Department of Informatics
University of Fribourg (Switzerland)

GENERIC DATA-DRIVEN APPROACHES TO TIME SERIES CLASSIFICATION

THESIS

presented to the Faculty of Science, University of Fribourg (Switzerland)
in consideration for the award of the academic grade of
Doctor Scientiarum Informaticarum

by

Christophe GISLER

from

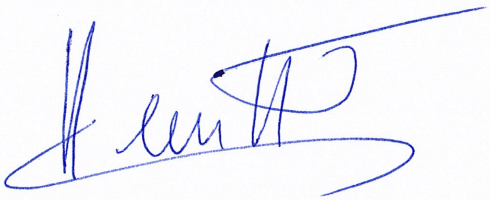
Belmont-Broye FR (Switzerland)

Thesis N° 2014
UniPrint, Fribourg
2017

Accepted by the Faculty of Science of the University of Fribourg (Switzerland) upon the recommendation of:

- Dr. Jean Hennebert, University of Fribourg, Switzerland & University of Applied Sciences Western Switzerland (Thesis Director)
- Prof. Rolf Ingold, University of Fribourg, Switzerland (Thesis Godfather & Examiner)
- Dr. Robert van Kommer, Innovation Mentor at Alliance, Innovation Park, EPFL, Switzerland (External Examiner)
- Prof. Andres Perez-Uribe, University of Applied Sciences Western Switzerland (External Examiner)
- Prof. Laurent Donzé, University of Fribourg, Switzerland (Jury President)

Fribourg, April 27th, 2017

Thesis Director	Faculty Dean
	
Dr. Jean Hennebert	Prof. Christian Bochet

I believe that the first objective of a thesis is to learn how to do research, and this implies working by yourself, developing your own ideas and discovering your mistakes.

I also believe that doing research is trying and trying again to open new doors. If you do not close this door, if you discover a wall behind it or if you don't explore entirely the room thus unveiled, it does not matter. The important, vital gesture, is to open the door.

Jean Hennebert, thesis director

Remerciements

Je tiens à remercier du fond du coeur:

- Jean Hennebert, mon directeur de thèse, mais avant tout cet homme qui est devenu mon ami, pour son précieux soutien, ses conseils toujours pertinents, son éternelle bonne humeur et toutes les chouettes discussions échangées autour d'une bière, si possible belge, dans les troquets alentours;
- Rolf Ingold, expert et parrain de cette thèse, qui m'a fait découvrir le monde fascinant du *machine learning* durant ses cours *Image Processing* et *Pattern Recognition* à l'université;
- Robert van Kommer et Andres Perez-Uribe, experts externes qui ont pris le temps de lire et juger cette thèse avec soin et attention;
- Antonio Ridi, mon cher collègue avec lequel j'ai travaillé sur tous ces passionnants projets de *machine learning*, pour sa bonne humeur, son soutien et les éclats de rires partagés;
- Mes collègues de l'école d'ingénieurs, de l'université et de l'entreprise Sensimed, pour tous les bons moments passés ensemble au travail ou en dehors;
- Ma famille et mon épouse Batoul, pour leur amour, leur présence et leur soutien au quotidien;
- Mes amis, pour leur soutien et leur humour me rappelant qu'il est bon de ne pas se prendre trop au sérieux.

Toutes ces personnes m'ont fait progresser, fait voir d'autres perspectives, d'autres mondes, tout aussi beaux et variés, voire empreints de notes de musique. Grâce à eux, je me souviens que...

Tout seuls, nous ne pouvons rien.

Résumé

Les séries temporelles (*time series*) représentent une grande partie des données acquises dans le monde informatique et de nombreuses tâches d'exploration de données (*data mining*), telles que la prévision et la classification, doivent les traiter. La plupart des algorithmes d'apprentissage automatique (*machine learning*) ne fonctionnent pas bien avec les séries temporelles en raison de leur structure de données unique qui exige des approches particulières utilisant des représentations de haut niveau.

Dans cette thèse, nous avons fait état de nos recherches dans le domaine des approches génériques pour la classification des séries temporelles. Après un survol des méthodes couramment utilisées dans l'exploration des données des séries temporelles, nous avons pris deux études de cas différents dans le but de présenter, appliquer et évaluer deux approches génériques pour la classification des séries temporelles multivariées : une globale et une locale. L'approche globale consiste à extraire un unique vecteur de caractéristiques globales de chaque série temporelle multivariée. La deuxième approche consiste à diviser chaque série temporelle en segments locaux et à extraire autant de vecteurs de caractéristiques que de segments.

La première étude de cas concernait la tâche de l'identification automatique des appareils ménagers, en fonction de leurs signatures de consommation électrique mesurées avec un capteur bas de gamme placé sur la prise électrique. Nous avons créé deux bases de données de signatures d'appareils, ACS-F1 et F2, disponibles librement pour la communauté scientifique. La deuxième étude de cas concernait la tâche de détection automatique de la maladie du glaucome chez des patients, en fonction de leurs profils de pression intraoculaire acquis 24 heures durant, au moyen d'un nouveau type de lentille de contact muni d'un capteur développé par l'entreprise suisse Sensimed.

Pour chaque approche et problème de classification, diverses caractéristiques génériques, dont certaines basées sur des représentations telles que la *Piecewise Aggregate Approximation* et la *Symbolic Aggregate Approximation*, ont été extraites, et trois algorithmes de classifica-

tion ont été testés et comparés : machine à vecteurs de support, perceptron multicouche et modèle à mixture de gaussiennes. Pour chaque problème, chaque base de données et protocole de test, les approches globale et locale ont donné des résultats plutôt similaires. Au travers de tests systématiques, nous avons démontré que l'approche globale fournit une représentation des données plus légère et un traitement plus efficace de la part des classifieurs, en termes de temps de calcul et d'évolutivité.

Enfin, nous avons présenté notre outil de classification générique de series temporelles (*Generic Time Series Classification Tool*) élaboré au cours des études de cas. Cette application JavaFX facile à utiliser respecte le processus d'exploration de données (*data mining process*) et permet notamment d'importer, visualiser, annoter, sélectionner, représenter et classifier les séries temporelles en utilisant des approches génériques. Il a été utilisé avec succès pour réaliser toutes nos expériences et démontrer que nos approches génériques sur les séries temporelles pouvaient être appliquées indépendamment à différents problèmes de classification. Ainsi, une contribution principale de cette thèse a été l'élaboration d'une méthodologie générique pour la classification des séries temporelles. Cette méthodologie implique un processus systématique et semi-automatisé permettant de trouver, grâce à l'outil développé, la meilleure configuration de caractéristiques extraites et de paramètres de l'algorithme de classification.

Abstract

Time series represent a large part of the data supply worldwide and many data mining tasks, such as prediction and classification, are concerned with them. Most machine learning algorithms do not work well on time series because of their unique data structure which requires particular approaches, using high level representations.

In this thesis, we have reported on our research in the field of generic approaches to time series classification. After a survey on the methodologies commonly used for mining time series data, we took two different case studies to present, apply and evaluate two generic approaches to multivariate time series classification: a global and a local one. The global approach consisted in extracting one global unique feature vector from each multivariate time series. The second approach consisted in dividing each time series in local segments, and extracting as many feature vectors as segments.

The first case study concerned the task of automatic identification of home appliances, based on their electric consumption signatures recorded with a low-end smart outlet sensor. We have built two databases of appliance consumption signatures, ACS-F1 and F2, made freely available for the scientific community. The second case study concerned the task of automatic detection of the glaucoma disease by patients, based on their 24-hour intraocular pressure profiles acquired with a new type of contact lens sensor developed by the Swiss company Sensimed.

For each approach and classification problem, various generic features, some based on representations like Piecewise Aggregate Approximation and Symbolic Aggregate Approximation, were extracted, and three classification algorithms were tested and compared: Support Vector Machine, Multilayer Perceptron, and Gaussian Mixture Model. For each problem, each database and test protocol, global and local approaches provided rather similar results. Through systematic tests, we have demonstrated that the global approach provides a lighter representation of the data, and a more efficient handling by the classifiers, in terms of com-

putation time and scalability.

At last, we presented the Generic Time Series Classification Tool elaborated during the case studies. This easy-to-use JavaFX application respects the data mining process and provides functionalities to import, visualize, annotate, select, represent, and classify time series using generic approaches. It was successfully used to perform all our experiments, and show that our generic approaches to time series could be applied to different classification problems independently. Hence, a main contribution of this thesis was the proposition of a generic methodology for time series classification. This methodology involves a systematic and semi-automated process to discover, thanks to the developed tool, the best configuration of extracted features and classifier parameters.

Contents

1

Introduction

Shoot for the moon. Even if you miss it, you will land among the stars.

Oscar Wilde

Contents

1.1 Context

The world is growing! Not only in terms of population, but also in terms of information. Everyday, the world is getting more and more interconnected, automatised, computerised and, some will even say, *smart*[1]. At the time of writing this thesis, it is the era of *Big Data*, *Internet of Things* and *Mobile Computing*. Along with the development of cloud and parallel computing, data storage capacities and computing performances have increased while their costs have come down. Everywhere, computers watch, record and store numerous pieces of information that they encounter, either directly from networks like the Internet, or by means of physical sensors of all sorts, like smart meters, building sensors or biomedical sensors.

[1]Many project calls have been using the adjective *smart*, e.g. *smart cities*, *home*, *living* or *sensors*.

Amongst all kinds of data, those recorded by sensors along with the time are the most widespread and commonly referred to as *time series* [Challapalli, 2014]. Often just stored and kept at disposal on servers, these data potentially contain useful information and knowledge which are not tangible at first sight. Thus, electric data recorded by smart meters may not only reflect the way people consume electricity in a house, but also their activities through the use of their various electric appliances. Intraocular pressure related data measured by a contact lens sensor[2] may not only reflect the way intraocular pressure varies during the day, but also the presence of a disease such as glaucoma and the way it evolves. Many people and companies now want to exploit all the data that they have piled up. However, this is a difficult task, not only because of the complexity and the quantity of data, but also because sometimes they don't even know what to look for in their data. This is where machine learning methodologies come into play. The purpose of machine learning algorithms is to learn knowledge from data, by building a model from examples and using it in order to solve problems such as classifying data, taking decisions or making predictions. That said, between traditional machine learning algorithms and raw data stands an important step, the *feature extraction*, which consists in building new related values (*features*) intended to be informative, non redundant, discriminative and facilitating the machine learning task to do. In general, features are new simpler representations of the data, potentially leading to a better human comprehension of the data and the problem to solve.

1.1.1 Time Series and Machine Learning Everywhere

As mentioned above, the data having a *time* dimension, that is for which measurements have been recorded repetitively as a function of time fall in the category of time series. Time series represents a large fraction of the data supply worldwide. A random set of 4'000 charts coming from 15 world famous newspapers published from 1974 to 1989 revealed that more than 75% of them were time series [Tufte, 1986]. Over the last years, there has been an growing interest in time series data mining and machine learning. Whether it be in the biomedical field (e.g. electrocardiogram, electroencephalogram, intraocular pressure, gene expression data), in biometry (e.g. speech, gait, handwriting, signatures), in finance (e.g. financial markets, capital flows), in security (e.g. intrusions, anomalies, abnormal activities), in ecology (e.g. biological system evolutions), sustainable development (e.g. electric consumption, energy saving) or in meteorology (e.g. weather forecast), time series are everywhere!

Although statisticians have been working with time series for more than a century, many of their methods have shown limitations when dealing with large and complex time series databases. Rather than analyzing some statistical properties on the time series, we are often more concerned about discovering useful and pertinent information from the data efficiently [Maimon and Rokach, 2010, Chapter 56]. As for time series data, machine learning

[2] Like the contact lens sensor Triggerfish® developed by the Swiss company *Sensimed* [Sensimed, 2014a].

algorithms are involved in quite a large number of applications and domains. From speech recognition to disease diagnosis, including user profiling and capital flows predictions, many tasks of data classification, regression, clustering, density estimation or dimensionality reduction are often figured out with machine learning methodologies. Therefore, the research on machine learning applied to time series is of great interest.

1.1.2 Old and New Trends in Machine Learning

Standard machine learning methods such as Support Vector Machine (SVM), Artificial Neural Network (ANN), Hidden Markov Model (HMM) or Gaussian Mixture Model (GMM) have proven to be powerful and efficient in a large number of domains and applications. Although they are still widespread and much used, they do have drawbacks. Such methods allow the computer to extract knowledge through a supervised experience, which typically requires a human operator to help the machine learn by giving it many preprocessed and selected training examples (*training samples*). Most of the time, this task is very time consuming and error-prone. Some machine learning experts say that standard approaches do not really reflect any form of machine intelligence, since they much rely on human cleverness to invent and encode the hand-engineered features that will permit the computer to learn [Dormehl, 2014]. However, thanks to this human involvement in a feature extraction phase, current algorithms need less data to learn. This is fondamental, notably when very few training data are available, as in the glaucoma disease recognition task of Chapter 4. Furthermore, the fact that the feature extraction phase is separated from the learning and classification phase constitutes a major advantage. Features are discernable, understandable and interpretable. Even the most mathematically abstract ones often have a signification! This criterion of interpretability is also required in some applications such as in the medical domain as explained below. Some will say that the human operator can be seen as a performant teacher helping his classifier (the student), learn rapidly, by making raw data (the subject matter) more approachable. From this point of view, such methods deserve to be qualified as machine learning methods, because at the end, this is the machine that learns and builds its model.

To address the drawbacks of standard machine learning algorithms and also with the desire to tackle the problem of learning representations rather than just simple classifiers, some researchers came up with new *deep learning* approaches allowing unsupervised training (or unsupervised pre-training, followed by supervised fine-tuning) [Piatetsky, 2014]. Deep learning algorithms can be seen as a rebranding of neural networks but differ from standard machine learning approaches by their ability to learn multiple and hierarchical representations of data (*features*) automatically, thus replacing handcrafted features that would have been engineered with difficulty, even by specialists of the domain [Bengio et al., 2013]. In general, these methods work well with structured data available in abundance. Therefore until now, they have been mostly applied to the classification of images, audio and video

data collected through the Internet, and rather not to time series data sparingly collected via physical sensors. However, some recurrent neural networks (RNN), such as simple Elman and Jordan networks trained by backpropagation through time or more complex Long-Short Term Memory (LSTM) networks, are well adapted to solve problems of natural language processing, sequence to sequence learning and time series prediction [Sutskever et al., 2014; Lipton et al., 2015]. LSTM are the state-of-the-art for these applications at the time of writing this thesis. Basic principles of deep learning methods are not recent (1980), but could efficiently be applied only recently (∼2010) as they require much data and computing power, such as GPU or parallel computing, to run. In addition to this, deep learning methods do have their drawbacks too, one major being the difficulty to represent and understand the features learnt in an unsupervised way, which can be really problematic in some cases, like disease diagnosis for which medical doctors and healthcare specialist really need to know and understand which feature carried weight in the decision process.

Thus, some computer scientists say that we currently stand at a point where deep learning approaches and standard machine learning approaches (with more and more efficient and advanced algorithms) tend to face/confront one another. In fact, they simply progress side by side. In this thesis, the framework and the nature of the considered time series problems, where few data were available and some interpretability on the features was required, pushed us to focus and use standard machine learning approaches.

1.1.3 Generic Machine Learning Approaches to Particular Problems

Whatever machine learning (or data mining) approach is adopted, some things are invariable. Each problem to solve has specificities and needs some dedicated engineering work to implement a machine learning algorithm. The task can take a lot of time and be expensive. Although the way of applying and using the machine learning algorithms themselves does not much depend on the nature of the data and the task to perform, the way of comprehending and preprocessing data highly depends on it. From this assessment has emerged the interest to head for generic approaches and develop dedicated generic machine learning frameworks or tools, which could constitute a real progress and benefit for machine learning specialists as well as customers. Furthermore, using a generic approach does not prevent the interpretation and the understanding of the returned output through the extracted features. For instance, some frequency band features generically extracted may be easily linked to a particular physiological trait in the case of an electrocardiogram signal analysis (with a medical advice of course).

In the field of time series data, one big advantage is their format. Although they potentially come from so different domains, they have all in common the property that their values are bound, ordered and aligned with the time. A large number of time series representations and features already exists and can thus be generically exploited to perform various

machine learning tasks, regardless of the data origin. Moreover, time series data often go along with various metadata of all kinds. Finding an automatic and generic way to combine them with the main time series data may bring supplementary knowledge to the machine learning process and improve the performance of algorithms.

1.2 Constitutive Hypothesis and Research Questions

From the preceding assessments and the actual State of the Art (see Chapter 2), the following constitutive hypothesis are made in our research work:

- Time series data represent a large part of the data supply worldwide.
- Many data mining tasks, such as classification, regression and prediction problems (usually solved by machine learning techniques) have to deal with time series and more specifically with multivariate time series.
- Most of the data mining tasks done on time series and the relative research work concern time series prediction (forecasting), indexing or anomaly detection problems, while much less concern classification problems [Aggarwal, 2015, Chapter 14].
- Existing machine learning algorithms and their performance do not only depend on the problem to solve, but also on the data themselves, in terms of quality, quantity and origin (field).
- Numerous high level representations of time series and dedicated features have been proposed and used (notably in the feature extraction phase of traditional machine learning processes) in various problems.
- New deep learning approaches have emerged, with abilities to learn features in an unsupervised way. They have advantages and disadvantages that traditional machine learning approaches don't have. While having shown to work well with big sets of structured data, like images and videos coming from the Internet, they shall not work with small sets of multivariate time series data sparingly collected via physical sensors. Moreover, features learnt by current deep learning methods are often difficult to represent and interpret, which is still required in numerous problems like disease diagnosis.
- In every machine learning process, the preprocessing phase and the representation of the data, especially time series, is fundamental.
- A majority of time series data come with meaningful metadata of various types (numbers, words, lists, enumerations). These metadata are sometimes left out from the machine learning process because they cannot be easily or automatically handled (i.e. without human manual intervention). To the best of our knowledge, there currently

exists no approach, nor tool, for handling them generically in order to combine them efficiently and automatically to time series features during the machine learning process.

- There currently exists no simple, intuitive and efficient dedicated machine learning framework, nor tool for handling time series data and solve related classification problems in a generic manner.

From these points, the research questions of this thesis are:

1. With standard machine learning approaches (notably involving a standard feature extraction phase) and among the existing high level time series representations and features, is it possible to set up procedures that automatically find efficient sets of features for a given time series problem?

2. Is it possible to elaborate and apply a generic machine learning approach to solve various classification problems concerning multivariate time series data? In other words, is it possible to create a generic time series classification tool notably able to automatically:

   (a) Import, visualize and handle all kinds of time series sets in a generic way?

   (b) Detect and handle problems specific to time series, such as jumps (abrupt changes), outliers, holes (missing values)?

   (c) Represent and adapt a time series problem to be automatically tackled by various general classification algorithms?

   (d) Extract efficient generic time series features during the machine learning process and build consistent feature vectors for the classification task, whatever the shape of time the classification problem will be?

   (e) Build efficient time series training and test sets, which are robust to variations and adapted to the classification problem?

   (f) Test and compare various machine learning algorithms for a given problem?

   (g) Represent and format the classification results obtained with the various approaches and algorithms?

In order to answer these questions, two rather different classification problems were analyzed as case studies in this thesis, namely:

1. A classification of appliance consumption signatures, with the purpose to detect appliance categories, and hence make energy savings or new smart domotics applications (see Chapter 3);

2. A classification of intraocular pressure profiles measured by using a new type of contact lens sensor, with the purpose to detect (diagnose) the glaucoma eye disease by patients, and hence help medical doctors to give them the appropriate treatment (see Chapter 4).

From these two specific classification problems have emerged the following questions:

1. Is it possible to apply the same generic time series classification approaches to these two different problems?

2. How will our generic time series classification methods perform and deal in such situations where few training data are available?

3. Will the results obtained be trustable and what conclusions will be made?

1.3 Contributions

The main contributions of this thesis are:

- The elaboration of two generic data-driven approaches to time series classification, namely a *global* and a *local* approach, either considering each time series as a global entity, or a set of local parts, taken then for generic feature extraction and classification;

- The application, evaluation, and comparison of these generic approaches for two classification problems (case studies), namely for appliance detection and glaucoma detection;

- The specific research work, results and knowledge brought in the particular fields of the two case studies, and notably:

  - The creation of two consequent databases of appliance consumption signatures (ACS) coming along with test protocols: ACS-F1 and ACS-F2;
  - The elaboration and integration of some specific algorithms (e.g. for eye blink and ocular pulse detection) and models in dedicated medical assistant softwares for intraocular pressure (IOP) analysis and glaucoma diagnosis in collaboration with the Swiss company Sensimed and eye specialists.

- The creation of an easy-to-use Generic Time Series Classification Tool (GTSCT) allowing to perform any task of time series classification by applying the preceding generic approaches, using state-of-the-art time series representations and features, and providing advanced visualization, tuning, and reporting functionalities;

- The creation of three simple Java libraries for Support Vector Machine, Multilayer Perceptron, and Gaussian Mixture Model, namely *SVM4J*, *MLP4J*, and *GMM4J*;

The **publications** related to this thesis are listed at the end of this document.

1.4 Thesis Outline

This thesis is organized as follows:

Chapter 2 – Fundamentals gives the key elements of theory which are required to tackle this thesis. All the fundamentals related to a generic data-driven approach to time series classification are introduced and detailed in this part. It describes the commonly used time series mining methods, the general data mining process, the classification task, as well as the classification algorithms used in our experiments. Everything described was effectively used often several times in the three next chapters of this thesis. This chapter is kind of "reference book" for all theoretic elements used, and allows to get to the point faster along the description of our experiments.

Chapter 3 – Case Study: Appliance Consumption Signature Classification presents a first case study used to present, test and evaluate our global and local generic approaches to time series classification. We applied them with three different classification algorithms, namely SVM, Multilayer Perceptron (MLP) and GMM, in the task of automatic identification/recognition of categories of home appliances, based on their electric consumption signatures. The created databases of appliance consumption signatures, as well as the appliance recognition test protocols specially elaborated for the machine learning problem, are also described.

Chapter 4 – Case Study: Intraocular Pressure Profile Classification presents a second case study used to implement, test and evaluate our global and local generic approaches to time series classification. We applied them with two different classification algorithms, namely SVM and GMM, in the task of automatic detection/recognition of the glaucoma disease by patients, based on their intraocular pressure profiles recorded with a new type of contact lens sensor during 24 hours. Two sequential and one genetic algorithms were applied to perform the feature selection.

Chapter 5 – A Tool for Generic Time Series Classification presents the Generic Time Series Classification Tool (GTSCT) which was created during this research work and implements both global and local generic approaches to time series classification. The specific time series problems pushing toward a generic time series data mining approach and tool are described. An overview of the application along with its functionalities to import, visualize, handle, represent, and classify time series is done. The three SVM, MLP and GMM Java libraries created for the tool (but not only) are also presented.

Chapter 6 – Conclusion closes this thesis by giving a general conclusion and presenting the perspectives related to this research work.

2

Fundamentals

Perplexity is the beginning of knowledge.

Khalil Gibran

Contents

2.1 Introduction

2.1.1 Purpose

This chapter gives the key elements of theory which are required to tackle this thesis. All the fundamentals needed to adopt a generic data-driven approach to time series classification are introduced and detailed in this chapter. This chapter is kind of "reference book" for all theoretic elements effectively used, often several times, in the three next chapters, and it allows to get to the point faster along the description of our experiments.

2.1.2 Outline

First, in Section 2.2, we introduce time series, which constitute the heart of this thesis, and describe the time series preprocessing and representation methods commonly used for the

different time series data mining tasks, such as the classification task. Then, in Section 2.3, we describe the general data mining process, which constitutes the standard approach commonly used by experts to tackle and solve data mining problems, such as classification problems. The data mining process was followed during the two case studies of this thesis (see Chapters 3 and 4). Many aspects of it were implemented in the tool that we developed to perform generic time series classification tasks (see Chapter 5). Finally, in Section 2.4, we present the classification task and detail the classification algorithms that we widely used in our experiments and implemented in our tool.

2.2 Mining Time Series Data

Time series[1] are a particular kind of data largely widespread in the numeric world. What are the best ways to tackle such data, as well as the tied data mining problems? Researchers have been working on these interesting questions in the last ten years. Many methodologies and algorithms have been proposed, used and tested in order to classify, cluster, index, segment, detect anomalies or particularities in time series. Although some methodologies can be very different, most of them have in common the fact that they rely on high level representations of the time series rather than on their raw original values. This process of getting high level time series representations is quite often a necessary step, not only for data mining tasks, like classification, but also for storage, transmission and computation on massive data sets.

2.2.1 Time Series

In order to settle on what we are talking about, let us begin by defining what are *time series*. Many data mining applications have to handle *temporal data*, which are typically collected by hardware or software monitoring devices through some continuously occurring processes. Examples of such data are those collected by various sensors, medical devices or financial markets monitoring tools. Temporal data can be *discrete* or *continuous*. For instance, an application log file may contain discrete events like date-error entries, whereas meteorologic data may contain continuous entries like the temperature. Continuous temporal data are called *time series*, while discrete temporal data are often called *sequences*. Though time series and discrete sequences are conceptually similar, the algorithmic methodologies used for each of them differ significantly. That said, time series are often transformed into discrete sequences via discretization (see Section 2.2.4) to take advantage of sequence mining methods.

Contrary to multidimensional data in which all attributes are handled equally, time series consist of data varying with the time context. More formally, time series have two kinds of attributes:

[1] This section was notably written based on two good data mining books and on article handling the topic of time series data mining [Maimon and Rokach, 2010; Aggarwal, 2015; Esling and Agon, 2012].

1. **Contextual attributes**, which represent the context in which the measurements were made. Time series owns a *unique contextual attribute* which is the *time*. Other data types may have multiple contextual attributes, like spatial data and their spatial coordinates. timestamps are usually represented either as raw time values or as *consecutive indices* (or *ticks*) at which behavioral attributes were acquired.

2. **Behavioral attributes**, which represent the behavioral values at given timestamps, like the different measures recorded by the Sensimed Triggerfish sensor every five minutes during 24 hours (see Chapter 4). Generally, to each contextual attribute corresponds (at least) one behavioral attribute. From the point of view of the problem to solve and the application to implement, behavioral attributes are the values of interest, but they cannot be correctly exploited without the information provided by contextual attributes.

Time series are either *univariate* or *multivariate*. In the first case, a unique measure (*behavioral attribute*) is associated to each timestamp (*contextual attribute*), while in the second case, there are several. Thus the dimensionality of a time series corresponds to the number of considered behavioral attributes.

Definition 2.1 (Multivariate Time Series) *A time series T of length $l > 0$ and dimensionality d contains d numeric feature values at each of the l timestamps $t_1, \ldots, t_l$. Each timestamp holds a component for each of the d series. Therefore, the set of values received at a timestamp t_i is $\mathbf{y}_i = (y_{i,1}, \ldots, y_{i,d})$ and $T = (\mathbf{y}_1, \ldots, \mathbf{y}_l)$. The value of the j^{th} series at timestamp t_i is $y_{i,j}$ (where $0 < i \leq l$ and $0 < j \leq d$) [Aggarwal, 2015].*

We speak of *multivariate time series*, when the value of d is greater than 1, and *univariate time series*, when the value of d is equal to 1. Such an univariate time series of length l is represented by a set of scalar values $(y_1, \ldots, y_l)$ associated with the timestamps $t_1, \ldots, t_l$[2]. Figure 2.1 shows an example of multivariate time series represented in one of its dimensions.

2.2.2 Time Series Preparation

Time series are frequently acquired by hardware or software monitoring devices in all kinds of domains, for example physical sensors, medical devices or financial market monitoring applications. These raw data often contain noise and/or errors that need to be handled before doing any data mining or machine learning task.

[2]In the literature, we sometimes see other definitions stipulating that the timestamps must be uniformly spaced, which may be too much limiting. With such definitions, the data recorded with the sensors of our two case studies wouldn't be considered as time series. Such sensors were not reliable and precise enough.

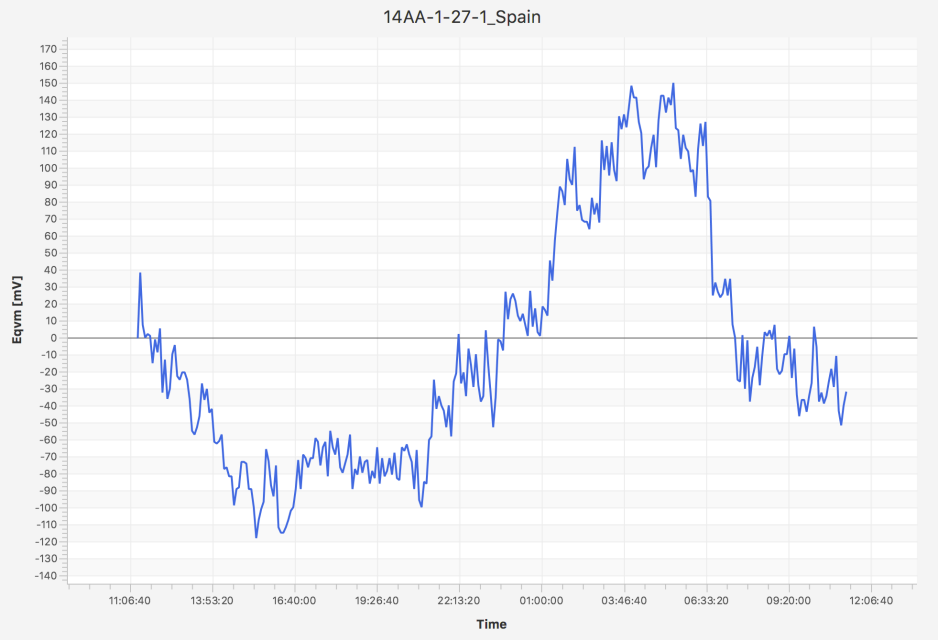


Figure 2.1: 24-hour evolution of a multivariate time series in one of its dimension, here the intraocular pressure (*Eqvm in [mV]*) by a glaucomatous patient.

Handling Noise, Errors and Outliers

Often collected with various types of physical sensors, time series may contain noise and errors, such as outliers, that can be detected, corrected or removed by some specific methods. A typical procedure is to remove short term fluctuations in the time series. For problems requiring to distinguish between noise and interesting outliers (which can be very problematic), noise removal methods should not be used. The most common methods are:

- **Binning**, which consists in dividing the time series data into time intervals of size k, i.e. bins (assuming that timestamps are equally spaced, yielding the same number of points in every bin), and taking the mean of each bin as smoothed values. Thus, the formula for a new binned value y'_{i+1} will be:

$$y'_{i+1} = \frac{\sum_{r=1}^{k} y_{i \cdot k+r}}{k}$$

  Instead of the mean, the median can be used. The median often provides better estimates as it is far less affected by outliers. The major drawback of this method is that is decrease the number of time series points. This method is also called *Piecewise Aggregate Approximation (PAA)*.

- **Moving-average smoothing**, which proceeds as the *binning* method but avoids the data reduction problem by using a *sliding window* to yield overlapping bins. Therefore

for each of the overlapping bins, the corresponding mean (or median) value is computed, producing a smoothed time series. The larger the size of the bin, the smoother the time series will be. The drawback of this method is the induced delay which can be a problem for real-time application.

- **Exponential smoothing**, which computes a smoothed value y'_i as a linear combination of the current value y_i and the preceding smoothed value y'_{i-1} as follows:

$$y'_i = \alpha \cdot y_i + (1 - \alpha) \cdot y'_{i-1}$$

where $\alpha \in [0, 1]$ is a smoothing parameter. The value of y'_0 is generally set to the one of y_0. If $\alpha = 1$, the original time series will not be smoothed. If $\alpha = 0$, the original time series will be entirely smoothed to the constant value of y'_0. This method is qualified as exponential because of the form of the non-recursive formula:

$$y'_i = (1 - \alpha)^i \cdot y'_0 + \alpha \cdot \sum_{j=1}^{i} y_i \cdot (1 - \alpha)^{i-j}$$

- **Hampel's outlier filter**, which is based on Hampel's outlier identifier [Hampel, 1974; Pearson, 2002; Chrominski and Tkacz, 2010; Leys et al., 2013]. In practice, this filter is often extremely efficient to detect and correct outliers. It works as follows. Let $Y = (y_1, \ldots, y_l)$ be a series of l values, then Hampel's identifier is applied to the values within a sliding window of size $2k+1$, where usually $k = 3$, and centred on the value y_i (where $0 < i < l$). When an outlier is detected, it is replaced with the median value med_i from the current window. In each window, Hampel's identifier is computed as follows. First, the median value med_i of the subset $Y_i = \{y_{i-k}, \ldots, y_{i+k}\}$ is computed, and then the *median absolute deviation* from the median $\texttt{MAD}_i = b\ \texttt{Median}(\{|y_j - med_i| \,|\, y_j \in Y_i\})$, where b is usually set to 1.4826 so that the expected $\texttt{MAD}_i$ is equal to the standard deviation of normally distributed data. Finally, the value y_i will be detected as an outlier if $|y_i - med_i| > t\ \texttt{MAD}_i$, where t is usually set to 3 for a medium tolerance.

Handling Missing Values

Time series may have missing values or values that were not synchronized in time by the sensor(s) during the acquisition. For the data processing, it is usually more convenient for time series to have equally spaced and time-synchronized values. The most common method to handle such missing, unsynchronised or unequally spaced values is *linear interpolation* which consists in computing estimated values at the wanted timestamps.

Let y_i and y_j be two values of a (univariate) time series at timestamps t_i and t_j (where $i < j$), and let $t_{interpol}$ be a timestamp in the interval $]t_i, t_j[$, the interpolated value at time

$t_{interpol}$ is given by:

$$y_{interpol} = y_i + \frac{t_{interpol} - t_i}{t_j - t_i} \cdot (y_j - y_i)$$

This linear interpolation formula is of course generalizable to multivariate time series. Other more complex methods, like *polynomial* or *spline interpolation* may be also applied. However, they require a greater number of points to compute the estimation.

Normalization

Time series generally need to be normalized, particularly multivariate time series whose various series are of different scales and therefore cannot be compared significantly at the same time. The most common methods are the *min-max normalization* and *Z-score standardization*, which are described in Section 2.3.3 (see Equations 2.1 and 2.2) for the more general case, where the data can be of different nature than time series.

2.2.3 Time Series Similarity Measures

Time series similarity measures are often used for specific application tasks, such as indexing and clustering (see next Section 2.2.5). The most common methods are the following:

Euclidean Distance & L_p-Norm

The euclidean distance is the simplest one and is a particular case of the L_p-Norm when $p = 2$. The L_p-Norm distance between two time series $X = (x_1, \ldots, x_l)$ and $Y = (y_1, \ldots, y_l)$ is defined by:

$$D(X, Y) = L_p(X, Y) = \left(\sum_{i=1}^{l} |x_i - y_i|^p \right)^{1/p}$$

The L_p-Norm has the main drawbacks that it requires the time series to have the same length l and it will not be suitable for time series which are similar but whose one is stretched or compressed in its y-axis. This can be addressed by applying normalization.

Dynamic Time Warping (DTW)

The Dynamic Time Warping (DTW) method aims at stretching time series along their time axis in a dynamic (i.e. varying) way over their different parts in order to obtain a better matching between them. Let $DTW(i, j)$ be the optimal distance between the first i and j values of two time series $X = (x_1, \ldots, x_l)$ and $Y = (y_1, \ldots, y_k)$ of potentially different lengths l and k, then:

$$DTW(i, j) = distance(x_i, y_j) + minimum \begin{cases} DTW(i, j-1) & \text{i.e. repeat value } x_i \\ DTW(i-1, j) & \text{i.e. repeat } y_j \\ DTW(i-1, j-1) & \text{i.e. repeat neither } x_i \text{ nor } y_j \end{cases}$$

The function $distance(x_i, y_j)$ can be defined in many ways, depending on the task.

