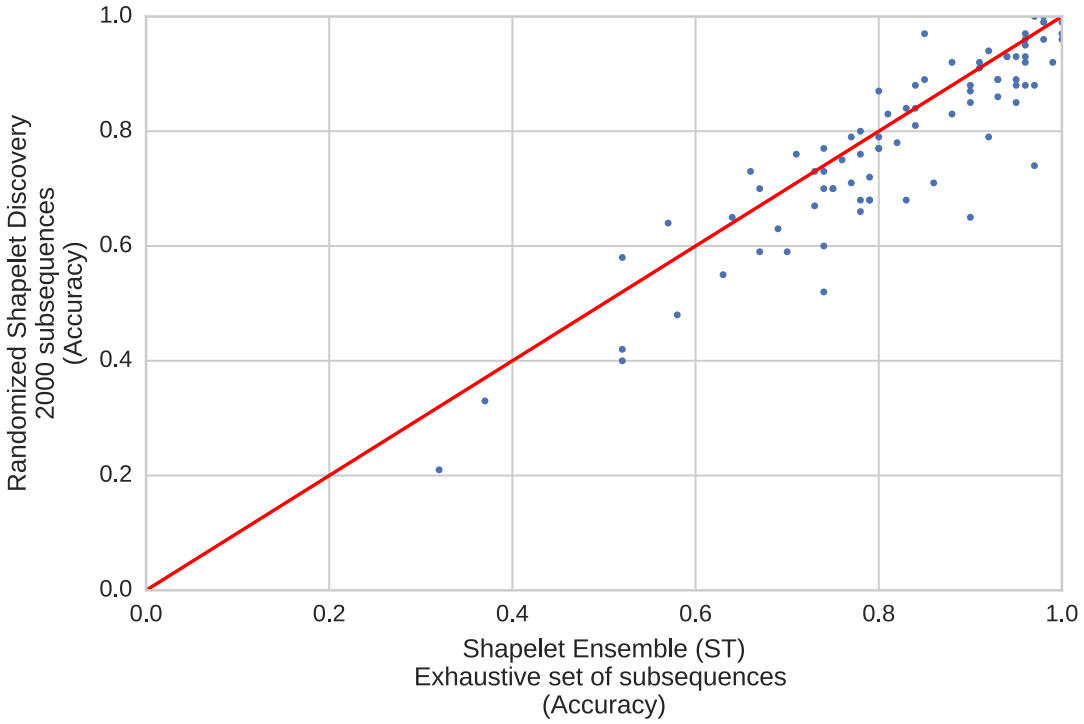


Figure 6.4: Classification accuracies of randomized shapelet discovery (using 2000 subsequences) versus the current leading shapelet approach (shapelet ensemble)

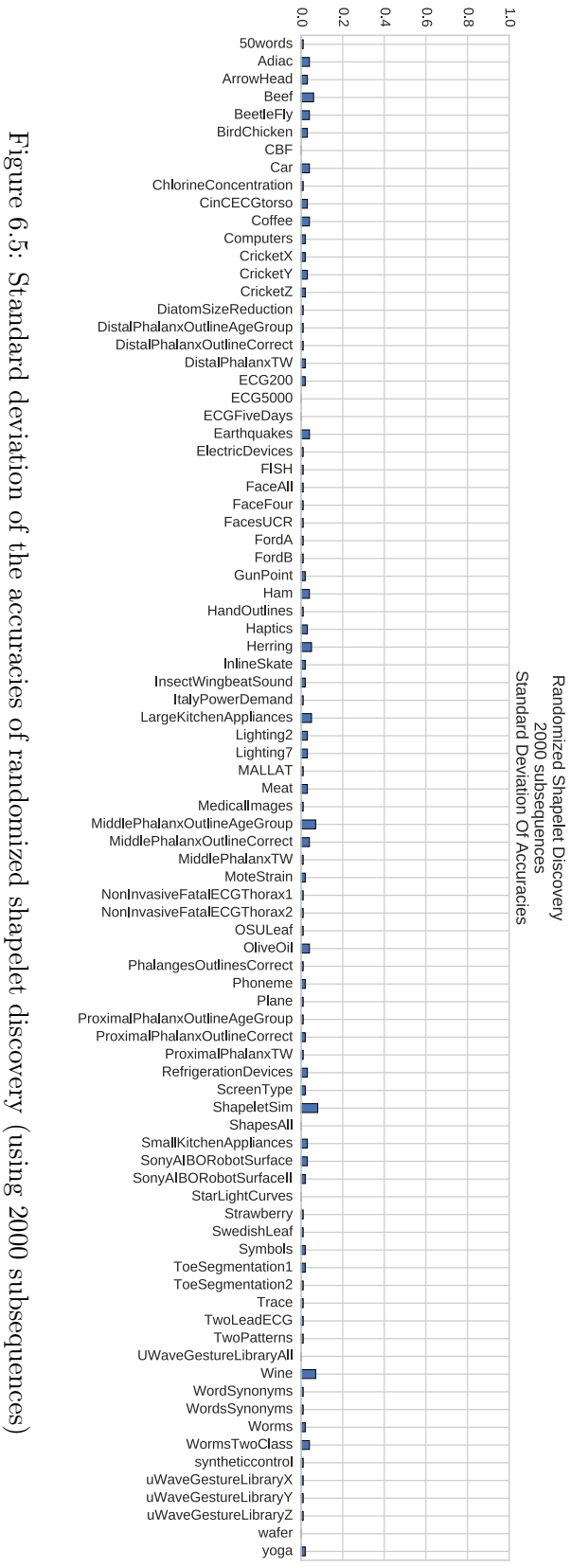


Figure 6.5: Standard deviation of the accuracies of randomized shapelet discovery (using 2000 subsequences)

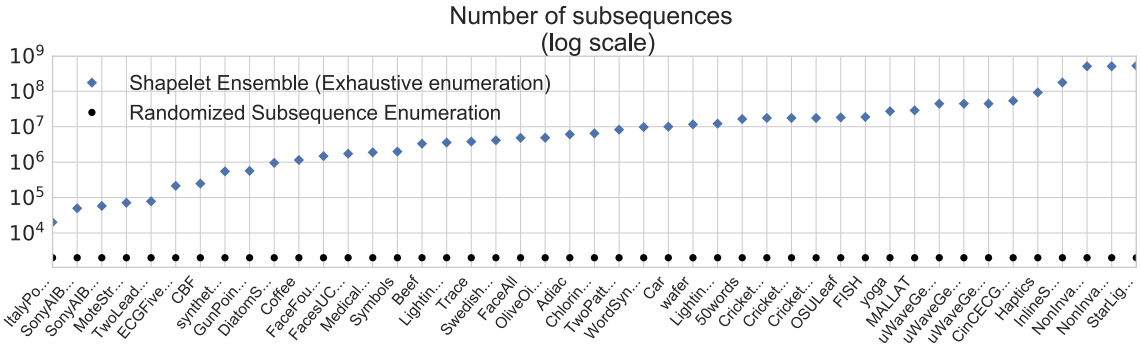


Figure 6.6: Number of subsequences evaluated with the randomization of the enumeration of the subsequences and the exhaustive shapelet discovery (log scale). Our approach enumerates a constant number of subsequences over the datasets with comparable classification accuracies than the shapelet ensemble that generates subsequences sets several orders of magnitude larger

6.6 Conclusions

In this chapter[1], we have shown that many subsequences in a time series dataset are redundant. We have demonstrated that the number of subsequences to draw from a time series dataset is independent of the number of time series in the dataset. This makes our approach scalable.

To overcome the redundancy and to benefit from these discoveries, we have proposed to draw randomly a very small number of subsequences among the exhaustive set of subsequences to reduce dramatically the time required to discover the time series shapelets. We have experimented the approach on all the datasets of the UCR repository, the usual literature benchmark for time series classification. We have shown the relevance of the approach in terms of classification accuracy while the standard deviation caused by the randomization remains very low. The proposed approach preserves the properties of the shapelet and can be combined with many proposed extensions. Speed-ups proposed in the literature to decrease the time complexity of the shapelet can be easily combined with our approach to go further in the time reduction of the discovery process.

Other approaches based on the randomization of the subsequence enumeration have provided good results [Wistuba et al., 2015, Karlsson et al., 2016].

The results of this chapter have another implication: we have demonstrated that we can drastically reduce the dimensionality of the feature vector $X^{\hat{\mathcal{Z}}}$. Since the number of features decreases from millions or billions to thousands, we can now consider the use of advanced techniques to select relevant set of subsequences to perform time series classification. This idea is developed in the next chapter.

[1] The ideas developed in this chapter have been published in [Renard et al., 2015]

Chapter 7

EAST-Representation: Casting the Discovery Into a Feature Selection Problem

In the previous chapter, we have shown that we can reduce by several orders of magnitude the computational complexity of the calculation related to the subsequence transformation Ψ to obtain a relevant feature vector $X^{\hat{\mathcal{S}}}$ while preserving the classification performances. The reduced dimensionality of the feature vector allows us to make use of state of the art machine learning techniques to discover $\hat{Z}$ from $\hat{\mathcal{S}}$ and produce the feature vector $X^{\hat{Z}}$ to train a classifier and perform time series classification.

In this chapter, we cast the discovery of a discriminant motif-based representation into a classical feature selection problem. The objectives are to avoid unnecessary subsequence redundancy while promoting complementarity of the subsequences.

7.1 Cast the discriminant motif-based representation discovery into a feature selection problem

In the last chapter, we have shown experimentally the relevance of the subsequence subsampling to decrease dramatically the dimension of the set of subsequences $\mathcal{S}$ while preserving the classification performances. Also, we have demonstrated the scalability of our framework to discover a discriminant motif-based representation based on the subsequences transformation principle Ψ. The resulting feature vector $X^{\hat{S}}$ holds a few thousands features in comparison to the billions of features in the exhaustive subsequence enumeration case. This enables us to make use of advanced machine learning techniques to discover $\hat{Z}$ among $\hat{S}$, with more expressiveness to take into account relationships between subsequences during the discovery.

To determine $\hat{Z}$ we propose to combine the random drawing of subsequences to produce $\hat{S}$ with a feature selection stage to learn $\hat{Z}$, a relevant set of subsequences for the classification task. We describe below the steps to build the representation.

Step 1: random sub-sampling to handle subsequence redundancy

The first step of the representation relies on the random sampling $\hat{S}$, among all the subsequences S, as discussed in the previous chapter. Each subsequence $s_{T_{n,m}}$ is given with the same probability to be picked, whatever its time series, variable, position and length.

Step 2: learning the representation by selecting a set of discriminant subsequences

Once $\hat{S}$ is drawn, we need to discover the set $\hat{Z} \subset \hat{S}$ that maximizes the classification performance. We have formalized in chapter 5 the problem as a standard feature-space classification task, and in chapter 6 we made the dimension of the problem tractable for common state of the art feature selection algorithms.

Here, to reduce $\hat{S}$ to $\hat{Z}$ and derive a feature space $X^{\hat{Z}}$ relevant to train a classifier $f(X_n^{\hat{Z}}) \mapsto y(T_n)$ we use the feature vector formalization of the problem to exploit classical feature selection approaches. They allow an efficient identification of the relevant attributes in a feature space with respect to a classification task. Advanced feature selection techniques offer the possibility to discover both single discriminant subsequence and sets of subsequences where each subsequence can be characteristic of a class or a subclass, while the whole set is discriminant. Numerous feature selection techniques exist, the approaches used in this work are presented in the experimentation section.

The overall principle of the proposed approach to discover a discriminant motif-based representation is summarized in figure 7.2. The classifier is trained on $X^{\hat{Z}}$. The result of the training is both a set $\hat{Z}$ of patterns and a classifier f. To perform the classification of new instances, time series are transformed into a feature vector according to $\hat{Z}$ using the subsequence transformation principle Ψ and the classification is performed with f.

The resulting representation is called EAST.

1: $\hat{S} \leftarrow$ Draw q subsequences from time series from D
2: $X^{\hat{S}} \leftarrow$ Perform subsequences transformations Ψ on time series from D using subsequences from $\hat{S}$
3: $X^{\hat{Z}}, \hat{Z} \leftarrow$ Perform feature selection on $X^{\hat{S}}$ with respect to labels of Y

Figure 7.2: Learning of the EAST representation in 3 key steps: random sub-sampling, subsequence transformation Ψ and feature selection

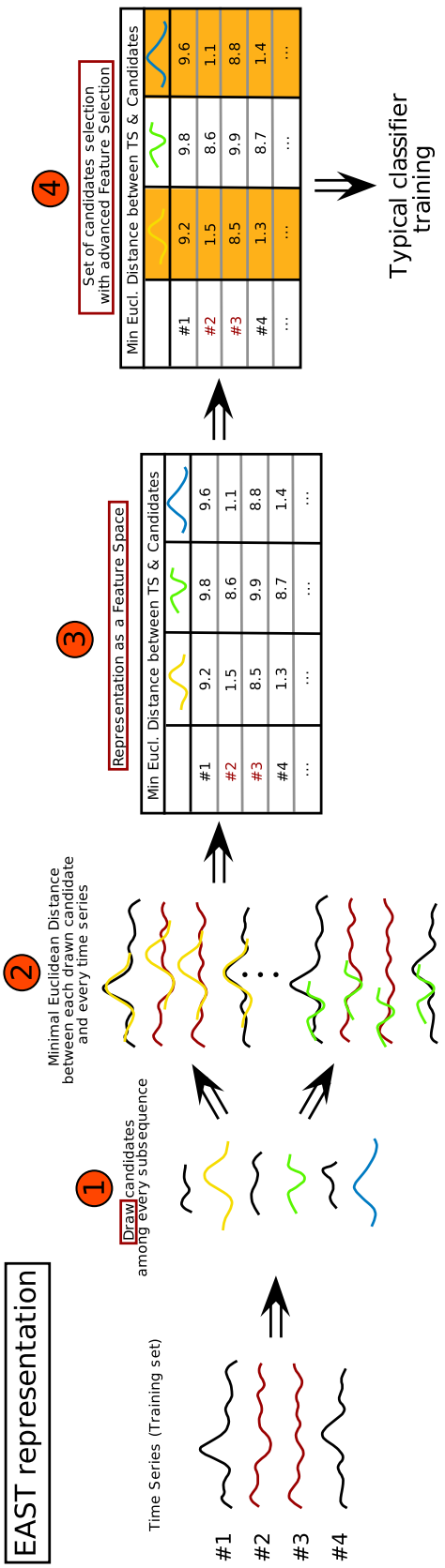


Figure 7.1: EAST principle workflow. After a drastic subsequences sub-sampling (1), the distances between subsequences and time series (2) form a feature space of reasonable size (3) on which advanced feature selection techniques can be applied to discover discriminant set of subsequences (4)

7.2 Experimentation

7.2.1 Objective

The objective of the experimentation is to evaluate the relevance of advanced feature selection in a standard feature space for the temporal pattern discovery over the classical selection scheme used by the shapelet (usually the information gain). The classification performances are observed with several configurations. For this purpose, EAST is **instantiated** with several feature selection approaches as well as different classifiers for various values q of subsequences drawn.

7.2.2 Setup

With EAST, the feature selection stage is open to any approach. The feature selection can even be skipped and the selection performed only by the classifier. For the experimentation we use a small set of feature selection approaches. Feature selection is an established field: we do not contribute to it but instead we rely on it. Also, we don't advocate one approach is better than another.

Feature selection methods are usually classified into three groups: *filters*, *wrappers* and *embedded methods* [Guyon and Elisseeff, 2003]. We avoid *wrappers* because they are too time consuming. We focus on *filters* and *embedded* methods. We select the Randomized Logistic Regression (RLR), the Random Forest (RF), the Robust Feature Selection (RBF) and a SVM with a linear kernel as multivariate feature selection approaches. These approaches are able to learn combinations or sets of features (*i.e.* subsequences). As univariate feature selection approaches we select the information gain (InfoGain) as used in [Renard et al., 2015] and the F-test (Fscore). Typically the shapelet approach makes use of the information gain with an independent evaluation of the subsequences, which is unable to learn sets or combinations of features. For each approach we use a SVM with a linear kernel to perform the classification. The SVM is trained using the features retained by the feature selection approaches. The Random Forest is the only one to be used for both the feature selection and the classification.

These approaches are tested on a small fraction randomly drawn from all the subsequences. The same number of subsequences is picked for each EAST instances. For the random draw of $\hat{S}$ several values $q = |\hat{S}|$ are tested: $q \in \{10, 50, 100, 500, 1000, 2000, 5000\}$.

For univariate datasets, the results are compared with the current leading shapelet approach, the shapelet ensemble (ST) [Bagnall et al., 2014b]. The authors state that shapelet ensemble performs identically or better than other shapelet approaches. We also compare the results with the Learning Shapelet (LS) approach [Grabocka et al., 2014] and the Fast Shapelet (FS) [Rakthanmanon and Keogh, 2013]. We reproduce here their results.

For multivariate datasets, results are compared with the SMTS approach [Baydogan and Runger, 2015] and a nearest neighbor with DTW distance (NNDTW).

A strict evaluation protocol is required to assess the EAST instances because they contain a random generation step. We rely on the evaluation protocol proposed in [Arcuri and Briand, 2014] for a proper way to analyze the performances of randomized algorithms. Each single configuration of the EAST instances is reproduced 10 times to evaluate the variability. 10 times is the minimum recommended in [Arcuri and Briand, 2014], we didn't perform more because the current number of tested configurations required weeks of calculations on a cluster.

The raw classification accuracies are presented in appendix in table 7.10 for univariate datasets $q = 2000$ and in table 7.9 for multivariate datasets with $q = 5000$. For randomized algorithms, the mean accuracy for each configuration is shown. The results for other values of q are available in the folder *results* of the website of this work [Renard et al., 2016b].

To evaluate the statistical significance of the performances between two randomized algorithms we use a non parametric paired Wilcoxon test to assess if the differences between their classification accuracies are centered around 0. The null hypothesis H_0 of this test is the absence of difference. The $p-value$ is conserved to get the probability to reject H_0 while the performances are actually identical (ie. *type I error*). We set up $\alpha = 0.05$ as the limit for the *type I error* in order to determine if the differences are statistically significant or not. To compare a randomized algorithm with a deterministic one (here the shapelet ensemble), we use a similar procedure with a non parametric one sample Wilcoxon test. In order to aggregate the results over all the datasets we use the procedure recommended in [Arcuri and Briand, 2014]. At the beginning of the procedure every configuration is given a score set up to 0. For each Wilcoxon test between two algorithms, if the difference is significant the score of the best performing algorithm is increased by 1, the other is decreased by 1. The result of the Wilcoxon tests are presented figure 7.4a for the univariate datasets and figure 7.7a for the multivariate datasets. Scores are sorted from the best performing overall the tested algorithms (with the highest score) to the worst performing (with the lowest score).

Finally, we compute the critical difference with the Nemenyi test with $\alpha = 0.05$ for the average ranks of the approaches on the datasets tested. Figure 7.3 shows the Nemenyi test with the critical difference for the univariate datasets and figure 7.6 for the multivariate datasets. Connected approaches don't have significantly different classification performances according to the performed Nemenyi test [Demsar, 2006]. Figure 7.4b and figure 7.7b show the average ranks overall the approaches for the univariate and multivariate datasets respectively.

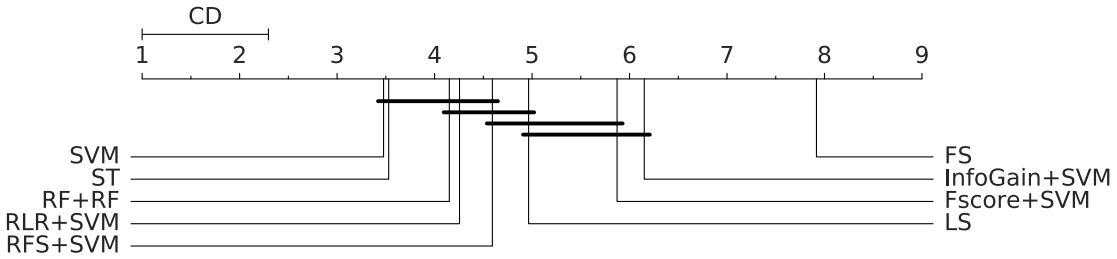


Figure 7.3: Comparison of the approaches with the Nemenyi test for **univariate datasets**. Groups of approaches not significantly different ($\alpha = 0.05$) are connected. CD is the critical difference

7.2.3 Datasets

The experimentation performed for this work uses two repositories of datasets to evaluate classification performances both for univariate and multivariate time series:

- The classical UCR repository for **univariate** time series classification framework. All the 86 datasets of this repository are used.
- The **multivariate** time series datasets gathered in [Baydogan and Runger, 2015]. The 15 datasets are used.

7.2.4 Results

Univariate datasets

For the UCR univariate datasets, the classification performances of the EAST instances based on multivariate feature selection are significantly similar to the ones obtained by the shapelet ensemble (figures 7.3 & 7.4a). These results are obtained with 2000 drawn subsequences that is an infinitesimal fraction of subsequences evaluated by the exhaustive shapelet ensemble: the largest tested dataset reaches 5.10^8 subsequences.

The relevance of multivariate feature selection approaches over univariate selection is shown by the experimentation. With the same number of subsequences drawn EAST instances based on multivariate feature selection systematically outperforms instances with univariate selection such as the classical shapelet evaluation procedure based on the information gain.

These results can be observed graphically in figures 7.5a and 7.5b.

Multivariate datasets

As for the univariate datasets, the classification performances of the best EAST instances based on multivariate feature selection are significantly similar to the reference that is the

	SVM	ST	**RF+RF**	**RLR+SVM**	**RFS+SVM**	LS	InfoGain+SVM	Fscore+SVM	FS
Score	**6**	6	**4**	**2**	**1**	-3	-4	-4	-8

(a) Wilcoxon tests scores. Higher is better

	SVM	ST	**RF+RF**	**RLR+SVM**	**RFS+SVM**	LS	Fscore+SVM	InfoGain+SVM	FS
Mean Rank	**3.5**	3.5	**4.2**	**4.3**	**4.6**	5.0	5.9	6.2	7.9

(b) Mean rank over the datasets

Figure 7.4: Statistics on classification accuracies for the 86 univariate datasets from the UCR repository. Results for instances of our approach are highlighted. 2000 subsequence candidates have been drawn for instances of our approach

SMTS approach (figures 7.6 & 7.7a) for $q = 5000$ (instead of 2000 because of the additional dimensions). Despite the gap in the average ranks (Fig. 7.7b), the information gain is not found significantly worse, which may be caused by a smaller number of datasets than the UCR repository. However the EAST instances based on multivariate feature selection almost systematically outperform the univariate selection over the multivariate datasets (figure 7.8a).

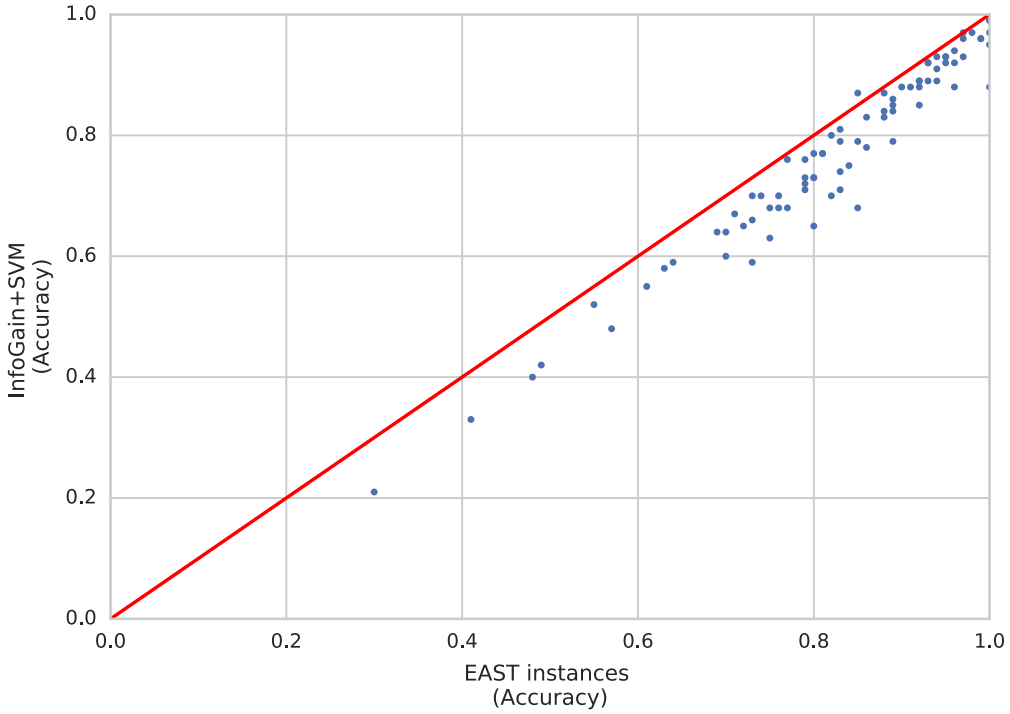
These results can be observed graphically in figures 7.8a and 7.8b.

Influence of the number of drawn subsequences

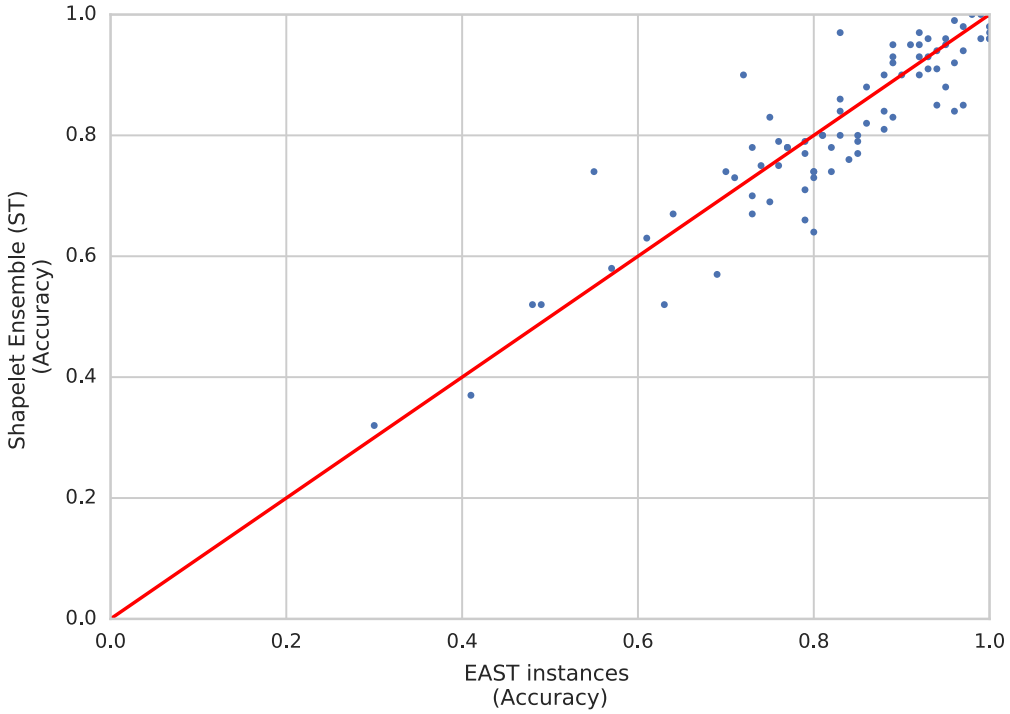
The parameter q is obviously critical, but only until a certain point as shown figure 7.11. Accuracies quickly reach a plateau: on UCR datasets this plateau starts from 500 to 1000 candidates. The mean rank of the EAST instances over the 86 UCR datasets together with the Nemenyi test can be observed figure 7.12.

Performance variability

Over the UCR univariate datasets for $q = 2000$, the EAST instances based on multivariate features selection have a mean standard deviation for their classification accuracies around 1% (from 0.8% for the Random Forest to 1.6% for the Randomized Logistic Regression



(a) Best EAST instance versus information gain selection



(b) Best EAST instance versus shapelet transform

Figure 7.5: Comparison of accuracies over the 86 UCR datasets

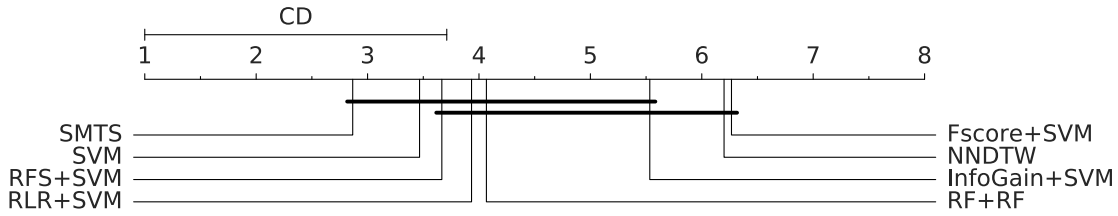


Figure 7.6: Comparison of the approaches with the Nemenyi test for **multivariate datasets**. Groups of approaches not significantly different ($\alpha = 0.05$) are connected. CD is the critical difference

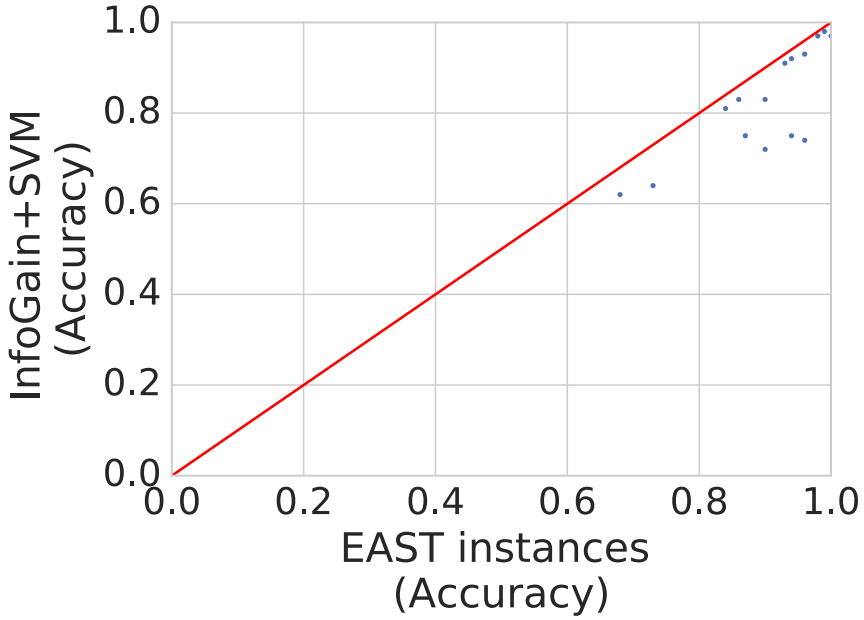
	SVM	SMTS	**RFS+SVM**	**RLR+SVM**	**RF+RF**	InfoGain+SVM	NNDTW	Fscore+SVM
Score	**3**	3	**3**	**3**	**3**	-5	-5	-5

(a) Wilcoxon tests scores. Higher is better

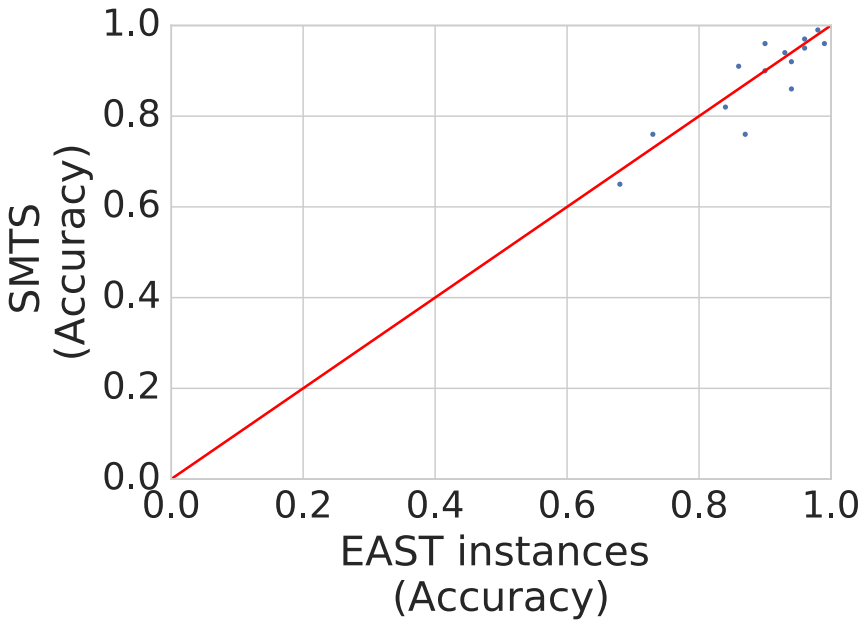
	SMTS	**SVM**	**RFS+SVM**	**RLR+SVM**	**RF+RF**	InfoGain+SVM	NNDTW	Fscore+SVM
Mean Rank	2.9	**3.5**	**3.7**	**3.9**	**4.1**	5.5	6.2	6.3

(b) Mean rank over the datasets

Figure 7.7: Statistics on classification accuracies for 15 multivariate datasets from the [Baydogan and Runger, 2015] repository. Results for instances of our approach are highlighted. 5000 subsequence candidates have been drawn for instances of our approach



(a) Best EAST instance versus information gain selection



(b) Best EAST instance versus SMTS

Figure 7.8: Comparison of accuracies over the multivariate datasets

dataset	**EAST**-SVM	**EAST**-RF+RF	**EAST**-RLR+SVM	**EAST**-RFS+SVM	SMTS	NNDTW	InfoGain+SVM	Fscore+SVM
AUSLAN	0.91	**0.96**	0.76	0.91	0.95	0.76	0.74	0.01
ArabicDigits	0.88	0.81	0.9	0.71	**0.96**	0.91	0.72	0.62
CMUsubject16	**1.0**	0.97	0.97	0.97	**1.0**	0.93	0.97	0.55
CharacterTrajectories	0.98	0.98	0.98	0.98	**0.99**	0.96	0.97	0.69
ECG	0.83	0.83	0.82	0.84	0.82	**0.85**	0.81	0.83
JapaneseVowels	0.96	0.92	0.96	0.95	**0.97**	0.65	0.93	0.71
LP1	0.9	**0.94**	0.85	0.93	0.86	0.72	0.75	0.79
LP2	0.66	0.72	0.62	0.73	**0.76**	0.53	0.64	0.70
LP3	**0.87**	0.75	0.83	0.73	0.76	0.5	0.75	0.84
LP4	0.82	**0.9**	0.85	0.87	**0.9**	0.81	0.83	0.88
LP5	0.58	**0.68**	0.63	0.59	0.65	0.52	0.62	0.62
Libras	0.85	0.82	0.86	0.86	**0.91**	0.8	0.83	0.73
PenDigits	**0.94**	0.93	**0.94**	**0.94**	0.92	0.91	0.92	0.10
Wafer	**0.99**	0.97	0.98	**0.99**	0.96	0.98	0.98	0.97
uWave	0.93	0.93	0.93	0.91	**0.94**	0.93	0.91	0.85

Figure 7.9: Classification accuracies on the multivariate datasets

approach). In the same conditions, the information gain has 2% of standard deviation in average. On the multivariate datasets for $q = 5000$, the average standard deviation are comparable around 1% for multivariate feature selection and 3% for the information gain. Complete results are available in additional material [Renard et al., 2016b].

Computation time

For the approaches used in this experimentation, most of the time is spent in the distance calculations between each subsequence enumerated and the time series. This fact is illustrated figure 7.13 and is especially true for large datasets and with increasing values of q: the time spent in the feature selection becomes insignificant. We use this specificity to compare the time complexity of the approaches and avoid implementation or hardware bias. EAST enumerates a fixed number of subsequences, in this work the maximal value is $q = 5000$. The typical shapelet approach performs an exhaustive enumeration. Figure 7.14 shows that the exhaustive shapelet discovery (ST) evaluates subsequences sets with several orders of magnitude larger than the EAST approach while having comparable classification performances. For the datasets used in the experimentation, the exhaustive number of subsequences to extract varies from thousands to 10^9. For all the datasets and for significantly similar classification performances our proposition uses $q = 2000$ subsequences. On the larger dataset this is less than 0.0002% of the exhaustive number of subsequences.

7.3 Discussion

A shapelet has been defined as a "primitive" to perform time series classification. In concrete terms, it is the association of a subsequence together with a metric and an aggregation