Edit Distance

The Edit Distance (also referred as the Levenshtein Distance) defines the distance between two time series (or any sequences or strings) as the least cost required to transform one time series into the other by applying a series of *edit* operations consisting in insertions, deletions and replacements of values. Generally, the replacement operation has a higher cost than insertion and deletion operations whose costs are often equal. Let $Edit(i, j)$ be the optimal matching cost between the first i and j values of two time series $X = (x_1, \ldots, x_l)$ and $Y = (y_1, \ldots, y_k)$ and I_{ij} an indicator which is equal to 0 when $x_i = y_j$ or 1 otherwise. Then the goal is find what operation to perform on the last x_i so that it either matches an element in $(y_1, \ldots, y_j)$ or is deleted, yielding three possible situations described by the following formula:

$$Edit(i,j) = minimum \begin{cases} Edit(i-1,j) + \text{Cost}_{\text{deletion}} \\ Edit(i,j-1) + \text{Cost}_{\text{insertion}} \\ Edit(i-1,j-1) + I_{ij} \cdot \text{Cost}_{\text{replacement}} \end{cases}$$

with the bottom recursion values $Edit(i, 0)$ set to the cost of i deletions of any value of i and $Edit(0, j)$ set to the cost of j insertions of any value of j. Note that because of the edit directionality and the different costs, the edit distance is not symmetric: Edit(X, Y) may be not equal to Edit(X, Y).

Longest Common Subsequence Similarity

The Longest Common Subsequence Similarity (LCSS) measure is a variation of the preceding Edit Distance. Its purpose is to match two time series by permitting some values to be unmatched. Let $LCSS(i, j)$ be the optimal longest common subsequence between the first i and j values of two time series $X = (x_1, \ldots, x_l)$ and $Y = (y_1, \ldots, y_k)$ (of potentially different lengths l and k), then the goal is either to match x_i and y_j or delete the last element in one of the two time series, yielding two possible situations described by the following formula:

$$LCSS(i,j) = maximum \begin{cases} LCSS(i-1,j-1)+1 & if\ x_i = y_j \\ LCSS(i-1,j) & if\ no\ match\ on\ x_i \\ LCSS(i,j-1) & if\ no\ match\ on\ y_j \end{cases}$$

with the boundary values $LCSS(i, 0)$ and $LCSS(0, j)$ set to 0.

2.2.4 Time Series Representations, Transformation and Reduction

Time series databases can be very large and extracting information and knowledge from them can be a challenge. Therefore, and depending on the data mining task, it is often useful to transform and/or reduce time series into more efficient and/or compact representations. In fact, representation and dimensionality reduction has become the heart of many time series

mining tasks in very large databases. Several methods exist and the output representations can be very different in nature, for instance discrete values or sequences of symbols. Moreover, these methods are sometimes applied on subsequences of time series obtained using analysis windows which are shifted (*sliding windows*).

Discrete Fourier Transform (DFT)

Discrete Fourier Transform (DFT) is typically efficient when the time series contains frequency information. Its purpose is to represent a signal of length l as a linear combination of $l-1$ sinusoidal series that are represented by complex numbers called *Fourier coefficients* (see Figure 2.2). When a time series is represented under this form, it is said to be in the *frequency domain*. Given a time series $T = (x_1, \ldots, x_l)$, each Fourier coefficient X_k of the DFT is expressed by the following formula:

$$X_k = \sum_{r=0}^{l-1} x_r \cdot e^{-irwk} = \sum_{r=0}^{l-1} x_r \cdot cos(rwk) - i \sum_{r=0}^{l-1} x_r \cdot sin(rwk), \qquad k = 0, \ldots, l-1$$

w is equal to $\frac{2\pi}{l}$ radians and i is the imaginary number (such that $i^2 = -1$). Because the coefficient X_{l-k} can be deduced from X_k simply by inverting the sign of the imaginary part for $k \geq 1$, only the first $l/2$ coefficients are kept. Time series data can be reduced by keeping only the coefficients $X_k = a_k + ib_k$ that have the biggest energy $a_k^2 + b_k^2$, in other words the greatest contribution in composing the original signal. Then the kept coefficients can be used to build an approximated representation of the time series. The original time series can be reconstructed from the Fourier coefficients according to the following formula:

$$x_r = \frac{1}{l} \sum_{k=0}^{l-1} X_k \cdot e^{irwk} = \frac{1}{l} (\sum_{k=0}^{l-1} X_k \cdot cos(rwk) + i \sum_{k=0}^{l-1} X_k \cdot sin(rwk)), \qquad r = 0, \ldots, l-1$$

Though each X_k is a complex number, the imaginary part of the equation will always be set to 0 to return a real value x_r. Because of some properties like *additivity*, DFT can be very useful in many data mining problems. From a computation point of view, DFT is very greedy. The best algorithm is called Fast Fourier Transform (FFT) and it requires the time series to have a length equal to a power of 2.

Discrete Wavelet Transform (DWT)

Discrete Wavelet Transform (DWT) is typically efficient when the time series contain variations that are mostly located in specific regions of the signal. Its purpose is to represent a signal of length l as a weighted sum of l "simpler" time series called *wavelets* that are orthogonal to each other (see Figure 2.2). In the decomposition, these wavelets are the basis vectors and the wavelets coefficients the weights of these basis vectors. This method allows

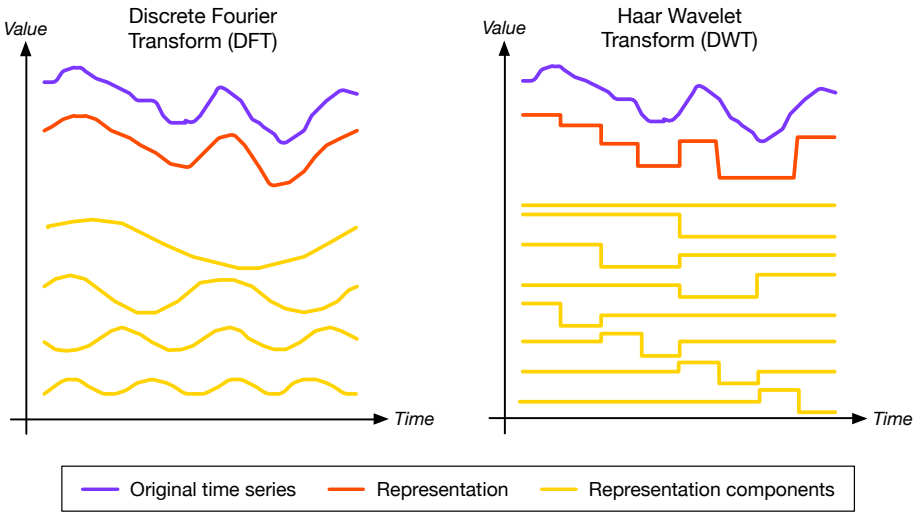


Figure 2.2: Time series representations with Discrete Fourier Transform (DFT) and Discrete Wavelet Transform (DWT).

to convert time series data to multidimensional data[3], in which the temporal continuity and dependancy of the values disappear. In other words, wavelet coefficients intrinsically describe properties of different contiguous temporal regions of the time series. This property of multidimensional data is an advantage for the data mining process. *Each coefficient is equal to half the difference in the average value of the behavioral attribute between a pair of carefully chosen contiguous segments of the series* [Aggarwal, 2015]. As for DFT, Time series data can be reduced by keeping only the first largest coefficients, which have the greatest contribution in generating the original signal (generally, they are not so much). Then the kept coefficients can be used to reconstruct an accurate approximated representation of the time series. The most commonly used set of DWT are the Haar wavelets and the Daubechies wavelets.

Singular Value Decomposition (SVD)

Singular Value Decomposition (SVD) is like Like DFT and DWT in the sense that it allows to represent the time series shapes as a linear combination of basis shapes (*singular vectors*), but is different in the sense that it is a global transformation on the entire set of time series. The whole database is rotated so that the first basis axis gets the maximum variance, the second one the maximum variance orthogonal to the first one, and so on. SVD is rather efficient and used in the task of time series indexing [Chan and Fu, 1999]. As this thesis work is focused on the task of classification, this method is not detailed here.

[3]One may say that multivariate time series are already multidimensional data. Dimensionality is a concept that can have two meanings for time series. While every behavioral attribute of a multivariate time series can be considered as a dimension, each value along the time axis in an univariate time series can also be seen as a dimension. The right definition to use depends on the context.

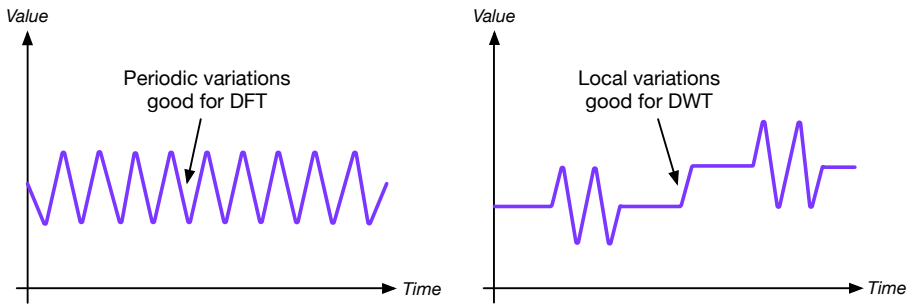


Figure 2.3: Typical scenarios to apply either DFT or DWT.

Piecewise Linear Approximation (PLA)

Piecewise Linear Approximation (PLA) aims to approximate a time series with piecewise linear segments [Pavlidis and Horowitz, 1974]. Noise removal and data compression are the main advantages of this approach. Several algorithms were proposed and used to perform PLA. One of its main drawbacks is the difficulty to choose the optimal number of segments to represent a set of time series. Figure 2.4 shows an example of this representation.

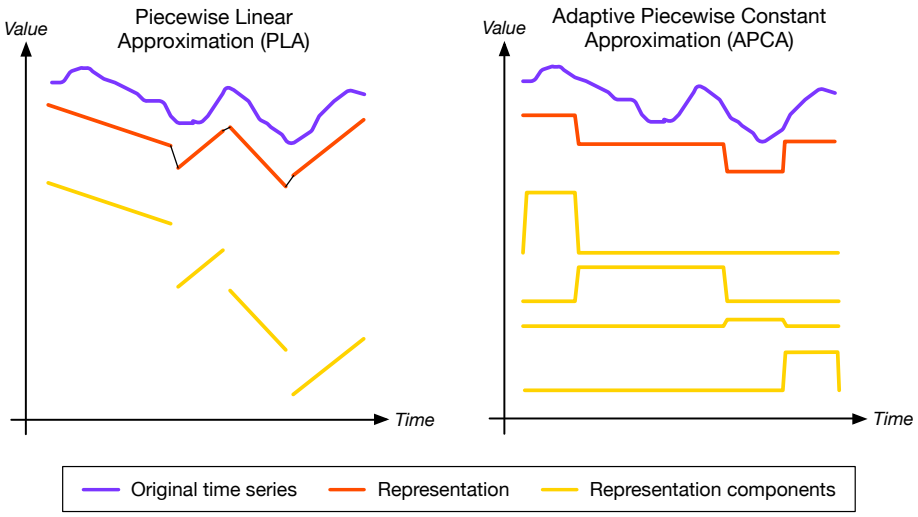


Figure 2.4: Examples of time series representations with PLA and Adaptive Piecewise Constant Approximation (APCA).

Piecewise Aggregate Approximation (PAA)

Piecewise Aggregate Approximation (PAA) aims to approximate a given time series by dividing it into k segments of equal length, taking the mean of the points contained in each segment, and aggregating all these mean values in a vector [Keogh et al., 2001b; Yi and

Faloutsos, 2000]. Thus, for a time series of length l, if $k = 1$, the PAA representation will be the mean of the whole time series, and if $k = l$, it will be the time series itself.

Adaptive Piecewise Constant Approximation (APCA)

APCA is an extension of the PAA method so that it allows the segments to have a varying length (see Figure 2.4), implying to have two parameters for each segment, namely the mean of the points contained in the segment and its length [Keogh et al., 2001a]. For a given time series, APCA has the advantage to be able to represent an area with low variability with a single segment and an area with high variability with a lot of segments, but the disadvantage to have generally twice as less segments as with PAA.

Symbolic Aggregate Approximation (SAX)

Symbolic Aggregate Approximation (SAX) is a recent method aiming at converting time series into discrete sequence of symbols (from a given alphabet of length α). Its purpose is to keep the meaningful information contained in the original time series [Lin et al., 2003]. The SAX transformation consists first in a step of normalization of the time series followed by two steps of discretization. More precisely, a time series of length l will be first divided by into k segments of equal length; the mean of the points contained in each segment will be computed, and all these mean values will be aggregated in a vector, resulting in a PAA representation. Then, to convert PAA coefficients to symbols, breakpoints dividing the time series into α equiprobable areas will be computed. Thus, any symbol and any substring of the final representation will be equiprobable. At last, each PAA coefficient will be mapped to a symbol from the given alphabet according to the areas in which they are located (see Figure 2.5). The SAX representation has revealed to be very efficient in the tasks of time series clustering and classification, and therefore is used in the case studies of this thesis.

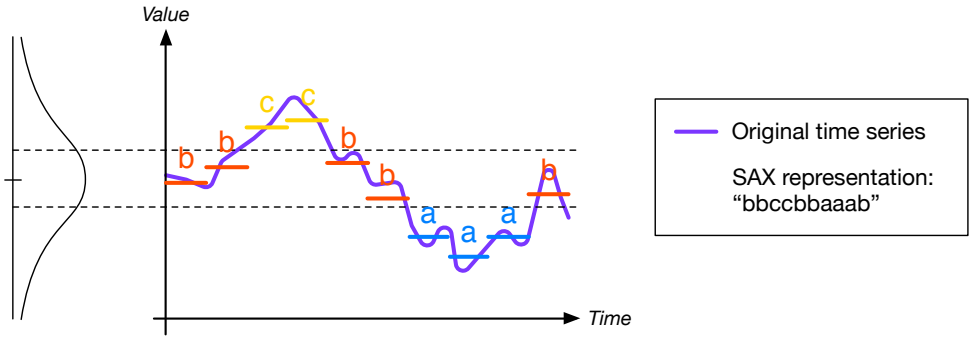


Figure 2.5: SAX representation of a time series using an alphabet of 3 letters and a length of 10.

Discussion

Amongst all these possible representations, one may ask which one is the best to use. Several works have been made to try answering this question and every one brings arguments in favour of their own method [Chakrabarti et al., 2002; Faloutsos et al., 1994; Popivanov and Miller, 2002]. There are many contradictory statements, such as "*Several wavelets outperform... DFT.*" [Popivanov and Miller, 2002], "*DFT-based and DWT-based techniques yield comparable results.*" [Wu et al., 2000], or "*Haar wavelets... perform better than DFT.*" [Kahveci and Singh, 2001]. Nevertheless, results of a large comparison on 50 different time series databases showed that all these representations seem to have more or less the same capacity to approximate time series, even if some perform better in certain cases, on certain databases [Keogh and Kasetty, 2002]. That said, differences between these methods can be pointed in their applicability along with the data mining tasks. Here are several examples. DWT and DFT (applied with FFT algorithm) are only defined for time series whose length is a power of 2. All DFT coefficients represent a contribution to the global shape of a time series [Faloutsos et al., 1994; Rafiei and Mendelzon, 1998]. First DWT wavelets coefficients contain information on the global shape of a time series, while the last ones contain information on local shapes [Popivanov and Miller, 2002; Shahabi et al., 2000]. APCA has the same resolution fidelity as DWT, but is defined for time series of any lengths [Chakrabarti et al., 2002]. All the above methods provide outputs that are real values, which can be an advantage or a disadvantage in certain cases where the use of symbolic representations such as SAX will be better [Lin et al., 2003].

2.2.5 Time Series Data Mining Tasks

The main tasks of *time series data mining* can be divided in the following categories. In this thesis, we focused on the topic of *time series classification*.

Classification

Let $\mathcal{Y}$ be a set of m predefined category (*class*) labels ($m \geq 2$) and T an unlabelled time series (or a part of it), assign T the most likely class label y from $\mathcal{Y}$.

Clustering

Let $D(T_i, T_j)$ be a (di-)similarity measure and $\mathcal{T}$ a time series data set, find natural groups (*clusters*) of time series in $\mathcal{T}$.

Motif Discovery

Let T be a time series, detect frequently occurring patterns or shapes (*motifs*) in T.

Anomaly (Outlier) Detection

Let T_1 be a time series supposed to be normal and T_2 an unannotated time series, detect all parts of T_2 suspected to contain unexpected, surprising or interesting events.

Prediction (Forecasting)

Let T be a time series of length l, predict the $l + 1^{\text{th}}$ value, i.e. at time t_{l+1}.

Indexing (Query by Content)

Let T be a time series, $D(T_i, T_j)$ a (di-)similarity measure[4] and $\mathcal{T}$ a time series data set, find the most similar time series T' in $\mathcal{T}$.

Summarization

Let T be a time series of length l (with l very large), create an approximation T' of T retaining its main features but of length $l' << l$.

Segmentation

Let T be a time series of length l, create a model T' approximating T' composed of k piecewise segments, where in general $k << l$.

2.3 Data Mining Process

Data mining is often associated with statistics. In statistics, the first step of an analysis is the setup of a methodology to acquire valuable data so that the analysis will be reliable. In a data mining problem, it often occurs that the data have already been collected, not necessarily in an appropriate way, and we have to make do. Statistics is generally concerned with testing hypotheses on the data while data mining is concerned with generating hypotheses from the existing data. From this have emerged all the varieties of data mining techniques and tools. However, each problem is different and today researchers rather agree that the data mining process cannot be fully automatised and a human intelligence is still needed to carry it out.

Data mining is also often associated with machine learning. Both topics overlap on many aspects. Machine learning comes directly from the field of computer science and particularly artificial intelligence. It is concerned with the design and development of computational and statistical methods that allow computers to *learn*, that is to infer knowledge from given

[4] Clustering and indexing *explicitly* use a distance measure, while classification, anomaly detection, prediction and summarization *implicitly* use a distance measure.

data and use it on other data automatically. It covers *supervised* and *unsupervised learning* methods, including *classification*, *clustering* and *regression* algorithms. Data mining is more tied to business applications. It has a larger scope and covers other methods, such as *motif discovery* and *indexing* (*query by content*) algorithms.

The *data mining process*[5] (or *data analysis process*) constitutes the standard approach commonly used by experts to tackle and solve data mining problems. It consists of six different phases which are depicted in Figure 2.6 by the CRoss Industry Standard Process for Data Mining (CRISP-DM) [Chapman et al., 2000].

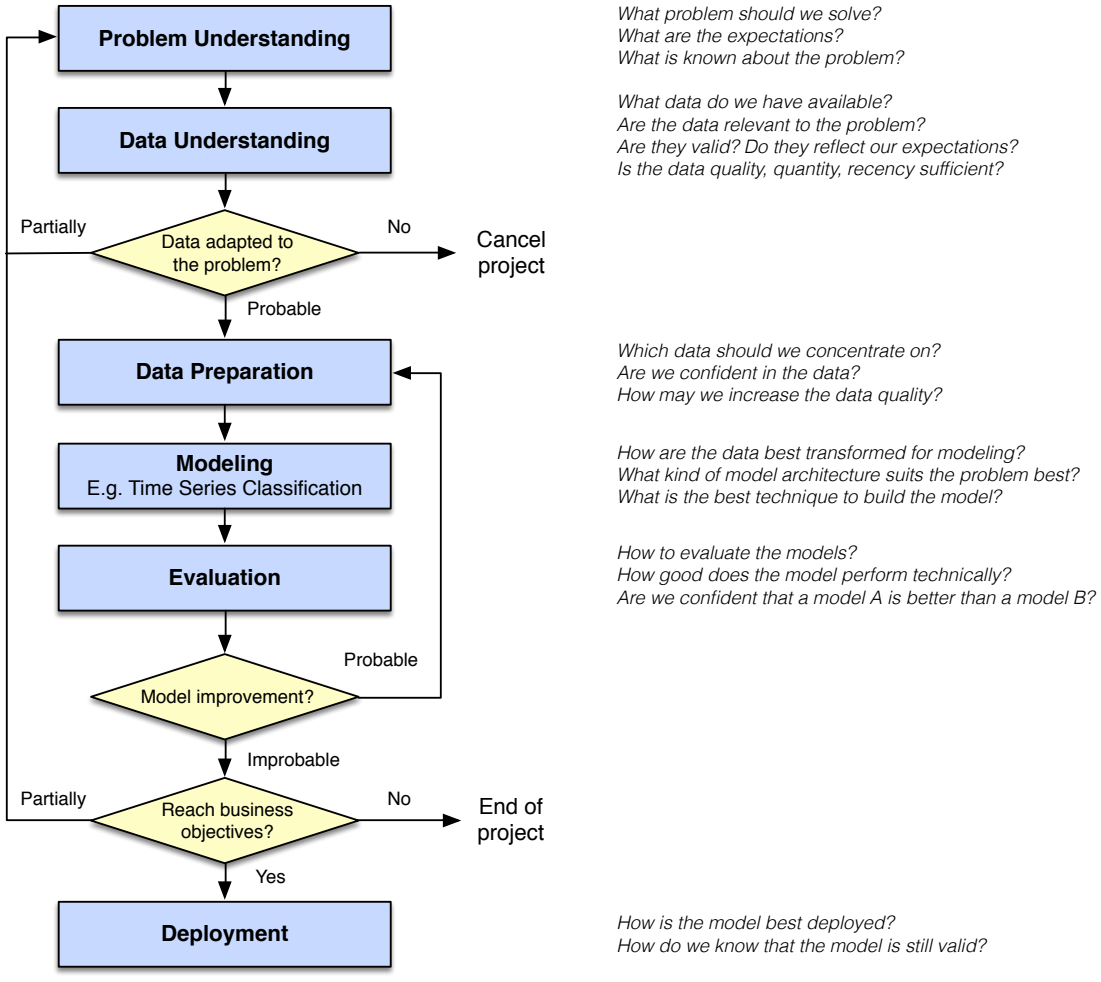


Figure 2.6: The CRISP-DM process with specific questions to ask at each phase [Chapman et al., 2000].

[5]Here, we mainly refer to an excellent book written by M. Berthold *et al.* handling the topic of data mining process [Berthold et al., 2010]. This section is a synthesis of chapters 3, 4, 5, 6 and 10.

2.3.1 Problem Understanding

The data mining process relies on the data. How can we be sure that the problem has chances to be solved with the available data? Why should we invest time and money, and take risks to try to solve it? What problem should be solved exactly? What is known about the problem? What are the expectations? Charles Elkan, a reviewer and expert for a data mining contest, remarked that "*when something surprising happens, rather than interrogating themselves about the expectations, people typically believe that they should have done something slightly different*" [Elkan, 2001]. Convinced that their problem can be solved with the available data, researchers often enters into an unending loop where they try to adapt and optimize their model continuously, rather than thinking that perhaps the data are not suitable for the problem. This initial phase allows not to fall in such traps by understand the problem defined by clearly stated objectives, typically aiming at crating new services or bringing value to an organization.

Requirements and Constraints

The requirements and constraints of the project must be considered. For example, if decisions have to be clearly justified, the generated models should be explanatory (*model requirements*). If the computation must not exceed a certain duration to return a decision, some modeling algorithms have to be avoided (*technical constraints*). Sometimes variables like gender, age or race must be left out (*ethical, political and legal issues*).

Assumptions and Risks

Several assumptions must be made on the data, and the ensuing risks be taken into account. For instance, the database should be representative of a specific target population to model (*representativeness*). Influencing factors should be represented by attributes in the database in order to have a chance to produce a valuable model (*informativeness*). The data should be relevant and of good quality, namely correct, complete and up-to-date (*quality*). The "external world" should be assumed as invariant during the project (*external factors*).

Problem Definition

The preceding reflexion must lead to a more technical problem definition, with an initial plan designed to achieve the goals. Even if the final choice of the modeling algorithm(s) will be made during the modeling phase, one should have a clear idea about the desired model properties, notably in terms of:

1. **Interpretability** – Some algorithms allow a better interpretability than others which act more like black boxes;

2. **Reproducibility and stability** – If experiments have to be repeated, they must deliver similar results and performances;

3. **Flexibility and adequacy** – Some problems and situations are less complex than others, and the models need to be more or less generalizing;

4. **Runtime** – Runtime constraints may dismiss some CPU consuming algorithms;

5. **Expertise availability** – Better models are produced when some domain expert knowledge is available.

2.3.2 Data Understanding

What data are available? Are the data relevant to the problem? Are they valid? Do they reflect our expectations? Are the data quality and quantity sufficient? Whatever the task on the data will be, the data must be tackled with caution and a neutral point of view. One should not trust blindly a given data set before having made some *plausibility* checks, i.e. credibility and consistency checks. This data understanding phase implies having a closer look at particular aspects of the available data.

Attribute Understanding

Data normally consist of a set of *samples* (also called *observations*, *instances* or *records*) containing *attributes* (also sometimes called *features*). Understanding the nature of these attributes is crucial for the next phases of the process. The *domain* of an attribute is the set of all of its possible values. An attribute is called:

1. **Categorical** (or *nominal*) when its domain is a finite set (e.g. an attribute *gender* with values *male* and *female*). If its domain has a linear order, it is called *ordinal* (e.g. an attribute *university degree* with values $BSc < MSc < PhD$). A categorical attribute may also have a dynamic domain, meaning that new values can be added (e.g. the *type* of an electric appliance). This may be a problem to handle in the modeling phase.

2. **Numerical** when its domain is composed of numbers. It can be *discrete*, usually taking integers as values (e.g. an *age* in years: 12, 27,...), or *continuous*, taking real numbers as values (e.g. a *duration* in seconds: 33.478, 87.541,...). The limited precision of their representations may cause rounding problems in the modeling phase, particularly if attributes have different precisions. Numerical attributes are sometimes defined in an *interval scale* (e.g. dates), a *ratio scale* (e.g. heights, distances or durations) or an *absolute scale* (e.g. counted objects). For each case, the zero value can be significative or not, and the potential usable mathematical operations can vary.

Data Quality

The quality of the data affects its ability to be usable for a task. It has three main dimensions:

1. The **accuracy**, which represents the distance between the measured value and the real value. It may be low in case of noise or insufficient precision in the data. We speak of *syntactic accuracy* when a value is correct relatively to the possible values of its attribute domain (e.g. *femalllle* is not in the domain of the attribute *gender* above), and of *semantic accuracy* when a value is correct relatively to the reality, in a semantic point of view (e.g. Mary's *gender* is not *male*). Detecting problems with semantic accuracy in the data is much harder than with syntactic accuracy. They depend on the way the data were generated, entered in the database or validated by humans.

2. The **completeness**, which can be related to attribute values, for example when there are missing values in the samples, or to the samples themselves, when there are not enough instances in the database to provide sufficient information for the modeling phase. This can occur with time series recorded with an insufficient sampling rate. It often occurs that the data are not representative, biased or unbalanced. For instance, in a disease detection problem, data set are often *unbalanced*, containing more samples of healthy people than ill people.

3. The **timeliness** refers to the data validity relatively to the time. If data are too old, the information that will be extracted will lead to inaccurate models. This is a problem especially for prediction tasks, often depending on dynamically changing attributes.

Data Visualization

The desire of viewing data graphically is natural and fundamental. "*There is no excuse for failing to plot and look*", when one has to tackle a data mining problem [Tukey, 1977]. A lot of methods exist to visualize data of all types. They will not be listed here, as in this thesis we focused on time series data, which fortunately can be often represented on simple *time-values* charts, contrarily to other types of data.

Data Correlation

It is recommended to compute some statistical measures of correlation between attributes to see if there are some dependencies, expected or not, between them. Common measures include *Pearson's correlation coefficient*, *Spearman's rank correlation coefficient* or *Kendall's tau ranks correlation coefficient*, notably detailed in [Berthold et al., 2010, Section 4.4].

Outliers

The definition for the term *outlier* is rather vague. An outlier is usually simply considered as an attribute value, or a sample, that is very different or distant from all the others. Outliers

often result from wrong measures or typing errors. Thus, the data quality depends on outliers which can be detected with various methods, but hardly corrected. While statistical tests like *Grubb's test*, *Dixon's test* or *Hampel's test* are commonly used on univariate data [Pearson, 2002; Chrominski and Tkacz, 2010], visualization, clustering, distance-, density- or projection-based methods are preferred for multivariate data [Maimon and Rokach, 2010, Chapter 7]. Nevertheless, multavariate data can often be decomposed into multiple univariate data.

Missing Values

Missing values can lead to similar problems as outliers. By simply ignoring them, we risk getting wrong outputs for our data mining task. Sometimes they can be ignored, sometimes not, and must be replaced by default values or approximated values like interpolated values.

2.3.3 Data Preparation

This phase includes all tasks needed to build from the raw data the final data set which will be used by the modeling algorithms and tools. These tasks may be performed several times not necessarily in a particular order. Of course, the preceding phase of *understanding* shall help a lot to perform this one. Four steps are usually required.

Data Selection

Even if there is plenty of data, a data selection is often needed to keep only the samples and attributes that are *relevant* for the given problem. This selection can be done at three levels:

1. **Sample selection** – Samples can be selected according to several criteria: the *temporality* (some tasks like prediction only need recent samples), the *representativeness* (some tasks need different target populations to be selected from the entire population in the database), the *rarity of events* (some tasks like anomaly detection need samples containing rare events to be selected), or the *runtime* (runtime problems may lead to reduce the data set to samples that are crucial for the given tasks and algorithms).

2. **Feature selection** – Attributes (or features) of samples can be selected to form an optimal subset of attributes to be used in the modeling phase. The greater the number of features there is, the greater the number of potential subsets there is (n attributes implies 2^n possible subsets). Feature selection methods generally requires to choose an *evaluation function* to compare two given subsets, and a *strategy* (often a heuristic) to select the subsets to compare using this evaluation function. The strategy adopted by *relief methods* consists in assuming that a subset will perform as well as features taken individually and therefore simply choose the k top ranked features. Another strategy, more CPU consuming, consists in choosing the top ranked subsets of features. It has been adopted by *cross product* methods, which select and test all possible attributes

combinations, *wrapper* methods, which select the subsets according to the performance of the fitted model, and other methods, which remove redundant or useless attributes. These methods are detailed in [Berthold et al., 2010, Section 6.1.1] and those used in our case studies are described later in Chapters 3 and 4.

3. **Dimensionality reduction** – Data dimension can be reduced by factor analysis methods like Principal Component Analysis (PCA). Their goal is to explain existing attributes as a linear combination of new, less numerous, attributes. Other methods like Independent Component Analysis (ICA) allow to get rid of restrictions to linear combinations. More details are given in [Bishop and Nasrabadi, 2007, Sections 12.1–12.4].

Data Cleaning (Preprocessing)

Also called *preprocessing* step in the field of machine learning, this step consists in cleaning/-correcting the attribute values which seem to be noisy, wrong or missing.

1. **Errors/noise** – The cleaning consist here in correcting simple errors called *noise*, like inconsistencies, spelling mistakes, spelling variants, abbreviations, varying formats (e.g dates, measures). Though humans are usually very good at detecting and correcting noise, the automatized task shall rapidly become very difficult for the computer (due to the potentially great number of occurring errors, coming from various sources like sensors, A/D converters, or operators).

2. **Missing values** – Why handling missing values? If information is missing, it can't be invented! Some feature extraction and modeling algorithms (see next sections) can't return an output if there are missing values. The ways to handle missing values are:

   - *Ignorance/deletion* – Samples or attributes with missing values are ignored or removed from the data set.
   - *Imputation* – Missing values are replaced by some estimated or predefined values
   - *Explicit value/variable* – In case of nominal attributes, missing values are replaced by special values, like `n/a`. The fact that an attribute value is missing can carry significant information about the attribute itself, depending on the task to perform.

Data Construction (Feature Extraction)

After selection and cleaning, the data often need to be adapted. Also called *feature extraction* step in the field of machine learning, this step consists in building some new attributes from the original ones, and this for several potential reasons:

1. **Operability** – Some tools and algorithms need the data to be adapted in order to work properly. This can be done by performing:

- *Scale conversion* of attributes values, for example to meet the requirements of some algorithms that accept only numeric values in a certain range.
- *Discretization* of the range of the possible numeric values into several contiguous non-overlapping labeled intervals.
- *Dynamic domain handling*, in which nominal attributes with unbounded domain must be converted into numeric values. One method consists in mapping these attribute values to a more general or abstract category whose value will be used to map new incoming values.
- *Problem reformulation*, for instance by transforming a multi-class classification problem to a binary one.

2. **Impartiality** – Without an *a priori* knowledge, every attribute should have the same weight and role during the modeling process. This can be done by normalizing the data. Most common methods are the:

   - *Min-max (or range-based) normalization*, which, for a numeric attribute $\mathbf{x} = \{x_1, ..., x_n\} \in \mathcal{X}$ with minimum min and maximum max computed across a set of n samples and a given sample attribute value x_i (with $i = 1, \ldots, n$), is defined as:

   $$f_{min\text{-}max} : \qquad \mathcal{X} \to [0, 1], \qquad x_i \mapsto \frac{x_i - min}{max - min} \tag{2.1}$$

   - *Z-score standardization* (or *standard score*), which, for a numeric attribute $\mathbf{x}$ defined as before with empirical mean μ and variance σ computed across a set of n samples and a given sample attribute value x_i, is defined as:

   $$f_{z\text{-}score} : \qquad \mathcal{X} \to \mathbb{R}, \qquad x_i \mapsto \frac{x_i - \mu}{\sigma} \tag{2.2}$$

   The new attribute values x_i will be centered around 0 (with a new $\mu = 0$), within $[-1, 1]$ with a probability of about 68% and in $[-2, 2]$ with a probability of 95%.

3. **Efficiency** – Depending on the context, modeling algorithms may have a facilitated task, notably for discovering potential dependencies. This can be achieved by:

   - *Building derived attributes (or features)* – From a context knowledge, new attributes can be derived from original attributes. For example, a new attribute *mean speed* may be built from two attributes *travel time* and *travel distance*.
   - *Changing the baseline of attributes* – An existing attribute can be adapted to share the same comparable baseline between samples. For example, an attribute *speed* expressed in *miles per hour* can be changed in *meters per second*.
   - *Grouping attributes* – Some attributes of similar nature often highly correlates and may be grouped. For example, the *mean* and *median* of several measures. Clustering attributes may help identifying attributes groups.

- *Hyperplanes* – New numeric (or binary if a threshold is provided) attributes may be created by linear combinations of existing attributes. For example, a binary attribute becoming true if certain conditions on attributes hold.
- *Nonlinearities* – New attributes may be created by performing nonlinear combinations of existing attributes, if no linear dependency was found between them.
- *Changing the granularity level* – Grouping attribute values according to a certain level of granularity corresponding to some aggregates, hierarchy or taxonomy may unveil or not interdependencies in the data.
- *Adding metadata* – New attributes like annotations indicating that some samples were modified may help identifying wrong assumptions.

Data Integration

Most existing modeling tools or implementations assume that the data are provided as a single database or table. If it is not the case, some integration tasks must be done. We distinguish *vertical data integration*, which aims at merging and concatenating tables holding essentially the same information (thus missing values and duplicates shall be handled), and *horizontal data integration*, which aims at enriching existing tables by adding and combining different types of information from other databases (a typically challenge is here to deal with *join* operations on tables).

The phase of *data preparation* is often neglected in the data mining process. Many data mining tools assume to have a nice data representation at disposal, e.g. in files or tables. When new raw data arrive, what should matter for such tools is not only to prepare the data, but also to reproduce easily the process needed to rerun a particular data mining task.

2.3.4 Modeling

This fundamental phase constitutes the heart of the *data mining process* (as well as *machine learning*). Here, modeling techniques must be selected, applied and evaluated, and their various parameters be tuned. There are often several algorithms that can be selected and tested for a given problem, but each one has its own particularities and requirements in terms of data inputs/outputs and parameters. This may lead back to the phase of *data preparation*.

Modeling Algorithms

Depending on the type of data and problem, various modeling algorithms can be chosen. Types of problems include regression, classification, clustering, prediction, and indexing (as presented in 2.2.5 for time series). Each one owns its set of appropriate algorithms, but some algorithms can be used for more than one problem types. In this thesis, we focused on the

classification algorithms that we applied to time series (see Section 2.4). Modeling algorithms can be selected based upon several criteria:

- **Simplicity** – Simple models, like linear models, are often easier to understand and interpret, and their computational complexity is usually lower. A model can often be seen as a summary of the data, and a complex one might just be more or less a one-to-one representation in the worst case. While complex models, like nonlinear models, are generally more efficient at fitting complex data, they also often fail at summarizing them and thus generalizing, i.e. revealing general structures and relations in the data. This issue is called *overfitting* (see also next section).

- **Interpretability** – The models provided by certain algorithms, like Decision Tree (DT) are more easily understandable and interpretable than others, which act more as black boxes, like Support Vector Machine (SVM) or Artificial Neural Network (ANN). When the goal is to "only" find the best model, black box algorithms may be a good choice, but when the goal is also to give an understandable description of the data and/or the model outcomes, simpler algorithms may be a better choice. Measuring this interpretability very much depends on the context.

- **Computational complexity** – Simple modeling algorithms that have few parameters to tune, often have a lower computational complexity and runtime than others.

Score Functions

A *score* (or *error*) *function* allows to evaluate and compare the generated models in order to find the best model, or at least a suitable one. Usually taking the form of an *objective function* $g : \mathcal{M} \to \mathbb{R}$, where $\mathcal{M}$ is the set of all generated models, the score function will be decisive in the tuning of the parameters of the modeling algorithm. For a regression problem, the most common error function is the *mean square error (MSE)*, while for a classification problem, specific performance measures based on *classification outcomes* are used, such as the *accuracy*, the *Receiver Operating Characteristic (ROC) Area Under Curve (AUC)* or the *confusion matrix* (see Section 2.4.7). Each score function has a different influence on the model fitting. Therefore, it must be carefully chosen according to the type of problem to solve and the objectives of the project (see Section 2.3.1). Score functions are often tied to a *cost function* (or *matrix*) measuring the consequences that a bad or good score will have in the context of the problem.

Model Fitting

The modeling algorithm needs some methods in order to make its parameters vary adequately, fit the model and converge toward a suitable solution to the optimization problem. Among the existing methods, the most known and applied are the following.

- **Gradient method** – If the model to fit is determined by k real parameters, the objective function has the form $f : \mathbb{R}^k \to \mathbb{R}$. The gradient is the vector of partial derivatives of the objective function relatively to the k parameters and points toward the greatest slope. This method can be used if the objective function is differentiable. Starting from a random point, that is a selection of parameter values, the idea is to go in the direction of the gradient when the objective function can be maximized (*gradient ascent*) and in the opposite direction, when it can be minimized (*gradient descent*). At each step, a new point is returned. The process is repeated until no progress is observed or after a certain number of steps.

- **Exhaustive search (or grid search)** – If the objective function is not differentiable or discrete (e.g. with a finite set of possible value parameters), an exhaustive search on the finite parameter domain can be done, although this is often unrealizable because the search domain may be too big with many parameters and possible values.

- **Random search** – This brute force method just randomly generates a finite set of points (selection of parameter values), tests them and returns the best one.

- **Greedy search** – Greedy search methods are heuristic methods. A simple example called *Hillclimbing* starts from a random point, tests all the points in its neighborhood and selects the best point as the starting point for the next step. The process is repeated until no progress is observed or after a certain number of steps.

- **Genetic and other evolution algorithms** – Such methods aim at combining different aspects of random and greedy search methods, on various sets of points, in order to explore the parameter space in a more efficient way.

Fitting Error

Once a model has been fitted to the given data, the fitting error can be computed. Four types of errors are usually distinguished:

- **The pure, experimental, or intrinsic error**, which is inherent to the data and due to noise, random variations, imprecise measures, or effects of hidden variables, which cannot be detected. The choice of a suitable model can not overcome this error. In a classification problem, it is also called *Bayes error* and is due to the fact that, in real problems, classes often overlap, so that it is impossible to classify all samples correctly.

- **the sample error**, which is due to an imperfect representation of the underlying data distribution because the number of samples is finite. According to the law of large numbers, the sample distribution tends to the real distribution generating the data as the number of samples tends towards infinity.

- **the model error**, which is due to a lack of fit, for instance when none of the considered models is able to fit the inherent structure of the data.
- **the algorithmic error**, which is caused by the method used to fit the model or algorithmic error model parameters. If an analytical solution exists for the optimum of the objective function, the algorithmic error is zero or only due to numerical problems.

The algorithmic and the model errors, which can be somehow controlled by choosing a suitable algorithm and model, are sometimes called *machine learning bias*. The intrinsic and the sample errors, which cannot be controlled, mostly if the data have already been collected, is sometimes called *variance*.

Model Validation

The error of a model measured on the data from which the model was built is usually smaller than the error measured on unknown and unseen data. A methodology is therefore needed to find the best model to the problem from the available data set.

- **Training, test and validation data sets** – In order to estimate the performance of a model on future unknown samples, the available data set is generally split into several different and independent parts. Samples of the training set are used to build/fit (*train*) the model, while those of the test set are used to evaluate (*test*) the trained model. A modeling algorithm often needs its parameters to be tuned to return the best model. Thus, the data set is often split in a third part: the *validation set*. The models trained with various parameter settings are evaluated (*validated*) on the validation set and the best one is returned for being evaluated on the test set. As the validation set serves to get the best model during the training phase, it is often said to be a part of the training set. In general, the width of the training set is voluntarily defined bigger than the width of the test set, for example 60% and 40% of the data set. For a classification problem, a common way to form these sets is to randomly select the samples for each set, so that the data distribution of the original data set shall be preserved. This can be ensured by doing a *stratification* of the data. In other words, the random selection is performed per class and not directly on the whole set.
- **Cross-Validation (CV)** – This method allows to give an averaged *estimate* of the fitting error and thus the performance of a given model on a given (training) data set. It also allows to determine the best model parameters.
  - *k-Fold Cross-Validation (CV)* – This resampling method coming from statistics randomly splits the data set of n samples into k subsets (where $2 \leq k \leq n$). Then, each of the k-folds are successively taken for the test set and the remaining $k-1$ for the training set. At each step, the performance, for example in terms of accuracy

or the ROC AUC, is measured and saved. At the end, the k measures are averaged to give the final performance estimation. A commonly chosen value for k is 10.

- *Leave-One-Out CV* (or *Jackknife*) – This method is just a particular case of the k-Fold CV with $k = n$, where n is the size of the data set. It is often applied when the data set is too small to perform a standard k-Fold CV.
- *Repeated Random Subsampling CV* – This method randomly splits the data set into i training and validation sets, and the i performance measures are averaged to give the final performance estimation. The advantage of this method over the k-Fold CV is that the number of samples in each training and validation set is not dependent on the number of iterations (folds). The disadvantage is that some samples may never be put in the validation set, whereas others more than once.

- **Bootstrap** (or **bagging**) – For a given data set, this resampling method does not directly aims at estimating the performance of a given model, but rather the variance of its parameters. The process consists in forming k *bootstrap* subsets of equal size $s < n$, whose samples are randomly chosen (with replacement) in the data set. Then, a model is trained on each of the k bootstrap sets, and evaluated on the test set to obtain k performance estimates for the given model and parameters. At last, the empirical standard deviation is computed for each parameter to assess the reliability of the parameter estimation.

2.3.5 Evaluation

The purpose of this phase is not to evaluate the trained model(s), which has been done during the *modeling* phase. Performance measures such as the accuracy or the ROC AUC were computed during the *modeling* phase in order to ensure a good convergence of the training process, by modifying and tuning the model until to obtain the best one that will hopefully solve the problem. The goal of this phase is to evaluate the performance that the best generated model(s) will have on the task(s) to be done in the context of the project and its business objectives before a deployment. Interpretations of data and models, as well as ensuing conclusions should be conform to the project owner's or a domain expert's knowledge.

Documentation

Not only the final model(s), but all evidences, findings and conclusions should be documented during the preceding phases of the process and reported during this phase. Such outputs represent fundamental resources for future potential projects. For example, encountered problems and drawbacks may highly contribute to improve the data quality and acquisition. All these outputs should then be discussed in order to decide if the implementation will be deployed or not, and this will constitute the output of this phase. The step of documentation is frequently neglected and generally leads to difficulties to reproduce what was done during

the project. During the mining process, several experiments and tests are conducted. Some come from sudden inspirations of analysts, some give promising results and others not. Many different models are rapidly generated from different data subsets and it becomes very problematic to keep track of what has been done and for what purpose. Important and good results remains usually in mind, but when they have to be reproduced, that's another kettle of fish! Dedicated tools should help documenting the data mining process.

Testbed Evaluation

When several efficient models are found during the *modeling* phase, a testbed evaluation may be useful (though often expensive) to assess the performances of the models in realistic conditions near to a deployment. As with former model evaluation, this evaluation should be done with a comparison baseline. Moreover, decision on deployment should not be taken depending on the performance of the models on extreme cases (worse or best).

2.3.6 Deployment

In this last phase, the final model which was selected and evaluated will be deployed and exploited to return the desired outputs, e.g. the classifications or predictions, on new incoming data. Then, the required information and knowledge will be inferred from these outputs.

Deployment Process

This process is generally realized under the form of the implementation and installation of a dedicated software system in a production environment. Deployment process can be done in several ways which are beyond the scope of the thesis and hence are not be covered here. Even if it is the data analyst that carries out the deployment process, it is fundamental that the project owner understands all the actions needed to actually exploit the provided models.

Monitoring

Are we confident that our deployed model will remain valid and efficient along with time? The purpose of monitoring is to detect situations where the model is still in use while it might have become inadequate or invalid. By having a closer look at the outputs returned from the model, performance degradations can be detected and the model eventually adapted and rebuilt. Various external factors may lead to invalidate the deployed model and should also be monitored. The project objectives may change depending on the business strategy of the company. Assumptions made during the *project understanding* phase may also change or be invalidated. The world evolves, and with it all sorts of things, such as data acquisition sensors which become more and more performant. New classes may appear in the acquired data. Like humans tend to do when they see something new, the model will potentially put a new class sample in a known class. Clustering methods can help prevent this.

2.4 Classification Task and Algorithms Used

Classification[6] is a branch of data mining and machine learning, which consists in classifying observations (*samples*) of objects into categories (*classes*) by assigning them *labels*. There are many applications, notably in the fields of biometry (e.g. signature, iris, fingerprints or speech recognition) and document analysis (e.g. multimedia file indexing or image classification). There are also many algorithms. Therefore, in this section, we only describe those which were studied and used in this thesis.

2.4.1 Classification Process

The classification task or process generally relies on a *model* built from a set of already labeled samples, the *training set*, and is ultimately applied on a set of unlabeled samples. This kind of learning strategy is called *supervised learning*. Labeled samples constitute a useful *a priori* knowledge on the class distributions, which facilitates the learning by the system. Learning can also be *unsupervised*, in the sense that the system doesn't have this *a priori* knowledge on the classes (no labeled samples are available). Instead, it determines the classes by itself on the basis of statistical regularities of the unlabeled samples. Such unsupervised strategy is adopted by *clustering* methods.

The classification performance of a model is generally estimated on an independent set of labeled samples, the *test set*. Thus, their correct classification is known and a performance measure can be computed (see Section 2.4.7). Another set called *validation set* is often used to avoid *overfitting* the model typically occuring when the number of parameters of the model is too large regarding the number of training samples. Overfitting begins when the classification performance measure increases on the training set while it decreases on the validation set.

Classification algorithms able to discriminate the samples from two classes only are called *binary* (or *binomial*) *classifiers*, while those able to discriminate the samples from m classes (where $m \geq 2$) are called *multi-class* (or *multinomial*) *classifiers*[7]. A multi-class problem can often be reduced to a binary problem. Therefore, binary classifiers shouldn't be immediately dismissed. Conversely, if a binary classifier would be the adequate choice, strategies exist to transform it into a multi-class classifier (see Section 2.4.4).

The classification process typically fits into the *data mining process* previously described, and its various algorithms take place in the *modeling* phase of the process (see Section 2.3.4 and Figure 2.6). It usually consists of the following phases:

[6] The subject being really well handled in Duda, Hart and Stork's book [Duda et al., 2001], as well as Bishop's [Bishop and Nasrabadi, 2007], we refer to them to have a more rigorous approach in this thesis.

[7] Classes are usually considered as mutually exclusive, otherwise we speak of *multi-label classification*.

1. **Data acquisition**, during which raw data are acquired means of physical sensors, when they come from the real world, or software utilities when they come from the digital world, i.e the Internet.

2. **Preprocessing**, during which *raw data* are prepared in terms of integration, selection and cleaning as described in Section 2.3.3 of the data mining process.

3. **Feature extraction**, during which meaningful numeric *feature vectors* are extracted from the cleaner raw data, as described in Section 2.3.3 of the data mining process. Its purpose is to reduce useless and redundant information, as well as increasing the discriminative capacity between the samples of the different classes. An often time intensive task called *feature engineering* consists in computing relevant features which will be similar for objects belonging to the same class, different for objects belonging to different classes, and insensitive to noise and distortion. Such features will tend to minimize the intra-class variability while maximizing the inter-class variability.

4. **Model training**, during which a model is learnt and built (*fitted*) from the feature vectors of the labeled *training samples*.

5. **Classification**, during which the model is used to compute classification scores for the feature vectors of the unlabeled *test samples*.

6. **Post-processing / Decision**, during which all classification scores are first merged if they come from several classification algorithms (*classifiers*), and returns for each one the most likely class label. These three last modules fall within Section 2.3.4 of the data mining process.

2.4.2 Bayesian Decision Theory

Bayesian decision theory is certainly the most basic and fundamental approach to the task of classification. The theory assumes that the decision problem is stated in probabilistic terms and that all relevant probabilities are known. Under these conditions, the Bayesian decision theory gives an *optimal decision* for the classification.

Here is the problem statement[8]. Let:

- $\mathcal{X}$ be a d-dimensional Euclidean space called *feature* space (generally $\mathcal{X} = \mathbb{R}^d$);
- $\mathcal{Y} = \{y_1, \ldots, y_m\}$ be a finite set of m categories (*classes*);
- $\mathbf{x} \in \mathcal{X}$ be a d-component *feature vector* (or *sample*) representing an object to classify;

[8] An upper-case $P(\ldots)$ usually denotes a probability mass function while a lower-case $p(\ldots)$ a probability density function

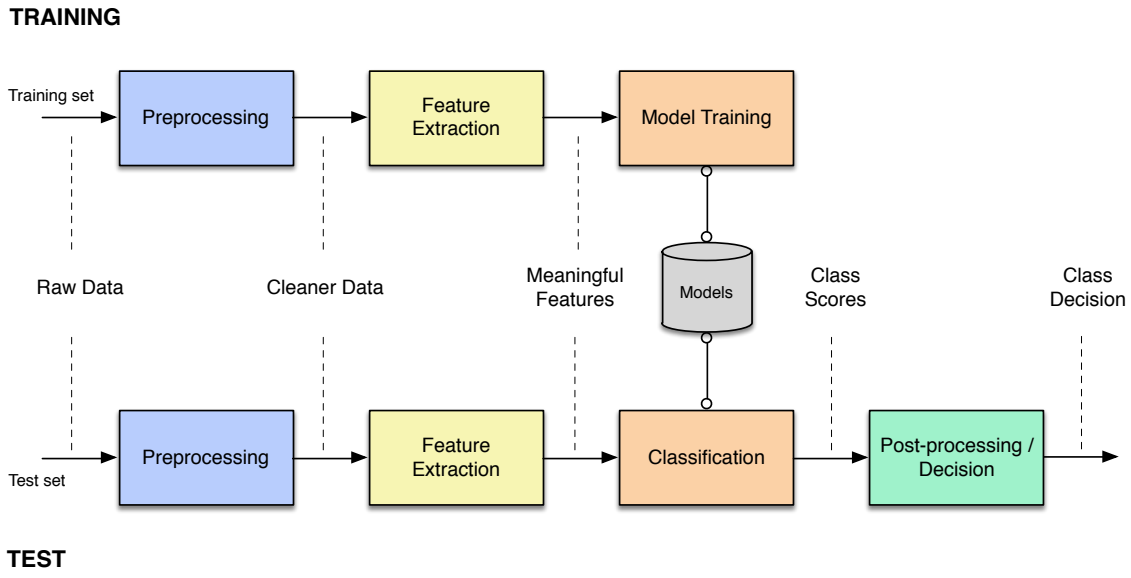


Figure 2.7: The standard process of classification in machine learning.

- $y \in \mathcal{Y}$ be the true class of the sample $\mathbf{x}$;
- $p(\mathbf{x}|y_j)$ be the class-conditional probability density function for the feature vector $\mathbf{x}$. It is also called *likelihood* of the class y_j with respect to $\mathbf{x}$ (where $j = 1, \ldots, m$);
- $P(y_j)$ be the *prior* (or *a priori*) probability of each class y_j.

Then, the *posterior* (or *a posteriori*) probability $P(y_j|\mathbf{x})$ can be calculated from $p(\mathbf{x}|y_j)$ by using the Bayes' formula:

$$P(y_j|\mathbf{x}) = \frac{p(\mathbf{x}|y_j)\, P(y_j)}{p(\mathbf{x})} \tag{2.3}$$

where the *evidence* $p(\mathbf{x})$ is:

$$p(\mathbf{x}) = \sum_{j=1}^{m} p(\mathbf{x}|y_j)\, P(y_j) \tag{2.4}$$

Bayes decision theory tells us that the optimal class $\hat{y}$ is given by the class y_j which has the highest *posterior probability* $P(y_j|\mathbf{x})$ given the evidence $p(\mathbf{x})$:

$$\hat{y} = \arg\max_{y_j \in \mathcal{Y}} P(y_j|\mathbf{x}) \tag{2.5}$$

Hence, a classification learning algorithm aims at minimizing the number of classification errors. In a more general framework, the theory proceeds by minimizing the overall risk of classifying a sample $\mathbf{x}$ in a class $\hat{y}$ when its true nature is the class y and the posterior probability $P(y|\mathbf{x})$ (see next Section 2.4.3).

Such an algorithm will compute an estimate of the posterior probability from a set of training samples $\{(\mathbf{x}_i, y_i) \mid \mathbf{x}_i \in \mathcal{X}, y_i \in \mathcal{Y}\}_{i=1}^n$ and some assumed prior knowledge. Generally, this prior knowledge comes from usual assumptions and any information that is known about the train data, like the forms of the probability density functions. The performance of a classification algorithm depends on how well it estimates the posterior probability and therefore on the validity of the prior knowledge. In any case, the lowest bound of classification errors corresponds to the true form of the posterior probability and no estimate will beat this value.

2.4.3 Classification Problem Generalization

In a more general framework, the classification problem can be stated as follows [Vapnik, 1999]. Let:

- $\mathcal{X}$ be an input *feature* space;
- $\mathcal{Y}$ be an output class (or category) *label* space;
- $\{(\mathbf{x}_i, y_i) \mid \mathbf{x}_i \in \mathcal{X}, y_i \in \mathcal{Y}\}_{i=1}^n$ be a set of n training labeled *samples* (*feature vectors*).

Then, the classification problem consists in finding a mapping $h : \mathcal{X} \to \mathcal{Y}$, called *classifier*, able to map any sample $\mathbf{x} \in \mathcal{X}$ to a classification label $\hat{y} \in \mathcal{Y}$, so that:

$$\hat{y} = h(\mathbf{x}) \tag{2.6}$$

Hence, a *classification algorithm* is a computational procedure which takes the training labeled samples as input and returns a classifier h as output. The function h can also be represented by a score function $g : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ so that h returns the label $\hat{y}$ yielding the highest score:

$$\hat{y} = h(\mathbf{x}) = \arg\max_{y \in \mathcal{Y}} \; g(\mathbf{x}, y) \tag{2.7}$$

The spaces $\mathcal{H}$ or $\mathcal{G}$ of all possible functions h or score functions g are called *hypothesis spaces*. $\mathcal{H}$ or $\mathcal{G}$ can be any space of functions. However, many classification algorithms are probabilistic models in which h takes the form of a conditional probability model $h(\mathbf{x}) = P(y|\mathbf{x})$ and g the form of a joint probability model $g(\mathbf{x}, y) = P(\mathbf{x}, y) = p(\mathbf{x}|y)\, P(y)$ (see Section 2.4.2).

In the framework of this problem definition, some assumptions are done:

- There exists a *joint probability distribution* P on $\mathcal{X} \times \mathcal{Y}$;
- Training samples $(\mathbf{x_1}, y_1), \ldots, (\mathbf{x_n}, y_n)$ are sampled independently from P (i.e. *independent and identically distributed*).

When a classifier h is used, a performance measure is needed. A *loss function* $l : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}_{\geq 0}$ measures the *cost* of classifying a sample $\mathbf{x} \in \mathcal{X}$ as $\hat{y} \in \mathcal{Y}$. For a training sample $(\mathbf{x}, y)$, the loss of attributing the label $\hat{y}$ is $l(y, \hat{y})$. The simplest loss function is called *0-1 loss* (or *misclassification error*) which is defined by:

$$l(y, \hat{y}) = l(y, h(\mathbf{x})) = \begin{cases} 1 & if\ h(\mathbf{x}) \neq y \\ 0 & otherwise. \end{cases} \tag{2.8}$$

The *risk* $R(h)$ of the function h is defined by the *expectation* of the loss function l:

$$R(h) = E[l(y, h(\mathbf{x}))] \tag{2.9}$$

Thus, the *best classifier* h_{best} is the one which minimizes the risk $R(h)$:

$$h_{best} = \arg\min_{h \in \mathcal{H}} R(h) \tag{2.10}$$

As already said, among all possible classifiers, the Bayes classifier constitutes the best one. However, it is impossible to compute it directly as the underlying probability function is unknown to the learner.

The two most common methods allowing to find h_{best}, or rather an approximation $\hat{h}_{best}$, are the *Empirical Risk Minimization* and the *Structural Risk Minimization*.

Empirical Risk Minimization (ERM)

In general, the underlying probability distribution P is unknown to the classification algorithm and attempting to estimate P from the training data usually doesn't work. Thus, the risk $R(h)$ can not be computed. Nevertheless, it can be estimated empirically by averaging the loss function over the training samples, as follows:

$$R_{emp}(h) = \frac{1}{n} \sum_{i=1}^{n} l(y_i, h(\mathbf{x}_i)) \tag{2.11}$$

Indeed, $\lim_{n \rightarrow \infty} R_{emp}(h) = R(h)$ [Vapnik, 1999]. The *Empirical Risk Minimization (ERM)* method consists in searching the function $\hat{h}_{best}$ that best fits the training set by minimizing the so-called *empirical risk* $R_{emp}(h)$. Therefore, a classification algorithm implementing the ERM method proceeds by solving this *optimization* problem to find:

$$\hat{h}_{best} = \arg\min_{h \in \mathcal{H}} R_{emp}(h) \tag{2.12}$$

When $\mathcal{H}$ contains many potential functions h or when the training set is too small, ERM will lead to high *variance*[9] and low *generalization*[10] by the model. The classification algorithm will just "memorize" the training samples and return a classification function, i.e. a model, unable to generalize well. This issue is called *overfitting*.

[9]The variance is the error due to the sensitivity to small variations of the training set. It shows how much the estimate varies around its average.

[10]The generalization is the ability of a model to predict classification outcomes for previously unseen samples.

Structural Risk Minimization (SRM)

The *Structural Risk Minimization (SRM)* method consists in including a regularization penalty function into the optimization process in order to control the *bias-variance tradeoff*[11] and thus prevent overfitting. The principle of SRM is to balance the complexity of the model against its capacity at fitting the training sample set. Many penalty functions corresponding to various definitions of complexity have been applied in the literature.

Let $C(h)$ be this penalty function. The optimization problem consists now in finding the function $\hat{h}_{best}$ which minimizes $J(h) := R_{emp}(h) + \lambda\ C(h)$ and defined by:

$$\hat{h}_{best} = \arg\min_{h \in \mathcal{H}} \left(R_{emp}(h) + \lambda\ C(h)\right) \tag{2.13}$$

The parameter λ controls the bias-variance tradeoff. If $\lambda = 0$, the classification algorithm will have low bias and high variance as in ERM. Inversely, if λ is big, the algorithm will have high bias and low variance. The value of λ is generally chosen empirically via *cross-validation* (see preceding Section 2.3.4).

The SRM principle was proposed in 1974 by Vladimir Vapnik and Alexey Chervonenkis. Vapnik is the inventor of the SVM algorithm mainly used in this thesis (see Section 2.4.4).

Discriminative and Generative Methods

A risk minimization method that tries to find a function h which separates well the samples of the different classes is said to be *discriminative*. From a Bayesian point of view, discriminative algorithms attempt to model the posterior probabilities directly and therefore not the underlying probability distributions. Discriminative methods notably include Support Vector Machine, Artificial Neural Network, and Random Forest.

In the particular case of probabilistic methods, where h is a conditional probability distribution $h(\mathbf{x}) = P(y|\mathbf{x})$, g a joint probability distribution $g(\mathbf{x}, y) = P(\mathbf{x}, y)$, and the loss function corresponds to the negative log likelihood $l(y, h(\mathbf{x})) = -\log p(\mathbf{x}|y)$, the method is said to be *generative* (or *conditional*) because g can be viewed as a model able to explain how the samples were generated and even able to generate new samples respecting the underlying class distributions. Thus, such algorithms attempt to model the likelihoods and prior probabilities, and are used in an intermediary step to compute posterior probabilities through Bayes' rule. Generative methods notably include Naive Bayes Classifier, Gaussian Mixture Model, and Hidden Markov Model.

[11] The bias is the expected error due to the model mismatch. It shows how far on average is the model from the truth. The bias-variance tradeoff is the problem of minimizing errors due to bias and variance at the same time, which prevent a supervised learning algorithm from generalizing.

We may say that discriminative methods aim at solving the particular problem of classification only, while generative methods attempt to solve the more general problem of modeling the underling probability distributions of the different classes beforehand. Both approaches are as much used as each other. Research and experiments actually states that "*so far there is no theoretically correct, general criterion for choosing between the discriminative and the generative approaches to classification of an observation* $\mathbf{x}$ *into a class* y*; the choice depends on the relative confidence we have in the correctness of the specification of either* $P(y|\mathbf{x})$ *or* $P(\mathbf{x}, y)$ *for the data*" [Xue and Titterington, 2008].

2.4.4 Support Vector Machine (SVM)

In this section, we give the theoretical background of Support Vector Machine (SVM) which is one of the most recent and efficient discriminative supervised method for classification. SVM is the first classification algorithm used in this thesis[12].

Linear SVM

The original SVM algorithm proposed by Vladimir Vapnik in 1963 is a linear binary classifier. Let $\{(\mathbf{x}_i, y_i) \mid \mathbf{x}_i \in \mathbb{R}^d, y_i \in \{-1, +1\}\}_{i=1}^n$ be a training set of n labeled *samples* from two classes. The samples are assumed *linearly separable*, i.e. there exists a linear decision boundary, that is a *hyperplane*, which separates *positive* samples from *negative* ones. We will initially assume that the two classes are linearly separable. The samples $\mathbf{x}_i$ that lie on this separating hyperplane satisfy the following equation[13]:

$$\mathbf{w} \cdot \mathbf{x}_i + b = 0 \tag{2.14}$$

$\mathbf{w}$ is the normal vector to this hyperplane, $\|\mathbf{w}\|$ its Euclidean norm, and $|b|/\|\mathbf{w}\|$ its perpendicular distance to the origin. Let d_+ and d_- be the shortest distances from the closest positive and negative samples to the separating hyperplane. The so-called *margin* is defined by $d_+ + d_-$. With linearly separable samples, the SVM algorithm will simply determine the equation of the hyperplane that has the largest margin. The fact that all samples $\mathbf{x}_i$ of both classes must be correctly classified beyond the two hyperplanes that are parallel and at distances (d_+ and d_-) from the separating one can be formulated by the following constraints:

$$\mathbf{w} \cdot \mathbf{x}_i + b \geq +a \quad \text{for } y_i = +1,\ a > 0 \tag{2.15}$$

$$\mathbf{w} \cdot \mathbf{x}_i + b \leq -a \quad \text{for } y_i = -1,\ a > 0 \tag{2.16}$$

The constant $a > 0$ can be set to $a = 1$, because multiplying the samples $\mathbf{x}_i$ by a value will just increase the margin accordingly, without really changing it. In fact, it is only the "units"

[12] This section is mainly based on the papers [Burges, 1998; Ben-Hur and Weston, 2010; Chang and Lin, 2013]. To go deeper into the subject of SVM, we also recommend the papers [Vapnik, 1999; Mak, 2000; Hearst et al., 1998; Smola et al., 2000] or the books [Duda et al., 2001] and [Bishop and Nasrabadi, 2007].

[13] $\mathbf{w} \cdot \mathbf{x}$ represents the *dot product* of vectors $\mathbf{w}$ and $\mathbf{x}$. In the literature we also see the matrix notation $\mathbf{w}^T\mathbf{x}$.

in which it is measured that change. Thus, the two preceding constraints can be combined into the following one:

$$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1, \quad i = 1, \ldots, n \tag{2.17}$$

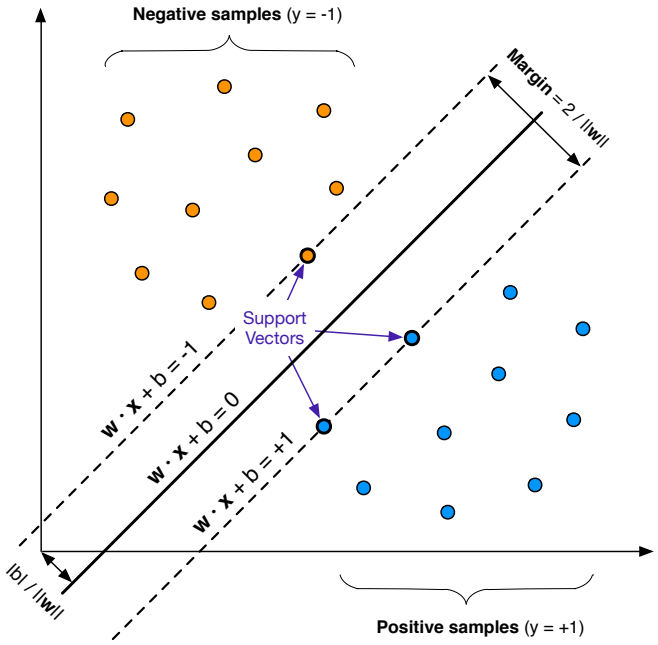


Figure 2.8: Principle of linear SVM dealing with linearly separable data (2D case).

Consider the samples $\mathbf{x}_i$ for which the equality in Equations 2.15 and 2.16 holds. These samples lie either on the hyperplane $\mathbf{w} \cdot \mathbf{x}_i + b = 1$ or $\mathbf{w} \cdot \mathbf{x}_i + b = -1$. Both hyperplanes have the same normal vector $\mathbf{w}$, but different perpendicular distance to the origin: $|1-b|/\|\mathbf{w}\|$ and $|-1-b|/\|\mathbf{w}\|$. Therefore, $d_+ = d_- = 1/\|\mathbf{w}\|$ and the margin is equal to $2/\|\mathbf{w}\|$. At this point, to determine the hyperplane (*classifier*) $\mathbf{w} \cdot \mathbf{x} + b$ that maximizes the margin $2/\|\mathbf{w}\|$ under the constraints that all the training samples are correctly classified, the SVM algorithm solves the following constrained optimization problem:

$$\begin{aligned} &\arg\min_{\mathbf{w},\, b} \ \frac{1}{2}\|\mathbf{w}\|^2 \\ &\text{subject to:} \quad y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1, \quad i = 1, \ldots, n \end{aligned} \tag{2.18}$$

Maximizing $2/\|\mathbf{w}\|$ is mathematically difficult, because of the square root involved in computing the norm of $\mathbf{w}$ ($\|\mathbf{w}\| = \sqrt{\mathbf{w} \cdot \mathbf{w}}$). Instead, we minimize $\|\mathbf{w}\|^2/2$ which gives the same solutions (the factor $1/2$ being used for mathematical convenience). The typical hyperplane solution for a 2-dimensional case is depicted at Figure 2.8. The samples $\mathbf{x}_i$ for which the equality in Equations 2.15 and 2.16 holds and whose removal would change the solution are called *support vectors*.

We now assume that the two classes are not linearly separable. Practically, the data are rarely linearly separable and even if they are, a larger margin can be obtained by allowing the classifier to misclassify some samples. In order to do this, we replace the contraints in Equation 2.17 with:

$$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1 - \xi_i, \quad i = 1, \ldots, n \tag{2.19}$$

$\xi_i \geq 0$ are called *slack variables* and permit a sample $\mathbf{x}_i$ to be either well classified ($\xi_i = 0$), located within the margin ($0 < \xi_i \leq 1$) or misclassified ($\xi_i > 1$). Because any misclassified sample has $\xi_i > 1$, $\sum_{i=1}^{n} \xi_i$ represents an upper bound on the number of misclassified samples. The SVM goal remains to maximize the margin by minimizing $\frac{1}{2}\|\mathbf{w}\|^2$, but additionally $\sum_{i=1}^{n} \xi_i$ must be minimized. The constrained optimization problem 2.17 becomes:

$$\begin{aligned} &\arg\min_{\mathbf{w},\, b} \ \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n} \xi_i \\ &\text{subject to:} \quad y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1 - \xi_i, \qquad \xi_i \geq 0 \end{aligned} \tag{2.20}$$

where $C > 0$ is a penalty parameter that controls the tradeoff between the margin size and the slack variable penalty. A small C value will yield a *smooth* hyperplane, while a high one will aim at classifying all training samples correctly. The algorithm that solves this version of the problem is called the *soft-margin* SVM algorithm. A typical 2-dimensional case is depicted at Figure 2.9.

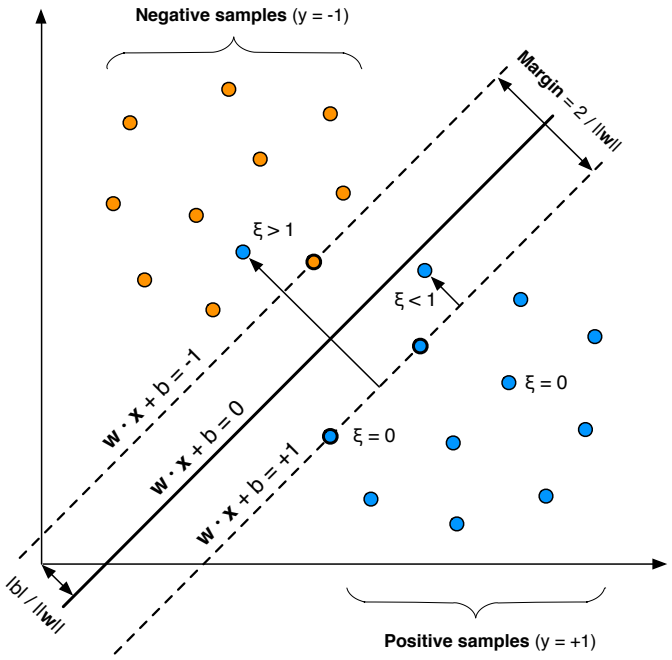


Figure 2.9: Principle of linear SVM dealing with not linearly separable data (2D case).

The quadratic constrained problem given by Equation 2.21 (or 2.18) is in its so-called *primal form*. The variable b to be found (along with the vector $\mathbf{w}$) is absent from the part to optimize; it only appears in the constraints 2.19. By introducing Lagrange multipliers $\alpha_i \geq 0$ (one for each constraint, and let $\alpha = [\alpha_1 \ \ldots \ \alpha_n]^T$), the optimization problem can be expressed in its *dual form* given by [Burges, 1998]:

$$\begin{aligned} &\arg\max_{\alpha} \ \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} y_i y_j \alpha_i \alpha_j \mathbf{x}_i \cdot \mathbf{x}_j \\ &\text{subject to:} \quad \sum_{i=1}^{n} y_i \alpha_i = 0, \quad 0 \leq \alpha_i \leq C \end{aligned} \tag{2.21}$$

There are two reasons for doing this. Firstly, the constraints in Equation 2.19 are replaced by constraints on the multipliers α_i only, which makes the problem easier to solve. Secondly, in the dual form, the training data only appear in the form of *dot product* between vectors, which is a crucial aspect to generalize SVM to the nonlinear case.