	EAST-SVM	**EAST**-RF+RF	**EAST**-RLR+SVM	**EAST**-RFS+SVM	ST	LS	FS	InfoGain+SVM	Fscore+SVM
50words	**0.79**	0.73	0.77	0.78	0.71	0.73	0.48	0.76	0.65
Adiac	0.73	0.73	0.71	0.72	**0.78**	0.52	0.59	0.66	0.38
ArrowHead	0.8	0.7	0.79	0.75	0.74	**0.85**	0.59	0.77	0.78
Beef	0.72	0.61	0.65	0.71	**0.9**	0.87	0.57	0.65	0.54
BeetleFly	0.88	**0.92**	0.9	0.9	0.9	0.8	0.7	0.85	0.86
BirdChicken	0.78	0.82	0.83	0.8	0.8	0.8	0.75	0.79	**0.84**
CBF	**1.0**	0.99	**1.0**	**1.0**	0.97	0.99	0.94	**1.0**	**1.0**
Car	0.85	0.77	0.88	0.89	**0.92**	0.77	0.75	0.79	0.85
ChlorineConcentration	**0.73**	0.65	0.62	0.66	0.7	0.59	0.55	0.59	0.6
CinCECGtorso	0.88	0.82	0.89	0.89	**0.95**	0.87	0.86	0.85	0.9
Coffee	**1.0**	0.93	0.95	0.95	0.96	**1.0**	0.93	0.97	0.94
Computers	0.56	0.7	0.59	0.61	**0.74**	0.58	0.5	0.6	0.62
CricketX	**0.79**	0.72	0.78	0.75	0.77	0.74	0.48	0.71	0.68
CricketY	0.77	0.72	0.75	0.71	**0.78**	0.72	0.53	0.68	0.66
CricketZ	**0.79**	0.76	**0.79**	0.74	**0.79**	0.74	0.46	0.72	0.71
DiatomSizeReduction	0.95	0.92	0.96	0.95	0.92	**0.98**	0.87	0.94	0.83
DistalPhalanxOutlineAgeGroup	0.78	**0.85**	0.77	0.74	0.77	0.72	0.65	0.79	0.82
DistalPhalanxOutlineCorrect	0.75	**0.82**	0.78	0.73	0.78	0.78	0.75	0.8	0.79
DistalPhalanxTW	0.74	**0.79**	0.72	0.73	0.66	0.63	0.63	0.73	0.78
ECG200	**0.89**	0.82	0.86	0.88	0.83	0.88	0.81	0.84	0.84
ECG5000	0.92	**0.94**	0.93	0.93	**0.94**	0.93	0.92	0.93	**0.94**
ECGFiveDays	**1.0**	**1.0**	**1.0**	**1.0**	0.98	**1.0**	**1.0**	**1.0**	**1.0**
Earthquakes	0.81	**0.82**	0.71	0.76	0.74	0.74	0.71	0.7	0.66
ElectricDevices	0.69	0.74	0.69	0.65	**0.75**	0.59	0.58	0.7	0.6
FISH	0.96	0.91	0.94	0.94	**0.99**	0.96	0.78	0.92	0.89
FaceAll	0.77	0.75	0.77	0.77	**0.78**	0.75	0.63	0.76	0.73
FaceFour	0.96	0.96	**0.97**	**0.97**	0.85	**0.97**	0.91	**0.97**	**0.97**
FacesUCR	**0.94**	0.92	0.93	0.93	0.91	**0.94**	0.71	0.91	0.9
FordA	0.92	0.92	0.89	0.9	**0.97**	0.96	0.79	0.88	0.88
FordB	0.88	0.87	0.87	0.86	0.81	**0.92**	0.73	0.83	0.85
GunPoint	0.97	0.97	0.98	0.99	**1.0**	**1.0**	0.95	0.96	0.97
Ham	0.69	**0.75**	0.66	0.68	0.69	0.67	0.65	0.63	0.67
HandOutlines	0.86	0.89	0.88	0.84	**0.93**	0.48	0.81	0.86	0.87
Haptics	0.47	0.48	0.45	0.43	**0.52**	0.47	0.39	0.4	0.41
Herring	0.6	0.57	0.63	0.64	**0.67**	0.62	0.53	0.59	0.64
InlineSkate	0.37	0.41	0.34	0.34	0.37	**0.44**	0.19	0.33	0.34
InsectWingbeatSound	0.59	0.61	0.59	0.55	**0.63**	0.61	0.49	0.55	0.58
ItalyPowerDemand	0.95	0.95	0.94	0.94	0.95	**0.96**	0.92	0.93	0.93
LargeKitchenAppliances	0.82	0.83	0.81	0.81	**0.86**	0.7	0.56	0.71	0.79
Lighting2	0.76	0.73	0.75	0.8	0.74	**0.82**	0.7	0.73	0.73
Lighting7	0.77	0.74	**0.8**	0.79	0.73	0.79	0.64	0.73	0.72
MALLAT	0.97	**0.99**	0.97	0.91	0.96	0.95	0.98	0.96	0.93
Meat	0.89	0.92	0.92	**0.94**	0.85	0.73	0.83	0.89	0.91
MedicalImages	0.72	0.69	**0.73**	0.71	0.67	0.66	0.62	0.7	0.69
MiddlePhalanxOutlineAgeGroup	0.7	**0.8**	0.7	0.65	0.64	0.57	0.55	0.65	0.77
MiddlePhalanxOutlineCorrect	0.71	0.76	0.65	0.67	**0.79**	0.78	0.73	0.68	0.64
MiddlePhalanxTW	0.59	**0.63**	0.59	0.57	0.52	0.51	0.53	0.58	**0.63**
MoteStrain	0.88	0.88	0.86	0.87	**0.9**	0.88	0.78	0.87	0.85
NonInvasiveFatalECGThorax1	0.91	0.88	0.91	0.9	**0.95**	0.26	0.71	0.88	0.89
NonInvasiveFatalECGThorax2	0.92	0.89	0.92	0.92	**0.95**	0.77	0.75	0.89	0.9
OSULeaf	0.83	0.76	0.8	0.78	**0.97**	0.78	0.68	0.74	0.68
OliveOil	0.89	0.87	**0.9**	**0.9**	**0.9**	0.17	0.73	0.88	**0.9**
PhalangesOutlinesCorrect	0.78	**0.84**	0.77	0.78	0.76	0.76	0.74	0.75	0.72
Phoneme	0.27	0.3	0.25	0.26	**0.32**	0.22	0.17	0.21	0.26
Plane	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	0.99	0.99
ProximalPhalanxOutlineAgeGroup	0.8	0.83	0.79	0.8	0.84	0.83	0.78	0.81	**0.85**
ProximalPhalanxOutlineCorrect	0.83	0.86	0.84	0.84	**0.88**	0.85	0.8	0.83	0.81
ProximalPhalanxTW	0.77	0.81	0.77	0.75	0.8	0.78	0.7	0.77	**0.82**
RefrigerationDevices	0.55	0.57	0.51	0.51	**0.58**	0.51	0.33	0.48	0.53
ScreenType	0.41	0.49	0.42	0.41	**0.52**	0.43	0.41	0.42	0.42
ShapeletSim	0.99	0.99	**1.0**	0.99	0.96	0.95	**1.0**	0.88	0.99
ShapesAll	0.87	**0.88**	0.85	0.84	0.84	0.77	0.58	0.84	0.79
SmallKitchenAppliances	0.67	**0.85**	0.69	0.66	0.79	0.66	0.33	0.68	0.66
SonyAIBORobotSurface	0.91	0.93	0.96	0.95	0.84	0.81	0.69	0.88	**0.97**
SonyAIBORobotSurfaceII	0.92	0.88	**0.93**	**0.93**	**0.93**	0.88	0.79	0.89	0.9
StarLightCurves	0.96	0.97	0.96	0.96	**0.98**	0.95	0.92	0.96	0.96
Strawberry	0.91	0.93	0.93	0.92	**0.96**	0.91	0.9	0.92	0.93
SwedishLeaf	0.92	0.9	0.92	0.91	**0.93**	0.91	0.77	0.89	0.89
Symbols	0.94	**0.95**	0.91	0.91	0.88	0.93	0.93	0.92	0.9
ToeSegmentation1	0.95	0.94	0.93	0.93	**0.96**	0.93	**0.96**	0.93	0.94
ToeSegmentation2	0.93	0.91	0.93	0.93	0.91	0.92	0.69	0.92	**0.94**
Trace	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**
TwoLeadECG	0.98	0.92	0.95	0.96	**1.0**	**1.0**	0.92	0.97	0.94
TwoPatterns	**1.0**	**1.0**	**1.0**	0.99	0.96	0.99	0.91	0.95	0.85
UWaveGestureLibraryAll	**0.97**	0.95	**0.97**	0.96	0.94	0.95	0.79	0.93	0.95
Wine	0.85	0.74	0.6	0.72	0.8	0.5	0.76	**0.87**	0.62
WordSynonyms	**0.69**	0.64	0.67	0.68	0.57	0.61	0.43	0.64	0.56
Worms	0.54	0.54	0.53	0.55	**0.74**	0.61	0.65	0.52	0.56
WormsTwoClass	0.75	0.72	0.73	0.73	**0.83**	0.73	0.73	0.68	0.68
syntheticcontrol	**1.0**	0.99	0.99	0.99	0.98	**1.0**	0.91	0.99	0.99
uWaveGestureLibraryX	**0.81**	0.8	0.79	0.77	0.8	0.79	0.69	0.77	0.77
uWaveGestureLibraryY	0.71	0.71	0.71	0.68	**0.73**	0.7	0.6	0.67	0.69
uWaveGestureLibraryZ	0.73	**0.76**	0.73	0.71	0.75	0.75	0.64	0.7	0.72
wafer	**1.0**	0.99	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	0.99
yoga	**0.86**	0.85	0.79	0.8	0.82	0.83	0.7	0.78	0.76

Figure 7.10: Classification accuracies on the UCR univariate datasets

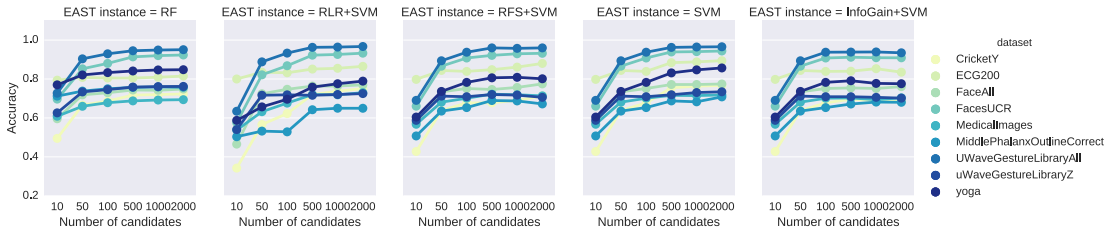


Figure 7.11: Accuracy with respect to the number of drawn candidate subsequences q

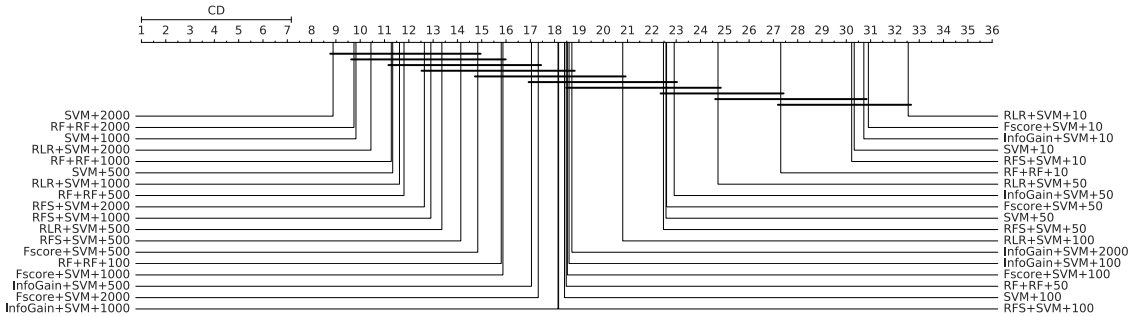


Figure 7.12: Comparison with the Nemenyi test and the mean rank of the EAST instances over the values of q, the number of drawn subsequences

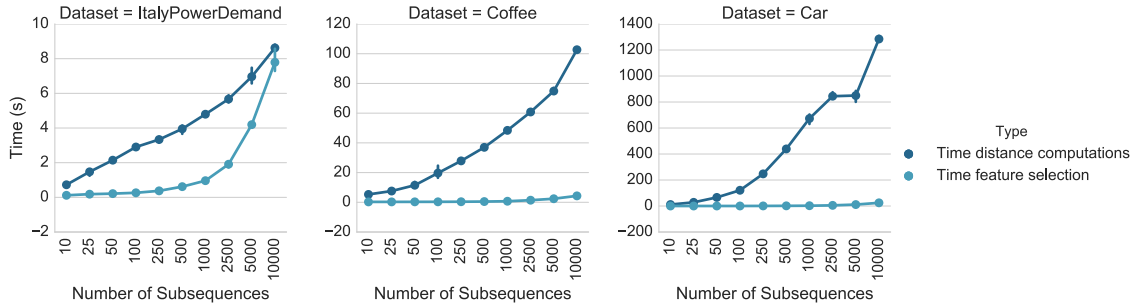


Figure 7.13: Time spent in the distance calculations vs. Time spent in the feature selection for EAST. For small datasets (ItalyPowerDemand), the feature selection requires a similar amount of time than the distance calculations. For larger datasets (Coffee, Car) feature selection becomes insignificant in front of distance computations. We use this specificity to compare the time complexity of the approaches based on the number of distance computations and avoid implementation or hardware bias

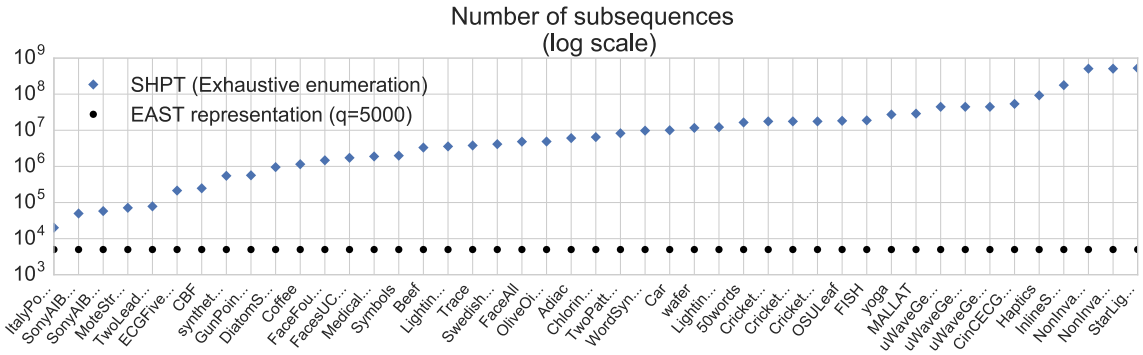


Figure 7.14: Number of subsequences evaluated by EAST and the exhaustive shapelet discovery (log scale). EAST enumerates a constant number of subsequences over the datasets with comparable classification accuracies than the shapelet ensemble that generates subsequences sets several orders of magnitude larger

function: usually the minimum of the Euclidean distance between a subsequence and time series.

Because of the very high number of subsequences of all lengths that can be enumerated from a set of time series, the shapelet discovery has been mostly seen as an iterative process where each subsequence is evaluated independently from one another, with a fast filter, typically the information gain. Despite heuristics, the discovery remains very long, especially on large datasets with long time series. Also, the independent evaluation of the subsequences can be a severe limitation.

From our point of view, a breakthrough was the discovery of the high redundancy of the subsequences in time series datasets [Gordon et al., 2012, Renard et al., 2015, Grabocka et al., 2015]. A drastic subsampling among the subsequences doesn't affect the time series classification performances while the discovery process is orders of magnitude faster. In this work, we have investigated further and demonstrated that the probability to draw a relevant subsequence is not related with the number of time series in a dataset. The immediate consequence is that a fixed number of subsequences around a few thousands is enough to get state-of-the-art classification performances using shapelet principle.

As stated previously in the proposition, the subsequences and more precisely their distances to the time series can be seen as a feature space describing the time series. This leads to the second advantage of the drastic reduction of the number of subsequences to consider. With a few thousands of features (ie. subsequences) it is imaginable to make use of the state-of-the-art machine learning techniques to discover relevant subsequences among the time series and open new possibilities. As a first instantiation of the EAST representation, we have shown in the experimentation that multivariate feature selection techniques or robust classifiers applied on such a feature space lead to very good performances.

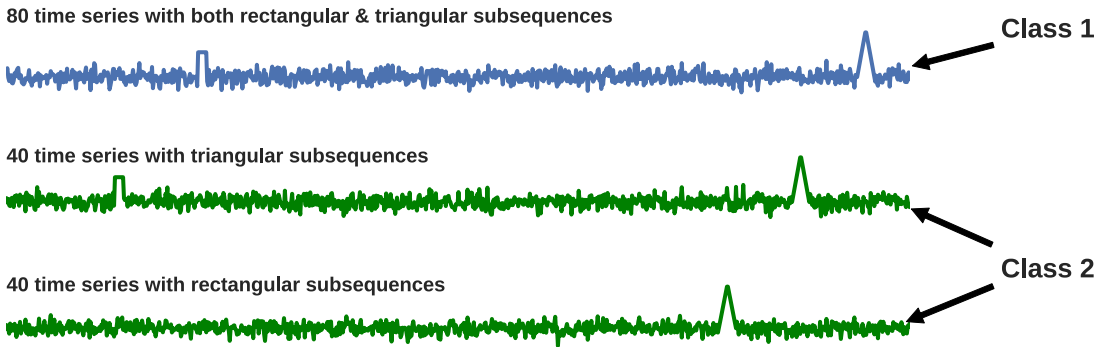
For a specific time series classification task, the practitioner now have the opportunity

to discover the properties and the relationships among the subsequences of its dataset. For instance, once the distances from subsequences to time series have been computed, it is possible to execute several feature selection techniques with distinct properties using cross-validation. By properties we mean for instance the ability to discover linear or non-linear relationships between subsequences, to be robust to noise and strong correlations or to discover complementary subsequences to handle distortions.

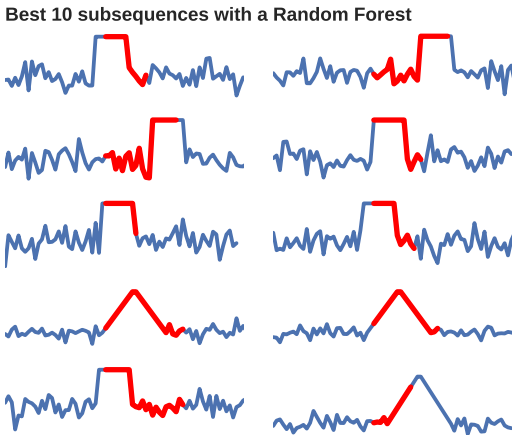
To illustrate the major argument of the learning of relationships between subsequences, we take an example inspired by [Mueen et al., 2011] to illustrate the logical-shapelets, which can only learn conjunction of subsequences. We generate a synthetic dataset of time series where time series of class 1 have both rectangular and triangular subsequences. Class 2 instances have two types of time series in equal proportions: one with only a triangular subsequence and the other one with only a rectangular subsequence. Rectangular and triangular subsequences can be at any position. Figure 7.15 illustrates this situation. By using the classical shapelet approach with information gain selection, most of high information gain subsequences are noise and don't contain a triangular or a rectangular subsequence that would have allowed the classification (figure 7.15c). Thus, using the classical shapelet approach based on univariate selection (subsequence information gain) as a selection process would mostly send to a classifier irrelevant and noisy features. On the contrary, using a multivariate feature selection process, for instance like the one performed in a random forest, triangular and rectangular subsequences are clearly identified as key subsequences (figure 7.15b).

Now that we have shown and demonstrated that only a few thousands of subsequence candidates is required to get a reliable set of subsequences to perform classification, the first stage of subsequence selection using information gain-like scores appears to be useless and limiting. The relevant subsequences are effectively learned by a classical supervised learning stack composed of a robust classifier and possibly, prior to the classifier, a state of the art feature selection technique. This need is especially prominent for multivariate time series.

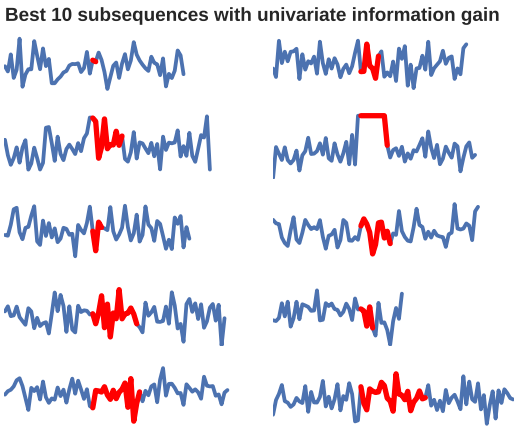
Finally, another advantage to consider the subsequences as a feature space is the addition of other information than the one related to the subsequences. Many use cases are not only composed by time series data. Imagine ECG time series associated with the age or the weight of the patient or in the industry time series describing the production of an object associated with the non-temporal object's specifications such as its dimension or chemical composition. The prediction of a class label may require to consider the whole context: temporal events with time series subsequences and non-temporal information. The EAST representation allows to add other features to the feature space prior to the feature selection or the classifier training. These additional information may even be other time series descriptors such as those extensively detailed in [Fulcher and Jones, 2014].



(a) Synthetic dataset generated to illustrate how advanced candidate selection techniques outperform simpler ones. The class 1 time series (in blue) contain both rectangular and triangular subsequences. The class 2 time series (in green) contain either a rectangular subsequence or a triangular subsequence



(b) Top 10 subsequences learnt by a random forest: they are useful to discriminate both classes



(c) Top 10 subsequences ranked by univariate information gain: none of them are relevant

Figure 7.15: Illustration of the relevance of advanced feature selection methods to detect relationships among subsequences

7.4 Conclusions

This chapter[1] has evaluated advanced feature selection relevance to discover a discriminant motif-based representation for time series classification. The approach is enabled by the observation that most subsequences in a time series are redundant and can be discarded. While the number of subsequences to draw is linked with the length of the subsequences, we have shown that at most a few thousands subsequences is sufficient.

We state that each subsequence represented by its aggregated distance to the time series (ie. subsequence transformation) is a feature in a larger feature vector on which classical feature selection techniques can be applied. Experimentation on 86 univariate datasets of the UCR and 15 multivariate datasets shows at least significantly similar classification performances to the state-of-the-art with a time complexity drastically reduced. The approach is flexible and allows to discover relevant subsequences with respect to other kinds of time series features than motif-based and also with respect to a "context" with typical static features. The proposed approach opens interesting perspectives to discover sophisticated patterns, for instance subsequences with relationships or multivariate patterns.

[1] The ideas developed in this chapter have been published in [Renard et al., 2016a]

Part III

Industrial Applications

In the previous part, we have presented our proposition to represent time series with discriminant set of subsequence transformations. We have shown its relevance both in terms of classification performances and computational complexity on a large set of datasets of the literature. In this part, we apply our proposition on the industrial use-cases provided by Arcelormittal.

In chapter 8, we present the context of the industrial application, we present a general overview of the industrial process and we introduce the use-cases and the datasets. In chapter 9, we benchmark our proposition with several configurations on the industrial datasets, together with the current classification configuration used by the company and other advanced features extracted from the time series. We will see that the relevance advanced time series representations for classification on the industrial use-cases is demonstrated. In particular, our approach based on motif provides both improved classification performances and meaningful insights, provided by subsequences, which are easily interpretable by process experts.

Chapter 8

Presentation of the industrial use cases

In this chapter, we present the industrial use cases at hand to benchmark our proposition. For this work, we have two datasets acquired from Arcelormittal's production lines. Both use cases concern defective products, however they have distinct nature and origin. The two cases' objective is the automatic detection of defective products using the production lines' data, in particular time series. With the trained models the final expected outcomes are first insights to get a better understanding of the conditions that lead to defective products and secondly process rules that could be used by the production lines to avoid such defects. Thus, the interpretability of our proposition is an advantage. A long term objective is the deployment of new online predictive models.

This chapter gives an overview of the use cases together with a brief description of the available data.

8.1 Context of the industrial use cases

The theoretical development of our work takes advantage of the industrial use cases provided by ArcelorMittal to test our proposition on real-world datasets.

Arcelormittal's production lines and R&D teams aim to improve the product quality while decreasing the operational costs, by a deep understanding of the process's behavior, the origin of defective products and also by developing corrective actions. The detection of defective products is complex: an appropriate sensor may not exist or the measurement procedure may not be compliant with an industrial environment.