This quadratic constrained problem can be solved by standard quadratic mathematical methods and computer algorithms. However it is complex and was much CPU time consuming until a dedicated and more efficient algorithm called *Sequential Minimal Optimization (SMO)* was proposed by John Platt in 1998 [Platt, 1998]. At the end, the solution for $\mathbf{w}$ will have the following form:

$$\mathbf{w} = \sum_{i=1}^{n} y_i \alpha_i \mathbf{x}_i \tag{2.22}$$

Hence, $\mathbf{w}$ is expressed as a linear combination of the training vectors. That said, only some α_i will be greater than 0. The corresponding training samples $\mathbf{x}_i$ lie on the margin if $0 < \alpha_i < C$ and are the *support vectors*, or within the margin if $\alpha_i = C$ when the soft-margin SVM algorithm is applied. Let n_{sv} be the number of support vectors and $n_{\alpha>0}$ be the number of samples for which $\alpha_i > 0$. The value of b can be determined from the constraints in Equation 2.19. For any support vector $\mathbf{x}_i$, the corresponding slack variable $\xi_i = 0$ and the equality holds in the constraints. Hence: $y_i(\mathbf{w} \cdot \mathbf{x}_i + b) = 1 \Leftrightarrow \mathbf{w} \cdot \mathbf{x}_i + b = 1/y_i = y_i \Leftrightarrow b = y_i - \mathbf{w} \cdot \mathbf{x}_i$. Averaging the n_{sv} values obtained for b gives a numerical stable solution:

$$b = \frac{1}{n_{sv}} \sum_{i=1}^{n_{sv}} (y_i - \mathbf{w} \cdot \mathbf{x}_i) \overset{(2.22)}{\downarrow} = \frac{1}{n_{sv}} \sum_{i=1}^{n_{sv}} \left(y_i - \sum_{j=1}^{n_{\alpha>0}} y_j \alpha_j \mathbf{x}_j \cdot \mathbf{x}_i \right) \tag{2.23}$$

Finally, in order to classify a new unlabelled sample $\mathbf{x}_t$ by using the trained model, the SVM algorithm just evaluates the sign of the obtained function $\mathbf{w} \cdot \mathbf{x} + b$ evaluated with $\mathbf{x}_t$ and returns the corresponding class label y_t as follows:

$$y_t = \text{sign} \ \ \mathbf{w} \cdot \mathbf{x}_t + b \overset{(2.22)}{\downarrow} = \text{sign} \ \sum_{i=1}^{n} y_i \alpha_i \mathbf{x}_i \cdot \mathbf{x}_t + b \tag{2.24}$$

Nonlinear SVM and Kernel Trick

In 1992, B. Boser, I. Guyon and V. Vapnik proposed a way to generalize linear SVM to nonlinear classifiers. A simple way to make a nonlinear classifier from a linear one is to the map the data from their input feature space $\mathcal{X}$ to a higher (potentially infinite) dimensional feature space $\mathcal{F}$ by using a nonlinear function $\Phi : \mathcal{X} \to \mathcal{F}$, such that $\mathbf{x} \mapsto \Phi(\mathbf{x})$. Hence, the resulting discriminant function will be nonlinear in $\mathcal{X}$ but linear (i.e. a hyperplane) in $\mathcal{F}$ (see Figure 2.10). It is given by:

$$\mathbf{w} \cdot \Phi(\mathbf{x}) + b \tag{2.25}$$

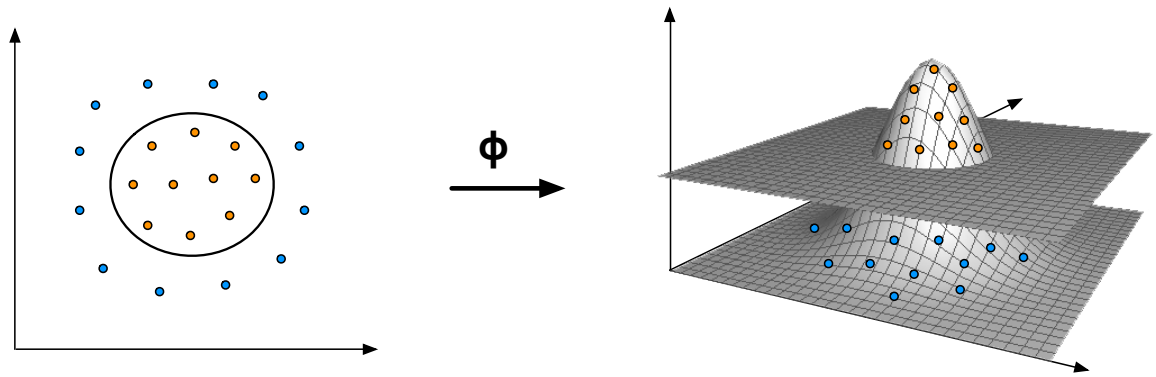


Figure 2.10: Principle of nonlinear SVM dealing with not linearly separable data (2D case). Samples are mapped in a higher dimensional space where they will be linearly separated.

In the linear SVM algorithm, the training samples only appear in the form of dot products $\mathbf{x}_i \cdot \mathbf{x}_j$. Therefore, to produce a nonlinear SVM, we just have to replace all dot products by $\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$. However, the dimension of the training samples being potentially very big, evaluating the output vector components of $\Phi(\mathbf{x})$, directly followed by the output value of $\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$ in the discriminating function, is unacceptable in terms of memory usage and CPU runtime. For example, consider a mapping $\Phi : \mathbb{R}^2 \to \mathbb{R}^3$, $\mathbf{x} = (x_1, x_2)^T \mapsto (x_1^2, \sqrt{2}x_1x_2, x_2^2)^T$. A quadratic increase will occur in memory usage and the CPU runtime to compute the discriminant function. The kernel trick consists in replacing each dot product by a known kernel function K that will avoid evaluating $\Phi(\mathbf{x})$ explicitely, as follows:

$$\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j) \overset{\text{Kernel trick}}{\underset{\downarrow}{=}} K(\mathbf{x}_i, \mathbf{x}_j) \tag{2.26}$$

In the preceding example, if we analytically develop and simplify $\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$, we obtain the kernel function that can be used instead, namely $K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i \cdot \mathbf{x}_j)^2$.

So, the resulting nonlinear algorithm remains formally similar as the linear one, except that every dot product is replaced by a nonlinear kernel function. The discriminant function

hyperplane is given by the following hyperplane equation:

$$\mathbf{w} \cdot \Phi(\mathbf{x}) + b = \sum_{i=1}^{n} y_i \alpha_i \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}) + b \overset{\text{Kernel trick}}{\downarrow} = \sum_{i=1}^{n} y_i \alpha_i K(\mathbf{x}_i, \mathbf{x}) + b \tag{2.27}$$

Some commonly used kernel functions $K(\mathbf{x}_i, \mathbf{x}_j)$ are given in Table 2.1. Because it often performs very well, one of the most frequently applied kernel functions is the *gaussian* one also *called radial basis function (RBF)*. Its γ parameter controls how much influence a single training sample has. The greater γ is, the closer other samples must be to be affected.

Kernel Type	**Kernel Function**	**Kernel Parameter(s)**
Linear	$K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i \cdot \mathbf{x}_j$	–
Polynomial	$K(\mathbf{x}_i, \mathbf{x}_j) = (\gamma\, \mathbf{x}_i \cdot \mathbf{x}_j + r)^d$	$\gamma > 0$
Gaussian (or RBF)	$K(\mathbf{x}_i, \mathbf{x}_j) = e^{(-\gamma \lVert \mathbf{x}_i - \mathbf{x}_j \rVert^2)}$	$\gamma > 0$
Sigmoid	$K(\mathbf{x}_i, \mathbf{x}_j) = tanh(\gamma\, \mathbf{x}_i \cdot \mathbf{x}_j + r)$	$\gamma > 0$, $r < 0$

Table 2.1: Kernel functions commonly used with SVM.

Multiclass SVM

The original SVM algorithm is a binary classifier. A simple way to make it multiclass consists in transforming the multiclass problem into several binary classification problems. Let m be the number of classes. The two prevalent approaches to do that are (see Figure 2.11[14]):

1. The **one-versus-all approach** – During the training phase, a separating hyperplane is built for each class against all the remaining classes taken together. Hence, m hyperplanes are computed from all training samples. During the test phase, the classification of a new unlabelled sample is done by a *winner-takes-all* strategy. The discriminant function that returns the highest output value will assign its class label to a given sample. There are several problems with this approach. First, a sample can be classified into multiple classes simultaneously, leading to inconsistent results. Then, the different classifiers are trained on different tasks; therefore it will not be granted that all discriminant functions will have suitable scales. Finally, the m training sets are unbalanced and strategies are needed to compensate for this.

2. The **one-versus-one approach** – During the training phase, a separating hyperplane is built for each pair of classes. Hence, $m(m-1)/2$ hyperplanes are computed from

[14]For the sake of clarity, only 4 hyperplanes over 6 are represented in the figure for the *one-versus-one approach* on the right.

the training samples of the concerned class pairs only. During the test phase, the classification of a new unlabelled sample is done by a *voting* strategy. Each discriminant function returns a class label for the given sample, corresponding to a vote for a class. All votes will be counted up and the class with the highest number of votes will assign its label to the sample (if two or more classes obtain the same number, the standard solution is to chose the first entry in the class name array). This is the most used approach (and the one that we used), because it doesn't have the drawbacks of the previous one.

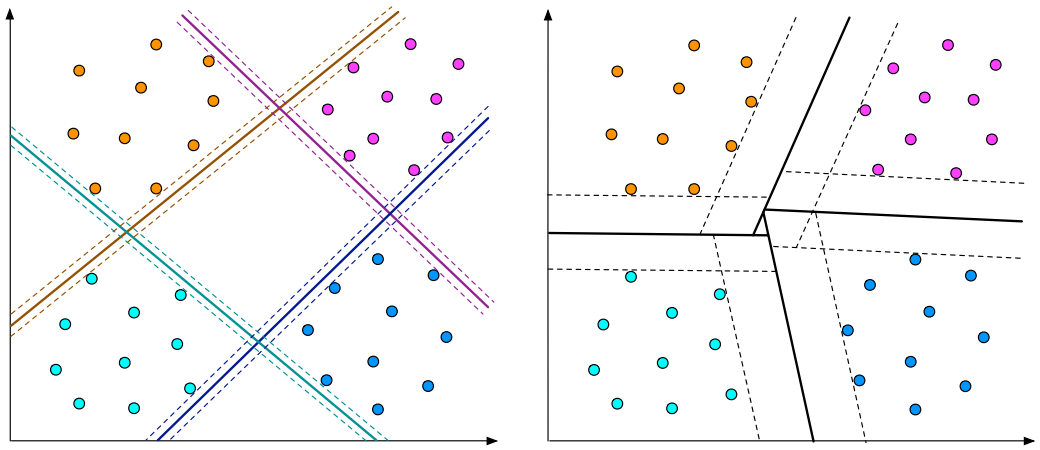


Figure 2.11: Principle of multiclass SVM with the *one-versus-all* (left) and *one-versus-one* (right) approaches (2D linearly separable case with 4 classes).

2.4.5 Multilayer Perceptron (MLP)

In this section, we give the theoretical background of the Multilayer Perceptron (MLP) which is a discriminative supervised method for classification, like SVM. MLP is the second classification algorithm used in this thesis[15].

Feedforward Operation and Classification

An Artificial Neural Network (ANN) is composed of several simple processing units connected together and called *neurons*. Directly inspired by the functioning of the central nervous system of living organisms, ANN can model complex and global behaviours thanks to their different neurons and connections (sometimes called *synapses*) determined by their (*synaptic*) weights. A Multilayer Perceptron is a particular type of ANN used to describe any general *feedforward network*, that is an ANN without recurrent connections. An MLP is under some conditions a *universal function approximator*, as shown in Cybenko's theorem [Cybenko, 1989]. Thus,

[15]This section is mainly based on the book [Duda et al., 2001].

an MLP can be trained and used as a classifier that will map input samples to output class labels (as explained in Section 2.4.3).

An MLP is a discriminant function $\mathbf{x} = g(\mathbf{x})$ that takes as input a d-dimensional vector (*sample*) $\mathbf{x}$ and returns as output a m-dimensional vector $\mathbf{z}$. This discriminant function can implement arbitrary decision boundaries in the d-dimensional space. The decision regions need neither to be convex, nor connected. As already said, an MLP is composed of simple processing units called neurons (see Figure 2.12). The output of a neuron is a function of the weighted sum of its inputs.

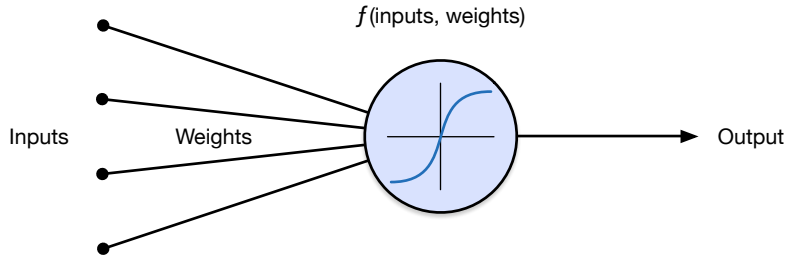


Figure 2.12: A neuron is a simple processing unit whose output is a function of the weighted sum of its inputs.

An MLP is a network of neurons composed of 3 (or more) successive layers: (1) one input layer, (2) one (or more) hidden layers, and (3) one output layer (see Figure 2.13). All neurons in a given layer are connected to all neurons in the following layer. Therefore, there are no cycle and the information can only transit forward, from the input neurons to the output neurons, via the hidden neurons. Then, the feedforward operation consists in presenting a vector (sample) to the input units and passing the values through the network, one layer after the other, until obtaining the final output from the output neurons.

During the process, each neuron transforms its input values to a single output value. First a net activation is computed by summing the weighted inputs. Then the *net activation* is transformed into an *output activation* by means of the activation function. Let d be the number of inputs and net_j the net activation of the j^{th} neuron in the hidden layer, then the net activation of this neuron is given by:

$$net_j = \sum_{i=1}^{d} x_i \, w_{ji} \tag{2.28}$$

Let the index i denote the neurons in the input layer and the index j those in the hidden layer, then w_{ji} denotes the weight of the input-to-hidden layer connection between input

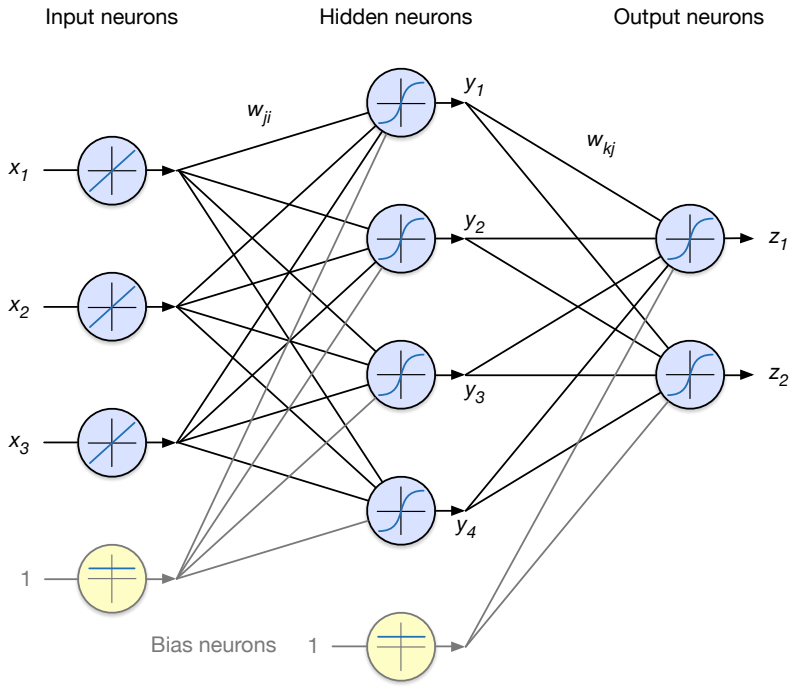


Figure 2.13: MLP with three layers formed by three input neurons, four hidden neurons and two output neurons.

neuron i and hidden neuron j. By analogy with neurobiology, these connections are sometimes called *synapses* and their values *synaptic weights*. The output activation of the j^{th} neuron of the hidden layer is a function of its net activation:

$$y_j = f(net_j) \tag{2.29}$$

The Equation 2.29 is generalizable for each neuron in any layer of an MLP. Let be an MLP with d input units, h hidden units, and m output neurons. In a classification problem, the m output neurons correspond to m different classes, and the signal of each output neuron is the discriminant function $g_k(\mathbf{x})$, such that:

$$g_k(\mathbf{x}) \equiv z_k = f(\sum_{j=1}^{h} w_{kj}\, f(\sum_{i=1}^{d} w_{ji}\, x_i + w_{j0}) + w_{k0}) \tag{2.30}$$

The MLP depicted in Figure 2.13 contains two *bias neurons* represented by the weights w_{j0} and w_{k0} in Equation 2.30. Such a bias neuron, whose output value is constantly equal to 1, is generally added to each layer (i.e. perceptron) of an MLP, except the output layer. In fact, a single perceptron needs an additional constant value (*bias*) in order to yield a separating hyperplane which does not pass by the origin of the space.

Backpropagation Algorithm and Training

The feedforward operation needs the connection weights of the network to be set. The backpropagation algorithm is a method that allows to infer these weights during the training phase thanks to a set of labeled samples. The algorithm computes the error between the actual output and the desired output of each neuron and updates the weights in such a way as to minimize this error. One strength of backpropagation is its simple and elegant way to compute the effective error for each hidden neuron, from which a learning rule may be derived to set the input-to-hidden weights.

Practically, the algorithm works as follows. A training sample is presented to the input layer. The signal spreads through the whole network using the feedforward method in order to compute the output values which are then compared with the target values. Their differences constitute the output errors. The general error is computed on a training sample. It is a scalar function of the weights, which is minimized when the network outputs match the desired outputs. Then, the synaptic weights are adjusted to reduce this error. Let $\mathbf{x}$ be a training sample, $\mathbf{w}$ the network weights, m the number of classes (output neurons), $\mathbf{t}$ the desired output, and $\mathbf{z}$ the actual output, then the training error $J(\mathbf{w})$ on this sample is given by the sum over the output neurons of the square difference between t_k and z_k:

$$J(\mathbf{w}) \equiv \frac{1}{2}\sum_{k=1}^{m}(t_k - z_k)^2 = \frac{1}{2}\left\|\mathbf{t} - \mathbf{z}\right\|^2 \tag{2.31}$$

The learning rule of the backpropagation is based on the gradient descent. Weights are changed in a direction that will reduce the error:

$$\Delta\mathbf{w} = -\eta\,\frac{\partial J}{\partial \mathbf{w}} \tag{2.32}$$

The so-called *learning rate* η allows to control the relative change size in weights. Backpropagation of the errors will lead to the update of the weights. Hence, the learning rules for updating hidden-to-output weights in a first time, and input-to-hidden weights in a second time, are given by:

$$\Delta w_{kj} = \eta\,(t_k - z_k)\,f'(net_k)\,y_j \tag{2.33}$$

$$\Delta w_{ji} = \eta\left(\sum_{k=1}^{m} w_{kj}\,(t_k - z_k)\,f'(net_k)\right)\,f'(net_j)\,x_i \tag{2.34}$$

These learning rules concern a three-layer MLP, nevertheless they can be easily generalizable for a MLP with a greater number of hidden layers.

The training (or learning process) of an MLP is performed by successively applying the backpropagation algorithm on a set of training samples. A backpropagation iteration is called learning *cycle* or *epoch*. The training phase starts with an untrained network having

its weights initialized with random values, and continues until the general training error is small enough. Three training protocols are commonly used: (1) stochastic, (2) batch, and (3) on-line training. In stochastic training, samples are picked randomly from the training set, and the network weights are updated after each sample presentation. In batch training, all the training samples are presented to the network before the weights are updated. In online training, each sample is presented only once to the network. Therefore samples don't need to be stored locally.

During the learning process, the training error becomes smaller as pass the epochs. This will lead to an asymptotic error on the training set (given that the learning rate is small enough). However, the average error computed in parallel on an independent sample set will generally be higher. Even worse, after a certain number of epochs, it will start to increase. Such a phenomenon is the result of an overtraining of the network whose decision boundaries overfit the training data (see Figure 2.14). In other words, the discriminative model built is too specific to the training samples, and is unable to generalize. To avoid overfitting, the training must stop after an appropriate number of epochs, when the error on a so-called *validation set* (whose samples are not used for training) has reached a minimum. Such a training method, based training and validation sets, is called *cross-validation*.

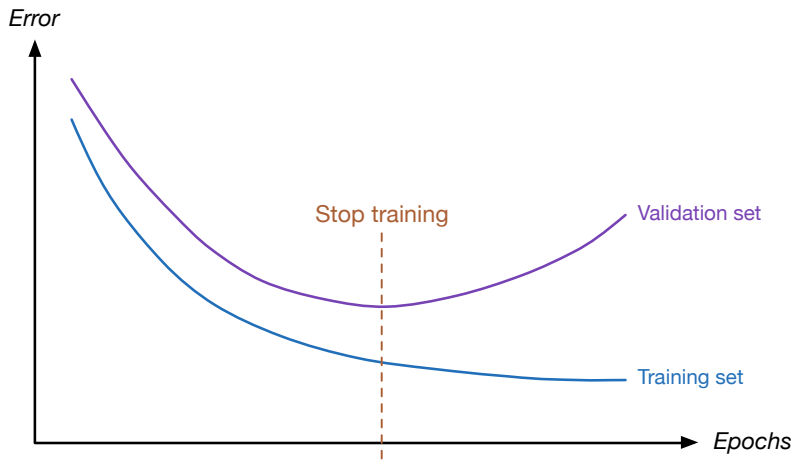


Figure 2.14: Evolution of the error of an MLP on both training and validation sets during the learning process.

The backpropagation algorithm usually performs better when the sample feature vectors are normalized (see Section 2.3.3). The network shall not be influenced by input features differing in magnitude. Moreover, small and balanced input values tend to avoid an excessive increase of the weights. Moreover, the normalization method to choose depends on the image domains of the chosen activation function as well as the target outputs.

Let $g_k(\mathbf{x}, \mathbf{w})$ be the output of the k^{th} neuron, i.e. the discriminant function corresponding to the class c_k, and suppose we train an MLP with m output neurons and a target signal t_k, such that:

$$t_k(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{x} \in c_k \\ 0 & \text{otherwise} \end{cases} \tag{2.35}$$

Then, in the limit of infinite input data, the network outputs of the trained MLP have been proven to approximate the true posterior probabilities in a least-squares sense; in other words, from a bayesian point of view, they represent the posterior probabilities: $g_k(\mathbf{x}, \mathbf{w}) \simeq P(c_k|\mathbf{x})$ (see Section 2.4.2). However, this is not true with limited training data and the output sum is not granted to be equal to 1. In the case of a significant sum difference, it may be an indication that the network topology is not adequate for modeling the posteriors, and should be modified. The common *exponential normalization* method allows to approximate the probabilities by applying the *softmax* function to the output neurons so that their values will lie in the range [0, 1] with their sum equal to 1.

MLP Parameters

An MLP has different parameters that should be tuned, and some techniques that may improve its performance for the classification task.

- The **activation function** must be continuous and differentiable to work with the back-propagation algorithm. It also has to be nonlinear in order to provide a 3-layer MLP its universal modeling power. The activation function should also be bounded in order to keep the weights and activations bounded, as well as to improve the training efficiency and time. This bound property is particularly useful when the network outputs are meant to represent probabilities. Other properties like monotonicity and linearity for small values of *net* are also desirable. Sigmoid functions, such as the *hyperbolic tangent* (see Equation 2.36) or the *logistic function* (see Equation 2.37), possess these properties. Figure 2.15 represents the shape of those functions.

$$f(net) = a \tanh(b\, net) = a \frac{e^{b\, net} - e^{-b\, net}}{e^{b\, net} + e^{-b\, net}}, \qquad a, b \in \mathbb{R} \tag{2.36}$$

$$f(net) = \frac{a}{1 + e^{-b\, net}}, \qquad a, b \in \mathbb{R} \tag{2.37}$$

- The **number of hidden layers** is usually set to 3. In fact, three layers have been proven to be sufficient to implement any function. Nevertheless, a greater number of layers shall be used for some problems. For example, a 4-layer MLP will have more facility to learn translations than a 3-layer one, because every layer is generally able to learn an invariance within a limited parameter range. Some functions shall also be

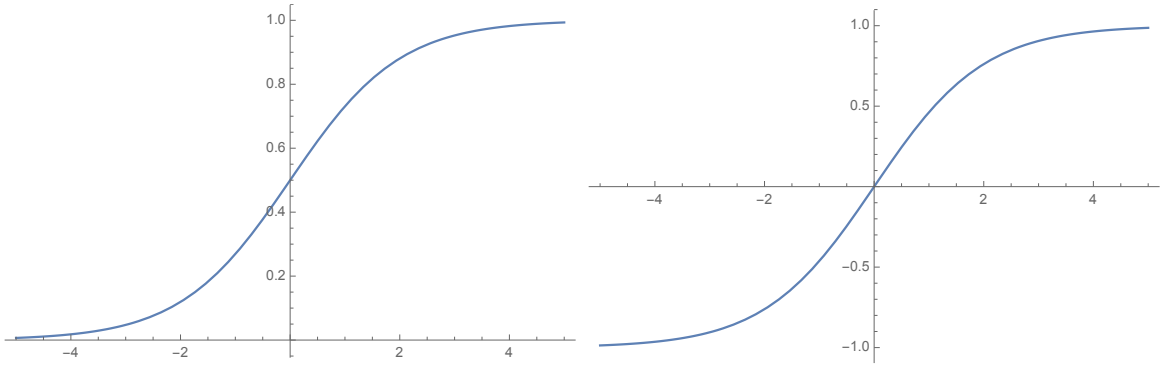


Figure 2.15: Representations of the standard logistic function (left) with $a = 1$ and $b = 1$, and the hyperbolic tangent (right) with $a = 1$ and $b = 1/2$. Both are sigmoid functions.

approximated more efficiently by a network that has less neurons in total, but more than one hidden layer. However, MLP with several hidden layers have shown to be more prone to fall in undesirable local minima, as well as to facilitate the learning of complex decision boundaries leading to overfitting. Therefore MLP with one unique hidden layer are often used.

- The **number of hidden neurons** is not as easily determined as the number of input and output neurons, which are determined by the dimensionality of the input feature vectors and the number of classes. Hidden neurons are tied to the complexity of the input densities, meaning that they must be numerous enough to model the corresponding decision boundary. That said, as for hidden layers, an excessive number of hidden neurons may lead to overfitting. Actually, there exists no general rule allowing to determine the adequate number of hidden neurons.

- The **initialization of the weights** with nonzero values is required in order to back-propagate the error and update the input-to-hidden weights. The initial weight values have a clear impact on the efficiency of the learning. Given normalized samples, the weights of an MLP are usually initialized to random values within the range of the input values and the bounds of the activation function. Let d and h be the numbers of input and hidden neurons, then choosing weights w_{ji} and w_{kj} in the following ranges will ensure a fast and uniform learning [Duda et al., 2001, Section 6.8.8]:

$$-1/\sqrt{d} < w_{ji} < 1/\sqrt{d} \tag{2.38}$$

$$-1/\sqrt{h} < w_{kj} < 1/\sqrt{h} \tag{2.39}$$

- The **learning rate** η determines the speed at which the network will reach a minimum in the error (criterion) function $J(\mathbf{w})$ (see Equation 2.31). If it is small enough, the convergence will in principle be granted. For a rapid and uniform learning, a particular

learning rate should be set for each weight according to the second derivative of the error function with respect to each weight. With MLP using sigmoid activation functions, a rate $\eta \simeq 0.1$ is often an adequate initial value. Then during the learning, the learning rate should be lowered when the error function diverges, or raised when the learning seems to be overly slow.

- The **momentum** will express the inertia of the weight update. During the learning, the error surface may have plateaus, i.e. regions in which the slope of $J(\mathbf{w})$ is very small. The momentum allows an MLP to learn faster when a plateau occurs in the error surface. At a given epoch t, the learning rule is changed by adding a percentage α of the previous weight update to the new one (typically $\alpha \simeq 0.9$), as follows:

$$\mathbf{w}_{t+1} = \mathbf{w}_t + (1-\alpha)\,\Delta\mathbf{w}_t + \alpha\,\Delta\mathbf{w}_{t-1} \tag{2.40}$$

- The **weight decay** permits to simplify the network and avoid overfitting by using the heuristic that the weights should be small. Its principle is to start with an MLP having "too many" weights and "decay" them during the training. Small weights favor models which are rather linear. After each update, every weight is simply "decayed" as follows:

$$w_{new} = w_{old}\,(1-\epsilon), \qquad \text{where } 0 < \epsilon < 1 \tag{2.41}$$

2.4.6 Gaussian Mixture Model (GMM)

In this section, we give the theoretical background of the Gaussian Mixture Model (GMM) which is a generative supervised method for classification. GMM is the third classification algorithm used in this thesis[16].

Mixture of Gaussians

Given a d-dimensional vector $\mathbf{x}$, a d-dimensional mean vector $\boldsymbol{\mu}$ and a $d \times d$ covariance matrix $\boldsymbol{\Sigma}$ with determinant $|\boldsymbol{\Sigma}|$, a multivariate *Gaussian density* is defined as:

$$\mathcal{N}(\mathbf{x}\,|\,\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}|^{1/2}} \exp\left(-\frac{1}{2}\,(\mathbf{x}-\boldsymbol{\mu})^T\,\boldsymbol{\Sigma}^{-1}\,(\mathbf{x}-\boldsymbol{\mu})\right) \tag{2.42}$$

The superposition of m Gaussian densities is called *mixture of Gaussians* and is given by:

$$p(\mathbf{x}) = \sum_{j=1}^{m} \pi_j\,\mathcal{N}(\mathbf{x}\,|\,\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j) \tag{2.43}$$

Each Gaussian density $\mathcal{N}(\mathbf{x}\,|\,\boldsymbol{\mu},\boldsymbol{\Sigma})$ is called *component* of the Gaussian mixture. To be valid probabilities, the so-called *mixing coefficients* π_j must satisfy:

$$0 \le \pi_j \le 1 \tag{2.44}$$

$$\sum_{j=1}^{m} \pi_j = 1 \tag{2.45}$$

[16]This section is mainly based on the books [Duda et al., 2001] and [Bishop and Nasrabadi, 2007].

Then, the *marginal density* is given by:

$$p(\mathbf{x}) = \sum_{j=1}^{m} P(j)\, p(\mathbf{x} \,|\, j) \tag{2.46}$$

This Equation 2.46 is equivalent to Equation 2.43 in which $\pi_j = P(j)$ corresponds to the probability of choosing the j^{th} component and $\mathcal{N}(\mathbf{x} \,|\, \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) = p(\mathbf{x} \,|\, j)$ to the probability of $\mathbf{x}$ conditioned on j. From Bayes' theorem (see Equation 2.3), the posterior probabilities $P(j \,|\, \mathbf{x})$, also called *responsabilities* $\gamma_j(\mathbf{x})$, are given by:

$$\gamma_j(\mathbf{x}) = P(j \,|\, \mathbf{x}) = \frac{p(j)\, p(\mathbf{x} \,|\, j)}{\sum_{k=1}^{m} p(k)\, p(\mathbf{x} \,|\, k)} = \frac{\pi_j\, \mathcal{N}(\mathbf{x} \,|\, \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}{\sum_{k=1}^{m} \pi_k\, \mathcal{N}(\mathbf{x} \,|\, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)} \tag{2.47}$$

where $\boldsymbol{\pi} = \{\pi_1, \ldots, \pi_m\}$, $\boldsymbol{\mu} = \{\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_m\}$ and $\boldsymbol{\Sigma} = \{\boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_m\}$ are the Gaussian mixture distribution parameters.

Maximum Likelihood

Let $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ be a training set of n samples coming from a given class y and $\mathbf{X}$ the representing $n \times d$ matrix in which the i^{th} row corresponds to $\mathbf{x}_i^T$. These data can be modelled with mixtures of Gaussians and later used for classification. From Equation 2.43, the logarithm of the so-called *likelihood function* is given by:

$$\ln p(\mathbf{X} \,|\, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi}) = \sum_{i=1}^{n} \ln \left(\sum_{j=1}^{m} \pi_j\, \mathcal{N}(\mathbf{x}_i \,|\, \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) \right) \tag{2.48}$$

The goal is to find the maximum of the this Equation 2.48 which constitutes a quite complex problem due to the presence of the summation over j inside the logarithm. Hence, there are no closed-form analytical solution for this problem and iterative numerical optimization methods have to be applied. The most famous and commonly used method is the *Expectation Maximisation (EM)* algorithm.

Expectation Maximisation (EM)

Given a GMM defined by its parameters $\boldsymbol{\mu}$, $\boldsymbol{\Sigma}$ and $\boldsymbol{\pi}$, the EM algorithm aims at maximising the (log) likelihood function with respect to these parameters. The EM steps are the following:

1. **Initialization step** – Initialize the m means $\boldsymbol{\mu}_k$, covariances $\boldsymbol{\Sigma}_k$ and mixing coefficients π_k (for $k = 1, \ldots, m$), generally by means of the *k-Means*[17] clustering algorithm, and evaluate the initial value of the log likelihood $\ln p(\mathbf{X} \,|\, \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$ (see Eqation 2.48).

[17] Given a sample set defined as above, the *k-Means* clustering algorithm aims at partitioning the n samples into k ($k \leq n$) sets (*clusters*) $S = \{s_1, \ldots, s_k\}$ such that, inside each cluster, the sum of squares of the distances between the samples and the centroid (i.e. the "mean sample") will be minimized. For further information, see [Duda et al., 2001] or [Bishop and Nasrabadi, 2007].

2. **Expectation step (E step)** – Evaluate the *responsabilities* (posterior probabilities) $\gamma_{ij} = \gamma_j(\mathbf{x}_i)$ by using the current parameter values:

$$\gamma_{ij} = \frac{\pi_j \mathcal{N}(\mathbf{x}_i \,|\, \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}{\sum_{k=1}^{m} \pi_k \mathcal{N}(\mathbf{x}_i \,|\, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)} \tag{2.49}$$

3. **Maximization step (M step)** – Reestimate the parameters using the current responsibilities:

$$\boldsymbol{\mu}_j^{\text{new}} = \frac{1}{n_j} \sum_{i=1}^{n} \gamma_{ij}\, \mathbf{x}_i \tag{2.50}$$

$$\boldsymbol{\Sigma}_j^{\text{new}} = \frac{1}{n_j} \sum_{i=1}^{n} \gamma_{ij}\, (\mathbf{x}_i - \boldsymbol{\mu}_j^{\text{new}})\, (\mathbf{x}_i - \boldsymbol{\mu}_j^{\text{new}})^T \tag{2.51}$$

$$\pi_j^{\text{new}} = \frac{n_j}{n} \tag{2.52}$$

where:

$$n_j = \sum_{i=1}^{n} \gamma_{ij} \tag{2.53}$$

4. **Evaluation step** – Evaluate the new value of the log likelihood:

$$\ln p(\mathbf{X} \,|\, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi}) = \sum_{i=1}^{n} \ln \left(\sum_{j=1}^{m} \pi_j \mathcal{N}(\mathbf{x}_i \,|\, \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) \right) \tag{2.54}$$

Check if either the parameters or the log likelihood have converged, and stop if it is the case. Otherwise return to step 2.

An example of the EM algorithm running on a set of samples is given in Figure 2.16[18]. Finally, the optimal number of gaussians m and potentially a variance floor value ν will have to be determined to provide an efficient model. In order to avoid the risk of overfitting which may occur in certain situations, e.g. when there is not enough data to represent the underlying class distributions, the variance values are limited to a floor value ν during the EM iterations [Melin and Lindberg, 1999].