The objective of the industrial application is to develop machine learning techniques to detect automatically defective products based on *existing* online sensor measurements. Many sensors monitor steel production over time, for instance physical properties of prod-

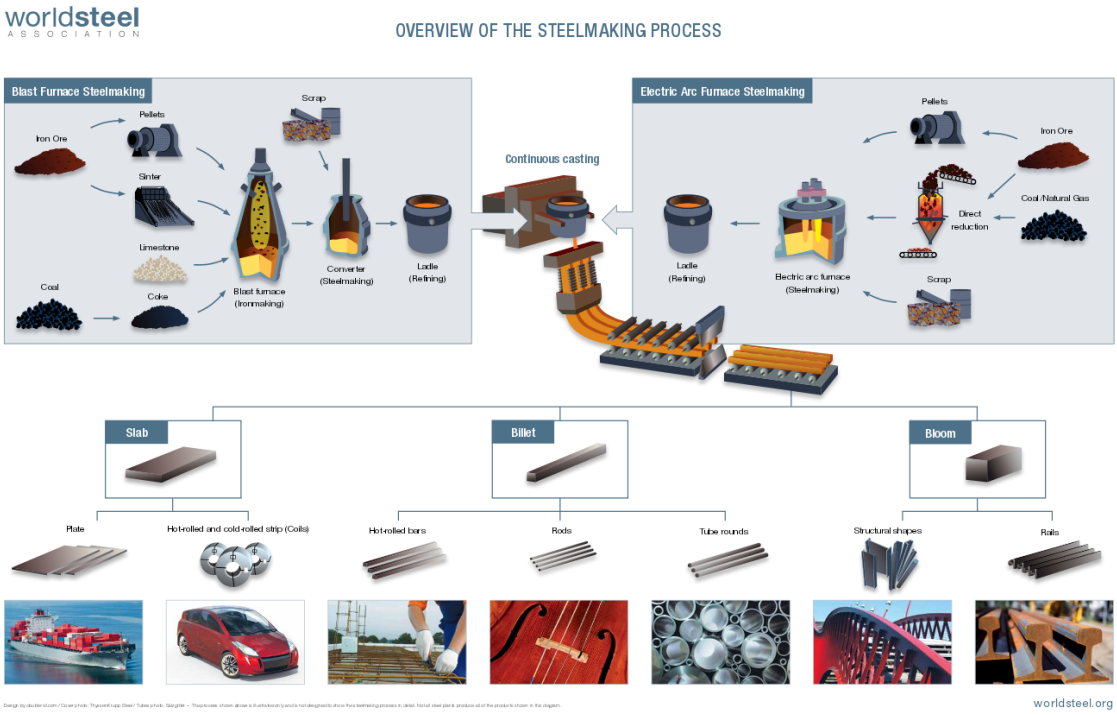


Figure 8.1: Overview of a steel production process, from blast furnace to end products

ucts or the state of production tools. In addition, the final quality of the products can be retrieved. We frame this industrial application into the time series classification task (with product quality as target), based on adequate time series representations of the time series.

In the next sections, we provide an overview on the context of the industrial use cases.

8.1.1 Steel production & Process monitoring

Arcelormittal's main activity consists in the transformation of ore, iron ore in particular, into steel end products such as coil, tube or wire for automotive applications, construction, domestic appliances and packaging. We describe very briefly the process to produce steel and how it is monitored.

Process description

Steel production involves a large scale process with several stages. Our objective here is to provide high level insights on the process to understand the industrial use cases.

Steel production process begins with steelmaking using raw materials, most of all iron ore. The primary steelmaking stage involves a blast furnace to smelt ore into liquid steel. The secondary steelmaking stage consists in refining the liquid steed by adding alloy el-

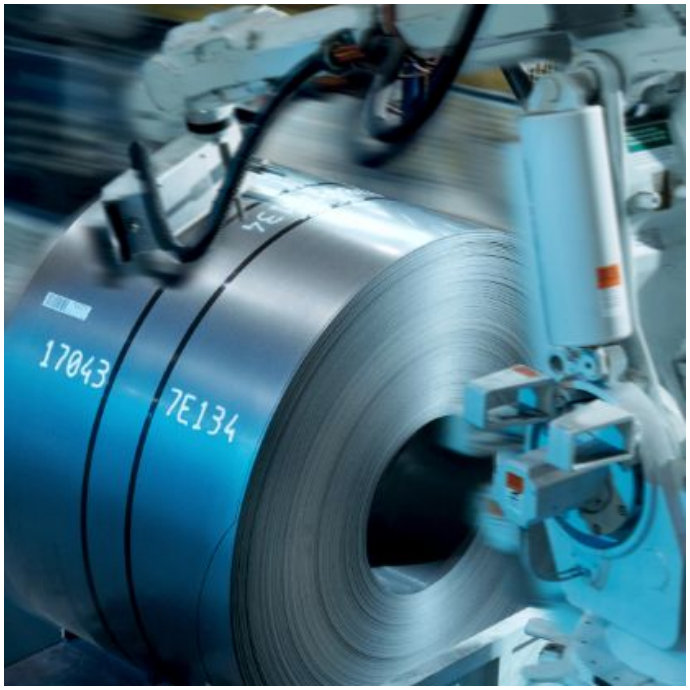


Figure 8.2: A coil, one example of end product from ArcelorMittal, used for instance to build cars or packages

ements to obtain the desired steel grade with specific mechanical or chemical properties. This step also involves impurities removal from liquid steel and the reduction of dissolved gases to ensure liquid steel quality.

Once liquid steel reaches the required chemical composition and quality, it is cast in the continuous casting stage. This operation consists in casting the liquid steel into a solid half-finished product, in our use cases a slab, a solid steel cuboid.

A slab is then processed to fit its desired shape: a coil in our datasets. This stage involves several rolling mills that successively flatten a slab into a coil that comes with a lengthening of the product. Then a surface treatment can be applied at the end of the product, before a final surface inspection of the product to control its quality. These steps involve controlled heating and cooling of the product together with controlled pressure in the rolling mills, to allow rolling but also to reach the desired structure of matter and obtain the required mechanical properties according to steel phase diagrams.

Process monitoring

Steel production is highly monitored and automated. Many physical parameters are continuously measured and controlled to maintain the process in the desired operational range. Measured parameters include physical state of the product being processed (temperature, composition, etc.) and states of the process tools (flow, temperature, pressure, etc.). Many

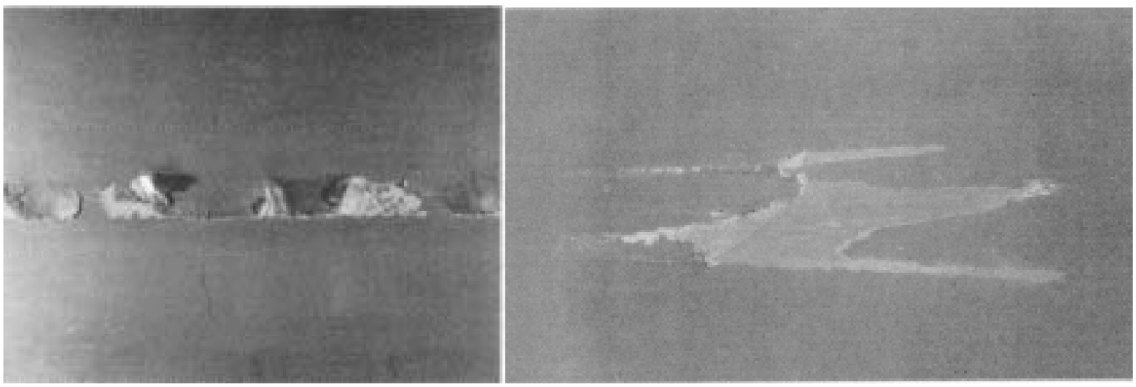


Figure 8.3: Two defects on the surface of a steel product (illustration from [Zhang and Thomas, 2003])

sensors of various types are disposed along the process.

Product quality is also controlled: for instance, the surface aspect is controlled with an automated surface inspection system (SIAS) at the end of the process. A SIAS is composed of a camera and specifically designed algorithms to detect continuously defects on the products. Other quality controls may not be automated and may even be destructive such as the assessment of the mechanical properties. Most sensor data measurements are centralized into factory databases, which usually facilitates the retrieval. Oftentimes, there are issues to gather, synchronize and clean the data because of the heterogeneity of the data structures, the different sampling rates of the sensors and most generally the complexity of the industrial environment.

For specific problematic, specific sensors can be used occasionally during a campaign of data collection to monitor particular parameters based on process expert's hypotheses.

Product quality

A steel product has an adequate quality when it has the desired properties with both precision and homogeneity. The desired properties concern the chemical composition, that should be reached at steelmaking, and the structure of matter related to the mechanical properties that is the result of the chemical composition and the processing (annealing for instance) at both steelmaking and rolling mills. The properties should be precise and homogeneous all along the product, which is a complex objective to reach since the process is long and complex and the product dimensions are large. Defect's origin can be related with imprecision in the process control rules that can themselves result from a lack in the body of knowledge or imprecision in process models.

Another source of non-quality product is the product aspect. Its degradation can be caused by a defective production tool or the inclusion of an external element.

Non-quality products are major issues since they induce waste of raw material and

energy while they keep busy the production line uselessly. This last point is particularly critical if a defect takes root at the beginning of the process but is observed at the end of it (see 1^{st} use case, section 8.2.1). The worse scenario with non-quality products is their non-detection and their shipment to clients. This scenario is handled effectively thanks to non-quality product detection at the end of the production line with for instance the automated surface inspection system. However, while this system gives us quality label for the products, it doesn't allow early detection of defective products.

The main objective of our work is to develop machine learning techniques that fully exploit the information contained in the time series data generated by online sensors.

8.1.2 Types of data

The datasets of the industrial use cases gather several types of data defined below.

Instance An instance is a set of measurements or values that characterizes a steel product (slab or coil) and gathers static data, dynamic data and a target (quality value).

Static variable It is the common datatype with one single value by instance and by variable. Several static variables may describe one instance, in this case their values form one vector by instance. Such static variables may be the length of the product or its concentration for a particular chemical component.

Dynamic variable Given one single instance, a dynamic variable evolves across time or space, producing one vector of values with the length bounded by the occurrence of the considered instance. When the values of the vector are numeric, which is the typical output of a sensor, we call it a time series. One instance can be described by several dynamic variables (several sensors), producing a multivariate time series $T_{n,m}$ (using the notation introduced previously). In our industrial use cases, dynamic variables have been preprocessed to obtain aligned sampling to gather them in a matrix where each column is a dynamic variable and each row corresponds to a timestep. The intersection of a row with a column receives the values of one dynamic variable at a given timestep.

Quality value Each instance of our industrial use cases is associated with a value that describes the quality of the steel product. This value can be a raw quality value like the number of defects for the product or their severity, or a refined value such as a precomputed class label, for instance "good product" or "bad product". In this work, raw quality values (such as the number of defects) are converted into discrete class labels to bring the problem back to a classification task. Classification is in line with the kind of decision taken on a production line with respect to the product quality: is the product defective or not ?

In our machine learning problem, static and dynamic variables are the input of our models and quality value is the target.

8.1.3 Industrial problematic formalization

The industrial context and the format of the data having been introduced, we can formalize the problematic.

The objective is to build classifiers able to predict the target (quality label: occurrence of a defect or not) of an instance (steel product) from static data (product characteristics) and dynamic data (measurements of online process sensors). Dynamic data is the focus of our work: performances of the classifiers should benefit from refined representations of the time series for machine learning models, based on state of the art approaches and our proposition.

The final models should help to understand the underlying phenomenon leading to defect occurrence, thus the time series representations should be as interpretable as possible, in particular for process experts with few expertise in time series analysis, statistics or machine learning. The time series representations together with the static data should enable the use of common machine learning algorithms. Finally, it must be highlighted that industrial datasets typically gather dozens of static and dynamic variables with few instances (from hundreds to thousands), which is not the easiest context to train machine learning models.

8.2 Description of the use cases

Note: additional information is provided in an undisclosed technical report for confidentiality issues.

8.2.1 1^{st} use case: sliver defect, detection of inclusions at continuous casting

The first use case is about the detection of one particular type of surface defect called *sliver*. This kind of defect is visible at the end of the steel production process during the automated surface inspection. A sliver defect takes root in the inclusion of external particles in the steel product: the inclusion of alumina (used for steel de-oxidation) or mould powder. These particles are trapped during the continuous casting process: they are caught in the liquid steel flow and get stuck inside the slab. They are revealed progressively along the process, when the thickness of the steel product is reduced in the rolling mills. When an inclusion is rolled, it produces a surface defect (a sliver) that can be severe enough to reject the product in order to avoid a non-quality issue.

This kind of defect is particularly critical. A coil with a surface defect is not tolerable for a client, especially if it is used at the surface of a final product like a car body. Inclusion defects are particularly inefficient for production lines: defects are generated at the continuous casting (beginning of the process) but revealed at the final automated surface inspection, which is at the very end of the process after the rolling mills and the surface treatments. The typical duration between the production of the defect and its observation is around 5 weeks. A downgraded product for inclusion is a particular waste of material, energy, production tool wear and unnecessary production planning occupation.

Sliver defect is the most costly defect. Many resources are dedicated to avoid this type of defects, both on production lines and R&D. In addition to the work of metallurgy process experts, machine learning techniques have been applied. Machine learning approaches rely on static data, such as product's chemical composition or simple global indicator of sensor measurements, such as mean or standard deviation. While these models are capable of some defect detections, improvement is expected with a refined exploitation of the dynamic of sensor measurements.

The main objectives are to improve the identification of the roots of these defects and the actuators to avoid them.

Available data to implement machine learning models consist in two datasets from two distinct plants with static data (global parameters of products like chemical composition), dynamic data (production parameters across time from sensors) and product quality (raw quality values or labels):

Sliver Dunkirk dataset The first dataset, from Dunkirk plant, comes with 2261 instances (coils), 11 static variables and 8 dynamic variables. The quality information is pre-computed for this dataset. Each product is labeled either *good* or *bad*. *Bad* products are linked with coils where serious sliver defects have been detected at the automated inspection lines. The class labels are almost homogeneous across the dataset with 1205 products labeled as *bad* and 1056 products labeled as *good*.

Sliver Bremen dataset The second dataset gathers 5 different measurement campaigns, assembled in one single dataset. The specificity of this dataset is the presence of specific sensors designed to measure physical parameters known to be related with sliver occurrence. A particular preprocessing of the quality values has been setup in coordination with process experts. The result is several type of targets, corresponding to different scenarios. In each of them the class are balanced.

8.2.2 2^{nd} use case: detection of mechanical properties scattering

The second use case is about issues of mechanical properties homogeneity. Mechanical properties are for instance the hardness, elasticity or compressive strength of the steel

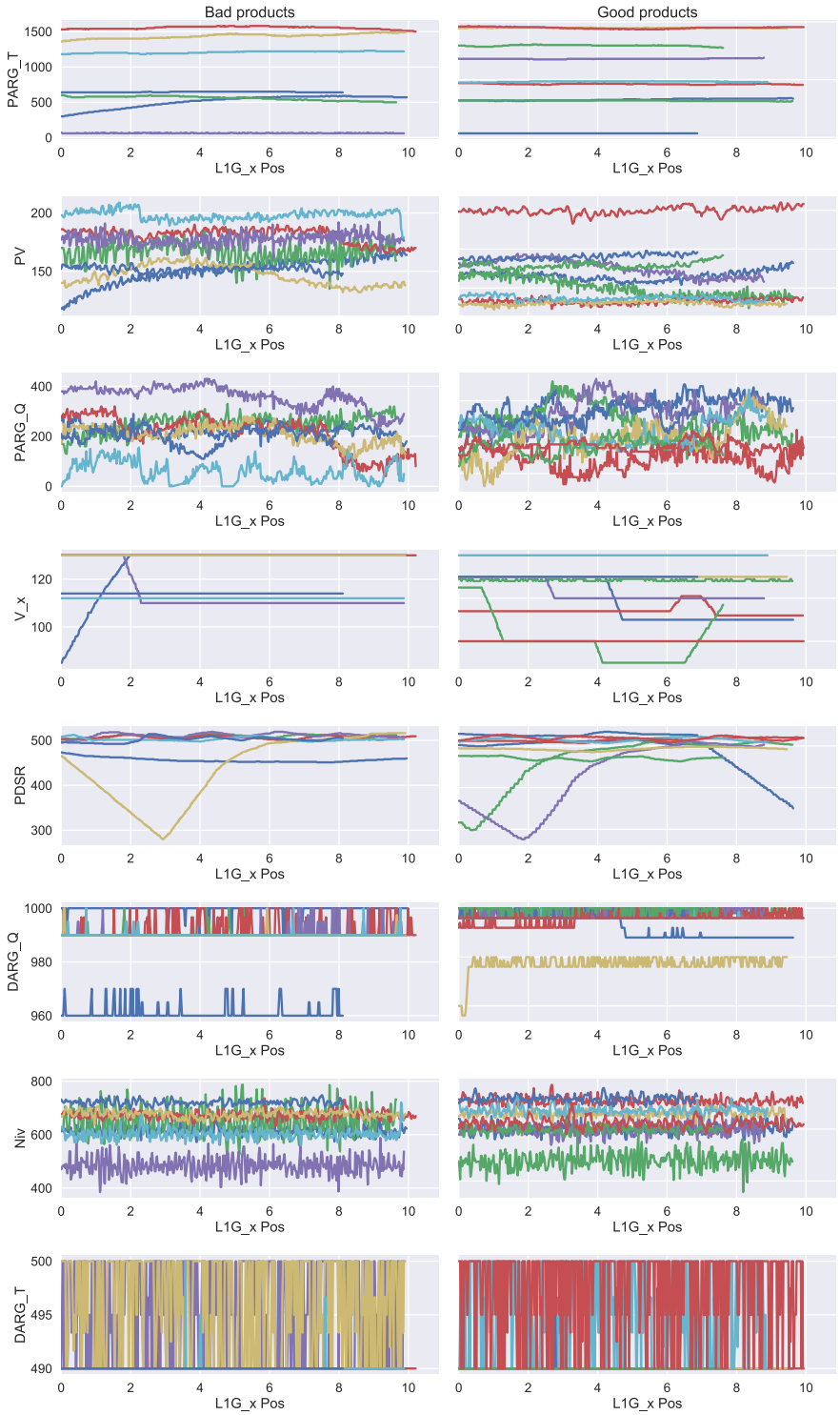


Figure 8.4: Sliver Dunkirk - Typical shapes of a small sample of instances, by quality label

product. Coils are usually more than $1000m$ long. It is essential to get the same desired mechanical properties all along a product, at least into tolerance bounds. Homogeneity is highly correlated with the ability to control the process parameters inside precise limits to avoid the scattering of such properties. For instance, product temperature is a critical parameter that influences steel structure and properties during the manufacturing. Temperature is influenced by various origins depending on the process stage like the product's temperature, furnace's temperature, cooling power or pressure applied on the product. The knowledge may not be sufficient to build models able to control perfectly the process parameters or actuators may not be efficient enough.

The objective of this use case is to gain insights on the origin of mechanical properties scattering.

Available data includes product's characteristics (chemical composition, etc.), sensor measurements at rolling mills and both mechanical properties (very low sampling rate because of the destructive nature of the trial) and proxies of the mechanical properties (with a higher sampling rate). In fact, a direct measure of the mechanical properties is destructive. Process experts have proposed to use proxy variables, known to be related with the mechanical properties of coils. These proxy variables are continuous. It is required to discretize them in order to perform a classification task and identify good from bad products. To do so, the standard deviation is computed for each of proxy variables, by coil. The distribution of the standard deviation is computed overall the coils. The median of the distribution is used as a threshold: coils with a standard deviation below the median are considered as good products (low variation of the mechanical properties), while products with a standard deviation greater than the median are considered as bad products.

The mechanical properties scattering dataset comes from one single plant 442 instances (coils), 21 static variables and 35 dynamic variables. The class labels are perfectly homogeneous across the dataset with 221 products labeled as *bad* and 221 products labeled as *good,* which is expected since the median is used during the calculation of the labels.

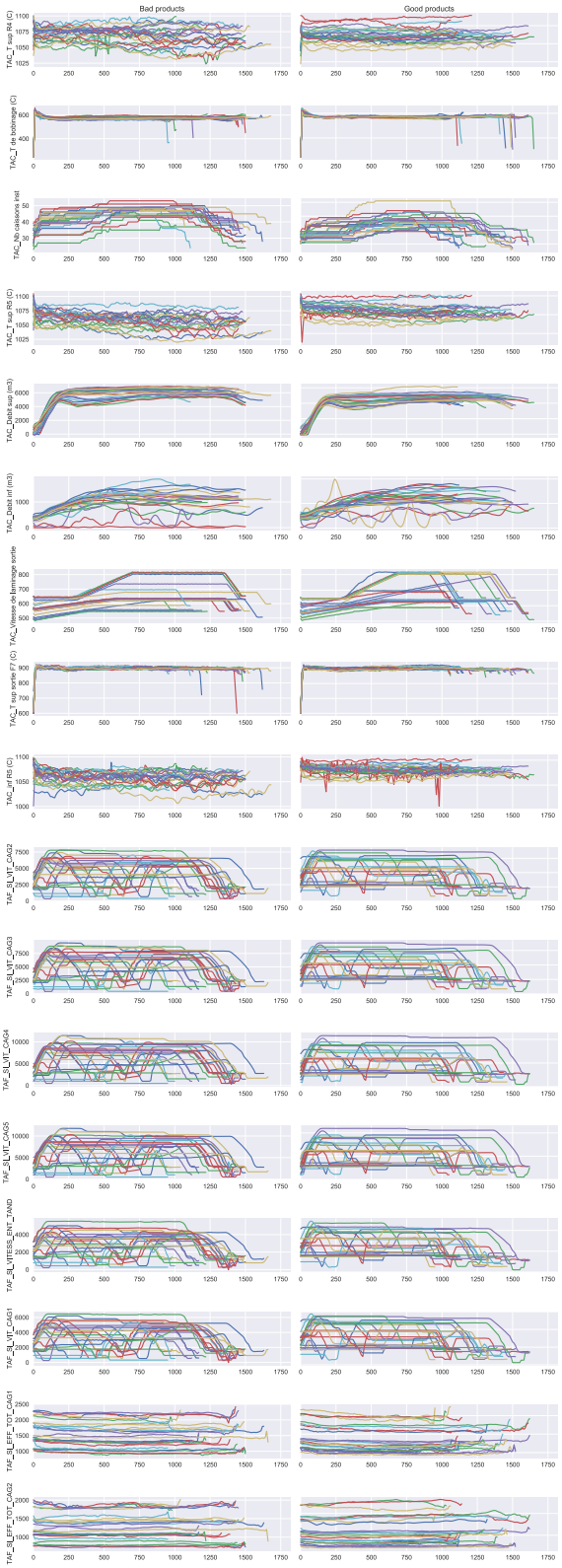


Figure 8.5: Mechanical properties scattering - Typical shapes of a small sample of instances, by quality label

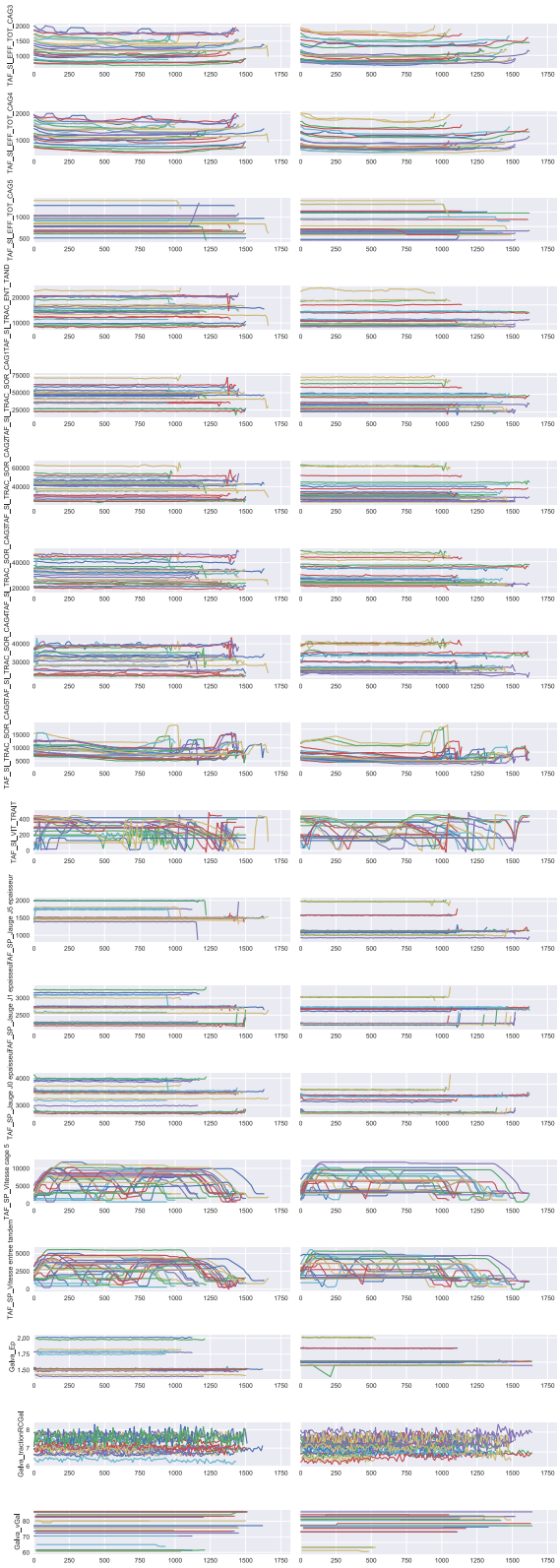


Figure 8.6: Mechanical properties scattering - Typical shapes of a small sample of instances, by quality label

8.3 Conclusions

In this chapter, we have presented the context of the applications, with a brief description of the industrial process and some insights on the challenges of process measurement and control and product quality. We have described the two use-cases at hand, both for product quality issues, and the datasets available: each instance (*ie.* product) is linked with static features of "context" to describe the product, time series data to describe the dynamic of the process and a label to describe the product quality. The next chapter describes the experimentation performed on these use-cases with our proposition and other approaches from the literature.

Chapter 9

Benchmark on the industrial use cases

In this chapter, we benchmark several time series representations, including our proposition, to feed a classifier trained on the industrial use cases datasets. A specific performance assessment is used to reflect the operational requirements of the production lines, which is the limitation of the number of wrong bad product detection (false positive rate).

First, the experimental framework is presented, including the detail of the representations extracted from the time series to build the feature vector. Then the classification performances are displayed followed by a sample of EAST-shapes discovered on the datasets to illustrate the potential of interpretation of our proposition.

9.1 Experimental procedure

For the experimentation, our focus is the comparison of several time series representations to train a classifier on the industrial use cases. In particular, we want to assess the impact in terms of classification performance and interpretability of our proposition based on discriminant set of subsequences. The choice of the learners (for feature selection or classifier) has not been optimized.

The experimentation is based on 3 steps. In a first time, the time series representations are computed. We use 3 types of time series representations: a baseline, composed of simple descriptors (mean and standard deviation), currently used by ArcelorMittal. A second type of time series representation is based on hundreds of advanced global time series descriptors from the time series analysis field. Then we compute several variations of the EAST representation. Combinations of these representations are also tested. The second step is the feature selection to reduce the dimension of the feature vector before feeding a classifier (third step). These 3 steps are described in the next sections. An

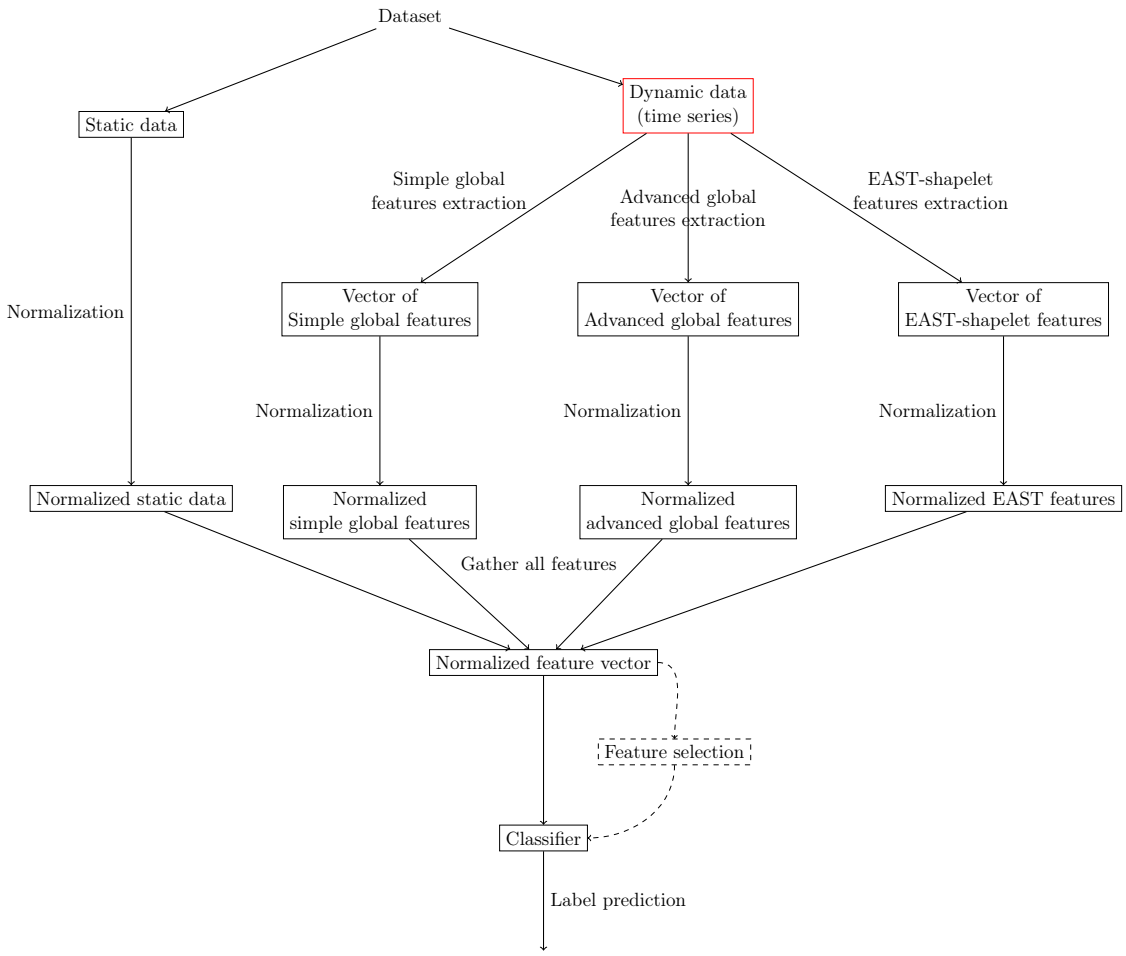


Figure 9.1: Feature vectors extraction from time series with several strategies and composition of the final feature vector to feed the model training stage

overview of the overall procedure is shown figure 9.1.

9.1.1 Feature vector engineering for the time series

Arcelormittal's datasets are composed of dynamic data (time series) from which we extracted features and static data. The final model takes in input the static data and subsets of the time series features. The system is described figure 9.1.

The following configurations of features are compared.

Baseline The baseline is a model trained with static data and basic global time series descriptors: the first 2 statistical moments (mean and standard deviation). It corresponds to the current usual setup to perform classification using time series from industrial dataset.

Advanced Global time series Features (AGF) The model is composed of the *Base-*

line with advanced global time series features thousands of time series descriptors much more refined than the simple mean or standard deviation are added. For instance, it adds information on periodicities (periodogram, autocorrelation, etc.), detailed information on the time series global distribution and also structural information (stationarity, etc.). However the extracted information from the time series remains at a global scale.
The list of advanced global time series features comes from an extensive work [Fulcher and Jones, 2014] to gather thousands of features from the whole time series analysis literature whatever the applications (physics, economics, medicine, etc.). These features were "manually" designed for specific applications to understand the structure and behavior of signals and the underlying phenomenons. In time series classification, it is desired to discover automatically the relevant features instead of manually designed them. In [Fulcher and Jones, 2014], the authors extract thousands of these time series representations that produce a feature vector used to feed a classical learning stack (feature selection and classifier). The complete list of features is detailed in the supplementary work of [Fulcher et al., 2013], but the general families of features are as follows [Fulcher and Jones, 2014]:

- Basic statistics of the distribution of the time series values (statistical moments, spread, gaussianity, spread, outlier properties, etc.)
- Linear correlations (autocorrelations, power spectrum, etc.)
- Stationarity measurements
- Information theory, entropy and complexity measures (auto-mutual information, approximate entropy)
- Methods from physical nonlinear time series analysis (Lyapunov exponent estimates, etc.)
- Fitting linear and non-linear models (ARIMA, gaussian processes, GARCH, etc.)
- And others (Wavelet transform coefficients, etc.)

EAST This model is composed of the static data and EAST-shapelets features from our proposition to extract local information through subsequences. One thousand subsequences are used as features $\psi(T_n, s_p)$ by dynamic variables.

EAST+AGF This model is a combination of the static data, AGF and EAST.

EAST/PreProc+AGF This model is the same as *EAST+AGF* except that dynamic variables are preprocessed prior to the EAST discovery and additional metrics and

aggregation functions are used. The idea is to generalize the EAST-shapelet discovery and highlighting relevant information in the time series using transformations and additional distance measures and aggregation functions. The principle was exposed chapter 5. For the experimentation on industrial datasets, the processing pipeline is detailed figure 9.1.
The preprocessing of each time series of the dataset is performed with the following transformations. An illustration of the decomposition of a time series using these transformations is shown figure 9.2.

- A Discrete Wavelet Transform (DWT) with the *db3* wavelet and 2 levels of decomposition
- An Empirical Model Decomposition (EMD) with *CEEMDAN* and 4 IMFs
- A cumulative integral
- A derivative with two distinct window sizes $\{1; 5\}$
- The autocorrelation function.

EAST/PreProc/Argument+AGF This model is the same as *EAST/PreProc+AGF* with the addition of the argument of $\psi(T_n, s_p)$ for each transformation involving each subsequence and each time series, as we will discuss in perspective chapter 10. The idea is to add the position were a EAST-shapelet matched with a time series to the feature vector, by recording the matching position in $[1 \dots |T_n|]$.

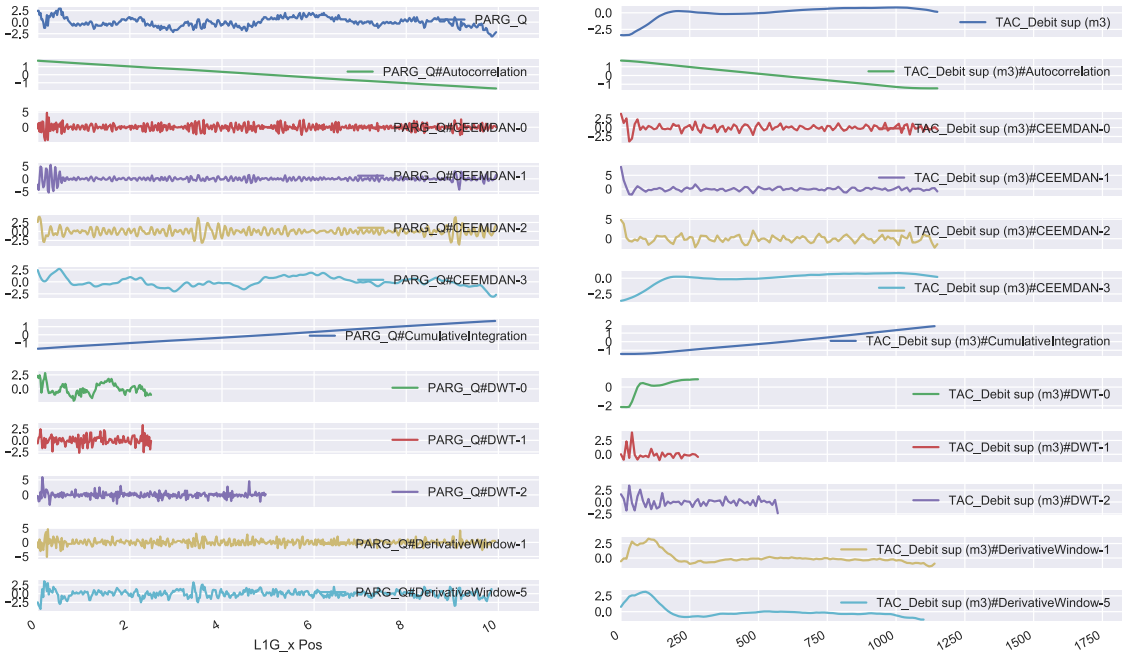
For EAST-shapelet, the number of subsequences to be extracted from the time series is set to 1000 by variables: for instance, the Sliver Dunkirk has 8 dynamic variables, we extract 8000 EAST-shapelet candidates.

9.1.2 Learning stack

The features from the previously described setups feed a learning stack, composed of a usual classifier and for some configurations a feature selection step that precedes the classifier. Our contribution is about the representation of the time series. Thus, for the trials on industrial datasets, we choose known one combination of efficient and interpretable algorithms for feature selection and classification. Any other configuration for the training stack could have been used.

Feature Selection

The drawback of feature extraction from time series to perform machine learning tasks is the resulting high dimensional feature vector. The number of features varies from 27



(a) (b)

Figure 9.2: Illustration of the decomposition of a time series. The relevant information can be highlighted or released from noise

Configuration name	Features					
	Static data	Dynamic data				
		Basic Glob. Feat.	Adv. Glob. Feat.	EAST-shapelets		
				Raw TS	Preproc.	Arg.
Baseline	✓	✓				
AGF	✓		✓			
EAST	✓			✓		
EAST+AGF	✓		✓	✓		
EAST/PreProc+AGF	✓		✓	✓	✓	
EAST/PreProc/Arg+AGF	✓		✓	✓	✓	✓

RLR: Randomized Logistic Regression
RF: Random Forest

Table 9.1: Summary of the different evaluated configurations

on the simpler feature vector configuration on Sliver Dunkirk (static data & simple global features) to 147 616 features on the mechanical scattering dataset with advanced global features and EAST-shapelet candidates over raw and preprocessed time series together with their arguments (matching positions of the EAST-shapelets).

In order to mitigate the curse of dimensionality problem, prevent overfitting, ease the classifier training, reduce the classification cost (by decreasing the number of sensors and the number of features to compute) and get a more interpretable final model, we use a dedicated feature selection stage before training the classifier. The feature selection used is the Randomized Logistic Regression, a stabilized version of the Lasso based on a L_1 regularization term that returns a sparse feature vector by setting at 0 coefficients of irrelevant features [Meinshausen and Buhlmann, 2010]. The stability is ensured by several training of the logistic regression on random sub-samples of the data.

Classifier

The classifier used is a Random Forest with 300 estimators [Breiman, 2001].

9.1.3 Classification performance evaluation

Statistical significance of the evaluation

To get a significant estimation of the classification performances, we use a 10-folds cross-validation on the whole dataset. For supervised feature extraction techniques such as the EAST-shapelets, a new set of features has to be drawn and computed for each fold, on the contrary to unsupervised features (like global statistics over time series) whose values remain constant across folds.

To assess the variability inter-folds, the standard deviation of the classification performances across folds is reported.

Classification performances assessment

The classification performances are evaluated with several indicators. The usual classification accuracy is computed, but industrial applications may require more refined details on the classifier behavior. The false detection rate (FDR) is particularly important: if most of the true defective products are detected at the price of a high rate of wrong detection (ie. good products labeled as defective) the classifier is likely to be useless for operations. To overcome this issue, an adequate measure is the True Positive Rate (TPR) of defective product detection under the constraint of a low False Positive Rate (FPR), typically 10% (TPR@10%FPR) or 15% (TPR@15%FPR). These particular values are observed on the Receiver Operating Characteristic (ROC) curve. This way, the bad product detection performance of the model is evaluated for reasonable operational conditions.

9.2 Results

In this section, we present the classification performances of the different configurations previously described and samples of subsequences discovered using EAST representation.

9.2.1 Classification performances

Classification performances are shown tables 9.2a and 9.2. In every case, the baseline configuration is outperformed by more refined representations of the time series, both advanced global features (*AGF*) and variations of the EAST-shapelets. We review classification performances by dataset below.

Sliver Dunkirk The *EAST+AGF* configuration has the highest accuracy (65%) and the best TPR@15%FPR (43%). At TPR@10%FPR, *EAST+AGF* is outperformed by *AGF* configuration by 1% (35% and 36% respectively - baseline at 28%). Complete results are shown table 9.2a.