Classification

Once the various GMM parameters have been determined from the samples of all classes to model, these mixture models can be used for the classification of a new unlabelled sample $\mathbf{x}_t$ as it is explained in the Bayesian decision theory (see Section 2.4.2, Equations 2.3 and 2.5).

[18] In this example, the Gaussian mixture parameters were not initialized using the k-Means algorithm.

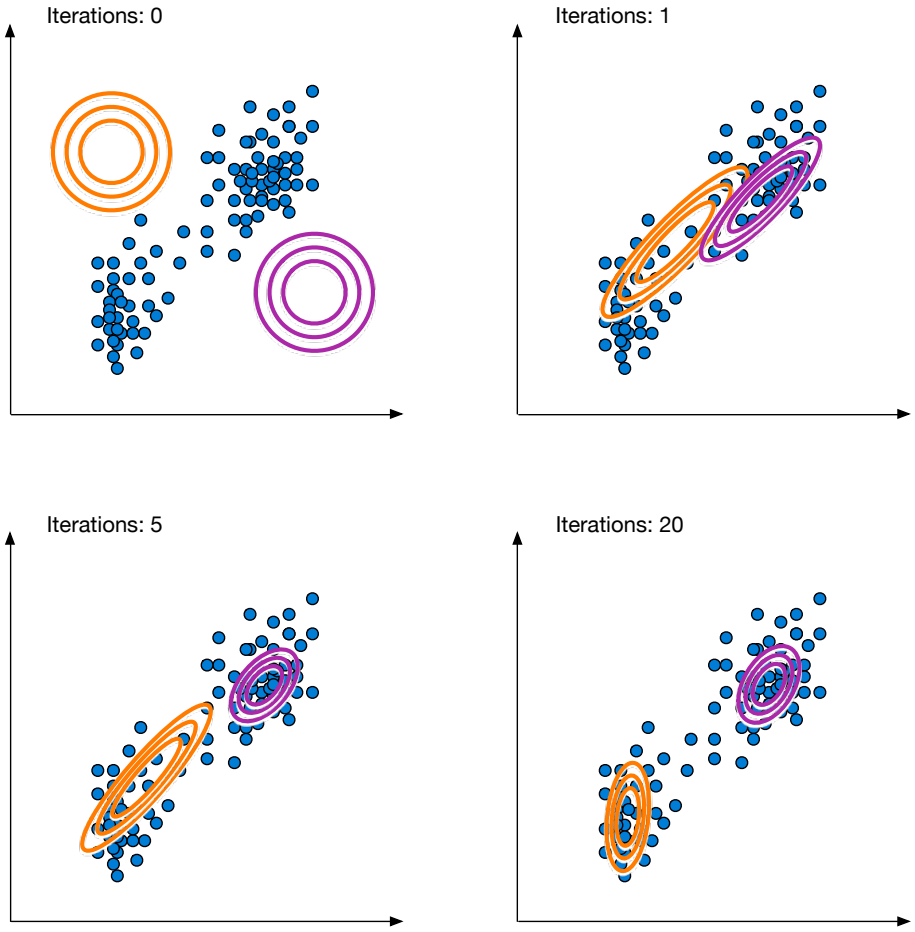


Figure 2.16: Illustration of EM algorithm running on a set of samples (2D case).

2.4.7 Classification Performance

As already explained, a *score function* is needed to evaluate and compare the generated models in order to find the best one. It will be decisive during the tuning of the parameters of the classifier. Various score functions can be used to compute various performance measures. Each one has a different sensibility to wrong and good *classification outcomes* (also called *errors* and *hits*). The score function is often tied to a *cost function* measuring the consequences of a certain classification outcome relatively to each class in the context of the problem. Thus, it must be carefully chosen.

Classification outcomes

For the t incoming samples to classify, there will be four possible classification outcomes:

- *True Positives (TP)*, representing samples which were correctly classified as belonging to the class;

- *True Negatives (TN)*, representing samples which were correctly classified as not belonging to the class;
- *False Positives (FP)*, representing samples which were classified as belonging to the class but should not have been;
- *False Negatives (FN)*, representing samples which were classified as not belonging to the class but should have been.

Confusion table and confusion Matrix

TP, TN, FP and FN are often represented in the so-called *confusion table* (see Figure 2.17). The *confusion matrix CM* allows to describe classification errors under another similar matrix form where the headers of the rows represent the predicted classes and those of the columns the true classes. At a given row r and column c intersection, an entry corresponds to the number of samples of the class r that were classified into the class c. A perfect classifier would have no classification errors and therefore positive numbers in the diagonal only, i.e. with a matrix trace $Tr(\mathrm{CM}) = n_t$, where n_t is the number of classified test samples. For a binary classification problem, both confusion table and matrix are equal. Sometimes percentages are given instead of absolute numbers.

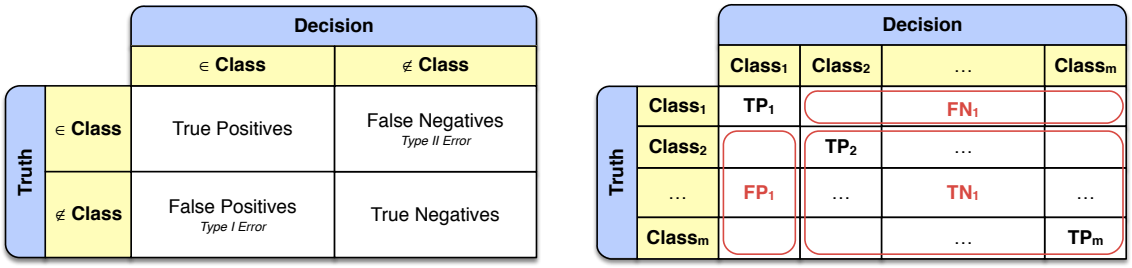


Figure 2.17: The four different types of errors represented in the confusion table and matrix. For a given class, the confusion table can be obtained by summing the corresponding values in the confusion matrix, as shown here by the red lines for Class_1.

Performance measures

In order to evaluate the performance of a binary classifier, various measures can be computed (see Table 2.2). The most known performance measures are probably the *accuracy*, the *precision* and the *recall*. The *accuracy* represents the proportion of correctly classed samples, i.e. true positives and negatives, relatively to the entire data set. The *precision* represents the proportion of positive tested samples which were correctly classed. The *recall* represents the proportion of samples belonging to the class which were correctly classed.

Performance Measure	**Formula**
Accuracy	$\frac{\#\text{Correct Samples}}{\#\text{Samples}} = \frac{TP+TN}{TP+FP+TN+FN}$
Precision (or Positive Predictive Value)	$\frac{TP}{TP+FP}$
Recall (or Sensitivity or True Positive Rate or Hit Rate)	$\frac{TP}{TP+FN}$
Negative Predictive Value	$\frac{TN}{TN+FN}$
Specificity (or True Negative Rate)	$\frac{TN}{TN+FP}$
False Positive Rate (or Fall-Out or 1−Specificity)	$\frac{FP}{FP+TN}$
False Discovery Rate	$\frac{FP}{FP+TP}$
Balanced Accuracy	$\frac{\text{Sensitivity} + \text{Specificity}}{2} = \frac{\frac{TP}{TP+FN}+\frac{TN}{TN+FP}}{2}$
F1 Score (or F-Measure or Balanced F-Score)	$2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$
Matthews Correlation Coefficient (or Phi Coefficient)	$2 \cdot \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$

Table 2.2: Performance measures for binary classification, i.e. with $m = 2$ classes.

Some other measures can be used in particular situations. The *balanced accuracy* allows to avoid inflated performance estimates on unbalanced data sets. The *F1 score* is the harmonic mean of precision and recall, i.e. the weighted average of the precision and recall. It reaches its best value at 1 and worst score at 0. The *Matthews correlation coefficient* is a kind of balanced measure that can be used even if the classes are of very different sizes. Related to the *chi-square* statistic, it is in essence a correlation coefficient between the observed and the predicted binary classifications. It ranges between -1 (total disagreement between prediction and observation) and +1 (perfect prediction), 0 being not better than random prediction.

In order to evaluate the performance of a multi-class classifier, *accuracy*, *precision* and *recall* have to be macro- or micro-averaged on the m classes (see Table 2.3).

Performance Measure	**Macro-Average Formula**	**Micro-Average Formula**
Accuracy	$\frac{1}{m}\sum_{j=1}^{m} \frac{TP_j+TN_j}{TP_j+FP_j+TN_j+FN_j}$	$\frac{\sum_{j=1}^{m} TP_j+TN_j}{\sum_{j=1}^{m} TP_j+FP_j+TN_j+FN_j}$
Precision	$\frac{1}{m}\sum_{j=1}^{m} \frac{TP_j}{TP_j+FP_j}$	$\frac{\sum_{j=1}^{m} TP_j}{\sum_{j=1}^{m} TP_j+FP_j}$
Recall	$\frac{1}{m}\sum_{j=1}^{m} \frac{TP_j}{TP_j+FN_j}$	$\frac{\sum_{j=1}^{m} TP_j}{\sum_{j=1}^{m} TP_j+FN_j}$

Table 2.3: Performance measures for multi-class classification, i.e. with m classes ($m \geq 2$).

ROC Curve & AUC

In case of a binary classification, the *ROC curve* can be obtained by plotting the points formed by the *true positive rate* (also known as *sensitivity*) and *false positive rate* (also equal to 1− *specificity*) values computed as the so-called *discrimination threshold* T varies from the minimal to the maximal score returned by the binary classifier. The *area under the ROC curve* is a good indicator of the classifier efficiency at solving the problem. The larger the area is, the better the performance is. The area is measured relatively to the area of the square (1.0). Hence, an ideal classifier with 100% TP and 0% FP for all T would yield a ROC curve that jumps immediately from 0% to 100%, corresponding to an AUC of 1.0, while a very bad classifier that has learned nothing from the data would yield a ROC curve close to the square diagonal which corresponds to pure random guessing and AUC of 0.5.

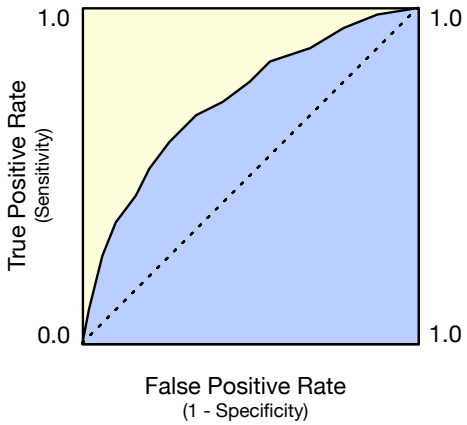


Figure 2.18: Receiver operating characteristic (ROC) curve and area under curve (AUC) of a binary classifier.

Confidence interval

Once the performance measures have been computed, the resulting values must be given with a confidence interval CI which, at a risk α for an estimated performance value p, is given by:

$$CI_{1-\alpha} = p \pm z_\alpha \sqrt{\frac{p\ (1-p)}{t}} \tag{2.55}$$

where t is the number of tested samples, $1-\alpha$ is the probability of being within the confidence interval and z_α follows from the inverse cumulative distribution function[19] and is equal to 1.96 for $\alpha = 5\%$. To be valid, the confidence interval must ensure that: $t * CI_{inf} \geq 5$ and $t * (1 - CI_{sup}) \geq 5$.

[19] `http://www.maths.manchester.ac.uk/~cds/internal/tables/normal.pdf`

3

Case Study: Appliance Consumption Signature Classification

Success in creating AI would be the biggest event in human history. Unfortunately, it might also be the last, unless we learn how to avoid the risks.

Stephen Hawking

Contents

3.1 Introduction

3.1.1 Context

Nowadays, growing prices of energy, political objectives of governments or simply personal convictions encourage people to look for solutions reducing their environmental impacts. In Switzerland, households use about 30% of the electricity consumed, industry, arts and crafts

33%, services 27%, transportation 8% and agriculture 2%. The electricity bill is established on the basis of the number of consumed kilowatt-hours. In the last years, the price of a kilowatt-hour varies from 0.1 to about 0.4 CHF depending on the location and use (e.g. domestic use, cooking, heating, business, or night rate). Electricity consumption emits CO_2 into the atmosphere. The quantity emitted per kWh depends on the facility producing the electricity. One kilowatt-hour provided by a coal power plant emits more than one kilogram of CO_2 while the one provided by a water turbine produces less than 50 grams. As most countries have several sources of electricity, we talk of *energy mix* to estimate the quantity of CO_2 which is emitted per kWh [Energie-environnement.ch, 2015].

From this perspective, a substantial progress can be made in the optimization and control of the electrical consumption of appliances in habitations. Everyday, new smart meters are installed in houses, and plug based energy monitoring devices are now available. Recent publications in the domain report on several mobile and web based energy monitoring systems aiming at providing pertinent information to the user [Weiss et al., 2009; Guinard et al., 2009]. Studies have shown that a continuous information feedback and fine-tuned automated management of home equipments would allow an energy bill reduction from 15% up to 30% [Neenan, 2009; Morel, 2008; Darby, 2006]. However, such solutions are still expensive, complicated to configure and often not well accepted due to their rudimentary automation strategy based on simple decision thresholds. To help those systems become more adaptive, efficient and easier to use, an important task is the automatic recognition of the appliances running in a house based on their electrical consumption profiles or *signatures*. Moreover, automatic appliance recognition has other applications to offer, such as the detection of defects, the localization of appliances in offices or hospitals, or the recognition of abnormal uses of appliances (intrusion detection or elderly surveillance).

This case study fits into the general field of *Green Computing* which is a generic term covering the topics of *Green-IT* and *IT-for-Green*. More formally, *Green Computing* can be defined as the set of all information technologies, computer resources and methodologies developed and applied in an efficient way (*Green-IT*) and/or for an ecological purpose (*IT-for-Green*) in order to respect, protect, preserve or restore the environment (see Figure 3.1).

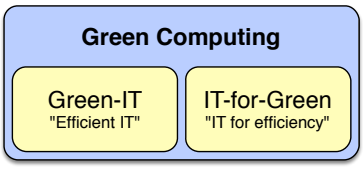


Figure 3.1: Relationship between Green Computing, Green-IT and IT-for-Green.

3.1.2 Objectives

In this case study, we exploited electrical consumption signals acquired using different sensors installed at the level of the electricity plugs in a house in order to automatically recognize various families of electrical appliances. Such sensors consist in affordable and low-consuming "smart" outlets able to measure at a low frequency several characteristics of the electrical consumption, like the power and current. We proposed algorithms for signal normalization, signal processing, signal modeling with the ultimate goal of a more efficient, more comfortable control of house appliances[1].

A first objective was to build a consequent database containing consumption signatures of various families of home appliances. In a previous paper [Zufferey et al., 2012], we showed that machine learning techniques can be used to perform appliance recognition. Such techniques are very powerful thanks to their ability to learn models from data and to generalize the recognition on unseen appliances. However, the performances of those algorithms highly depend on the quality of the training data set. Thus, a challenging part of this database creation was the settling of a data acquisition protocol that could allow us to elaborate and propose recognition test protocols flexible and precise enough to benchmark the classification performances of future algorithms. For example, to obtain representative appliance signatures, we had to deal with all possible running states of the different appliances. Finally, we proposed two classification test protocols and made the database freely available to the whole scientific community on a website [Watt-ICT, 2013-2016]. Hence, researchers wanting to perform machine learning tasks on those appliance consumption signature databases will have a common basis to do their tests and performance comparisons.

A second objective was to apply data-driven classification methodologies to build models of electrical appliances in order to recognize them when they are plugged in a power outlet in a given room. This can be used to compute probabilities of using the different appliances, or probabilities of user activities at a certain time and place in the house. Thanks to these models, added-value information can be computed and exploited for the energy management, for example via actuators in the house. The signals being temporal and state based, this case study was a typical case of time series classification where a generic data-driven approach to time series could be applied. We tested three classifiers, namely SVM, GMM and MLP, with two different approaches: a global and a local one.

[1]This work was part of the project Green-Mod supported by the research grant of the Hasler Foundation and by the University of Applied Sciences Western Switzerland (HES-SO).

3.1.3 Outline

This chapter is organized as follows. First, in Section 3.2, we give the specific elements of theory related to electrical energy consumption. In Section 3.3, we present the State of the Art in the field of appliance monitoring and recognition. Then in Section 3.4, we describe the two created appliance consumption signature databases in terms of data acquisition protocol, database contents and formats. In this section, we also describe the test protocols that were proposed to researchers wanting to perform machine learning tasks like appliance recognition on our databases. In Section 3.5, we describe the experiments and discuss the results obtained in the task of appliance recognition on our databases, applying our test protocols and generic approach using both discriminative (SVM, MLP) and generative (GMM) classifiers. Finally, in Section 3.6, we conclude and give some research perspectives in the domain.

3.2 Specific Elements of Theory

Before talking of *appliance consumption signature classification*, some specific elements of theory about electricity, energy and consumption must be introduced[2].

3.2.1 Electrical Signal

The electrical[3] signal is measurable and characterized by the following physical quantities:

- The **voltage** U is the potential energy source in an electrical circuit. It is sometimes called *electromotive force (EMF)* and its unit of measure is the volt [V]. Voltage pushes or pulls current, and makes it flow in a conductor (wire).

- The **current** I is the flow of electrons in a conductor. Electrons are pushed and pulled by voltage through an electrical circuit (closed-loop path). The electrons flowing in a conductor return to their voltage source. Current is measured in amperes (or amps) [A]. One ampere represents 6.28×10^{18} electrons flowing in a conductor per second. The number of electrons never decreases in a circuit but its flow produces heat due to the resistance of the conductor.

- The **electric power** P is the rate at which electric energy is transferred by an electric circuit. It represents the quantity of energy consumed each second. Its unit of measure is the watt [W] and one watt is equal to one joule per second [J/s]. Voltage or current by itself does not do any real work. However, voltage and current together can produce real work. Equal to the product of voltage times current ($P = UI$), power thus produces

[2]Here, we mainly refer to a very good book written by S. W. Blume on the subject [Blume, 2007].

[3]The adjective *electrical* usually refers to anything related to electricity in a general sense, while *electric* refers to some specific thing that runs on electricity. Both terms have a large semantic overlap and are indifferently used with many nouns like *appliance*.

real work. Note that watts are also used to describe the power produced by a device, even if it is not electrical, for example the traction power of a car motor.

- The **electrical energy** E is the product of electric power and time. It is the amount of time an appliance is ON (i.e. in which the current is flowing) times the amount of power (watts) consumed by the appliance. Electric energy is measured in watt-hours [Wh], and more commonly in kilowatt-hours [kWh] for residential usage and megawatt-hours [MWh] for large industries.

- **AC/DC current and voltage** – In the case of a direct current (DC), electrons flow in the same direction in a circuit, because the voltage is kept constant. Electrons transit from the negative to the positive terminal. In the case of an alternating current (AC), the terminals of the potential energy source (i.e. the voltage) alternate between positive and negative making the current alternate between positive and negative in the circuit. The variation of the voltage over the time mathematically describes a sine wave.

- The **frequency** represents the number of cycles, i.e. the number of times the voltage source alternates per second. It is measured in hertz [Hz] (or [s^{-1}]). In most of the world including Europe, the frequency is 50 Hz, although in USA, Canada and some parts of Asia it is 60 Hz.

3.2.2 Alternating Current Circuits

In the case of AC circuits, several supplementary physical quantities have to be distinguished:

- The **active power** (also called **real power** or **true power**) P corresponds to the average power consumed over one period (i.e. a complete cycle of the AC waveform). It is expressed in watts [W]. In other words, it is the portion of power resulting in a net transfer of energy in one direction over one period. For a current $i(t)$ and a tension $v(t)$ of period T, its expression is:

$$P = \frac{1}{T} \int_T v(t)i(t)dt \tag{3.1}$$

  For a sinusoidal tension of effective value U and a sinusoidal current of effective value I out of phase relatively to the tension (see Figure 3.2), this expression becomes:

$$P = U * I * cos(\varphi) \tag{3.2}$$

  In alternating current circuits, the ratio between maximum and effective value is worth $\sqrt{2}$. It is the only power to have a direct physical meaning. At last, it is the first feature to use to discriminate appliances with dissimilar consumptions.

- The **reactive power** Q corresponds to the portion of power, due to stored energy, which returns to the source during each period. It is expressed in volt-amperes reactive [var]. It is also the imaginary part of the complex apparent power (see below).

- The **complex power** is the vector sum of the active power P and the reactive power Q. Understanding the relationship between real, reactive and apparent powers, that are noted P, Q and S, lies at the heart of understanding power engineering. The mathematical relationship among them can be represented by vectors or expressed by using complex numbers, $S = P + jQ$, where j is the imaginary unit (see Figure 3.3).
- The **apparent power** (or **total power**) S is measured as the magnitude of the complex power. It is expressed in volt-amperes [VA] and: $S^2 = P^2 + Q^2$. The current associated with reactive power does no work in the circuit, but it heats the wires and wastes energy. Hence, circuits must be sized to carry the total current (which is associated with apparent power), not just the current that does the work.
- The **phase angle** φ is the angle of difference, in degrees [°], between the voltage U and the current I.
- The **power factor** is defined as the ratio between the active power P and the apparent power S in a circuit. It is a practical measure of the *efficiency* of a power distribution system. In the case of a sinusoidal waveform, the power factor is equal to the cosine of the phase angle φ and $cos(\varphi) = \frac{P}{S}$. This feature is useful to discriminate appliance types, namely resistive, capacitive and inductive appliances.

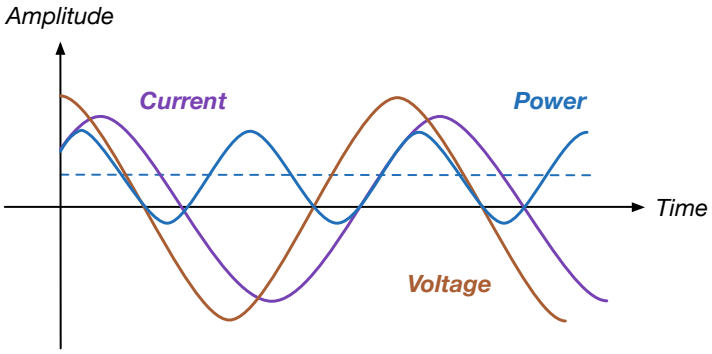


Figure 3.2: Relationship between current, voltage and power in an inductive AC circuit.

3.2.3 Electrical Energy Consumption

The electrical energy consumption corresponds to the energy used by the appliances plugged on the power system. It also includes the energy used to transport and provide the energy, for example the energy lost via heating of the circuitry. The overall consumption is constantly growing with years. Measuring the electrical energy consumption depends on the context of use often classified into *residential*, *commercial* and *industrial*.

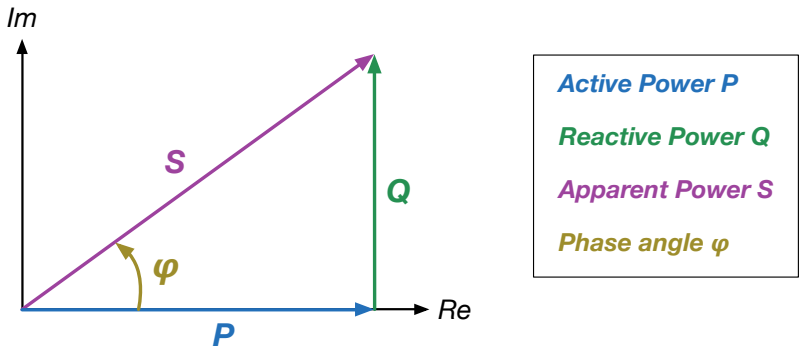


Figure 3.3: Relationship between active power, reactive power, apparent power and phase angle.

Electrical energy consumption also depends on the circuits contained in the appliances connected to the power system. We distinguish three kinds of circuits: *resistive*, *inductive* and *capacitive*. Each one behaves and interacts with others in a different way. When a circuit is plugged to an AC voltage source, a *phase angle* may appear between the voltage and current. The phase angle is a time difference usually measured in degrees, with 360° per period. Figure 3.4 shows the Voltage-Current (V-I) relationship in the three types of circuits.

1. In a **resistive** circuit, the current is in-phase with the voltage.

2. In an **inductive** circuit, the current lags the voltage.

3. In a **capacitive** circuit, the current leads the voltage.

If we add inductive to capacitive circuits, their phase angles will oppose each other. In most cases, their sum will result either in a lagging inductive or a leading capacitive phase angle. In the best case, it will result in a null resistive phase angle, as all phase angles entirely cancel each other. The *efficiency* of a power system is maximized when the combination of all connected appliances results in a circuit which is purely resistive. In this case, the efficiency of the system is maximal, while current losses and requirements (i.e. the total power which has to be produced) are minimal. The *total power* becomes *active power* only (i.e. *Watt power*), and extra capacity is made available in the transmission lines, distribution lines and substation equipment. As said before, the efficiency of a power system is determined by computing its *power factor*. It depends on the total power required to do the real work. We talk of a *good* (or *high*) power factor when it is greater than 95%, and a *poor* (or *low*) power factor when it is less than 90%. When the power factor of a circuit (like in some motors) is too low (between 80% and 85%), capacitors are added to improve its overall efficiency.

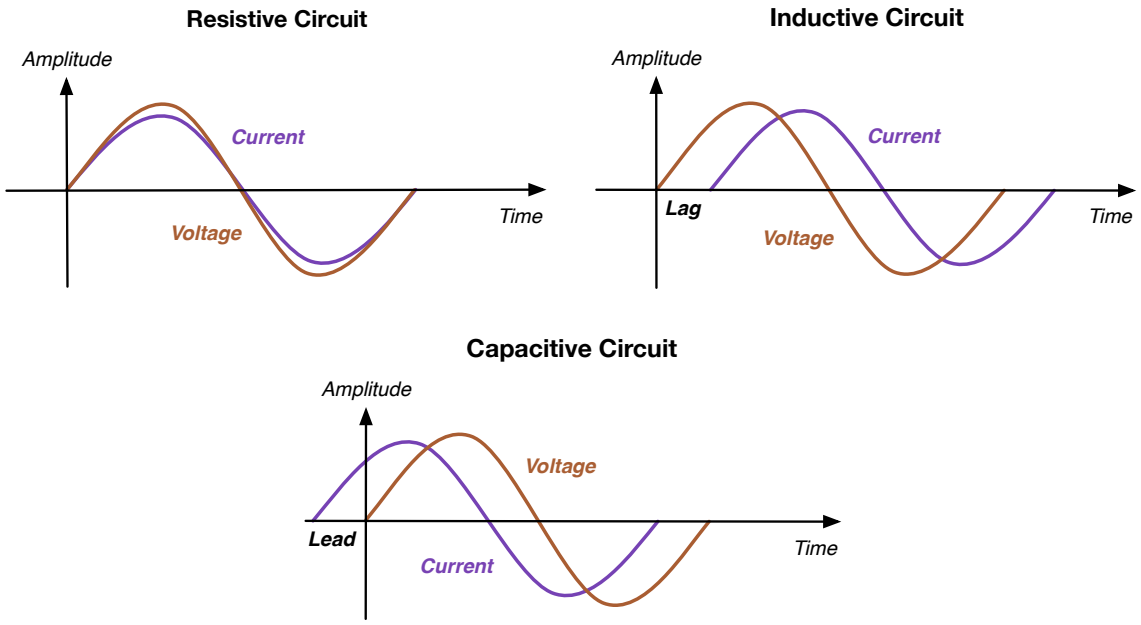


Figure 3.4: Relationship between voltage and current in resistive, inductive and capacitive circuits.

3.2.4 Appliance Consumption Signature

In general, an electrical consumption signal has a relative long lifetime and may be recorded for a relative long period (from a few seconds to several hours). The acquisition process consists of making regular measures of the electrical consumption of a given appliance. Thus, dedicated sensors can sample the electric consumption signal of running appliances and measure the various physical quantities presented above over the time. The resulting data structure is what we call an *Appliance Consumption Signature (ACS)* and, according to definition 2.1, it is a (potentially multivariate) *time series*. It can be represented by a curve on a chart. Figure 3.5 shows an example of such a signature. Appliance Consumption Signature (ACS) constitute the raw time series inputs of this case study.

3.3 State of the Art

Appliance signature classification (or recognition) is a particular field of a bigger domain called Appliance Load Monitoring (ALM). Over the last few years, many researchers have been working on this thematic in order to understand electrical energy consumption, improve the efficiency of the various appliances installed in buildings, and save electricity/energy[4].

[4]In the literature, the terms *electricity* and *energy* are so often either exchanged or mixed when they are associated to the term *energy*. The nature of energy can be of different natures, not only electrical. Therefore the expression *electrical* consumption is preferred in this document.

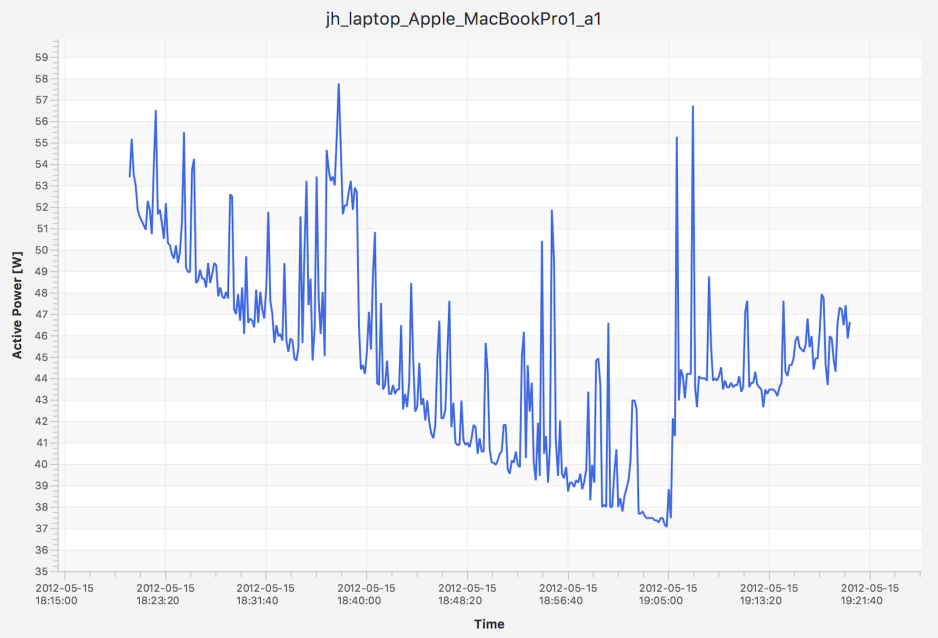


Figure 3.5: Appliance consumption signature (power curve) of a laptop.

One of our contributions is a survey on that topic, which we summarized in this section [Ridi et al., 2014a].

3.3.1 Approaches to Appliance Load Monitoring (ALM)

The two main approaches to the problem of Appliance Load Monitoring (ALM) are Non-Intrusive Load Monitoring (NILM) and Intrusive Load Monitoring (ILM) [Hart, 1992]. As our case study places itself in an ILM context, we put the emphasis on that approach.

Non-Intrusive Load Monitoring (NILM)

Sometimes also called *single-sensor metering*, Non-Intrusive Load Monitoring (NILM) relies on the recent smart meters to monitor the overall load of all appliances running on the electrical network at a single entry point of a house (see Figure 3.6). Featured by a single piece of hardware and a unique spot to collect data, this approach is qualified as *non-intrusive* and generally permits to obtain electrical consumption data at a minimal cost and installation time. In NILM, the appliance recognition process is usually divided into three steps: the feature extraction, the event detection and the load appliance disaggregation. In fact, multiple independent appliance consumption signatures are recorded simultaneously and the overall load corresponds to the sum of all electrical consumption signals (see Figure 3.7). Hence, the main objectives are the recognition of typical *events* produced by some appliances in the overall electrical consumption signal as well as the determination of the contribution of each

single appliance. This last point (*disaggregation phase*) constitutes the main difficulty of the approach.

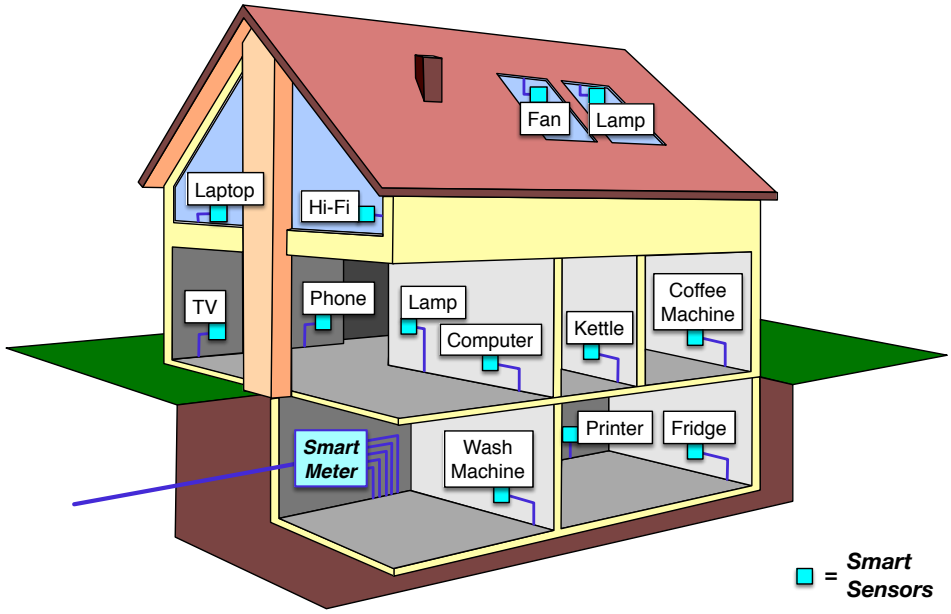


Figure 3.6: House equipped with a smart meter allowing a NILM (II-III) approach and smart sensors at the plugs allowing a ILM approach.

Two methods are generally applied to analyse the appliance consumption signals:

1. The **steady-state analysis**, which consists in analyzing the stable state of the power consumption. The *stable state* is defined as *a difference between any two samples of a sequence that does not exceed a given tolerance value* [Figueiredo et al., 2012, 2011]. For the steady-state analysis, the most applied technic is the *power change analysis* consisting in analyzing the active or even the reactive power of the total power consumption. Its main drawback is the difficulty to separate appliances with similar consumption signatures. Other methods based on the analysis of the current, the voltage or other time and frequency features are also applied. Trajectories in the voltage-current domain or the voltage noise can also be exploited [Zoha et al., 2012].

2. The **transient analysis**, which consists in looking for relevant changes in the power signal and analyzing the portion of the signal considered as unstable. When a given appliance is switched on or off, some specific oscillations appear on the voltage and current signals and they permit to recognize the appliance. The main drawback of this method is that more expensive analog-digital converters are needed. They must be insensitive to the general noise generated by the remaining appliances in the power network and

must operate at high sampling frequency to capture transients. Then, a frequency domain analysis is usually done to extract features representing the transitions [Leeb et al., 1993, 1995; Chan et al., 2000; Patel et al., 2007].

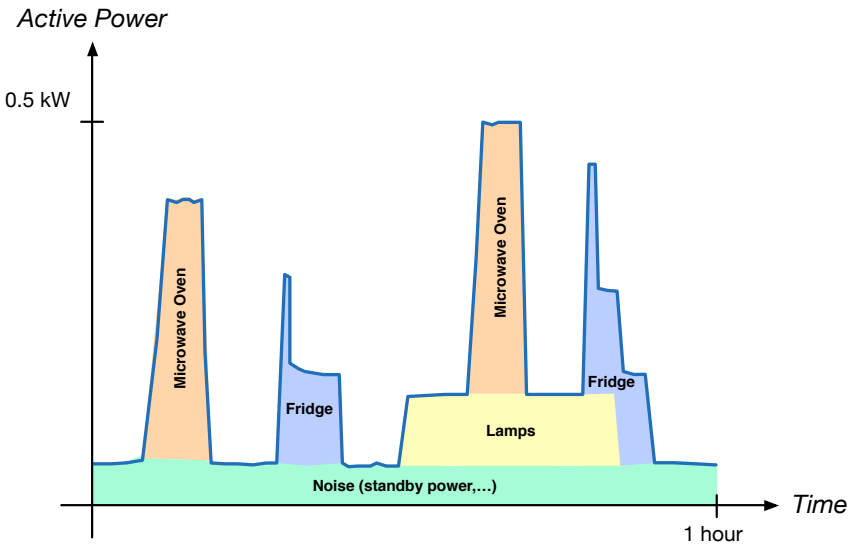


Figure 3.7: NILM relies on smart meters capturing the electrical consumption signals all together.