Mechanical Properties Scattering (PreHeating) The *EAST+AGF* configuration has the highest accuracy (73%) and *EAST/PreProc/Arg+AGF* configuration (EAST-shapelets variation with time series preprocessing and argument) clearly outperforms other configurations with 44% of TPR@10%FPR and 56% of TPR@15%FPR (baseline at 31% and 45% respectively). Complete results are shown table 9.2a.

Mechanical Properties Scattering (Soaking) In this case, *AGF* configuration outperforms the other configurations, with little difference. The *AGF* configuration reaches 72% of accuracy (equivalent to *EAST/PreProc+AGF* configuration) and 49% of TPR@10%FPR and 57% of TPR@15%FPR (baseline at 46% and 57% respectively). Complete results are shown table 9.2b.

The mechanical properties scattering (soaking) case is the only case where the classification performances of advanced time series representations are not drastically better than the baseline. It is also the only case where a configuration including *EAST-shapelets* representation doesn't lead the board.

ROC curve (Receiver Operating Characteristic) is an important tool to assess the performance of a predictive model for industrial applications: it allows to compare the overall performances of the models and to know its classification performances (True Positive Rate *vs.* False Positive Rate for the positive class) at precise areas of false positive detection. This is important for business application to setup the right operating point of the classifier: a too high False Positive Rate is expensive and impact the productivity. Too few detections and the model is useless.

Configuration name	Accuracy	TPR @ 10% FPR	TPR @ 15% FPR
Baseline	61% (σ 1%)	28% (σ 9%)	36% (σ 8%)
AGF	64% (σ 3%)	**36% (σ 7%)**	**43% (σ 7%)**
EAST	62% (σ 2%)	30% (σ 6%)	38% (σ 5%)
EAST+AGF	**65% (σ 3%)**	35% (σ 5%)	**43% (σ 6%)**
EAST/PreProc+AGF	63% (σ 2%)	31% (σ 6%)	39% (σ 6%)
EAST/PreProc/Arg+AGF	63% (σ 2%)	30% (σ 6%)	39% (σ 7%)

(a) 1^{st} Use Case: Sliver Dunkirk

Configuration name	Accuracy	TPR @ 10% FPR	TPR @ 15% FPR
Baseline	71% (σ 7%)	31% (σ 11%)	45% (σ 14%)
AGF	71% (σ 5%)	41% (σ 16%)	**56% (σ 12%)**
EAST	70% (σ 4%)	37% (σ 17%)	52% (σ 13%)
EAST+AGF	**73% (σ 6%)**	39% (σ 12%)	51% (σ 15%)
EAST/PreProc+AGF	71% (σ 4%)	39% (σ 13%)	54% (σ 9%)
EAST/PreProc/Arg+AGF	72% (σ 5%)	**44% (σ 15%)**	**56% (σ 10%)**

(a) 2^{nd} Use Case: Mechanical Properties Scattering (Quality label proxy: pre-heating)

Configuration name	Accuracy	TPR @ 10% FPR	TPR @ 15% FPR
Baseline	71% (σ 6%)	46% (σ 10%)	**57% (σ 11%)**
AGF	**72% (σ 5%)**	**49% (σ 12%)**	**57% (σ 12%)**
EAST	68% (σ 4%)	38% (σ 14%)	51% (σ 11%)
EAST+AGF	70% (σ 5%)	47% (σ 17%)	52% (σ 18%)
EAST/PreProc+AGF	**72% (σ 5%)**	47% (σ 14%)	54% (σ 18%)
EAST/PreProc/Arg+AGF	71% (σ 5%)	43% (σ 21%)	53% (σ 13%)

(b) 2^{nd} Use Case: Mechanical Properties Scattering (Quality label proxy: soaking)

Table 9.2: Classification performances for the 2^{nd} use case (mechanical properties scattering). Each table is a distinct quality label proxy. Each line is a distinct set of features. The learning stack is composed by a Randomized Logistic Regression for the feature selection and a Random Forest (300 estimators) for the classification

We present ROC curves of classifiers figure 9.3 for some configurations based on our proposition and AGF. While the exact performances of the actual classifiers are not disclosed, the performances of the classifiers based on our proposition have good performances.

9.2.2 Illustration of discovered EAST-shapelets

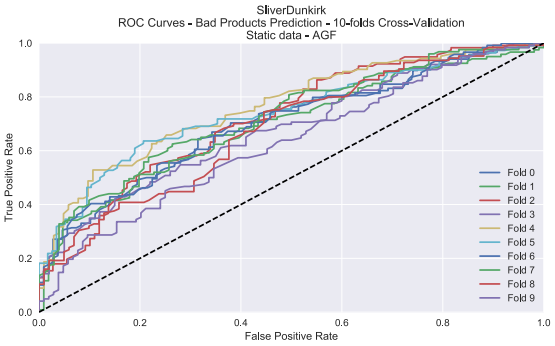
A major advantage of our proposition to represent the time series is the good interpretability of the features. The subsequences learned to be meaningful for the classification task are extracted from the time series dataset: they represent actual behaviors of the industrial process.

Together with the process experts we have assessed dozens of subsequences learned by the classifiers: the experts do appreciate the interpretability of the representation. They have also validated that subsequences are effectively relevant: either they are in line with known physical behavior or they report an interesting behavior worth investigating.

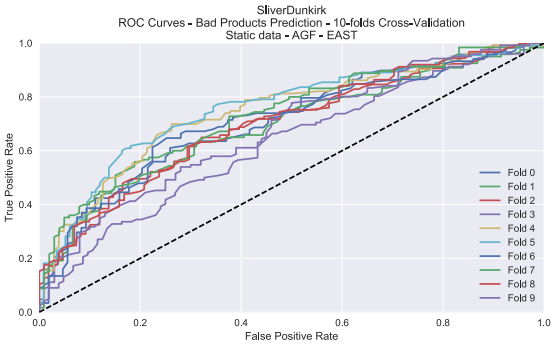
After several meetings with the experts, we refined the representation of the meaningful subsequences until getting the graphs shown figures 9.4 and 9.5. For each case, the graph on the left represents the subsequence in context (in its time series of origin). The graphs at the center represent (1) the closest time series of the dataset to the subsequence (in the meaning of the used distance measure) and (2) random time series from the dataset to compare the time series in (1) with random time series. The time series colors represent distinct labels. The histograms on the right represent the distribution of the distances of all the time series of the dataset to the subsequence. Time series with the same label are gathered in the same histogram. The histograms allow to assess the "strength" of the discriminatory power of the subsequence: if the distribution are significantly dissimilar, the subsequence is highly discriminant of one class.

9.3 Computational performances

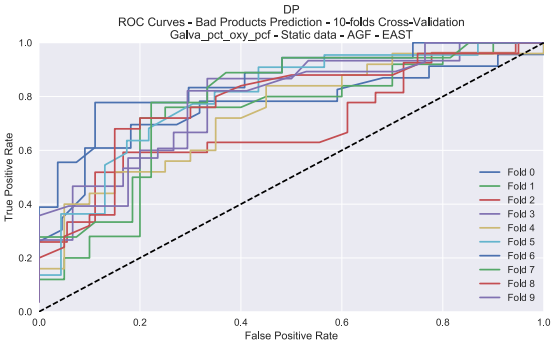
The time required to compute the EAST-Representation on the industrial datasets with multivariate time series is drastically reduced in comparison with the exhaustive shapelet approaches. On a cluster, the computation took a couple of hours: the time needed for the exhaustive approaches would have been prohibitively higher.



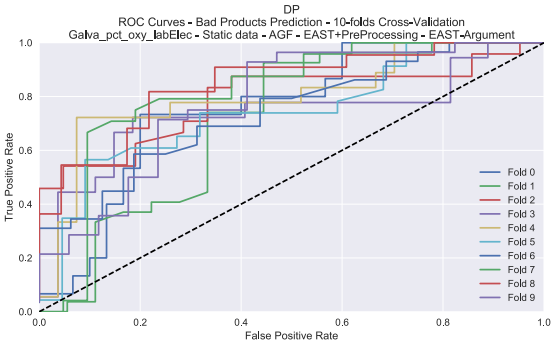
(a) Sliver Dunkirk - AGF configuration



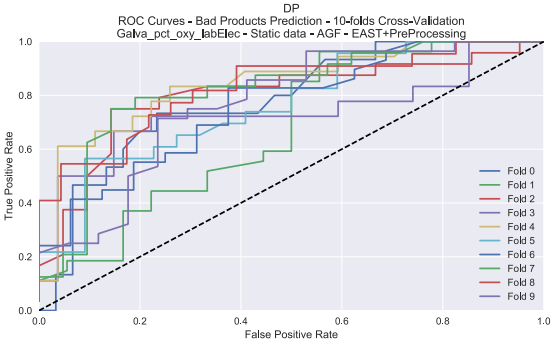
(b) Sliver Dunkirk - EAST+AGF configuration



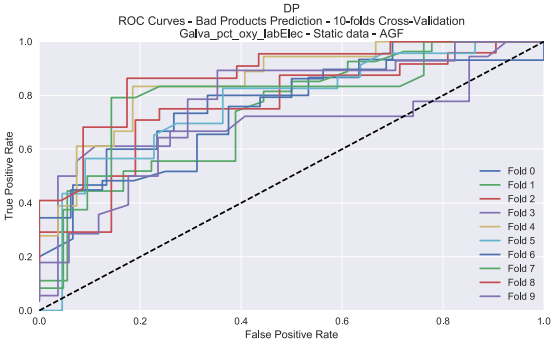
(c) Mechanical Properties scattering - Pre-heating EAST+AGF configuration



(d) Mechanical Properties scattering - Pre-heating EAST/PreProc/Arg+AGF configuration

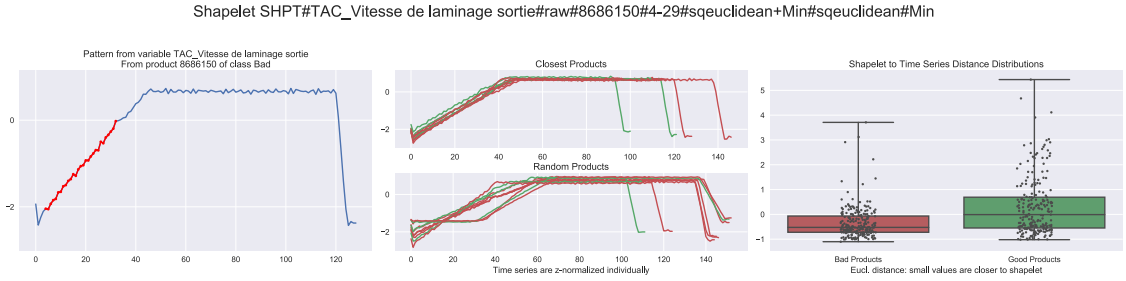


(e) Mechanical Properties scattering - Soaking EAST/PreProc+AGF configuration

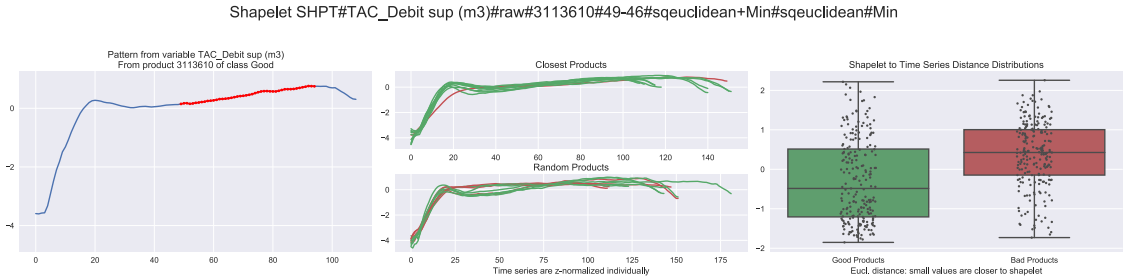


(f) Mechanical Properties scattering - Soaking AGF configuration

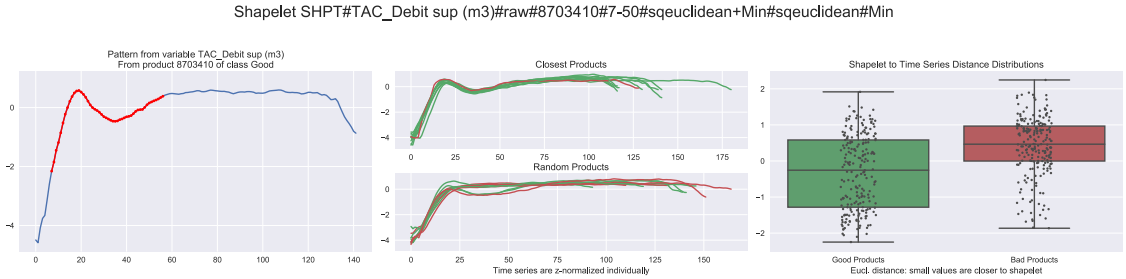
Figure 9.3: ROC Curves of best performing configurations for each use case



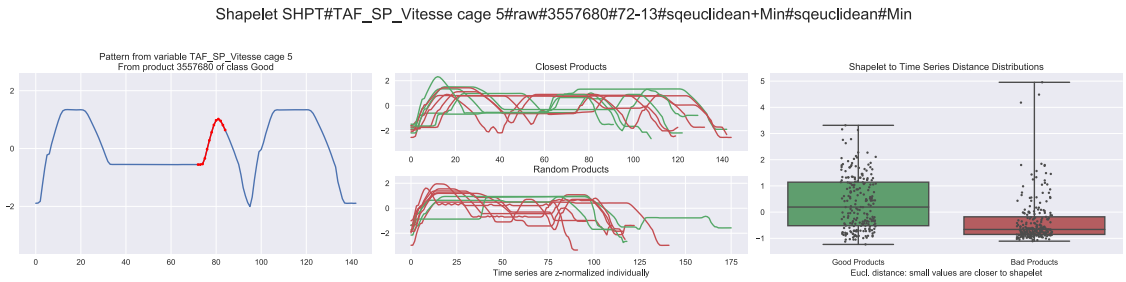
(a)



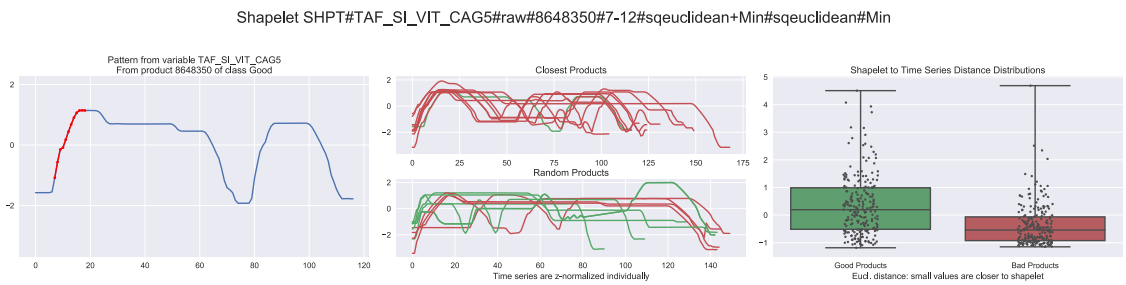
(b)



(c)

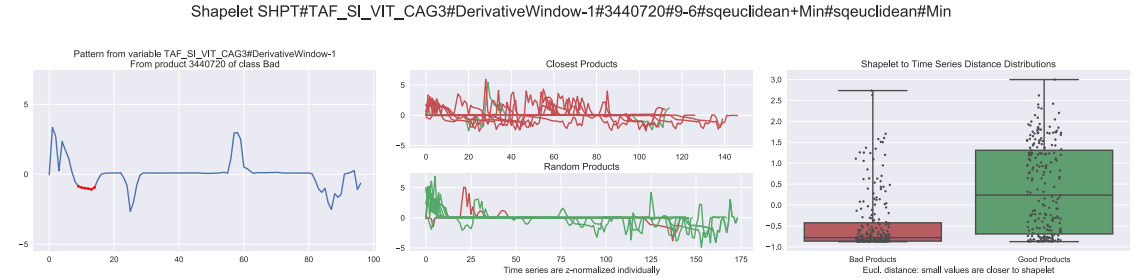


(d)

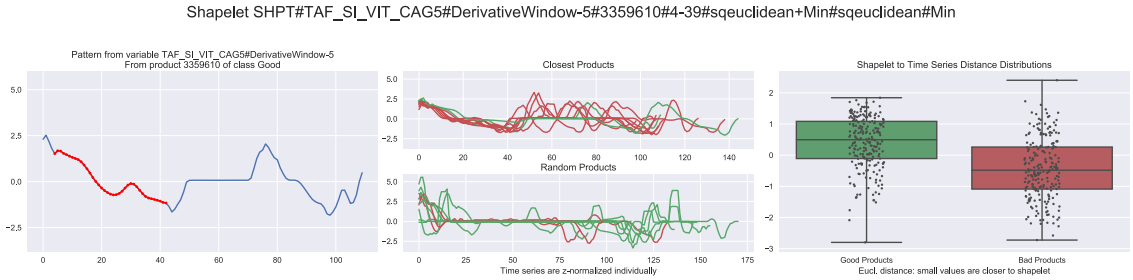


(e)

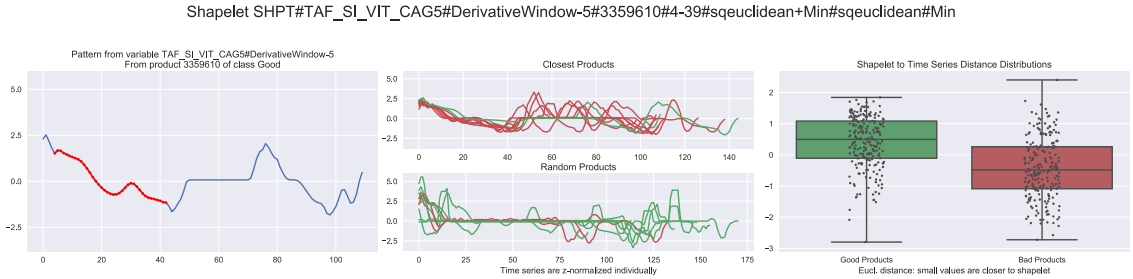
Figure 9.4: Some EAST-shapelets discovered on ***raw*** time series from the 2^{nd} use case (DP) with pre-heating as proxy quality label



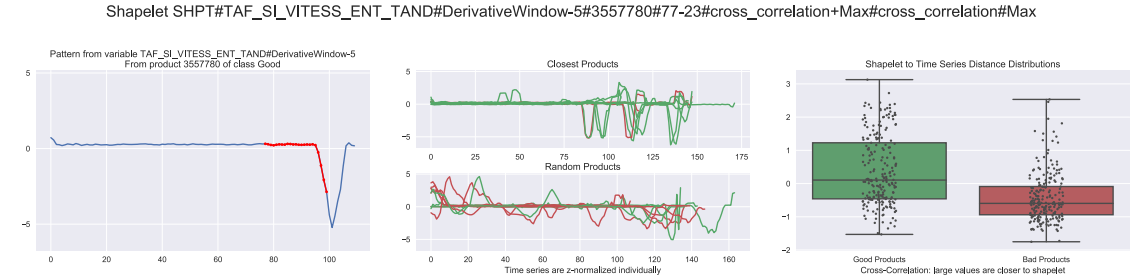
(a)



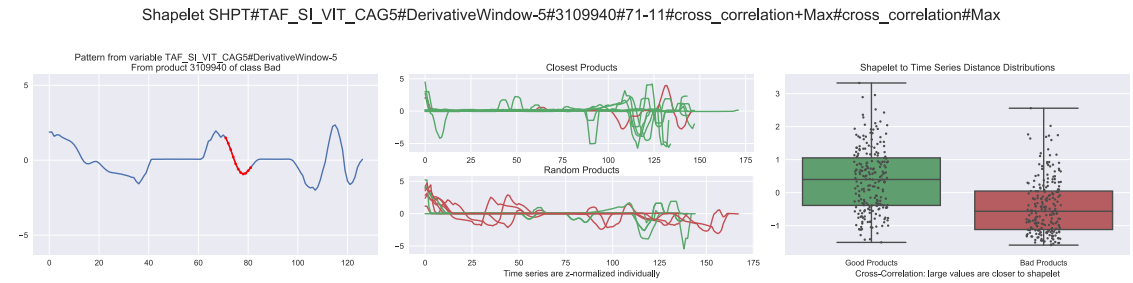
(b)



(c)



(d)



(e)

Figure 9.5: Some EAST-shapelets discovered on ***preprocessed*** time series, from the 2^{nd} use case (DP) with pre-heating as proxy quality label

9.4 Conclusions

In this chapter[1], we have seen that state of the art time series representation and EAST-shapelets systematically outperform simple representation of time series, currently used by the company, on all the industrial uses-cases and for every relevant classification metrics. The "advanced global features" (AGF) by [Fulcher and Jones, 2014] has good classification performances. However its hundreds of features are complex to interpret into industrial process insights. Many EAST-shapelets discovered have been validated by process experts as relevant: it is a clear advantage of motif-based representations confirmed by our proposition. The computational performances and the scalability of our approach have also been shown on industrial datasets.