Intrusive Load Monitoring (ILM)

Sometimes also called *multi-sensor metering* [Weiss et al., 2012], *Intrusive Appliance Load Monitoring systems (IALMs)* [Robert and Liszewski, 2013; Jiang et al., 2012], *Hardware based Sub-Metering* [Paradiso et al., 2013; Berges et al., 2008] or *Distributed System/Metering* [Huang and Lai, 2013; Reinhardt et al., 2012b], Intrusive Load Monitoring (ILM) relies on low-end plug based devices (*sensors*) installed between appliance power plugs and electricity outlets. Such devices measure the electric consumption of each appliance individually (see Figure 3.8). The term *intrusive* means that the sensors are located in the house. The multi-sensor metering approach allows a fine-grained analysis, given that appliance consumption signatures are directly available. Once the feature extraction is done, appliance classifiers can directly be trained and new unknown data can be classified into the predicted appliance category. As explained in our survey [Ridi et al., 2014a], ILM can be divided in three domains:

1. **ILM I** in which the sensors measure the electric consumption after the primary utility meter. Such sensors are typically dedicated to monitor a zone of the house and are often placed at the circuit breaker level.

2. **ILM II** in which the sensors are located at the plug level in order to monitor appliances which are connected to the outlet or multi-outlet.

3. **ILM III** in which the sensors are directly embedded in the equipment or dedicated smart outlet to monitor each single appliance (see Figure 3.6).

Approach Comparison

Research in the NILM domain started during the 80s [Sultanem, 1991; Hart, 1992]. From then, it has mainly concerned NILM and ILM III domains, and publications in NILM outnumber those in ILM. Table 3.1 shows the main differences between NILM and ILM approaches.

	NILM	**ILM I**	**ILM II**	**ILM III**
Sensor(s)	One: 1/house	Few: 1/zone	Several: 1/plug	Many: 1/appliance
Intrusiveness	None	Low	Moderate	High
Costs	High	Moderate	Low	Low
Data Granularity	Very rough	rough	Fine	Very fine
Modeling	Very difficult	Difficult	Medium	Easier

Table 3.1: Comparison between the NILM and the 3 ILM approaches.

As compared with NILM, the ILM approach has several advantages. As sensors are more numerous and directly close to single appliances, more precise data are acquired and the ACS modeling is easier. The recognition of low power appliances or appliances in standby is also made possible [Marchiori et al., 2011], an important fact given that standby power has become one of the largest source of consumption in residences recently (up to 26% of the total electricity consumption) [Harrington et al., 2007; Ross and Berkeley, 2000].

ILM has also some disadvantages. Due to numerous sensors, the installation procedure takes more time, is more difficult and potentially expensive. That said, ILM sensors are cheaper as NILM ones, and with time and technological progress, their prices are coming down. Plug-based sensors can sometimes not be used with big hard-wired appliances like electric heaters. However, there exists current transducers rolling up the wire that can be used in such cases [Cease and Johnston, 1990].

Hence, for both approaches, the deployment of sensors in a real home environment still yields many technical challenges. To be popularized by the residents, such systems will have to be rather cheap, not intrusive, easy to install, configure and maintain and green (i.e. ecological, not polluting, with low energy consumption and no electromagnetic radiations).

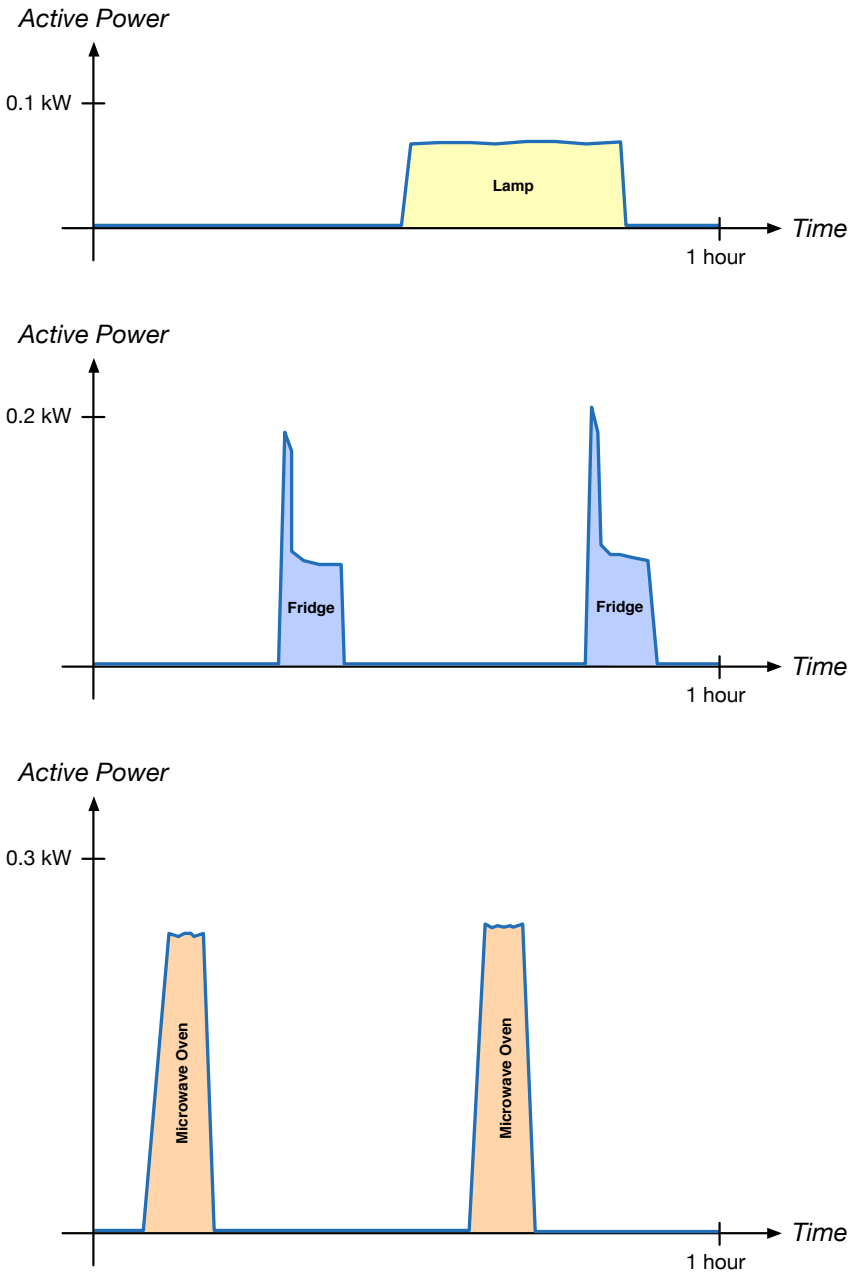


Figure 3.8: ILM relies on several smart sensors which capture the electrical consumption signals separately. Such signals can be more easily analyzed than aggregated signals acquired in NILM (see Figure 3.7).

Independently to the NILM/ILM distinction, two modes of usage may be applied for a recognition system during its modeling phase [Hart, 1992]:

1. A **Manual Setup (MS)** mode, which requires each appliance to be manually turned on and off by the user in order to understand its specific contribution in the electric signal. Those user actions permit to label *on-site* the consumption signatures of the various appliances, improving the model quality. However, such a system may have difficulty to deal with signals coming from unknown (*unseen*) appliances. We talk of *MS-NILM* and *MS-ILM* systems.

2. An **Automatic Setup (AS)** mode, in which exclusively *a priori* information about appliances is exploited, without touching any running appliance to make the system learn its specific contribution. Being *a priori* trained, such a system may be better at dealing with signals coming from *unseen* appliances. Generally speaking, its performance will depend on the signal variability inside each class of appliances as well as between the classes. The so-called *intra-class* variability is due to the differences between models and brands inside a given appliance class. We talk of *AS-NILM* and *AS-ILM* systems.

For both approaches, manual setup provides better performances than automatic setup, although in the case of NILM it contributes to make the system more expensive and intrusive [Najmeddine et al., 2008; Kato et al., 2009; Paradiso et al., 2013; Gisler et al., 2013; Ridi et al., 2013]. Mixed approaches (MS-AS) can also be envisioned where a manual setup procedure will improve previously trained models of an AS system with, for example, incremental learning.

Applications

Until now, appliances consumption signatures have been modeled for several applications:

- **Local electrical consumption understanding**, to provide the residents with an energy consumption feedback for each appliance in the house. Feedbacks are generally given either directly via the plug or through a static or mobile display [Plugwise, 2015]. A further step to this application is even to predict the electricity consumption of the various appliances [Basu et al., 2012].

- **Global electrical consumption understanding**, to better understand the monthly electricity bill and even predict its amount. The common process is to compute the contribution of each appliance to the global electrical consumption by aggregating the data of all sensors [Jiang et al., 2009].

- **Appliance monitoring**, to detect the abnormal or deviating electricity consumption of potential faulty appliances [Marcu and Stancovici, 2011; Serra et al., 2005].

- **Appliance localization**, to identify and localize the different appliances in a building via the consumption signatures provided by the attributed sensors [Cho and Hahn, 2009; Cho et al., 2009].
- **Human activity recognition**, indirectly through appliance monitoring [Lee et al., 2013, 2012; Ridi et al., 2015]. Such systems often use additional heterogenous signals such as presence or temperature sensors. More precisely, we can distinguish:
  - *User activity recognition*, which aims at identifying activities being done by one or several users by recognizing the set of running appliances which are associated to a specific user activity. Determining which users are involved in an activity that was recognized is another task.
  - *Non-essential appliance recognition*, which aims at identifying appliances not involved in current user activities.
  - *User presence detection*, which aims at detecting the presence of one or more persons in a room by detecting some (unexpected) running appliances (alarm system).
  - *User identification*, which aims at recognizing users by their habits, the way they use their appliances.
- **NILM system evaluation**, to evaluate NILM system disaggregation performances. The general process is to compare the output of the disaggregation algorithm of the system at the smart meter level to the ground truth provided by the various ILM sensors at the plug level [Schoofs et al., 2010a,b].
- **NILM system simulation**, to simulate a NILM system by artificially aggregating the consumption signals provided by the ILM sensors, and perform the evaluation of a disaggregation algorithm as presented in the previous point [Prasad and Semwal, 2013].

The main purpose of all these specific applications is certainly the elaboration of applications that will make the people save energy and consequently money. Although this topic is related to smart home environment and domotics, and not directly to ACS classification, we can mention that there are usually three ways for such applications to make people save energy:

- **Energy monitoring and feedback**, by showing detailed energy information to the users, so that they are well-informed of their energy usage, for example the operating states and the cumulative power consumption of each appliances in the household[5].

[5]In a paper [Gisler and Barchi, 2012], we reported on the elaboration of a wireless lamp dedicated to the feedback of energy consumption. Its principle was to provide a simple and intuitive feedback to residents through color variations of the lamp depending on the amount of energy consumed in a house. Our system relied on inexpensive components piloted by a gateway storing and processing the energy data in a web of things (WoT) framework. Different versions of the color feedback algorithm were presented.

- **Energy saving tips**, by providing energy saving advices to the users according to their current situation of use of appliances;
- **Energy control**, by using an automatic control system to manage the appliance running states appropriately.

Some big companies even developed their own services :

- *Google PowerMeter*, which was[6] an energy monitoring tool that gave consumption information to people based on smart meters and other monitoring sensors. Users could visualize their current energy usage and compare it with those in the past.
- *Microsoft Hohm*, which was[7] a web service that roughly predicts the energy distribution in a given house based on its profile (build year, square footage, etc.) and give suitable energy saving tips to its residents.

3.3.2 Data Acquisition and Databases

System Architecture

A classical ILM system architecture is usually composed of four layers (see Figure 3.9):

1. The **sensor layer**, which constitutes the network of sensors of category ILM I, II or III. Sensors can often act as actuators to switch appliances on or off.
2. The **gateway layer**, which contains the gateways conveying the information from the sensors to the server. Gateways have the necessary drivers and protocols to allow the communication between the sensors and the network. Several sensor networks are available, such as PLC, IP, Enocean or EIB/KNX [Lee et al., 2013; Duarte et al., 2012]. This layer is optional when the sensors directly connect to the server, for instance over IP.
3. The **server layer**, which communicates with the gateways and/or the sensors, stores and process the data, and provide the information to the devices in the view layer.
4. The **view layer**, which is composed of the various devices (e.g. computers, tablets, smart phones) allowing the users to see useful information and manage the system.

Examples of such architecture are well described in [Ito et al., 2004; Kato et al., 2009; Marcu and Stancovici, 2011; Huang and Lai, 2013; Lee et al., 2013].

[6]The project has been retired in 2011 because it didn't scale as Google had hoped. Thus, no more reference is available, except a Wikipedia page.

[7]The service was discontinued in 2012 due to a lack of consumer uptake. Hence, no more reference is available, except a Wikipedia page.

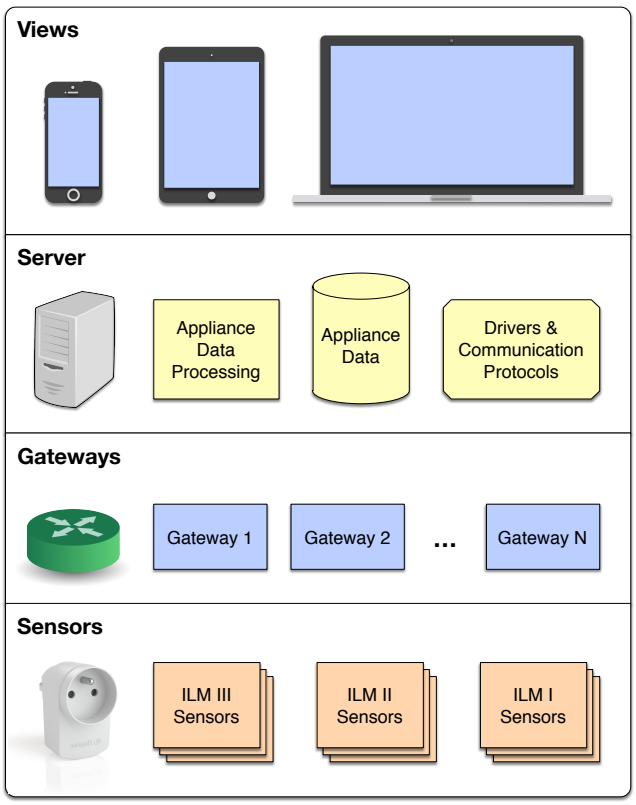


Figure 3.9: Classical ILM system architecture.

Appliance Categories

There exist more than 50 different categories of home appliances. Inside a given category, the existing brands and models may present various features and functionalities. Thus, electric consumption signatures may significantly differ not only between different categories, but also inside the same one[8]. Moreover, appliances can have several running states (modes) depending on their use. This makes the classification task hard for machine learning algorithms which need big ACS databases to train complex and efficient models. Hence, appliances can be grouped in four categories [Hart, 1992; Baranski and Voss, 2003; Zoha et al., 2012]:

1. **Type I – ON/OFF**, containing appliances whose state can only be ON or OFF, like light bulbs or toasters.

2. **Type II – Finite States**, containing appliances with a finite number of modes of energy consumption, for example televisions, microwave oven, fans, dishwashsers or fridges.

3. **Type III – Continuously Variable**, containing appliances whose consumption varies

[8]Reminder: we speak of *inter-class* and *intra-class* differences.

with the time, for example laptops or smart phones, whose batteries can be charging or not.

4. **Type IV – Permanently ON**, containing appliances whose consumption remains constant, like alarms, landline phones or even smart meters.

Table 3.2 shows the variability of the appliance categories used for the task of classification in various research projects. Only categories containing at least five instances were considered. By way of comparison, we added at the end of the table two references (in blue) of our databases ACS-F1 and F2 which are presented in Sections 3.4 and 3.5. Many different approaches were also proposed and the results are not easily comparable.

	[Ito et al., 2004]	[Lam et al., 2007]	[Saitoh et al., 2008]	[Kato et al., 2009]	[Zaidi et al., 2010]	[Reinhardt et al., 2012b]	[Reinhardt et al., 2012a]	[Zufferey et al., 2012]	[Englert et al., 2013]	[Huang and Lai, 2013]	[Paradiso et al., 2013]	[Gisler et al., 2013]	[Ridi et al., 2014b]
Coffee Machines			✓		✓		✓	✓			✓	✓	✓
Computers	✓	✓	✓	✓	✓		✓	✓				✓	✓
Dishwashers			✓		✓		✓				✓		
Fans		✓	✓	✓					✓	✓			✓
Fridges		✓	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓
Hair Dryers	✓	✓	✓	✓									
Irons	✓			✓			✓				✓		
Lamps	✓	✓	✓	✓		✓	✓		✓	✓	✓	✓	✓
Laptops		✓	✓			✓	✓	✓	✓	✓		✓	✓
Microwave Ovens		✓		✓	✓		✓				✓	✓	✓
Monitors		✓	✓			✓	✓		✓				✓
Phone Chargers	✓	✓	✓					✓	✓			✓	✓
Printers		✓	✓		✓		✓					✓	✓
Shavers	✓		✓									✓	✓
Televisions	✓	✓	✓	✓		✓	✓		✓		✓	✓	✓
Vacuum Cleaners	✓	✓	✓	✓			✓						
Wash Machines			✓	✓			✓				✓		
Others	✓	✓	✓	✓		✓	✓			✓	✓		✓

Table 3.2: Appliance categories used for the task of classification in various research projects.

Extracted Features

The main electrical features that are recorded and used for the task of appliance recognition are usually the current, voltage, active power, reactive power, and phase angle (see Sections 3.2.1 and 3.2.2). From them, various features have been extracted and used. Power based features are the most popular, notably because most of commercial sensors can record them. Other features include V-I trajectories [Lam et al., 2007] or current peak values [Ito et al., 2004]. From the current form can be extracted features like the crest factor [Englert et al., 2013], the form factor, their combination [Ito et al., 2004], or the inrush current related to the steady state current [Reinhardt et al., 2012b]. When the sampling frequency of the recorded electric signal is relatively high (between 1 and 100 kHz), frequency features are typically extracted with DFT or Fast Fourier Transform (FFT) [Lam et al., 2007; Kato et al., 2009; Englert et al., 2013]. DFT based features have been reported to be less efficient on low sampled signals [Zaidi et al., 2010], and FFT based features have shown varying performances when coupled with other current based features [Reinhardt et al., 2012b]. Performing FFT on signals sampled at least 20 kHz allows to extract unique harmonic features for each appliance, and even appliances from the same category can be discriminated [Phelan et al., 2012]. However, high sampling acquisition equipment is more expensive [Kang et al., 2013], therefore researchers tend to use cheaper low-consuming sensors with a low signal sampling frequency, generally less than 10 Hz [Paradiso et al., 2013]. Some preprocessing tasks are sometimes performed: normalization and phase shifting [Saitoh et al., 2008], Principal Component Analysis (PCA) to reduce the dimension of the feature space [Kato et al., 2009]. Statistical features are also extracted within fixed time intervals, for example the maximum, the minimum or the mean of the power [Lam et al., 2007; Paradiso et al., 2013], the number of transitions within a given power interval [Paradiso et al., 2013], the number of edges [Zaidi et al., 2010], or the number of times that the signal crosses a given threshold [Lam et al., 2007].

Existing Databases

A growing interest for ILM applications has been shown by the scientific community. Many researchers have been using their own data acquired with commercial sensors or specific hardware [Serra et al., 2005; Tapia et al., 2006; Lifton et al., 2007]. However, the existing public ACS databases dedicated for ILM are less numerous than for NILM. The main are:

- *REDD (Reference Energy Disaggregation Data-set)*, which contains measures of (1) the whole home electricity signal sampled at 15 kHz, (2) up to 24 single circuits labeled each with its appliance category and sampled at 0.5 Hz, and (3) up to 20 plug-level monitors sampled at 1 Hz. The database was dedicated to the analysis of the contribution of single appliances to the aggregated signal [Kolter and Johnson, 2011].
- *AMPds (Almanac of Minutely Power dataset)*, which contains measures recorded by 21

sensors positioned at breaker level on the power panel. During one year, 11 electrical features were recorded every minute [Makonin et al., 2013].

- *BLUED (Building-Level fUlly-labeled dataset for Electricity Disaggregation)*, which contains voltage and current measures of the whole home electricity signal sampled at 12 kHz. A list with the appliance state transition timestamps is given as ground truth [Anderson et al., 2012].
- *GREEND*, which is an appliance energy consumption database of 9 households in Italy and Austria. Various use scenarios are proposed for disaggregation, occupancy detection and appliance usage modeling [Monacchi et al., 2014].
- *UK-DALE (UK Domestic Appliance-Level Electricity)*, which contains measures of (1) the whole house consumption signal sampled at 16 kHz, and (2) individual appliance consumption signals sampled at 1/6 Hz. The data come from 4 houses recorded for several days [Kelly and Knottenbelt, 2014].
- *Smart**, which contains energy measures from 3 houses. These measures were made every second on the whole house at the main panel, on nearly every circuit at plug level, and on electricity generation devices like solar panels and wind turbines. Moreover, data from outdoor weather, indoor temperature and humidity were recorded, as well as binary events from wall switches, the HVAC system, doors and motion sensors. [Barker et al., 2012].
- *iAWE (Indian data for Ambient and Electricity Sensing)*, which contains measures of the electrical consumption of 10 appliances in a smart home environment. The electric signals were sampled at 1 Hz at the meter level, the circuit level and the appliance level, and the following features were recorded: voltage, current, frequency, active and apparent powers, energy and phase angle [Batra et al., 2013].
- *Tracebase*, which contains more than 1000 power consumption signatures recorded from 122 appliances of 31 categories, and acquired over a period of 24 hours at a sampling frequency within 1 and 10 Hz [Reinhardt et al., 2012a]. This database was dedicated to the appliance recognition task.
- *PLAID (Plug-Level Appliance Identification Data-set)*, which contains more than 1000 measures from 235 appliances of 11 categories. Voltage and current signals were sampled at 30 kHz. The database is public and crowd-sourced [Gao et al., 2014].
- Other databases, which are rather dedicated to statistical analysis [De Almeida and Fonseca, 2006; Zimmermann, 2009].

Most of these databases are strongly unbalanced, in terms of number of recorded appliances by category. This pushed us to create and publish a new ACS database of quality, with balanced appliance categories.

3.3.3 Appliance Consumption Signature Classification

Different Machine Learning Tasks and Challenges

At a low level of information analysis, the classification of appliance consumption signatures only relies on raw electrical signals. In this domain, several machine learning tasks must be distinguished:

1. **Appliance type recognition**, which consists in identifying the types/families of appliances, like appliances with batteries or heating/cooling cycles.
2. **Appliance (category) recognition**, which consists in identifying (the categories of) appliances, for example televisions, lamps or coffee machines.
3. **Appliance state recognition**, which consists in identifying the use state of appliances, for example ON, OFF, in standby, idle or in a working cycle.

Scientific challenges in the domain of ACS classification lie principally in realizing a system able to [Ruzzelli et al., 2010]:

- Discriminate appliances with similar electrical consumption signatures;
- Discriminate appliances with multiple running states and settings like standby mode or cycles;
- Deal with long appliance cycles, which may imply long profiling periods.
- Disaggregate parallel appliances by identifying each contribution in the total electric consumption (in ILM I and II);
- Cope with external factors such as unseen appliances;

Related Work

In the ILM domain, most related works are based on supervised machine learning techniques. In [Paradiso et al., 2013], multilayer perceptrons trained with back propagation were used to classify 8 different devices with an accuracy of 95.3% in a MS-ILM system. Low sampling frequency signals were used and power based features such as the number of samples and transitions between power intervals were extracted. In [Reinhardt et al., 2013, 2012a,b], Bagging, Bayesian Networks, J48, JRip, LogitBoost, Naive Bayes, Random Committee, Random Forests and Random Trees were compared on the task of classifying consumption signatures of 33 appliances from the *Tracebase* database in a MS-ILM approach. Current based and harmonics features were extracted from a 1.6 kHz sampled signal and the Random Committee algorithm returned the best accuracy of 95.5%. In [Englert et al., 2013], a hardware and software system handling high frequency (96 kHz) sampled electric consumption signals was

presented. Temporal and frequency features were extracted from the voltage and current data. Various tasks including category recognition, equipment identification and operation mode recognition were performed. Random Forests provided the best accuracy of 99.8% in the task of recognizing 14 appliance categories. In [Zufferey et al., 2012], an AS-ILM system is presented. GMM and k-Nearest Neighbours (k-NN) were compared on the task of classifying 5 appliance categories yielding a best accuracy of 85% with k-NN. The electric consumption signals were sampled at 10^{-1} Hz using the low-end *Plogg* sensor and power based features were extracted. In [Zaidi et al., 2010], DTW and HMM were used to classify 6 categories of appliances whose signatures were sampled at 10^{-1} Hz. Several features were extracted and benchmarked. HMM were reported to perform better than DTW. In [Kato et al., 2009], SVM with Gaussian kernels were used to classify 16 appliances with an accuracy of 99.9% in a MS-ILM case, and 25 appliances with an accuracy of 95.8% in an AS-ILM case. The features were extracted on voltage and current signals a 6 kHz sampling frequency. In [Saitoh et al., 2008], k-NN were used to identify 35 appliances in a MS-ILM approach. The appliance consumption signatures were sampled at 4.4 kHz and current based features were extracted. Best accuracies of 85.5% for category recognition, 80.5% for appliance recognition and 76.3% for state recognition were obtained. In [Fitta, 2010], k-NN were also used to recognize switch-on and off events on appliance consumption signatures acquired at a 6.4 kHz sampling rate. Features such as the active power and current harmonic were extracted. Correct recognition rates of 87.5% for switch-on events and 90.6% for switch-off events were reported. In [Lam et al., 2007], dendrograms were used to cluster 30 appliances into 13 categories showing similar operating characteristics. Shape features of trajectories in the V-I space were extracted. Only few works report having used unsupervised methods in an ILM context. In [Huang and Lai, 2013; Ito et al., 2004], other works and algorithms details are reported on ILM tasks.

Although it is not in the scope of this State of the Art, some works in the NILM domain deserve to be mentioned. In [Chen et al., 2010], Naive Bayes, Bayes Networks, ANN and SVM were used to predict the energy usage based on sensor data collected and generated by the people moving and doing activities in a smart home environment. Several features were extracted: activity label, duration, previous and next activity, number of kind of motion sensors involved, number of triggered motion sensors, states of motion sensors. In [Ruzzelli et al., 2010], a low-cost system able to recognize in real time the most contributing appliances in each electric spikes in the load was reported. Called *RECAP* (RECognition of electrical Appliances and Profiling in real-time), the system used a single wireless (*ZigBee*) energy monitoring sensor attached to the main electrical unit, and an ANN to recognize appliance activities. In [Lee et al., 2012], the system was upgraded to take user behaviours into account and improve the recognition accuracy. Appliances not involved in current user activities could be detected and turned off. Accuracies of 96.4% in recognizing the operating state of appliances and 72.7% in detecting nonessential appliances were reached. In [Lin et al.,

2009], a Bayes filter approach was used to recognize appliances running states based on the total power consumption measured by a smart meter at the distribution board level in a house. In [Lee et al., 2004], the current and voltage signatures of common appliances were observed and analysed in various operating modes. The research focused on methods for measuring and representing appliance characteristics, signal processing techniques and estimation algorithms for signal filtering, signal disaggregation and appliance recognition. In [Marceau and Zmeureanu, 2000], a program for nonintrusive disaggregation of the total electricity consumption of a house into major end uses, such as water heater, baseboard heater and refrigerator, is described. The errors in the energy estimate were less than 10% for most evaluation scenarios. Appliance recognition from combined load was investigated without performance evaluation with HMM in [Kolter and Jaakkola, 2012; Zia et al., 2011; Durand et al., 2004] and ANN in [Prudenzi, 2002]. In [Zeifman, 2012; Liang et al., 2010], other works and algorithms details are also reported on NILM tasks.

Discussion

The growing number of papers published over the last 4 years specially in the ILM approach shows a real interest from the scientific community. Several conclusions can be formulated:

- Machine learning algorithms are indifferently used and are quite influenced by the appliance categories used and the features extracted. A major difficulty for all classification algorithms is the confrontation of its model(s) trained from already seen signals with incoming unseen signals.

- As compared to compared to NILM, ILM is currently lacking of large and free public databases. Furthermore, the work done by the various research teams can only be compared with difficulty as they are all trying to build their own private data sets and test protocols. The release of new public databases will hopefully incite the scientific community to compare their results.

- Classification results reported until now were probably optimistic as existing databases are relatively small in terms of categories and number of instances. New, bigger and more varied databases may affect significantly the classification performances. Moreover, appliances with similar resistive, inductive and capacitive characteristics are more difficult to discriminate. Therefore appliance categories used and listed in the scientific papers should be carefully analyzed to interpret fittingly the reported results.

- Consumption signatures acquired at a higher sampling frequency give generally better classification results. More detailed signals permit to extract frequency features, which are particularly useful for the classification of appliances of the same category. That said, high frequency sensors are more expensive, more energy consuming than low ones, and require more computation power.

- Developing AS systems is more difficult and challenging than MS systems.

3.4 Appliance Consumption Signature Databases

The lack of free public databases and protocols allowing researchers to perform common classification tests and result comparisons, as well as the difficulty to obtain suitable ACS data in terms of quality and quantity pushed us to create a consequent database for our case study. So was born the database called *Appliance Consumption Signatures – Fribourg*. Its first version (ACS-F1) contains 10 appliance categories [Gisler et al., 2013] and its second version 15 appliance categories [Ridi et al., 2014b].

The databases are currently hosted on the website of the *Watt-ICT* project [Watt-ICT, 2013-2016] and are available for free to the scientific community in order to make the research progress. People interested in using the databases must accept the use conditions and fill a licensing form (see Appendix A). At the time this document was written, our databases were already distributed more than 130 times across the world.

In this section, we first present the acquisition protocol and the data recorded as appliance consumption signatures in our ACS-F1 and F2 databases. Then, we give an overview of the overall content of the resulting databases.

3.4.1 Data Acquisition

Acquisition Protocol

As we have seen in the related work (see Section 3.3.3), many acquisition protocols and settings have been used, notably concerning the appliance categories, the signal sampling frequency, or the electric features to record, to name a few.

In order to create a valuable ACS database for appliance consumption signature classification, an acquisition protocol describing all acquisition modalities and parameters had to be defined:

- The **electric signal sampling frequency**: 10^{-1} Hz – As previously stated, in most approaches, the electrical signals generated by the appliances were sampled at a high frequency. In our green approach aiming energy saving applications, we wanted to acquire data at a low sampling frequency by using cheap and low energy consuming sensors. The whole challenge lies in the ability to model and classify appliances from such low sampled signals.
- The **electric signal features to record**: power [W], reactive power [var], current [A], voltage [V], and phase angle [°] – We wanted at least the most standard electrical

features that most low end sensors can record.

- The **acquisition duration**: 1 hour – Given our low sampling frequency, one hour of acquisition seemed to be reasonable to capture the electrical consumption signal of any appliance in all its possible running states (see next paragraph on Acquisition Scenario).
- The **number of acquisition sessions per appliance**: 2 – Two acquisition sessions were done for each recorded appliance in every category. We wanted to have different recording times and uses. Thus, we have 2 × 1 hour of data for each recorded device.
- The **set and number of appliance categories** – The first version of the database was completed with 5 more categories in its second version:
  1. 10 for ACS-F1: coffee machines, computer stations (with monitors), fridges and freezers, Hi-Fi systems (with CD players), lamps (compact fluorescent), laptops (via chargers), microwave ovens, mobile phones (via chargers), printers and televisions (LCD or LED);
  2. 15 for ACS-F2: idem as for ACS-F1 + fans, kettles, lamps (incandescent), monitors and shavers (via chargers).
- The **number of instances per appliance category**: 10 for ACS-F1, 15 for ACS-F2 – All instances of a given category were recorded from exclusively different brands and/or models. Hence, our databases contain no duplicates. The categories present in ACS-F1 were completed with 5 more instances per category in ACS-F2.
- The **number of appliances recorded per acquisition device**: 1 – We wanted to record appliances separately and have disaggregated signals (for an ILM approach).
- The **acquisition places**: home and work (office) buildings. One will roughly find the same categories of appliances in both places.

Acquisition Device

We used inexpensive and low energy consuming sensors of the brand *PLOGG* [PLOGG, 2014]. Such devices (see Figure 3.10) can periodically measure the principal features of the electricity consumption signal at the appliance plug level (see Table 3.3).The maximal signal sampling frequency of our acquisition device is limited to 1Hz. Therefore every recorded feature value is timestamped with a precision of 1 second. The recorded ACS data are wirelessly transmitted to a dedicated computer equipped with a ZigBee[9] transceiver (USB dongle) of the brand *Telegesis* [Telegesis, 2015]. There, the acquisition program formats and stores the

[9]ZigBee is a wireless transmission technology well suited for low data rate (max. 250 kbit/s) applications requiring long battery life, secure networking, and intermittent data transmissions from a sensor or input device.

new incoming consumption signatures in our ACS-F database.

Feature	Unit	Precision
Active power	Watt [W]	0.001W
Reactive power	Volt-ampere reactive [var]	0.001var
Current	Ampere [A]	0.001A
Voltage	Volt [V]	0.001V
Phase angle	Degree [°]	1°
Frequency	Herz [Hz]	0.1Hz

Table 3.3: Electrical features recorded by the *PLOGG* sensors.

Figure 3.10: The *PLOGG* sensor and the *Telegesis* ZigBee transceiver (USB dongle).

Acquisition Scenario per Appliance Category

The acquisition scenario for each appliance category is very important. It specifies the use procedures and operations to do for each appliance category during the duration of the recording session. The main advantage of using scenarios is that they permit to reduce the acquisition duration and ensure that all appliance use states are effectively recorded. The disadvantage is the bias which is introduced given that a recorded signature do not always represent a real use case. Thus, we defined an acquisition scenario for each appliance category, as follows:

- Coffee machines must be put in operating mode, in standby mode and turned off from time to time. The size of the coffee cup should vary.
- Computer stations (with monitors) must be put in operating mode, sleep mode or turned off. The use must vary. Therefore users must change their activities, for example surf on the web, watch movies, perform high and low CPU tasks. The brightness can be varied.
- Fans (with helix) must be put in operating mode, in standby mode and turned off from time to time.
- Fridges and freezers can never be turned off. Their door must sometimes be opened during a variable time. Foodstuff (at ambient temperature) should be removed and added.
- Hi-Fi systems (equipped with CD players) must be turned on, turned off, used and put in standby mode from time to time. Users must listen to music. Music tracks must be changed, and the sound volume varied.
- Kettles must be turned on and off from time to time. The quantity of water, as well as the temperature (if possible) must vary.
- Lamps (compact fluorescent) must be put in operating mode and in standby mode from time to time.
- Lamps (incandescent) must be turned on and off from time to time.
- Laptops must be put in operating mode, in sleep mode and turned off from time to time. As for computer stations, the use must vary. Therefore users must change their activities, for example surf on the web, watch movies, perform high and low CPU tasks. Their battery can be fully charged or charging. The brightness can be changed.
- Microwaves ovens can be either in operating mode, in standby mode or turned off. The cooking time, the power and the eventual modes (normal, convection, grill) should vary.
- Mobile phones (via chargers) must be turned on, turned off, put in standby mode and used form time to time. They must be put in charge (starting with any charge level) and even unplugged from time to time.
- Monitors must be put in operating mode, in standby mode and turned off from time to time. The displayed image must change. The brightness can be varied.
- Printers must be turned on, turned off, put in standby mode, and print some documents from time to time.

- Shavers (via chargers) must be put in operating mode, in standby mode and turned off from time to time.
- Televisions (LCD or LED) must be turned on, turned off and put in standby mode from time to time. Channels must be changed and the sound volume varied.

3.4.2 Database Description

Database Contents

The first version of the ACS-F database (ACS-F1) contains 200 electrical consumption signatures acquired from 100 appliances uniformly distributed in 10 categories. The second version of the database (ACS-F2) contains 450 signatures acquired from 225 appliances uniformly distributed in 15 categories. ACS-F2 can be considered as an extension of the ACS-F1. The contents of the ACS-F1 and F2 databases are listed in Table 3.4.

		ACS-F1	**ACS-F2**
1.	**Coffee machines**	10×2	15×2
2.	**Computer stations (with monitors)**	10×2	15×2
3.	**Fridges and freezers**	10×2	15×2
4.	**Hi-Fi systems (with CD players)**	10×2	15×2
5.	**Lamps (compact fluorescent)**	10×2	15×2
6.	**Laptops (via chargers)**	10×2	15×2
7.	**Microwave ovens**	10×2	15×2
8.	**Mobile phones (via chargers)**	10×2	15×2
9.	**Printers**	10×2	15×2
10.	**Televisions (LCD or LED)**	10×2	15×2
11.	**Fans**	—	15×2
12.	**Kettles**	—	15×2
13.	**Lamps (incandescent)**	—	15×2
14.	**Monitors**	—	15×2
15.	**Shavers (via chargers)**	—	15×2
Total (# classes × # instances × # sessions)		$10 \times 10 \times 2 = 200$	$15 \times 15 \times 2 = 450$

Table 3.4: Contents of the ACS-F1 and F2 databases.

Appliances can often be discriminated on the basis of real and reactive power [Gonçalves et al., 2011]. Figure 3.11 shows the distribution of the instances of the ACS-F2 database relatively to their mean active and reactive powers.

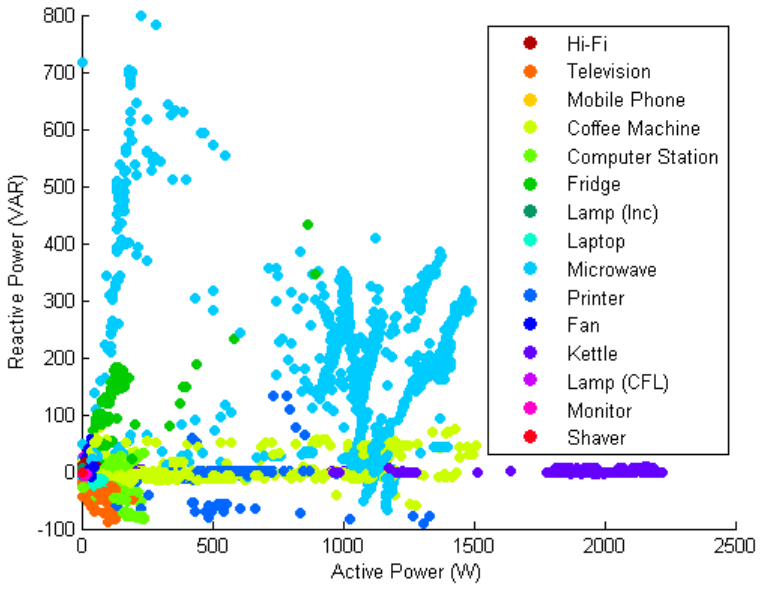


Figure 3.11: Distribution of ACS-F2 instances relatively to their mean active and reactive powers.

Both database instances are organized in category directories and for each instance several files are available:

- An **XML file** containing all recorded data and metadata of the instance (its format is described right after). It is the main file which all others were generated from.
- A **MAT file** containing the raw data only. It was specially created to be directly imported and used with softwares like MATLAB or Octave.
- An **EPS** and an **SVG image file** for each electrical feature. Each image file contains the time series represented on a time-value chart to help visualize and comprehend the corresponding feature. SVG chart examples are given in Figure 3.12.

The generated files were denominated according to the following rule (see example below): (a) the initial letters of the recording person, (b) the appliance category, (c) the appliance brand, (d) the appliance model, (e) the acquisition session number, and optionally (f) the feature name (only present in EPS/SVG image filenames). All parts are separated by an underscore character ('_'). Some parts are missing when the brand or the model was not available for some appliances.

$$\overbrace{\text{cg}}^{\text{Recorder initials}}\_\underbrace{\text{coffeeMachine}}_{\text{Appliance category}}\_\overbrace{\text{Jura}}^{\text{Brand}}\_\underbrace{\text{ImpressaC5}}_{\text{Model}}\_\overbrace{\text{a1}}^{\text{Acquis. session}}\text{.xml}$$

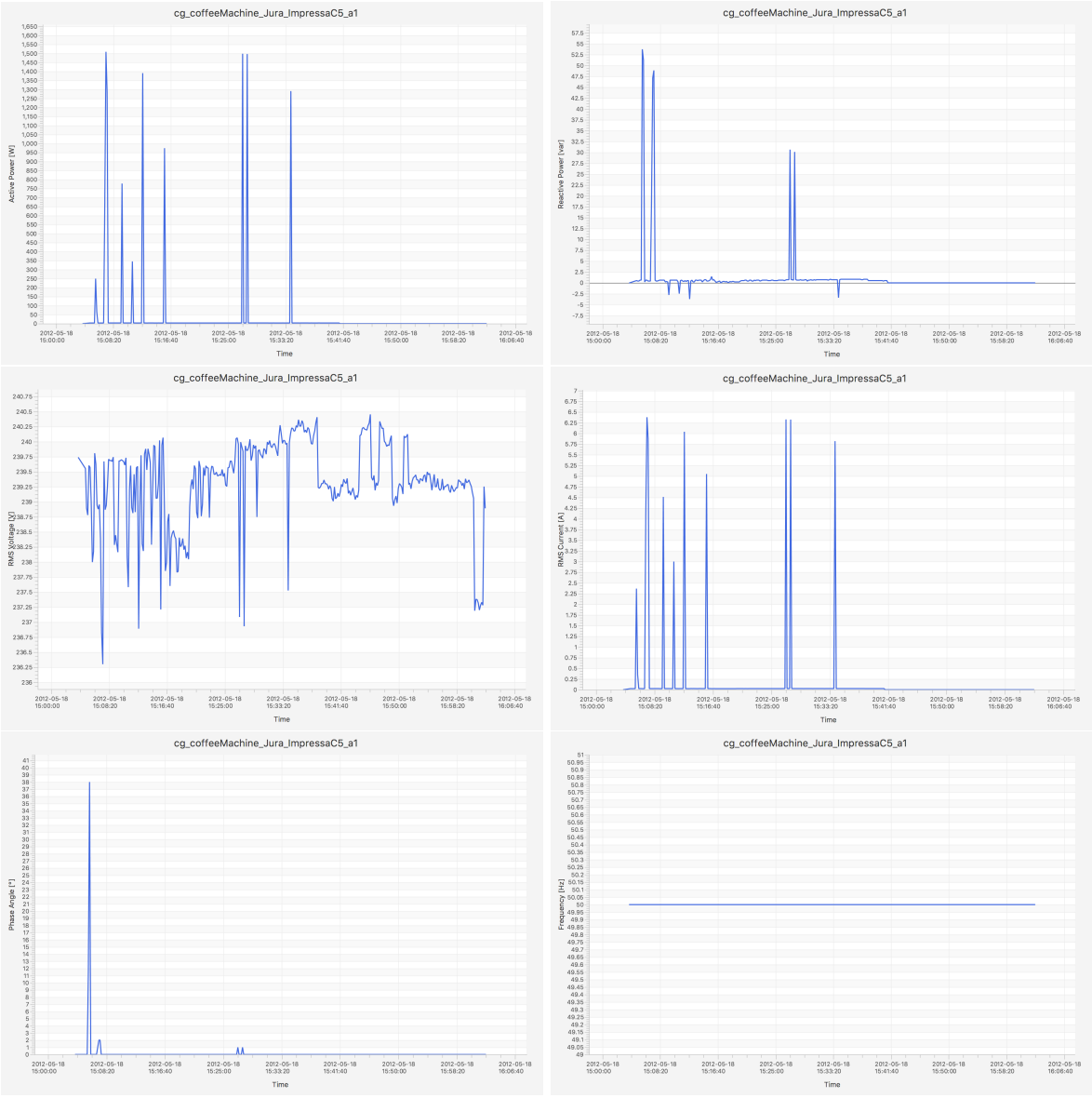


Figure 3.12: Appliance consumption signature of a coffee machine. All electrical features are represented, namely from top left to bottom right: active power, reactive power, RMS voltage, RMS current, phase angle, and frequency.

Data Formats

An XML data structure was designed to store the ACS raw data and the descriptive metadata (including the ground truth) concerning the appliances recorded. The format includes two main parts. First, the header part covers the description of the database (name, version, acquisition session), the place where the acquisition was done, the acquisition device (i.e. the sensor used), the target appliance, and the electrical signal features recorded. Finally, the body part contains the time series values for all electrical features. This generic and self-described format allows to deal with different types of sensors and signals (including future potentially non-electrical ones). As shows the following example, this XML format was designed to be as flexible as possible.

```
<?xml version="1.0" encoding="UTF-8"?>
<signalData>
  <acquisitionContext database="ACS-F" databaseSet="1" session="1" version="1" />
  <acquisitionPlace name="home" type="leavingRoom">
    <feature name="buildingName" value="christophe" />
    <feature name="buildingType" value="residential" />
    <feature name="constructionYear" value="1982" />
    <feature name="rooms" value="6" />
    <feature name="surface" value="?" />
    <feature name="address" value="CH-1772 Grolley" />
  </acquisitionPlace>
  <targetDevice type="Coffee Machine" brand="Jura" model="Impressa C5" energyClass="A" comment="
      My Jura coffee machine" />
  <acquisitionDevice type="electricity_socket" brand="PLOGG" model="PLG-ZGB-CH"
      samplingFrequency="0.1" comment="Sampling frequency is expressed in Hz.">
    <channel name="time" type="date" precision="1" units="s" />
    <channel name="freq" type="float" precision="0.1" units="Hz" />
    <channel name="phAngle" type="integer" precision="1" units="deg" />
    <channel name="reacPower" type="float" precision="0.001" units="var" />
    <channel name="rmsCur" type="float" precision="0.001" units="A" />
    <channel name="rmsVolt" type="float" precision="0.001" units="V" />
    <channel name="power" type="float" precision="0.001" units="W" />
  </acquisitionDevice>
  <signalCurve>
    <signalPoint time="2012-05-18 15:03:55" freq="50.0" phAngle="0" reacPower="0.0" rmsCur="0.0"
        rmsVolt="239.733" power="0.0" />
    <signalPoint time="2012-05-18 15:04:05" freq="50.0" phAngle="0" reacPower="0.517" rmsCur="0.026
        " rmsVolt="239.553" power="3.416" />
    <signalPoint time="2012-05-18 15:04:15" freq="50.0" phAngle="0" reacPower="0.414" rmsCur="0.025
        " rmsVolt="238.894" power="3.312" />
    <!-- ...other signal points... -->
    <signalPoint time="2012-05-18 16:02:25" freq="50.0" phAngle="0" reacPower="0.0" rmsCur="0.0"
        rmsVolt="238.901" power="0.0" />
  </signalCurve>
</signalData>
```

A MAT file containing only the raw values is also available. Its advantage is to be easily usable with softwares like MATLAB or Octave. Here is an example of such a MAT file:

```
# name: phAngle
# type: matrix
# rows: 1
# columns: 347
0 0 0 0 0 0 12 38 0 0 0 0 0 0 1 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 ...
# name: freq
# type: matrix
# rows: 1
# columns: 347
50.0 50.0 50.0 50.0 50.0 50.0 50.0 50.0 50.0 50.0 50.0 50.0 50.0 50.0 50.0 50.0 ...
# name: reacPower
# type: matrix
# rows: 1
# columns: 347
0.0 0.517 0.414 0.414 0.621 0.621 53.727 51.242 0.207 0.621 0.621 0.414 0.414 ...
# name: power
# type: matrix
# rows: 1
# columns: 347
0.0 3.416 3.312 3.416 3.416 3.416 249.691 65.838 3.312 3.312 3.312 3.312 3.312 ...
# name: rmsVolt
# type: matrix
# rows: 1
# columns: 347
239.733 239.553 238.894 238.78 239.603 239.571 238.989 237.997 238.174 239.808 ...
# name: rmsCur
# type: matrix
# rows: 1
# columns: 347
0.0 0.026 0.025 0.025 0.026 0.026 2.366 0.356 0.025 0.026 0.026 0.025 0.025 ...
```

3.4.3 Machine Learning Test Protocols

As said before, ACS-F1 and F2 databases are now available for free to the whole scientific community [Watt-ICT, 2013-2016]. Hence, every interested researcher will be able to perform various machine learning tasks on them. In this section, we present the two test (or evaluation) protocols that will allow to reiterate experiments and make performance comparisons, notably in the task of appliance recognition. As the databases and the protocols are publicly available, we speak of an *in-house evaluation with existing benchmark* [Hannani and Hennebert, 2008]. Table 3.5 shows a summary of two proposed protocols and Figure 3.13 a graphical representation of the splitting between the training and test sets for both protocols.

Test Protocol 1 – Intersession

In this first test protocol called *intersession* protocol, all ACS instances of the first acquisition session are included in the train set and all ACS instances of the second acquisition session in the test set. In other words, cardinalities of both train and test sets are equal, namely 100 in the case of ACS-F1 and 225 in the case of ACS-F2. For each given instance, the whole duration of the signal, namely 1 hour, can be used. Classification results must be presented in the form of a confusion matrix and the overall accuracy rate. The classification task linked to this first protocol is relatively simple because the appliances to recognize during the test phase have been already "seen" during the training phase. Thus, this protocol corresponds to a MS-ILM recognition system.

Test Protocol 2 – Unseen Instances

In the second test protocol called *unseen instances* protocol, all ACS instances of both sessions are taken to perform a k-fold cross-validation, where $k = 10$ in the case of ACS-F1 and $k = 15$ in the case of ACS-F2. Thus, with these values of k, each fold contains exactly one instance of each session 1 and 2 for each appliance category (see Figure 3.13). As explained in Section 2.3.4, the cross-validation process consists in taking successively each of the k-folds as test set, and the remaining $k-1$ folds as training set. As in the first test protocol, the whole duration of the signal, namely 1 hour, can be used. Classification results must be presented in the form of a confusion matrix and the overall recognition rate averaged over the k-folds (in order to give a single estimation). The classification task linked to this second protocol is much harder because the appliances to recognize during the test phase have been "unseen" during the training phase. Thus, this protocol corresponds to a AS-ILM recognition system.

	Protocol 1 – Intersession	**Protocol 2 – Unseen Instances**
ML task	Appliance recognition,...	Appliance recognition,...
Train set	Instances of session 1	In turns, all instances of the $k-1$ folds
Test set	Instances of session 2	All instances of the k^{th} remaining fold*
Train duration	$\leq$ 1 hour (whole instance)	$\leq$ 1 hour (whole instance)
Test duration	$\leq$ 1 hour (whole instance)	$\leq$ 1 hour (whole instance)
Result form	Confusion matrix, accuracy	Confusion matrix, accuracy

* $k = 10$ for ACS-F1 and $k = 15$ for ACS-F2

Table 3.5: Machine learning test protocols: (1) *intersession* and (2) *unseen instances*.

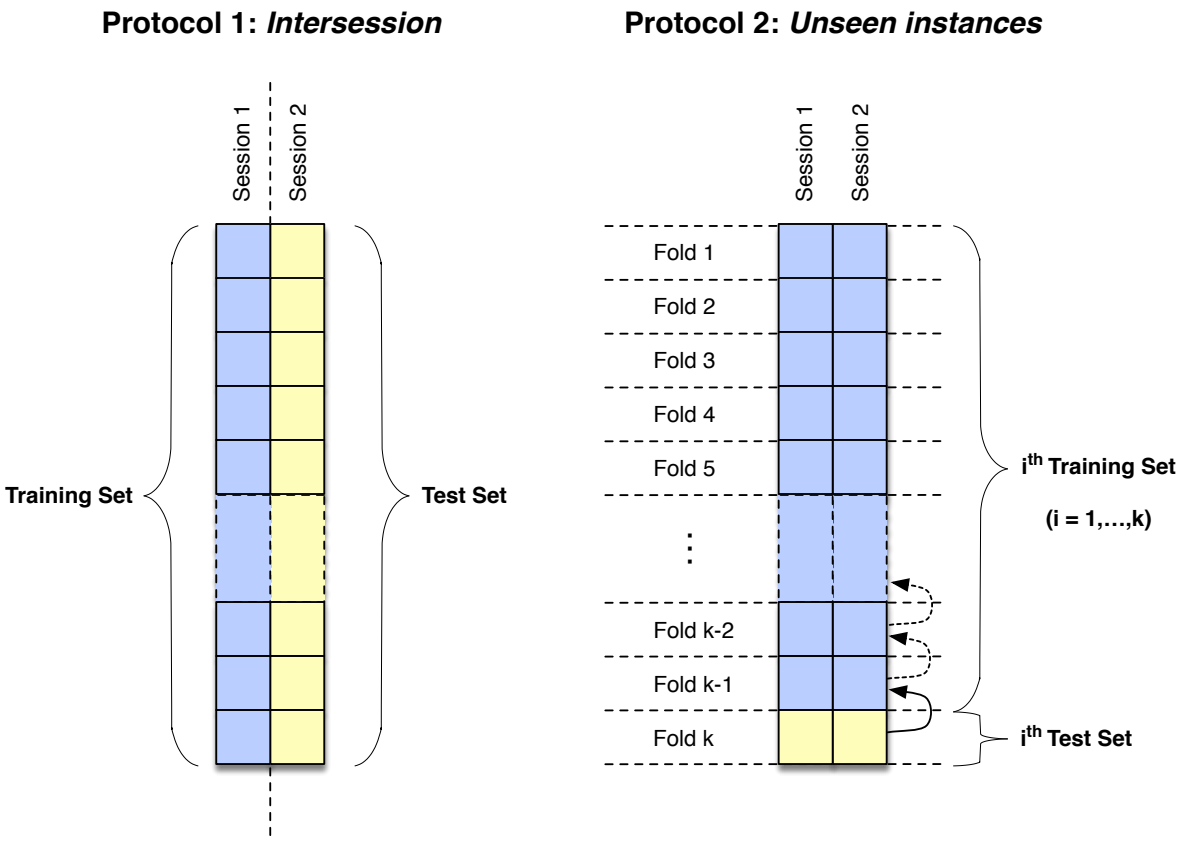


Figure 3.13: Train and test set splitting (in blue and yellow) for both machine learning test protocols: (1) *intersession* and (2) *unseen instances*.

3.5 Appliance Consumption Signature Classification

In the literature, the term *classification* is often mixed up with *recognition*, *identification*, *detection* or *discrimination*. In fact, one must simply remember not to mix the final objective of the task with its object. In this case study, our final objective was to *recognise/identify/detect* appliance categories, and in order to perform this task, we *classified/discriminated* ACS recorded using *PLOGG* devices.

In this section, we present the experiments and results obtained in the task of classifying the ACS of the ACS-F1 and F2 databases, by adopting two generic approaches to time series data, namely a global (or holistic) and a local approach. Various features were tested and tuned, including statistical features, or other SAX based features. Three classification algorithms were tested, namely SVM, MLP and GMM.

3.5.1 Data Preparation

Following the standard data mining process (see Section 2.3.3), a data preparation was made.

Data Selection

Data attributes (or features) should be relevant and not redundant for a machine learning task, such as classification. A feature is considered as relevant if it is correlated with the classes and redundant if it is correlated with the other features [Yu and Liu, 2003]. Thus, not all electrical features recorded by the PLOGG sensors were relevant for our classification tasks. *Frequency* and *voltage* are imposed by the electrical network and remains practically stable, whatever is the activity of the running appliances. Therefore these two basic features could be left out. Concerning redundancy, physics and its theory of electricity tell us that active power, reactive power and phase angle are redundant, as the one can be computed from the two others (see Section 3.2.2 and Figure 3.3). Current, voltage and active power are also redundant, as shows Equation 3.2. This first phase of data selection has the advantage to restrain the input data which all others features will be extracted from. Then during the modeling phase (see Section 2.3.4), all extracted features were selected for example by performing an exhaustive grid search (or other more targeted methods when all possibilities could not be tested for reasons of computation time).

Data Cleaning

In this case study, we had the chance to have led the data acquisition phase carefully by means of an acquisition protocol and a dedicated acquisition software. Thus, the data contained no missing values or noisy artefacts which needed to be cleaned. We only had to unify or correct some class labels that were differently entered or misspelled by the person who performed the recording, for example “TV” or “Television”, “HiFi” or “HI-FI”.

Data Construction

From an *operability* point of view, we had to take the sine of the phase angle converted in radiants. Because of the cyclic nature of the phase, some appliances may have successive samples that shift from 0 to 360 degrees and vice versa with time. Computing the sine of the phase angle in radiants permits to avoid this shift in the phase, and the bijection problem is excluded by the positivity of the cosine of the phase angle in radiants.

From an *impartiality* point of view, all extracted features were normalized according to the *Z-score standardization* formula (see Equation 2.2 in Section 2.3.3) ensuring most of the feature values to be centered around 0 and within $[-1, 1]$. Typically, an MLP using a logistic activation function whose image is $[-1, 1]$ needs the features to be normalized that way.

From an *efficiency* point of view, we extracted several new features, which are described in the following section, and added some running state annotations in the metadata of the ACS recordings.

3.5.2 Feature Extraction

As presented in Chapter 2, we aimed here to extract and use features which are independent of the nature of the data (i.e. the ACS), making this time series classification approach generic and data-driven. All features were extracted on the four dimensions previously mentioned, namely the *active power*, the *reactive power*, the *RMS current* and the *phase angle* of the ACS multivariate time series. Although the phase angle has been qualified as redundant and could be left aside, we noticed during our experiments that it could be helpful in some cases, and thus we kept it.

Global and Local Features

Global features are extracted by applying one or several (mathematical) operations on a whole time series, i.e an ACS in our case. Hence, global operations take all the values of the time series as input to return one (or several) *global* value(s) as output (see left part of Figure 3.14).

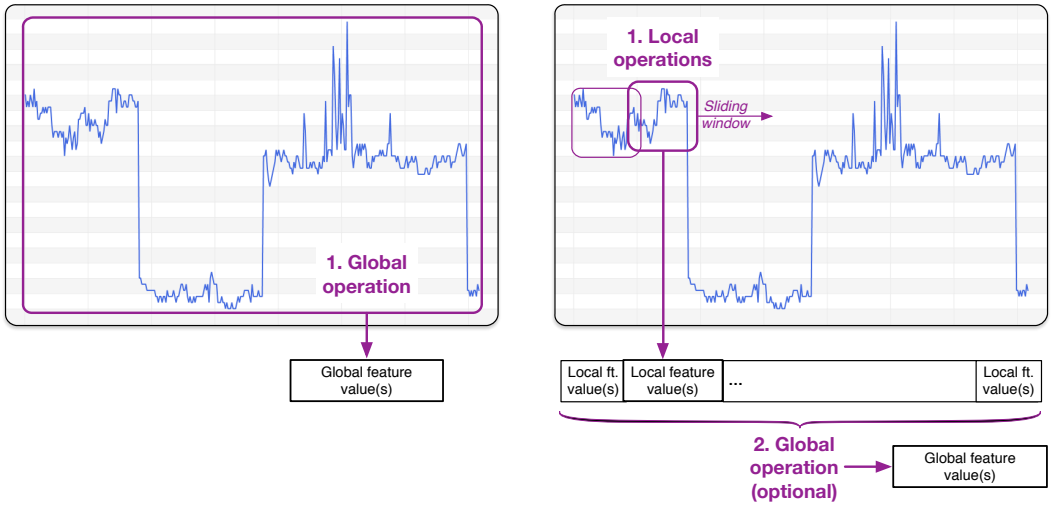


Figure 3.14: Extraction of global (left) and local(-global) (right) features on time series.

Local features are extracted by applying one or several (mathematical) operations on subparts of a time series, by using a *sliding window* (whose size and overlap will be tuned). Hence, local operations take separately the values of the different subparts of the given time series as input in order to return an array of local values as output. From there, global features can also be obtained by applying global operations directly on the obtained local features (see right part of Figure 3.14).

As we shall see in the next section, global and local features extracted from multivariate time series have to be handled in a different way in order to be processed by the classifiers.

Statistical Features

Several statistical operations were computed on the values of each dimension of the ACS (multivariate) time series, namely the minimum, the maximum, the amplitude, the median, the mean, the standard deviation, the skewness and the kurtosis.

Skewness is measure of asymmetry of a value set. A value set (or distribution) $x_1, \ldots, x_n$ is symmetric if it looks the same to the left and to the right of its center point. Skewness formula is given by $\frac{\sum_{i=1}^{n}(x_i-\mu)^3/n}{\sigma^3}$, where μ is the mean and σ the standard deviation of the values. Kurtosis is a measure of whether the data are heavy- or light-tailed relatively to a normal distribution. A data set with a high kurtosis tends to have heavy tails, i.e. outliers, while a data set with a low kurtosis tends to have light tails, i.e. lack of outliers. Kurtosis formula is given by $\frac{\sum_{i=1}^{n}(x_i-\mu)^4/n}{\sigma^4}$. More details are given in [NIST/SEMATECH, 2013, Section 1.3.5.11].

From all these operations, three types of features were extracted:

1. *Global statistical features*, computed on the entire time series, for example the global amplitude of the signal. Such features are *global features*.
2. *Local statistical features*, computed on parts of the time series which has been divided, for example local amplitudes (see Figure 3.15). Such features are *local features*.
3. *Local-global statistical features*, computed on the values of the preceding local statistical features, for example a histogram of the local amplitudes (see Figure 3.15). Such features are *global features* computed from *local features*.

Dynamic Coefficients

Coming from the field of speech recognition [Hennebert, 1998], dynamic coefficients of 1st and 2nd order, namely *delta* and *delta-delta* coefficients, were extracted. They aim to express the local dynamics of the signal in a close neighbourhood of each point x, and thus are *local features*. They can be seen as a kind of derivative of the signal, expressing the *velocity* for the first order, and the *acceleration* for the second one. Such features are likely to characterize variations within time series.

Let T be a (univariate) time series of length $l > 0$ and dimensionality d (as described in Definition 2.1). For each point $y_i = (y_{i,1} \ \ldots \ y_{i,d})$ of T, where $i = 1, \ldots, l$, *delta coefficients* are defined as follows:

$$\Delta y_i = \sum_{j=-k}^{k} j \cdot y_{i-j} \tag{3.3}$$

where k is commonly set to 2. At the extremities of the time series, the y_i values are simply repeated when the i values are out of the bounds. *Delta-delta coefficients* are defined similarly

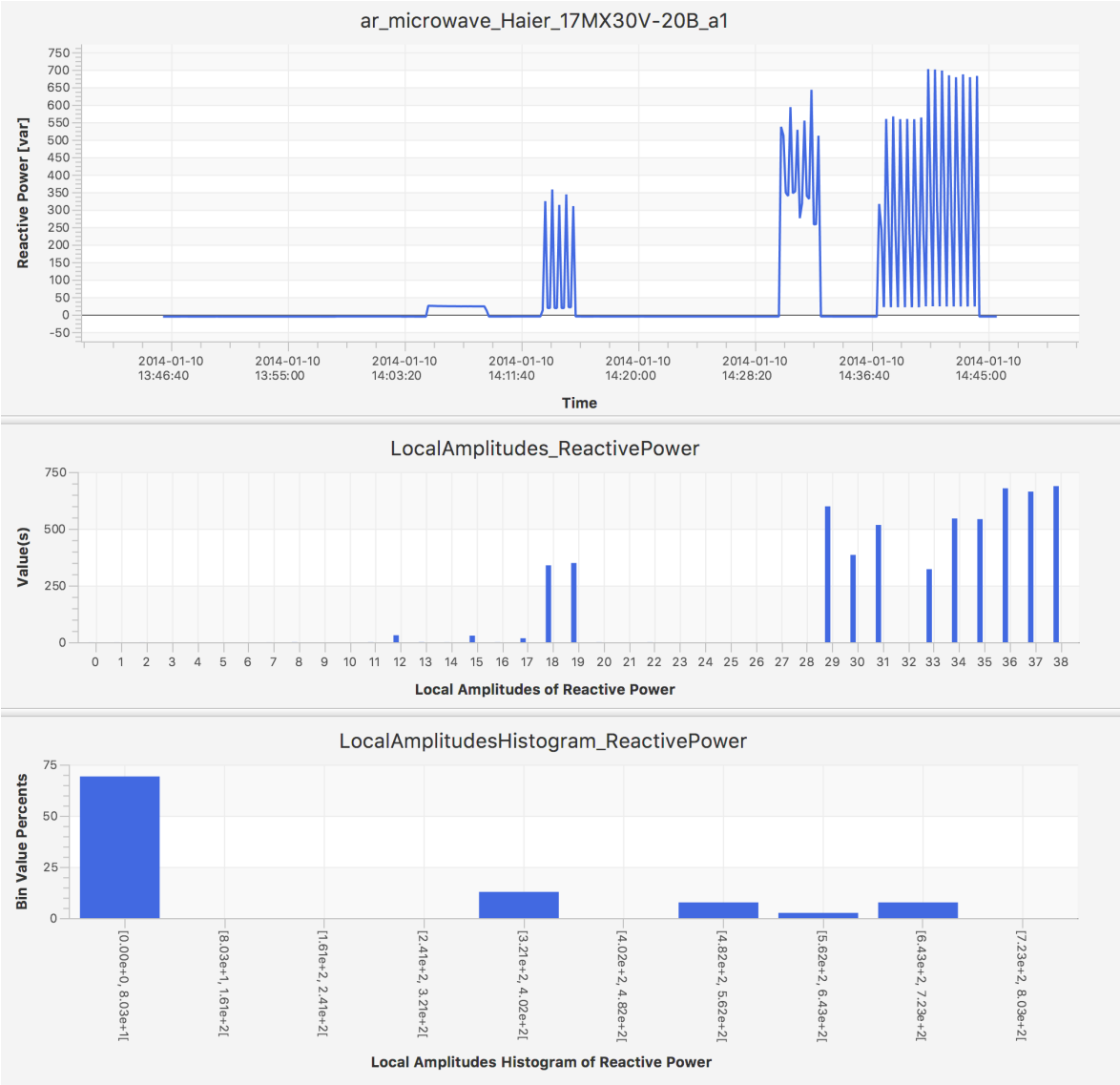


Figure 3.15: ACS of a microwave oven (reactive power) with the extracted local amplitudes and local amplitude histogram (from top to bottom).

or recursively from the delta coefficients as follows:

$$\Delta\Delta y_i = \Delta y_{i+1} - \Delta y_{i-1} \tag{3.4}$$

The half window size k was set to the common value 2, but other values may be tested in future work. Figure 3.16 shows the curves of the dynamic coefficients extracted from the active power curve of an ACS.

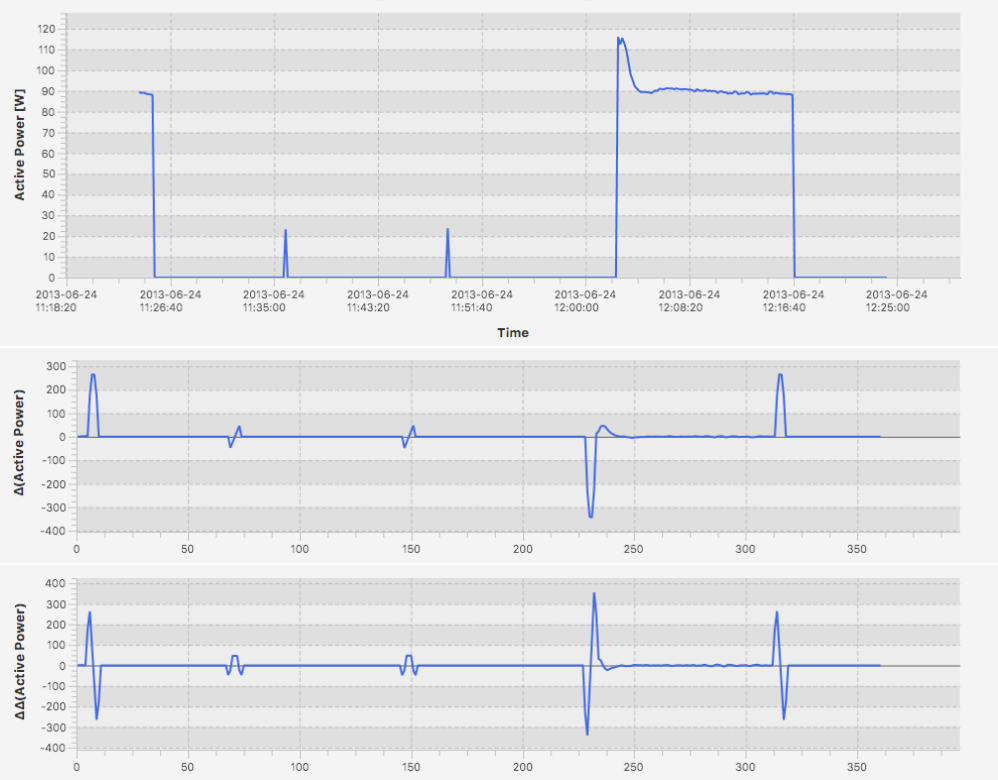


Figure 3.16: ACS of a fridge (active power) with the corresponding delta coefficients of 1st and 2nd order (from top to bottom).

PAA and SAX Features

PAA segment values were extracted and taken as local features (see Section 2.2.4). These values are obtained by dividing a time series int k segments of equal length and taking their means. The PAA representation preserves the information about the absolute position.

SAX letter values were also extracted as local features. The letter values are obtained from the PAA segment values of a time series that has been *normalized* (see procedure in Section 2.2.4). The normalization permits to compare the shape of time series independently of their scale and absolute position (offset).

SAX word frequencies are global features computed from the preceding SAX letter representation. Inspired by the so-called Bag-of-Words (BoW) process, such features are obtained as follows. First, the SAX letter representation of a time series is decomposed into "words" of varying lengths by means of a sliding window of varying length and overlap. Then, the word counts (*frequencies*) are computed over the whole time series set. Finally, all words whose frequency is below a given threshold T are removed. This threshold ensures that each word is represented at least a certain number of times in the set and has a chance to be significant. Experiments have shown that, even with small thresholds, most of the words are discarded. A similar method was reported in [Senin and Malinchik, 2013].

Figure 3.17 shows an example of these three features computed from a given time series. On the SAX word frequencies chart, words have been ordered from the most to the least present word in the time series set.

Frequency Domain Features

In the frequency domain, we extracted both DFT- and DWT-based features (by means of the FFT and FWT algorithms). As explained in Section 2.2.4, the first is rather efficient when the time series contains global periodicities, while the second is more efficient when the time series contain variations located in specific regions. We decided to extract both types of features because, even if the ACS recordings have a limited duration of 1 hour, depending on the appliance category, the signal may contain both periodical or local variations. For instance, the cooling events of a fridge are cyclic, while the door opening events (with the light turning on) of the same fridge are local. All features coming from the frequency domain are by definition not spacial and can only be global features.

As FFT based features, we extracted the frequency magnitudes and filter banks with varying bank numbers and overlap ratios (see two middle charts in Figure 3.18). Independent classification tests allowed us to determine the optimal number of banks (10) and the overlap ratio between the banks (0.1).

As FWT based features, we extracted the Haar wavelets coefficients (see bottom chart in Figure 3.18).