Nevertheless, we have identified several issues that can be addressed in perspectives. First, there is a feature selection issue: sometimes two different subsets of features perform better independently than together. It seems there is room for improvement to obtain a feature selection more robust to the high dimensionality, feature correlation and noisy features. Second, complex configurations of the subsequence transformations (with time series preprocessing, heterogeneous distance measures and aggregation functions) show potential on the industrial datasets, in particular with noisy time series. This is also a lead in perspective: learn relevant a set of time series preprocessing, distance measures and aggregation functions for the subsequence transformation and quantify the interest of the approach on literature datasets.

[1]The ideas developed in this chapter have been published in [Fricout et al., 2017]

Part IV

Conclusions

Chapter 10

Conclusions & Perspectives

Conclusions

Time series data is everywhere and can be encountered in every scientific or industrial fields. Time series raises specific challenges for common machine learning algorithms and requires specific techniques gathered in the time series mining field.

To handle these issues, most time series mining techniques have in common two solutions: distance measures and time series representations. In this work we have focused on time series representations based on relevant motifs for time series classification. More precisely we have proposed a framework to learn flexible time series representations that aims to discover discriminant sets of subsequence transformations from a time series dataset. The representation is based on motifs extracted from the time series and thus is particularly interpretable for domain experts. We have shown the performances of our proposition on an extensive experimentation using the classical time series mining benchmark with more than one hundred datasets and also on two industrial datasets.

Contributions

We can summarize our contribution in four points:

- We proposed a new motif-based time series representation. Inspired by the time series shapelet principle, our representation is learned from the data and supervised by the classification task at hand.

- The high computational complexity of the discovery of motif-based representations is well known in the literature. We proposed to decrease drastically the time required using a random sampling of the subsequences, taking advantage of their high redundancy in the time series. We have shown the validity of our approach on one hundred datasets from the literature with a rigorous experimental evaluation. We have also

demonstrated the scalability of our approach (the computational complexity of the discovery is not dependent of the number of time series instances in the dataset).

- We proposed to consider the motif-based time series representation as a classical feature space, where each feature is based on the concept of subsequence transformation: a subsequence, a distance measure and an aggregation function are used to transform a set of time series into a feature vector, describing to what extent the subsequence belongs to the time series.
  The obtained feature space allows us to use state of the art feature selection techniques to learn the relevant subsequences to perform the classification. We called the representation obtained "EAST-Representation". The classification and the computational performances have been assessed on one hundred univariate and multivariate time series datasets from the literature: the classification performances are at least as good as the state of the art while the computational complexity is much lower.

- We benchmarked our proposition on several industrial datasets with promising results: the low computational complexity of our approach allows a fast discovery of a motif-based representation on industrial datasets, the classification performances are improved in comparison with current approaches and the learned motifs are relevant according to process experts.

First part: state of the art to learn from time series

The first part of the manuscript is dedicated to an introduction to the time series mining: we review the task, the issues and the solution developed in the literature to overcome them.

In **chapter 2,** we define the main concepts used in this manuscript and the notation. We propose an overview of the time series mining tasks (in this thesis we focus on time series classification). Then we describe the main issues and challenges when training machine learning algorithms on time series. In particular, we see that a time series is not a suitable feature vector for several reasons including the fact that a time series usually suffers from distortions. Strategies have been developed in the literature to overcome these issues: most time series algorithms rely on specific distance measures and time series representations. Two main time series classification approaches exist in the literature: time-based classification (the whole time series is considered and the focus is mainly on the distance measure) and feature-based classification (the focus is mainly on the time series representations to extract classical static features).

Our work is focused on time series representations extracted from the time series to train common classifiers. In **chapter 3,** we perform an overview of the time series representations. We propose a taxonomy to group time series representations into three groups:

time-based representations (a raw time series is transformed into another time series that can be denoised, compressed and with the meaningful information highlighted), feature-based representations (a raw time series is transformed into a classical feature vector with features mainly derived from the feature analysis field, to characterize the structure of the time series for instance) and motif-based representations (meaningful subsequences are discovered in the time series: subsequences can be recurrent, surprising or discriminant).

Second part: our proposition to learn a motif-based representation for time series classification

In this part, we focus on our proposition to learn a motif-based representation for time series classification.

In **chapter 4,** we perform a specific state of art on the discovery of meaningful motif from time series to perform classification. Currently, the major principle is the time series shapelet, whose principle is to discover discriminant subsequences. This principle has produced many variations to handle its drawbacks: the computational complexity to discover the shapelets is very large. This contributes to the second main drawback: the shapelet principle often requires specific algorithms (from the shapelet discovery to the time series classification), which affects its flexibility and its expressiveness.

In **chapter 5,** we formalize our framework for the discovery of a discriminant motif-based representation based on set of motifs meaningful to perform time series classification. We use the concept of subsequence transformation to represent time series: a subsequence transformation $\psi_{a,d}(T_{n,m}, s_{T_{n,m}})$ is setup with a subsequence $s_{T_{n,m}}$ extracted from the time series dataset $\mathcal{D}$, a distance measure d to assess the similarity of the subsequence with the time series and a aggregation function a to summarize the similarity of the subsequence to the time series, such that $\psi_{a,d}(T_{n,m}, s_{T_{n,m}}) = a \circ d(T_n, s_{T_{n,m}})$ whose output is a classical feature. Typically, many subsequence transformations should be computed with different subsequences, distance measures and aggregation functions to form a feature vector X_n to train a classifier f such that $f(X_n) \mapsto y(T_n)$.

However the main issue is the dimensionality of the problem: from moderate size datasets, billions subsequences can be enumerated from $\mathcal{D}$. In **chapter 6**, we discuss this issue and we show that most of the subsequences that we can enumerate from $\mathcal{D}$ are redundant. We show that with a random sampling to draw a small fraction of the exhaustive set of subsequences we preserve the classification performances while dramatically reducing the computation complexity. We demonstrate that the number of subsequences to draw is not linked with the size of the dataset (*ie.* the number of time series in the dataset $\mathcal{D}$): our proposition is scalable. The experimentation shows that a few thousands of subsequences allow to reach state of the art classification performances (instead of the millions or billions used by most literature approaches on time series shapelets).

The large dimensionality reduction of the problem permitted by the random sampling allows in **chapter 7** to cast our proposition into a classical feature selection task on the feature space generated from the subsequence transformations. With this strategy we reach state of the art classification performances with low computation complexity while learning potential complex relationships between the subsequences and possibly with respect to other types of time series features and even typical static features (*eg.* context of the time series measurements).

Third part: industrial applications

Our proposition is benchmarked on industrial datasets of use-cases provided by Arcelormittal.

In **chapter 8**, we present the context of the use-cases, the industrial environment and how the time series are produced. We also describe the content of the datasets.

In **chapter 9**, we benchmark our proposition on the industrial datasets and the classification performances are compared with the current strategy used by the company and other types of time series features proposed in the literature. We show that our proposition has good classification performances and provides insights in the form of subsequences, which have been assessed by the process experts as meaningful.

Perspectives

The relevance of our proposition has been shown in terms of classification performances, computational efficiency and flexibility, which opens several interesting perspectives. In particular, we see three possible ways to improve the time series representation: by enriching the subsequence transformation concept to take advantage of its flexibility (in particular in terms of distance measure and aggregation function), by decreasing the computation complexity by switching from a pure random sampling of the subsequences to a reinforcement learning framework for instance, and finally by improving the robustness of the selection of the set of subsequences.

Enrichment of the general subsequence transformation concept $\psi_{a,d}$

In this work we have formalized a scalable framework for the discovery of a time series representation based on discriminant set of motifs for univariate and multivariate time series. The framework is based on a subsequence transformation $\psi_{a,d}$ of the time series to produce a feature vector suitable to apply classical attribute-based machine learning algorithms. The framework is flexible in that it relies on generic tools that can be changed or gathered to produce a rich representation of the time series and capture heterogeneous

shapes in the time series. The subsequence transformation $\psi_{a,d}$ accepts variations for the following parameters:

- The subsequence transformation $\psi_{a,d}$ can be applied directly on the *raw* time series or any other *time-based* transformation of the data either to highlight specific information or to reduce the dimensionality of the data and accelerate the computations.
- The distance measure d of $\psi_{a,d}$ can be changed for any relevant measure beyond the common Euclidean distance to take advantage of specific properties, such as invariances (for example cross-correlation for scale invariant, dynamic time warping for local warping invariance, Hamming distance, etc.).
- The aggregation function a of $\psi_{a,d}$ can be changed to adapt to different distance measures while preserving the meaning "is the subsequence present in the time series ?" as it is for the pair Euclidean distance and minimum: for instance, for the cross-correlation, the relevant aggregation to get a similar result would be the maximum. The result expected by the aggregation function may also be different, for instance we may be interested in:
  - the number of times the subsequences appears in the time series instead of the value of its closest match. This could be achieved by a count of the number of peaks or troughs in the distance measures d.
  - the position of the subsequence in the time series (where the distance is minimal or maximal depending of the distance measure). This could be achieved with the argument of the aggregation function. This information may be particularly relevant for applications where the position of the subsequence is relevant, for instance peaks in spectrograms.

It may be desirable to combine different configurations of these parameters.

In the theoretical part of this work, we instantiated the EAST-representations with the classical association of distance and aggregation function to evaluate the relevancy of the representation (random sampling, feature space and global feature selection) all other things being equal.

We plan to take advantage of the flexibility of EAST to build more robust and accurate representations of the time series. The first step would be to identify relevant pools of *time-based* representations, distance measures and aggregation functions for the subsequence transformation $\psi_{a,d}$. Then the enriched subsequence transformation should be benchmarked on the literature time series datasets.

In chapter 9, about the benchmarking on industrial applications, we have introduced a first simple enriched subsequence transformation $\psi_{a,d}$, with some preprocessing on the data (EMD, DWT, trivial derivation and integration), an additional distance measure and

aggregation function pair (minimum cross-correlation) and the position of the subsequences (argument). We have shown, on these complex and real-world datasets, the relevance of the enriched subsequence transformation.

Reinforcement learning to lead the subsequence sub-sampling

Even if we demonstrated that the EAST-representation scales with the size of the dataset, if we sum up the additional dimension provided by multivariate time series, additional time-based transformations (they can be seen as new variables of a multivariate time series) and additional pair of distance-aggregation, the search space to discover a discriminant set of subsequence transformations $\psi_{a,d}$ is becoming very large.

Every variable of multivariate time series may not contain subsequences to include in discriminant set of motifs. Also, every *time-based* transformation of the *raw* time series and every pair distance-aggregation may not be meaningful.

The random sampling of the subsequences have been shown to be very efficient to discover discriminant subsequences on univariate time series with a fixed configuration of subsequence transformation $\psi_{a,d}$, in particular when we don't have prior information where the relevant information stands in the time series. However, it may be helpful to develop a mechanism to learn from previously drawn subsequence transformations $\psi_{a,d}$ to lead the discovery and bias the drawing of further subsequence transformation candidates. The idea is to focus on areas of the search space that have been found promising (ie. where meaningful subsequence transformations have been drawn), in particular for the variable and *time-based* preprocessing m, the distance measure d and the aggregation function a.

Reinforcement learning seems to be an interesting option. We can illustrate the overall idea:

1. We have a set of multivariate time series and we look for a discriminant set of subsequence transformations. For the sake of simplicity we only consider a single pair of distance-aggregation: it simplifies the search space. Then the hyper-parameters to draw a subsequence are: the variable m and the parameters to locate a subsequence in a time series (time series T_n, starting position i and length l). We have no clue on the variables that will provide meaningful subsequences for our task: in a first iteration we draw uniformly a set of subsequences among the variables.

2. We evaluate the relevance of each drawn subsequence given an objective function related with our classification task. It is likely that we begin to have an indication on the variables that provide meaningful subsequences.

3. Based on our new knowledge, we bias the draw at the next iteration to focus more on the promising variables, without completely forgetting the remaining variables, which may provide meaningful subsequences in the next iterations.

4. The process iterates until a budget is reached or until the convergence of a particular criterion.

The multi-armed bandit may be suitable for this problem, while other approaches are worth considering.

Improvement of the feature selection stage

In chapter 7, we have cast the discovery of discriminant set of motifs into a feature selection problem. In the feature space generated from subsequence transformations $\psi_{a,d}$, most features are irrelevant and redundant. We have found during the experimentation that a few thousand subsequences drawn from univariate time series datasets is usually sufficient to provide state-of-the-art classification performances. With multivariate time series, complex subsequence transformation configurations (several distance measures and aggregation functions) and additional time series representations the number of features can reach dozen of thousands.

A reinforcement learning strategy to sample more on relevant variables, as discussed in the previous section, will certainly help to reduce the number of features while reducing the number of irrelevant features. But a specific study on the feature selection part may provide significant enhancement. In particular, the problem has specificities that can be useful for the feature selection. For instance, there is a structure in the features: set of subsequences are drawn from the same variables. If some variables, distance measures or aggregation functions are meaningless, all the subsequence transformations $\psi_{a,d}$ are meaningless: by providing their relationship to the feature selection algorithm, is is likely we obtain a faster convergence of the feature selection and possibly a more accurate solution. For instance, approaches like the *group lasso* meet the particular requirements [Friedman et al., 2010].

It will be interesting to investigate further feature selection or dimensionality reduction techniques to solve the issues of our problem: high redundancy, many irrelevant features, learn relationships between features (subsequence transformation based on subsequences), structure in the features, criterion to optimize and computation complexity to allow fast iterative calls to the algorithm.

Bibliography

Bibliography

[Agrawal et al., 1993a] Agrawal, R., Faloutsos, C., and Swami, A. (1993a). Efficient similarity search in sequence databases. *Foundations of data organization and algorithms*, pages 69–84.

[Agrawal et al., 1993b] Agrawal, R., Imieliński, T., and Swami, A. (1993b). Mining association rules between sets of items in large databases. *ACM SIGMOD Record*, 22(2):207–216.

[An et al., 2003] An, J., Chen, H., Furuse, K., Ohbo, N., and Keogh, E. (2003). Grid-based indexing for large time series databases. *IDEAL*, pages 614–621.

[Analyis et al., 2010] Analyis, S., Arima, F., Box-jenkins, T., and Series, T. (2010). 1 What are Time Series ? *Hilary Term, USA*, (1990):1–66.

[André-Jonsson and Badal, 1997] André-Jonsson, H. and Badal, D. Z. (1997). Using signature files for querying time-series data. *European Symposium on Principles of Data Mining and Knowledge Discovery (PKDD)*, 1263:211–220.

[Arathi and Govardhan, 2014] Arathi, M. and Govardhan, A. (2014). Performance of Mahalanobis Distance in Time Series Classification Using Shapelets. *International Journal of Machine Learning and Computing*, 4(4):339–345.

[Arcuri and Briand, 2014] Arcuri, A. and Briand, L. (2014). A Hitchhiker's guide to statistical tests for assessing randomized algorithms in software engineering. *Software Testing Verification and Reliability*, 24(3):219–250.

[Åström, 1969] Åström, K. J. (1969). On the choice of sampling rates in parametric identification of time series. *Information Sciences*, 1(3):273–278.

[Bagnall et al., 2014a] Bagnall, A., Hills, J., and Lines, J. (2014a). Finding Motif Sets in Time Series. *arXiv preprint arXiv:1407.3685*.

[Bagnall et al., 2014b] Bagnall, A., Lines, J., Hills, J., and Bostrom, A. (2014b). Time-Series Classification with COTE: The Collective of Transformation-Based Ensembles. *IEEE Transactions on Knowledge and Data Engineering*, 27(9):1–10.

[Bao and Yang, 2008] Bao, D. and Yang, Z. (2008). Intelligent stock trading system by turning point confirming and probabilistic reasoning. *Expert Systems with Applications*, 34(1):620–627.

[Basu and Meckesheimer, 2007] Basu, S. and Meckesheimer, M. (2007). Automatic outlier detection for time series: An application to sensor data. *Knowledge and Information Systems*, 11(2):137–154.

[Batal et al., 2016] Batal, I., Cooper, G. F., Fradkin, D., Harrison, J., Moerchen, F., and Hauskrecht, M. (2016). An efficient pattern mining approach for event detection in multivariate temporal data. *Knowledge and Information Systems*, 46(1):115–150.

[Batista et al., 2011] Batista, G., Wang, X., and Keogh, E. (2011). A Complexity-Invariant Distance Measure for Time Series. *SIAM International Conference on Data Mining*, pages 699–710.

[Baydogan and Runger, 2015] Baydogan, M. G. and Runger, G. (2015). Learning a symbolic representation for multivariate time series classification. *Data Mining and Knowledge Discovery*, 29(2):400–422.

[Baydogan et al., 2013] Baydogan, M. G., Runger, G., and Tuv, E. (2013). A bag-of-features framework to classify time series. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(11):2796–2802.

[Berndt and Clifford, 1994] Berndt, D. and Clifford, J. (1994). Using dynamic time warping to find patterns in time series. *Workshop on Knowledge Knowledge Discovery in Databases*, 398:359–370.

[Bettaiah and Ranganath, 2014] Bettaiah, V. and Ranganath, H. S. (2014). An Analysis of Time Series Representation Methods. *Proceedings of the 2014 ACM Southeast Regional Conference*, page 16.

[Breiman, 2001] Breiman, L. (2001). Random forests. *Machine Learning*, 45(1):5–32.

[Caiado et al., 2006] Caiado, J., Crato, N., and Pena, D. (2006). A periodogram-based metric for time series classification. *Computational Statistics and Data Analysis*, 50(10):2668–2684.

[Castro and Azevedo, 2010] Castro, N. and Azevedo, P. (2010). Multiresolution Motif Discovery in Time Series. *Proceedings of the 2010 SIAM International Conference on Data Mining*, pages 665–676.

[Castro and Azevedo, 2012] Castro, N. C. and Azevedo, P. J. (2012). Significant motifs in time series. *Statistical Analysis and Data Mining*, 5(1):35–53.

[Cetin et al., 2015] Cetin, M., Mueen, A., and Calhoun, V. (2015). Shapelet Ensemble for Multi-dimensional Time Series. *Proceedings of the SIAM International Conference on Data Mining*, pages 307–315.

[Chakrabarti et al., 2002] Chakrabarti, K., Keogh, E., Mehrotra, S., and Pazzani, M. (2002). Locally adaptive dimensionality reduction for indexing large time series databases. *ACM Transactions on Database Systems*, 27(2):188–228.

[Chen et al., 2015] Chen, Y., Keogh, E., Hu, B., Begum, N., Bagnall, A., Mueen, A., and Batista, G. (2015). The UCR Time Series Classification Archive.

[Chiu et al., 2003] Chiu, B., Keogh, E., and Lonardi, S. (2003). Probabilistic discovery of time series motifs. *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining KDD 03*, 304:493.

[Das et al., 1998] Das, G., Lin, K.-I., Mannila, H., Renganathan, G., and Smyth, P. (1998). Rule discovery from time series. *Knowledge Discovery and Data Mining*, 98:16–22.

[DeBarr and Lin, 2007] DeBarr, D. and Lin, J. (2007). Time Series Classification Challenge Experiments. *proceedings of the Workshop and Challenge on Time Series Classification, at the 13th ACM SIGKDD International Conference on.*, pages 1–5.

[Demsar, 2006] Demsar, J. (2006). Statistical Comparisons of Classifiers over Multiple Data Sets. *Journal of Machine Learning Research*, pages 1–30.

[Deng et al., 2013] Deng, H., Runger, G., Tuv, E., and Vladimir, M. (2013). A time series forest for classification and feature extraction. *Information Sciences*, 239:142–153.

[Ding et al., 2008] Ding, H., Trajcevski, G., and Scheurmann, P. (2008). Querying and mining of time series data: experimental comparison of representations and distance measures. *Proceedings of the VLDB Endowment*, 1(2):1542–1552.

[Esling and Agon, 2012] Esling, P. and Agon, C. (2012). Time-series data mining. *ACM Computing Surveys*, 45(1):1–34.

[Faloutsos et al., 1997] Faloutsos, C., Jagadish, H., Mendelzon, A., and Milo, T. (1997). A Signature Technique for Similarity-based Queries. *Proceedings on Compression and Complexity of Sequences*, pages 2–20.

[Faloutsos et al., 1994] Faloutsos, C., Ranganathan, M., and Manolopoulos, Y. (1994). Fast subsequence matching in time-series databases. *ACM SIGMOD Record*, 23(2):419–429.

[Feng et al., 2013] Feng, Z., Liang, M., and Chu, F. (2013). Recent advances in time-frequency analysis methods for machinery fault diagnosis: A review with application examples. *Mechanical Systems and Signal Processing*, 38(1):165–205.

[Fradkin and Morchen, 2015] Fradkin, D. and Morchen, F. (2015). Mining sequential patterns for classification. *Knowledge and Information Systems*, 45(3):731–749.

[Fricout et al., 2017] Fricout, G., Arnu, D., Neuer, M., Renard, X., Gallinari, P., Leger, J.-B., and Mocci, C. (2017). Data mining continuous sensor data for training plant wide defect models. *European Steel Technology and Application*.

[Friedman et al., 2010] Friedman, J., Hastie, T., and Tibshirani, R. (2010). A note on the group lasso and a sparse group lasso. *arXiv:1001.0736 [math, stat]*, page 8.

[Fu et al., 2008a] Fu, A. W.-C., Keogh, E., Lau, L. Y. H., Ratanamahatana, C. A., and Wong, R. C.-W. (2008a). Scaling and time warping in time series querying. *The VLDB Journal*, 17(4):899–921.

[Fu, 2011] Fu, T. C. (2011). A review on time series data mining. *Engineering Applications of Artificial Intelligence*, 24(1):164–181.

[Fu et al., 2008b] Fu, T. C., Chung, F. L., Kwok, K. Y., and Ng, C. M. (2008b). Stock time series visualization based on data point importance. *Engineering Applications of Artificial Intelligence*, 21(8):1217–1232.

[Fujimaki et al., 2009] Fujimaki, R., Nakata, T., Tsukahara, H., Sato, A., and Yamanishi, K. (2009). Mining abnormal patterns from heterogeneous time-series with irrelevant features for fault event detection. *Statistical Analysis and Data Mining*, 2(1):1–17.

[Fulcher and Jones, 2014] Fulcher, B. D. and Jones, N. S. (2014). Highly comparative feature-based time-series classification. *IEEE Transactions on Knowledge and Data Engineering*, 26(12):3026–3037.

[Fulcher et al., 2013] Fulcher, B. D., Little, M. a., and Jones, N. S. (2013). Supplementary - Highly comparative time-series analysis: the empirical structure of time series and their methods. *Journal of The Royal Society Interface*, 10(83):20130048–20130048.

[Geurts, 2001] Geurts, P. (2001). Pattern extraction for time series classification. *Principles of Data Mining and Knowledge Discovery*, 2168:115–127.

[Ghalwash et al., 2013] Ghalwash, M. F., Radosavljevic, V., and Obradovic, Z. (2013). Extraction of interpretable multivariate patterns for early diagnostics. *Proceedings - IEEE International Conference on Data Mining, ICDM*, pages 201–210.

[Ghalwash et al., 2014] Ghalwash, M. F., Radosavljevic, V., and Obradovic, Z. (2014). Utilizing temporal patterns for estimating uncertainty in interpretable early decision making. *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '14*, pages 402–411.

[Gordon et al., 2012] Gordon, D., Hendler, D., and Rokach, L. (2012). Fast Randomized Model Generation for Shapelet-Based Time Series Classification. *arXiv*.

[Grabocka et al., 2014] Grabocka, J., Schilling, N., Wistuba, M., and Schmidt-Thieme, L. (2014). Learning Time-series Shapelets. *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 392–401.

[Grabocka et al., 2015] Grabocka, J., Wistuba, M., and Schmidt-Thieme, L. (2015). Fast classification of univariate and multivariate time series through shapelet discovery. *Knowledge and Information Systems*, pages 1–26.

[Guo et al., 2010] Guo, C., Li, H., and Pan, D. (2010). An Improved Piecewise Aggregate Approximation Based on Statistical Features for Time Series Mining. *KSEM*, pages 234–244.

[Guyon and Elisseeff, 2003] Guyon, I. and Elisseeff, A. (2003). An Introduction to Variable and Feature Selection. *Journal of Machine Learning Research*, 3:1157–1182.

[Hastie et al., 2009] Hastie, T., Tibshirani, R., and Friedman, J. (2009). The Elements of Statistical Learning. *Springer*, 1.

[He et al., 2012] He, Q., Zhi, D., Zhuang, F., Shang, T., and Shi, Z. (2012). Fast time series classification based on infrequent shapelets. *Proceedings - 2012 11th International Conference on Machine Learning and Applications, ICMLA 2012*, 1:215–219.

[Hegger et al., 1998] Hegger, R., Kantz, H., and Olbrich, E. (1998). Problems in the Reconstruction of High-dimensional Deterministic Dynamics from Time Series. *Nonlinear Analysis of Physiological Data*, pages 23–47.

[Hills et al., 2014] Hills, J., Lines, J., Baranauskas, E., Mapp, J., and Bagnall, A. (2014). Classification of time series by shapelet transformation. *Data Mining and Knowledge Discovery*, 28(4):851–881.

[Hu et al., 2013] Hu, B., Chen, Y., and Keogh, E. (2013). Time series classification under more realistic assumptions. *Proceedings of the thirteenth SIAM conference on data mining (SDM)*, (1):578–586.

[Huang et al., 1998] Huang, N., Shen, Z., Long, S., Wu, M., SHIH, H., ZHENG, Q., Yen, N., Tung, C., and Liu, H. (1998). The empirical mode decomposition and the Hilbert

spectrum for nonlinear and non-stationary time series analysis. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 454(1971):995, 903.

[Hughes, 1968] Hughes, G. (1968). On the Mean Accuracy of Statistical Pattern Recognizers. *IEEE Information Theory Society*, I:55 – 63.

[Hugueney, 2006] Hugueney, B. (2006). Cadre général et algorithmes de constructions pour des représentations symboliques adaptatives de séries temporelles. *Revue MODULAD*, 34:1–12.

[Kadous and Sammut, 2005] Kadous, M. W. and Sammut, C. (2005). Classification of multivariate time series and structured data using constructive induction. *Machine Learning*, 58(2-3):179–216.

[Kandhari, 2009] Kandhari, R. (2009). Anomaly detection. *ACM Computing Surveys*, 41(3):1–6.

[Karlsson et al., 2016] Karlsson, I., Papapetrou, P., and Bostrom, H. (2016). Generalized random shapelet forests. *Data Mining and Knowledge Discovery*, 30(5):1053–1085.

[Keogh et al., 2001] Keogh, E., Chakrabarti, K., Pazzani, M., and Mehrotra, S. (2001). Dimensionality Reduction for Fast Similarity Search in Large Time Series Databases. *Knowledge and Information Systems*, 3(3):263–286.

[Keogh and Kasetty, 2002] Keogh, E. and Kasetty, S. (2002). On the need for time series data mining benchmarks. *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '02*, page 102.

[Keogh and Lin, 2005] Keogh, E. and Lin, J. (2005). Clustering of time-series subsequences is meaningless: Implications for previous and future research. *Knowledge and Information Systems*, 8(2):154–177.

[Keogh et al., 2005] Keogh, E., Lin, J., and Fu, A. (2005). HOT SAX: Efficiently finding the most unusual time series subsequence. *Proceedings - IEEE International Conference on Data Mining, ICDM*, pages 226–233.

[Keogh et al., 2007] Keogh, E., Lin, J., Lee, S.-H. H., and Van Herle, H. (2007). Finding the most unusual time series subsequence: Algorithms and applications. *Knowledge and Information Systems*, 11(1):1–27.

[Keogh et al., 2002] Keogh, E., Lonardi, S., and Chiu, B. (2002). Finding surprising patterns in a time series database in linear time and space. *In KDD*, pages:550–556.

[Keogh and Smyth, 1997] Keogh, E. and Smyth, P. (1997). A probabilistic approach to fast pattern matching in time series databases. *Proceedings of the 3rd International Conference of Knowledge Discovery and Data Mining*, M(1994):52–57.

[Keogh and Pazzani, 2000] Keogh, E. J. and Pazzani, M. J. (2000). Scaling up dynamic time warping for datamining applications. *Knowledge discovery and data mining*, In 6th ACM:285–289.

[Lee et al., 2002] Lee, S., Kwon, D., and Lee, S. (2002). Efficient Pattern Matching of Time Series Data. *Developments in Applied Artificial Intelligence*, pages 586–595.

[Leng, 2009] Leng, M. (2009). Time series representation for anomaly detection. *2009 2nd IEEE International Conference on Computer Science and Information Technology*, pages 628–632.

[Li et al., 1998] Li, C.-S., Yu, P. S., and Castelli, V. (1998). Malm. *Proceedings of the seventh international conference on Information and knowledge management - CIKM '98*, pages 267–272.

[Li et al., 2002] Li, T., Li, Q., Zhu, S., and Ogihara, M. (2002). A survey on wavelet applications in data mining. *ACM SIGKDD Explorations Newsletter*, 4(2):49–68.

[Liao, 2005] Liao, W. (2005). Clustering of time series data survey. *Pattern Recognition*, 38(11):1857–1874.

[Lin et al., 2002] Lin, J., Keogh, E., Lonardi, S., and Patel, P. (2002). Finding motifs in time series. *Proc. of the 2nd Workshop on Temporal Data Mining*, pages 53–68.

[Lin et al., 2005] Lin, J., Vlachos, M., Keogh, E. J., Gunopulos, D., Liu, J., Yu, S., and Le, J. (2005). A MPAA-based iterative clustering algorithm augmented by nearest neighbors search for time-series data streams. *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 333–342.

[Lin et al., 2012] Lin, J., Williamson, S., Borne, K., and DeBarr, D. (2012). Pattern Recognition in Time Series. *Advances in Machine Learning and Data Mining for Astronomy*, pages 617–645.

[Lines and Bagnall, 2012] Lines, J. and Bagnall, A. (2012). Alternative quality measures for time series shapelets. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7435 LNCS:475–483.

[Lines and Bagnall, 2015] Lines, J. and Bagnall, A. (2015). Time series classification with ensembles of elastic distance measures. *Data Mining and Knowledge Discovery*, 29(3):565–592.

[Lines et al., 2012] Lines, J., Davis, L., Hills, J., and Bagnall, A. (2012). A shapelet transform for time series classification. *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 289–297.

[Lkhagva et al., 2006] Lkhagva, B., Suzuki, Y., and Kawagoe, K. (2006). Extended sax: Extension of symbolic aggregate approximation for financial time series data representation. *DEWS2006 4A-i8*.

[Man and Wong, 2001] Man, P. W. P. and Wong, M. H. (2001). Efficient and robust feature extraction and pattern matching of time series by a lattice structure. *International Conference on Information and Knowledge Management, Proceedings*, pages 271–278.

[Meinshausen and Buhlmann, 2010] Meinshausen, N. and Buhlmann, P. (2010). Stability selection. *Journal of the Royal Statistical Society. Series B: Statistical Methodology*, 72(4):417–473.

[Minnen et al., 2007] Minnen, D., Isbell, C. L., Essa, I., and Starner, T. (2007). Discovering multivariate motifs using subsequence density estimation and greedy mixture learning. In *Aaai '07*, volume 22, pages 615–620.

[Moerchen, 2006] Moerchen, F. (2006). Algorithms for time series knowledge mining. *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '06*, (2):668.

[Morchen, 2003] Morchen, F. (2003). Time series feature extraction for data mining using DWT and DFT. Technical report.

[Mörchen and Ultsch, 2005] Mörchen, F. and Ultsch, A. (2005). Optimizing time series discretization for knowledge discovery. *Proceeding of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining - KDD '05*, page 660.

[Moskovitch and Shahar, 2015] Moskovitch, R. and Shahar, Y. (2015). Classification-driven temporal discretization of multivariate time series. *Data Mining and Knowledge Discovery*, 29(4):871–913.

[Mueen, 2013] Mueen, A. (2013). Enumeration of Time Series Motifs of All Lengths. In *2013 IEEE 13th International Conference on Data Mining*, pages 547–556.

[Mueen et al., 2015] Mueen, A., Hamooni, H., and Estrada, T. (2015). Time Series Join on Subsequence Correlation. *Proceedings - IEEE International Conference on Data Mining, ICDM*, 2015-Janua(January):450–459.

[Mueen et al., 2011] Mueen, A., Keogh, E., and Young, N. (2011). Logical-shapelets: an expressive primitive for time series classification. *the 17th ACM SIGKDD international conference*, pages 1154–1162.

[Mueen et al., 2009] Mueen, A., Keogh, E., Zhu, Q., Cash, S., and Westover, B. (2009). Exact Discovery of Time Series Motifs. *Proceedings of the 2009 SIAM International Conference on Data Mining*, pages 473–484.

[Nanopoulos et al., 2001] Nanopoulos, A., Alcock, R., and Manolopoulos, Y. (2001). Feature-based classification of time-series data. *Information processing and Management*, 0056:49–61.

[Nason and von Sachs, 1999] Nason, G. U. B. and von Sachs, R. U. L. (1999). Wavelets in time series analysis. *Philosophical Transactions of the Royal Society of London A*, 357(1760):1–16.

[Ohsaki and Sato, 2002] Ohsaki, M. and Sato, Y. (2002). A rule discovery support system for sequential medical data, in the case study of a chronic hepatitis dataset. *Proceedings of the International Workshop on Active Mining (AM '02) in International Conference on Data Mining (ICDM '02)*, pages 97–102.

[Patel et al., 2008] Patel, D., Hsu, W., and Lee, M. L. (2008). Mining relationships among interval-based events for classification. *Proceedings of the 2008 ACM SIGMOD international conference on Management of data - SIGMOD '08*, page 393.

[Patri et al., 2014] Patri, O. P., Sharma, A. B., Chen, H., Jiang, G., Panangadan, A. V., and Prasanna, V. K. (2014). Extracting discriminative shapelets from heterogeneous sensor data. *2014 IEEE International Conference on Big Data (Big Data)*, pages 1095–1104.

[Perng et al., 2000] Perng, C.-S. C.-S., Wang, H., Zhang, S. R., and Parker, D. S. (2000). Landmarks: a new model for similarity-based pattern querying in\ntime series databases. *Proceedings of 16th International Conference on Data Engineering*, pages 33–42.

[Pratt and Fink, 2002] Pratt, K. B. and Fink, E. (2002). Search for Patterns in Compressed Time Series. *International Journal of Image and Graphics*, 02(01):89–106.

[Qu et al., 1998] Qu, Y., Wang, C., and Wang, X. S. (1998). Supporting fast search in time series for movement patterns in multiple scales. *Proceedings of the seventh international conference on Information and knowledge management - CIKM '98*, pages 251–258.

[Quoc et al., 2008] Quoc, N., Hung, V., and Anh, D. T. (2008). An Improvement of PAA for Dimensionality Reduction. *PRICAI 2008: Trends in Artificial Intelligence*, pages 698–707.

[Rakthanmanon and Keogh, 2013] Rakthanmanon, T. and Keogh, E. (2013). Fast shapelets: A scalable algorithm for discovering time series shapelets. *Proceedings of the thirteenth SIAM conference on data mining (SDM)*, pages 668–676.

[Ratanamahatana and Keogh, 2004] Ratanamahatana, C. A. and Keogh, E. (2004). Making Time-series Classification More Accurate Using Learned Constraints. *Proceedings of the 2004 SIAM International Conference on Data Mining*, pages 11–22.

[Ratanamahatana et al., 2010] Ratanamahatana, C. A., Lin, J., Gunopulos, D., Keogh, E., Vlachos, M., and Das, G. (2010). Mining Time Series Data. In *Data minin and knowledge discovery handbook*, pages 1069–1103. Springer.

[Renard et al., 2015] Renard, X., Rifqi, M., Erray, W., and Detyniecki, M. (2015). Random-shapelet : an algorithm for fast shapelet discovery. *IEEE International Conference on Data Science and Advanced Analytics*, pages 1–10.

[Renard et al., 2016a] Renard, X., Rifqi, M., Fricout, G., and Detyniecki, M. (2016a). EAST Representation: Fast Discriminant Temporal Patterns Discovery From Time Series. *ECML/PKDD Workshop on Advanced Analytics and Learning on Temporal Data.*

[Renard et al., 2016b] Renard, X., Rifqi, M., Fricout, G., and Detyniecki, M. (2016b). https://github.com/xrenard/EAST-Representation.

[Shatkay and Zdonik, 1996] Shatkay, H. and Zdonik, S. B. (1996). Approximate queries and representations for large data sequences. *Proceedings of the Twelfth International Conference on Data Engineering*, (March):536–545.

[Struzik and Siebes, 1999] Struzik, Z. R. and Siebes, A. (1999). The Haar wavelet transform in the time series similarity paradigm. *Principles of Data Mining and Knowledge Discovery*, pages 12–22.

[Tanaka et al., 2005] Tanaka, Y., Iwamoto, K., and Uehara, K. (2005). Discovery of time-series motif from multi-dimensional data based on MDL principle. *Machine Learning*, 58(2-3):269–300.

[Wang and Megalooikonomou, 2008] Wang, Q. and Megalooikonomou, V. (2008). A dimensionality reduction technique for efficient time series similarity analysis. *Information Systems*, 33(1):115–132.

[Wang et al., 2013] Wang, X., Mueen, A., Ding, H., Trajcevski, G., Scheuermann, P., and Keogh, E. (2013). Experimental comparison of representation methods and distance measures for time series data. *Data Mining and Knowledge Discovery*, 26(2):275–309.

[Wang et al., 2005] Wang, X., Smith, K. A., and Hyndman, R. J. (2005). Dimension reduction for clustering time series using global characteristics. *Proceedings of the International Conference on Computational Science*, 3516:792–795.

[Weiss, 2004] Weiss, G. M. (2004). Mining with Rarity: A Unifying Framework. *SIGKDD Explorations*, 6(1):7–19.

[Wistuba et al., 2015] Wistuba, M., Grabocka, J., and Schmidt-Thieme, L. (2015). Ultra-Fast Shapelets for Time Series Classification. *CoRR abs/1503.05018*, cs.LG.

[Xi et al., 2007] Xi, X., Keogh, E. J., Wei, L., and Mafra-Neto, A. (2007). Finding Motifs in Database of Shapes. In *Proceedings of the 2007 SIAM international conference on data mining*, pages 249–260.

[Xing et al., 2010] Xing, Z., Pei, J., and Keogh, E. (2010). A brief survey on sequence classification. *ACM SIGKDD Explorations Newsletter*, 12(1):40.

[Xing et al., 2011] Xing, Z., Yu, P. S., and Wang, K. (2011). Extracting Interpretable Features for Early Classification on Time Series. *SIAM International Conference on Data Mining*, pages 247–258.

[Yankov et al., 2007] Yankov, D., Keogh, E., Medina, J., Chiu, B., and Zordan, V. (2007). Detecting time series motifs under uniform scaling. *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining KDD 07*, page 844.

[Ye and Keogh, 2009] Ye, L. and Keogh, E. (2009). Time series shapelets: a new primitive for data mining. *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 947–956.

[Ye and Keogh, 2011] Ye, L. and Keogh, E. (2011). Time series shapelets: a novel technique that allows accurate, interpretable and fast classification. *Data Mining and Knowledge Discovery*, 22(1-2):149–182.

[Yi and Faloutsos, 2000] Yi, B.-K. and Faloutsos, C. (2000). Fast Time Sequence Indxing for Arbitrary Lp Norms. In *VLDB*.

[Zhang and Thomas, 2003] Zhang, L. and Thomas, B. G. (2003). Inclusions in continuous casting of steel. *Proceedings of the XXIV National Steelmaking Symposium*, pages 138–183.

[Zhang et al., 2009] Zhang, X., Wu, J., Yang, X., Ou, H., and Lv, T. (2009). A novel pattern extraction method for time series classification. *Optimization and Engineering*, 10(2):253–271.