3.5.3 Manual and Automatic Annotations

Time series often need to be annotated. By *annotating* the time series, we mean that some of their parts (sets of multivariate points) are given one or several labels. The ACS of our databases ACS-F1 and F2 have been manually annotated with three possible working states (*labels*), namely: *on*, *off*, and *fridge door open* (only for the fridge category). From the annotated parts, all features presented in the preceding section can be extracted, resulting either

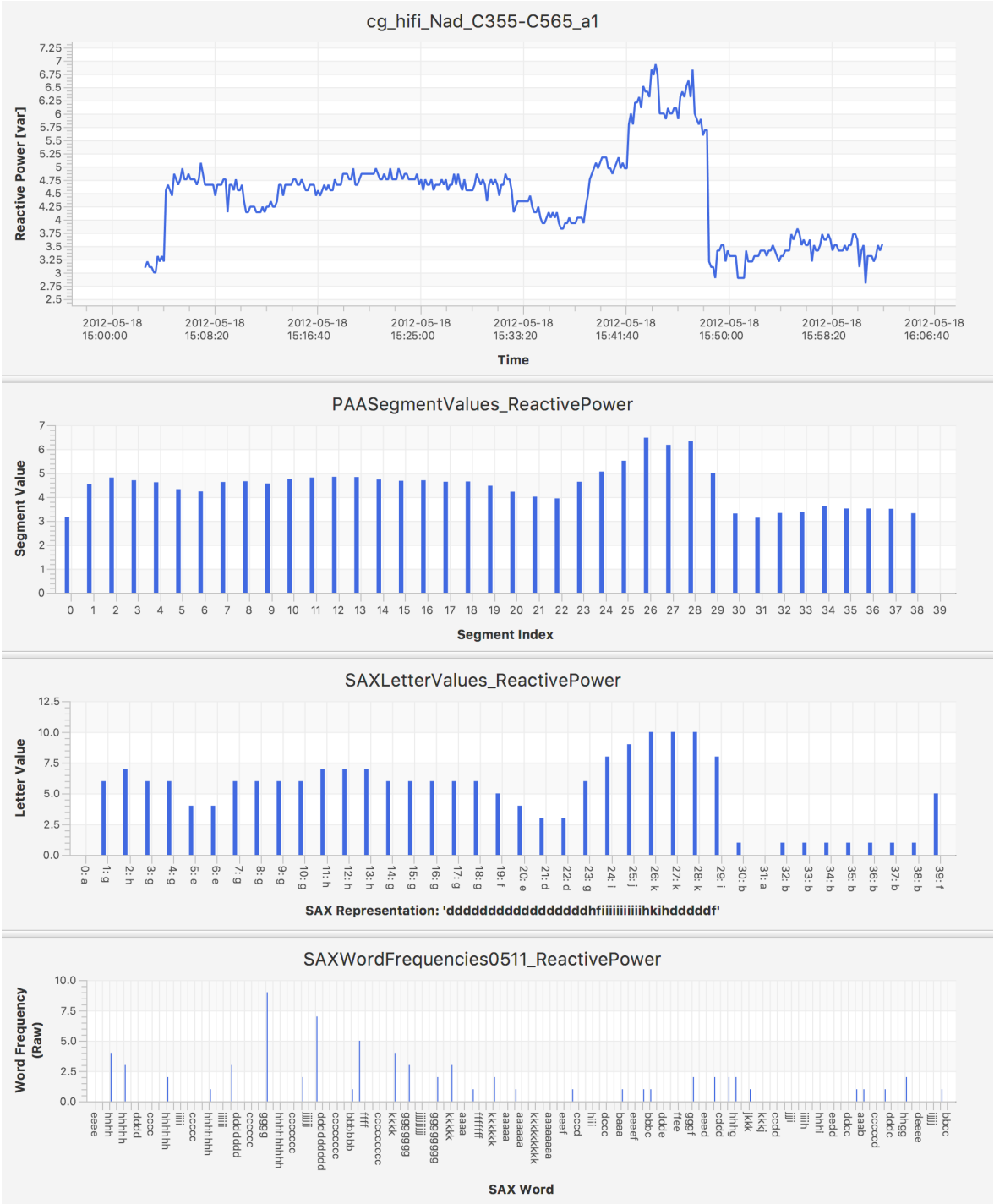


Figure 3.17: Representations of the PAA segment values, the SAX letter values and SAX word frequencies features extracted from a given Hi-Fi ACS (reactive power).

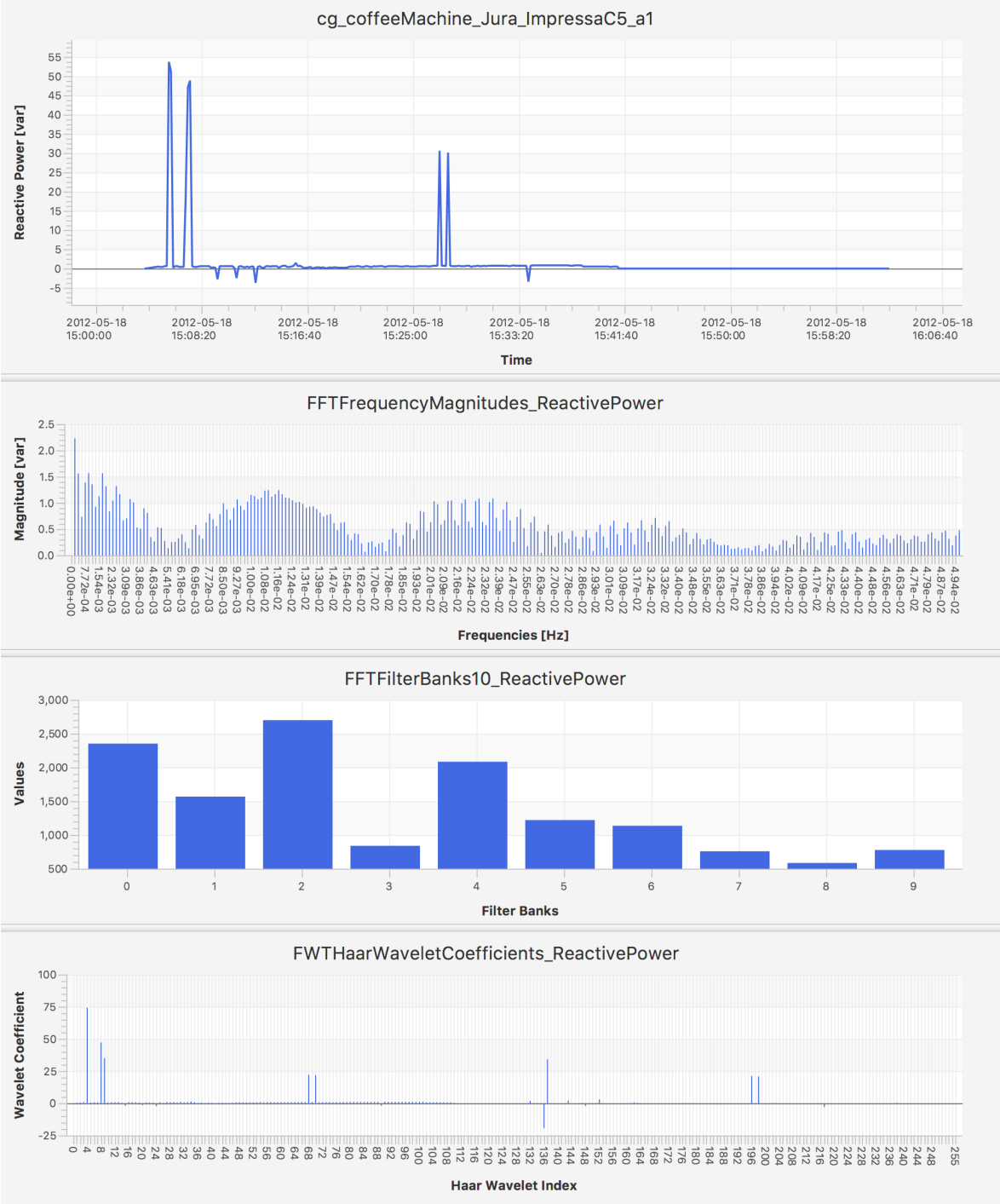


Figure 3.18: Representations of frequency domain features (from top to bottom: frequency magnitudes, filter banks 10 and Haar wavelet coefficients) of an ACS of a coffee machine (reactive power).

in global or local features. However, as the annotated parts of a given time series may be not adjacent, it makes no sense to extract features from the frequency domain or others like the dynamic coefficients. Statistical features remain appropriate.

The ACS having been annotated with their running states, various features could be extracted. We extracted a feature that we called *Working State Percents* representing the relative *time* percentage that each appliance was in each working state (see Figure 3.19), and another feature consisting of various statistical measures such as the minimum, the maximum, the amplitude, the median, the mean, or the standard deviation.

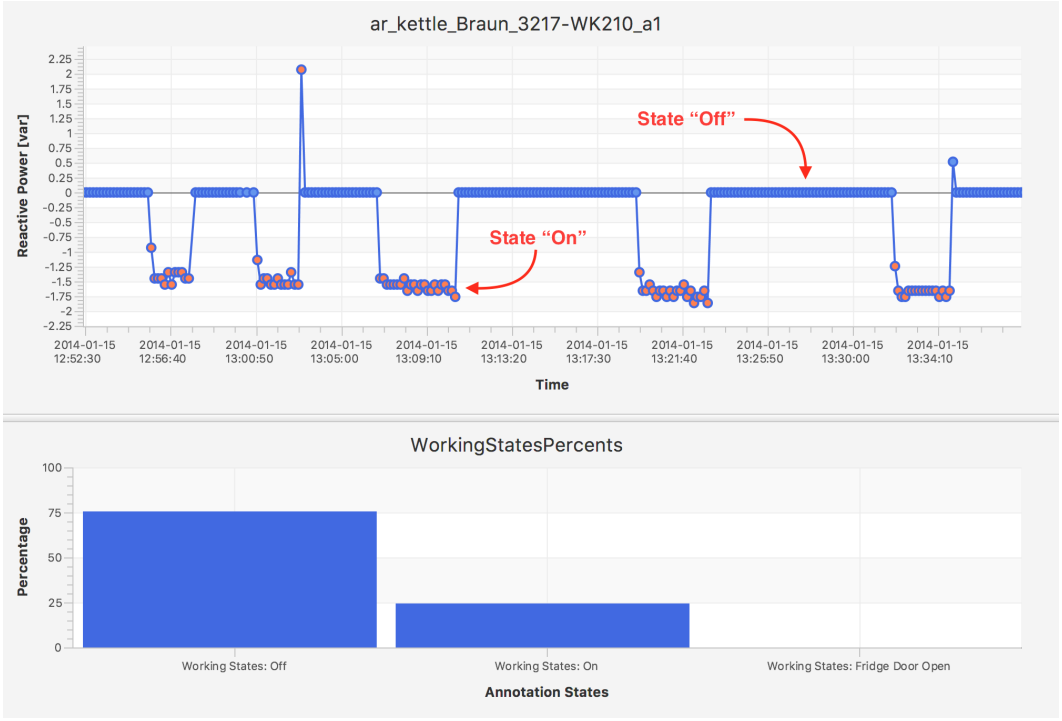


Figure 3.19: Annotation feature representing the working state percents of a kettle (blue points are in *off* state and orange ones in *on* state).

Annotating time series can be a very hard and time consuming task, mostly if the time series database is big. Moreover, manual annotation (done by humans) is subject to errors and to different interpretations by people. For instance, two different persons may not perform the same annotations, because it is quite difficult to tell if a given point of a (multivariate) time series is in such or such a state, just by looking at the different dimensions of the time series. Furthermore, humans are only able to see obvious states, like *on*, *off*, or *fridge door open*, but some unexpected states may arise from the intrinsic functioning of appliances. For these reasons, we decided to perform automatic annotation with the computer.

We may reasonably define a state as an accumulation of (multivariate) points somewhere in their relative dimensions *independently* of the time, because states may occur at any moment (see Figure 3.20). Even if this is not true in some cases and applications, this is true most of the time for ACS. For example in our case study, most appliances can be switched on or off, or have a failure at any time. The solution that we adopted to automatically detect these states is to cluster the time series points. We used the k-Means++ algorithm, which is a k-Means (see Section 2.4.6) improved with a more efficient method to initialize the cluster centroids. In order to be comparable amongst all time series, the obtained clusters are sorted according to their centroid position, from the lowest to the highest. A difficulty with this algorithm is to *a priori* determine the number of clusters. However this can be done empirically for example by assessing the classification performances from features that have been extracted from different cluster configurations. In the case of electrical appliances, we could easily tell that the number of working states should not be high (< 10), notably because of their total duration, which has to be significant in order to be detected as a state, and interesting from an application point of view.

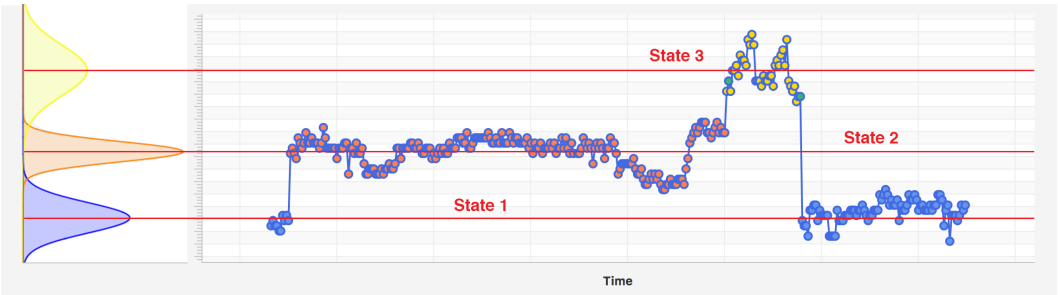


Figure 3.20: A state in a multivariate time series is defined by an accumulation of points at certain levels in its relative dimensions *independently* of the time.

In our case study, we extracted the same statistical features (minimum, maximum, amplitude, median, mean, and standard deviation) from different clusterizations, that is with a varying number of clusters (see Figure 3.21), and the best classification results were obtained with a number of clusters equal to 4 (followed by 3 and 2). Therefore we kept the statistical features extracted from these 4 clusters in our set of relevant features for the experiments presented in the next section.

3.5.4 Experiments and Results

Two types of experiments were realized following two different approaches to time series classification, namely a global and a local approach.

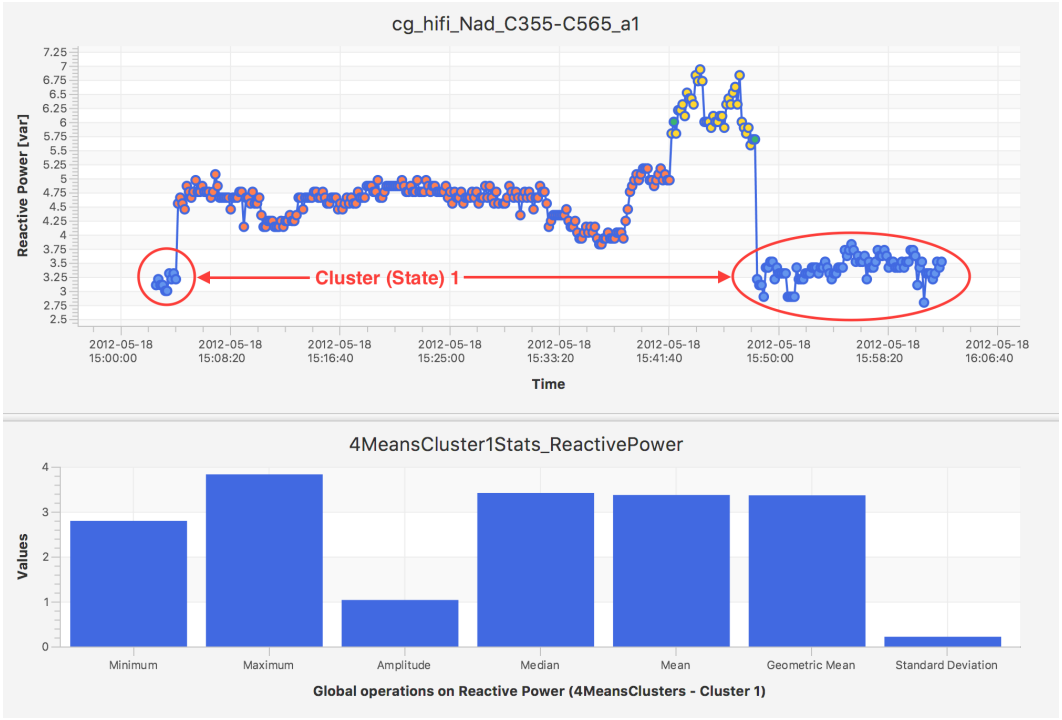


Figure 3.21: Statistical features (minimum, maximum, amplitude, median, mean, and standard deviation) extracted from the reactive power values of the first (lowest) cluster in the ACS time series.

Global Approach

With this global approach (also called *holistic*), we built *global* feature vectors, meaning that one unique feature vector was extracted from the entire multivariate ACS time series. Such global feature vectors are built in a *horizontal* (or *longitudinal*) way, by extracting feature components one after the other on each dimension (i.e. univariate series) of interest, and then simply concatenating them (see Figure 3.22).

Depending on the length and the dimension of the time series, the potential global features to find out, extract and test can be very plentiful and varied. Hence, with this global approach, the various classifiers had to deal with a small quantity (one per multivariate time series) of feature vectors of relative big dimension (between 40 and 500 components).

Four types of features were used for this approach: (1) *global statistical features*, including the minimum, the maximum, the amplitude, the median, the mean, and the standard deviation (SD) (the classification performance was optimal by taking all values together), (2) the *local amplitudes median*, (3) 10 *filter banks* computed on the frequency magnitudes obtained with the FFT, and (4) local statistics including the minimum, the maximum, the amplitude, the median, the mean, the geometric mean, and the SD, computed on each clus-

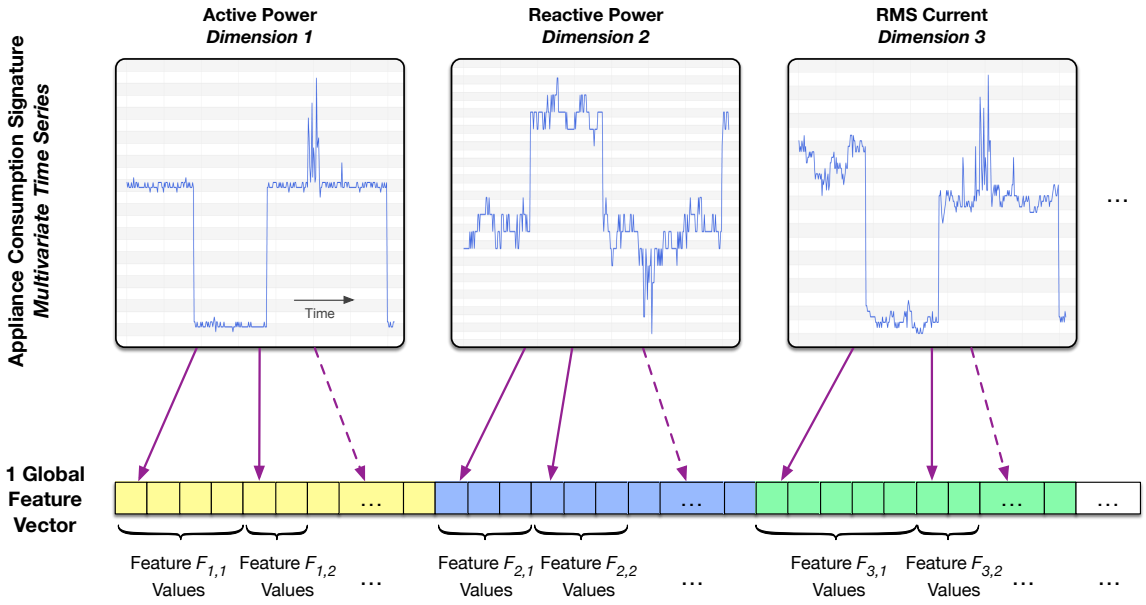


Figure 3.22: Global approach for ACS classification: one global feature vector extracted per multivariate time series. For example, feature $F_{1,1}$ values can be global statistics extracted from active power series, and $F_{2,2}$ values filter banks extracted from reactive power series.

ter obtained with the k-Means++ clustering method. After individual classification tests with this latter feature only, the optimal value 4 for k was found (the next best value being 3), which bears out our idea that there are not plenty of running states for each appliance.

Three classification algorithms (*classifiers*) were implemented and tested for the task of appliance category recognition, namely Support Vector Machine (SVM), Multilayer Perceptron (MLP), and Gaussian Mixture Model (GMM) (see Section 2.4). SVM and MLP are both discriminative modeling algorithms, and GMM is a generative one (see Section 2.4.3). The nature of the classification and the relative small quantity of data available led us to use SVM at first, because they are a recent and efficient algorithm, and then MLP and GMM to compare the performance with another discriminative algorithm and a generative one.

The results obtained for the task of appliance recognition (i.e. ACS classification) with the three above-mentioned classifiers on both ACS-F1 and F2 databases are presented in Table 3.6 and the corresponding confusion matrices in Tables 3.7 and 3.8. The best classifier parameters and feature combinations were obtained using a grid search on targeted subsets of values and feature sets[10]. Each classifier has its own parameters: cost C and RBF kernel pa-

[10]The number of possibilities to test is too large and would require a more powerful computing resource. Nevertheless, in the case the obtained results are not the optimal, they constitute at least a lower bound and give a pessimistic idea of the real values.

rameter γ for the SVM, number of gaussians m and variance floor value ν for the GMM, and ratio between the number of hidden neurons and input neurons[11] h/i and initial learning rate η for the MLP. Results are given with their confidence interval ($CI_{95\%}$) (see Equation 2.55).

Protocol	**Database**	**Classifier**	**Best Classifier Parameters**	**Global Statistics** Min, max, amp, median, mean, SD	**Local Amplitudes Median**	**FFT: Filter Banks** Bank number: 10, bank overlap ratio: 0.1	**4-Means: 1st Cluster Stat.** Min, max, amp, median, mean, geo.mean, SD	**4-Means: 3rd Cluster Stat.** Min, max, amp, median, mean, geo.mean, SD	**4-Means: 4th Cluster Stat.** Min, max, amp, median, mean, geo.mean, SD	**Feature Vector Dimension**	**Classifier Performance:** Accuracy [%] with $CI_{95\%}$
1. Intersession	ACS-F1	SVM	$C = 1e5$, $\gamma = 1e{-}3$	arc	—	arc	arc	—	arcp	97	91.0 ± 5.6
		GMM	$m = 2$, $\nu = 1e{-}5$	arc	arcp	—	—	—	—	22	91.0 ± 5.6
		MLP	$h/i = 2.0$, $\eta = 0.1$	arc	arc	—	—	—	—	21	89.0 ± 6.1
	ACS-F2	SVM	$C = 1e6$, $\gamma = 1e{-}4$	arc	arcp	arc	—	—	arc	73	90.7 ± 3.8
		GMM	$m = 3$, $\nu = 1e{-}4$	arc	—	—	—	—	arc	39	90.7 ± 3.8
		MLP	$h/i = 1.5$, $\eta = 0.1$	arc	—	—	—	—	arc	39	85.8 ± 4.6
2. Unseen Instances	ACS-F1	SVM	$C = 1e4$, $\gamma = 1e{-}2$	arc	—	—	—	arc	—	39	81.5 ± 5.4
		GMM	$m = 3$, $\nu = 1e{-}4$	arc	—	arc	—	arc	—	69	81.0 ± 5.4
		MLP	$h/i = 1.5$, $\eta = 0.1$	arc	arc	—	—	arc	—	42	77.5 ± 5.8
	ACS-F2	SVM	$C = 1e4$, $\gamma = 1e{-}2$	arc	arc	arc	—	—	arc	73	80.4 ± 3.7
		GMM	$m = 6$, $\nu = 1e{-}2$	arc	arc	arc	—	arc	arc	93	76.7 ± 3.9
		MLP	$h/i = 1.5$, $\eta = 0.1$	arc	arc	—	—	arc	arc	63	75.8 ± 4.0

N.B.: arcp ⇒ feature extracted on (a)ctive power, (r)eactive power, (c)urrent and (p)hase angle

Table 3.6: Global approach: appliance recognition (ACS classification) performances.

With both protocols P1 and P2, namely *Intersession* and *Unseen Instances*, and both ACS-F1 and F2 databases, the global statistics features containing the minimum, maximum, amplitude, median, mean and standard deviation of each ACS were always included in the best feature set after each feature selection. This shows that, at the end, the whole thing is a story of statistics. Furthermore, with the ACS-F2 database, which has more instances and classes, global statistics features extracted from the 4^{th} cluster appeared to be always pertinent. This is not surprising as this upper cluster tends to model the highest running states.

[11]The number of input neurons is equal to the feature vector dimension also given in the result tables.

	Coffee machine	Computer station	Fridge / Freezer	Hi-Fi system	Lamp CFL	Laptop	Microwave oven	Mobile phone	Printer	TV
Coffee machine	100	0	0	0	0	0	0	0	0	0
Computer station	0	90	0	0	0	10	0	0	0	0
Fridge / Freezer	0	0	80	20	0	0	0	0	0	0
Hi-Fi system	0	0	0	100	0	0	0	0	0	0
Lamp CFL	0	0	0	0	100	0	0	0	0	0
Laptop	0	0	0	0	0	90	0	0	0	10
Microwave oven	10	0	0	0	0	0	90	0	0	0
Mobile phone	0	0	0	0	0	0	0	100	0	0
Printer	20	10	0	0	0	0	0	0	**70**	0
TV	0	0	0	0	0	10	0	0	0	90

(a) P1 – ACS-F1 – SVM – Accuracy: $91.0 \pm 5.6\,\%$

	Coffee machine	Computer station	Fan	Fridge / Freezer	Hi-Fi system	Kettle	Lamp CFL	Lamp inc.	Laptop	Microwave oven	Mobile phone	Monitor	Printer	Shaver	TV
Coffee machine	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Computer station	0	87	0	0	0	0	0	0	7	0	0	0	0	0	7
Fan	0	0	93	0	0	0	0	7	0	0	0	0	0	0	0
Fridge / Freezer	0	0	7	93	0	0	0	0	0	0	0	0	0	0	0
Hi-Fi system	0	0	0	0	93	0	0	0	0	0	0	7	0	0	0
Kettle	0	0	0	0	0	**100**	0	0	0	0	0	0	0	0	0
Lamp CFL	0	0	7	0	0	0	93	0	0	0	0	0	0	0	0
Lamp inc.	0	0	0	0	7	0	0	93	0	0	0	0	0	0	0
Laptop	0	0	0	0	0	0	0	0	93	0	0	0	0	0	7
Microwave oven	7	0	0	0	0	0	0	0	0	93	0	0	0	0	0
Mobile phone	0	0	0	0	0	0	0	0	0	0	87	0	0	13	0
Monitor	0	0	0	0	0	0	0	0	0	0	0	93	7	0	0
Printer	13	0	0	0	0	0	0	0	0	7	0	0	80	0	0
Shaver	0	0	0	0	7	0	7	0	0	0	0	0	0	87	0
TV	0	13	0	0	0	0	0	0	0	0	0	13	0	0	**73**

(b) P1 – ACS-F2 – SVM – Accuracy: $90.7 \pm 3.8\,\%$

	Coffee machine	Computer station	Fridge / Freezer	Hi-Fi system	Lamp CFL	Laptop	Microwave oven	Mobile phone	Printer	TV
Coffee machine	90	0	0	0	0	0	10	0	0	0
Computer station	10	80	0	0	0	0	0	0	0	10
Fridge / Freezer	0	0	100	0	0	0	0	0	0	0
Hi-Fi system	0	0	0	90	0	0	0	0	0	10
Lamp CFL	0	0	0	0	90	10	0	0	0	0
Laptop	0	0	0	0	0	100	0	0	0	0
Microwave oven	0	0	0	0	0	0	100	0	0	0
Mobile phone	0	0	0	10	0	0	0	90	0	0
Printer	0	0	0	10	0	0	10	0	**80**	0
TV	0	0	0	0	0	0	0	0	10	90

(c) P1 – ACS-F1 – GMM – Accuracy: $91.0 \pm 5.6\,\%$

	Coffee machine	Computer station	Fan	Fridge / Freezer	Hi-Fi system	Kettle	Lamp CFL	Lamp inc.	Laptop	Microwave oven	Mobile phone	Monitor	Printer	Shaver	TV
Coffee machine	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Computer station	0	80	0	0	0	0	0	0	0	0	0	0	0	0	20
Fan	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0
Fridge / Freezer	0	0	7	73	0	0	0	0	0	20	0	0	0	0	0
Hi-Fi system	0	0	0	0	80	0	0	0	0	0	7	0	0	13	0
Kettle	0	0	0	0	0	**100**	0	0	0	0	0	0	0	0	0
Lamp CFL	0	0	0	0	7	0	93	0	0	0	0	0	0	0	0
Lamp inc.	0	0	0	0	7	0	0	93	0	0	0	0	0	0	0
Laptop	0	0	0	0	0	0	0	0	93	0	0	7	0	0	0
Microwave oven	0	0	0	0	0	0	0	0	0	100	0	0	0	0	0
Mobile phone	0	0	0	0	0	0	0	0	0	0	93	0	0	7	0
Monitor	0	0	0	0	0	0	0	0	0	0	0	93	0	0	7
Printer	7	0	0	0	0	0	0	0	0	0	0	0	87	7	0
Shaver	0	0	0	0	0	0	0	0	0	0	7	0	0	93	0
TV	0	0	0	0	0	0	0	0	7	0	0	13	0	0	**80**

(d) P1 – ACS-F2 – GMM – Accuracy: $90.7 \pm 3.8\,\%$

	Coffee machine	Computer station	Fridge / Freezer	Hi-Fi system	Lamp CFL	Laptop	Microwave oven	Mobile phone	Printer	TV
Coffee machine	90	0	0	0	0	0	10	0	0	0
Computer station	0	90	0	0	0	0	0	0	0	10
Fridge / Freezer	0	0	90	0	0	0	10	0	0	0
Hi-Fi system	0	0	0	90	0	0	0	0	10	0
Lamp CFL	0	0	0	0	90	10	0	0	0	0
Laptop	0	0	0	0	0	90	0	0	0	10
Microwave oven	0	0	0	0	0	0	100	0	0	0
Mobile phone	0	0	0	0	0	0	0	100	0	0
Printer	0	10	0	0	10	0	10	0	**70**	0
TV	0	0	0	0	0	10	0	0	10	80

(e) P1 – ACS-F1 – MLP – Accuracy: $89.0 \pm 6.1\,\%$

	Coffee machine	Computer station	Fan	Fridge / Freezer	Hi-Fi system	Kettle	Lamp CFL	Lamp inc.	Laptop	Microwave oven	Mobile phone	Monitor	Printer	Shaver	TV
Coffee machine	93	0	0	7	0	0	0	0	0	0	0	0	0	0	0
Computer station	0	73	0	0	0	0	0	0	7	0	0	13	0	0	7
Fan	0	0	80	0	0	0	13	7	0	0	0	0	0	0	0
Fridge / Freezer	0	7	13	73	0	0	0	7	0	0	0	0	0	0	0
Hi-Fi system	0	0	0	0	87	0	0	7	0	0	0	0	7	0	0
Kettle	0	0	0	0	0	**100**	0	0	0	0	0	0	0	0	0
Lamp CFL	0	0	7	0	7	0	87	0	0	0	0	0	0	0	0
Lamp inc.	0	0	0	0	13	0	0	87	0	0	0	0	0	0	0
Laptop	0	7	0	0	0	0	0	0	93	0	0	0	0	0	0
Microwave oven	0	0	0	0	0	0	0	0	7	93	0	0	0	0	0
Mobile phone	0	0	0	0	0	0	0	0	0	0	87	0	0	13	0
Monitor	0	0	0	0	0	0	0	0	0	0	0	100	0	0	0
Printer	7	0	0	0	0	0	0	0	7	0	7	0	73	7	0
Shaver	0	0	0	0	0	0	7	0	0	0	0	0	0	93	0
TV	0	0	0	0	0	0	0	0	7	0	0	13	13	0	**67**

(f) P1 – ACS-F2 – MLP – Accuracy: $85.8 \pm 4.6\,\%$

Table 3.7: Global approach: best classification confusion matrices (values are in %) obtained with protocol P1 *Intersession* on both databases ACS-F1 and F2.

	Coffee machine	Computer station	Fridge / Freezer	Hi-Fi system	Lamp CFL	Laptop	Microwave oven	Mobile phone	Printer	TV
Coffee machine	90	0	0	0	0	0	10	0	0	0
Computer station	0	80	0	0	0	15	0	0	0	5
Fridge / Freezer	0	0	100	0	0	0	0	0	0	0
Hi-Fi system	0	0	0	85	0	5	0	0	10	0
Lamp CFL	0	20	0	0	**70**	0	0	5	0	5
Laptop	0	0	0	10	0	80	0	0	0	10
Microwave oven	15	0	0	0	0	0	85	0	0	0
Mobile phone	0	0	0	0	0	0	0	**100**	0	0
Printer	20	0	0	10	0	0	5	0	60	5
TV	0	15	0	0	5	5	0	0	10	65

(a) P2 – ACS-F1 – SVM – Accuracy: $81.5 \pm 5.4\,\%$

	Coffee machine	Computer station	Fan	Fridge / Freezer	Hi-Fi system	Kettle	Lamp CFL	Lamp inc.	Laptop	Microwave oven	Mobile phone	Monitor	Printer	Shaver	TV
Coffee machine	93	0	0	0	0	0	0	0	0	7	0	0	0	0	0
Computer station	0	**70**	0	3	0	0	0	0	10	0	0	0	0	0	17
Fan	0	0	80	0	0	0	3	3	0	0	0	3	0	0	10
Fridge / Freezer	0	0	7	90	0	0	0	0	0	0	0	0	0	0	3
Hi-Fi system	0	0	0	0	73	0	0	3	0	0	7	7	0	7	3
Kettle	0	0	0	0	0	100	0	0	0	0	0	0	0	0	0
Lamp CFL	0	0	3	0	7	0	83	0	0	0	0	0	3	3	0
Lamp inc.	0	0	3	0	10	0	0	83	0	0	0	0	0	3	0
Laptop	0	7	3	0	0	0	0	0	77	0	0	3	3	0	7
Microwave oven	7	0	0	0	0	0	0	0	0	93	0	0	0	0	0
Mobile phone	0	0	0	0	3	0	0	0	0	0	83	0	0	13	0
Monitor	0	0	0	0	0	0	3	0	0	0	0	80	3	0	13
Printer	17	0	0	0	3	0	0	0	0	10	10	0	**53**	7	0
Shaver	0	0	0	0	7	0	3	0	0	0	7	0	0	83	0
TV	0	20	0	0	0	0	0	0	0	0	0	17	0	0	**63**

(b) P2 – ACS-F2 – SVM – Accuracy: $80.4 \pm 3.7\,\%$

	Coffee machine	Computer station	Fridge / Freezer	Hi-Fi system	Lamp CFL	Laptop	Microwave oven	Mobile phone	Printer	TV
Coffee machine	90	0	0	0	0	0	10	0	0	0
Computer station	0	85	0	0	5	0	0	0	0	10
Fridge / Freezer	0	0	50	5	10	0	5	0	15	15
Hi-Fi system	0	0	0	70	5	15	0	0	10	0
Lamp CFL	0	10	10	0	**65**	10	0	5	0	0
Laptop	0	0	0	0	0	85	0	0	10	5
Microwave oven	0	0	0	0	0	0	100	0	0	0
Mobile phone	0	0	0	0	0	0	0	**100**	0	0
Printer	0	0	0	10	0	0	5	0	85	0
TV	0	5	0	0	5	5	0	0	5	80

(c) P2 – ACS-F1 – GMM – Accuracy: $81.0 \pm 5.4\,\%$

	Coffee machine	Computer station	Fan	Fridge / Freezer	Hi-Fi system	Kettle	Lamp CFL	Lamp inc.	Laptop	Microwave oven	Mobile phone	Monitor	Printer	Shaver	TV
Coffee machine	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Computer station	0	**67**	0	0	0	0	0	0	10	0	0	0	0	0	23
Fan	0	0	70	13	0	0	7	7	3	0	0	0	0	0	0
Fridge / Freezer	0	0	3	83	3	0	0	0	0	3	0	0	0	0	7
Hi-Fi system	0	0	0	0	73	0	0	0	3	0	7	3	3	7	3
Kettle	0	0	0	0	0	93	0	0	0	7	0	0	0	0	0
Lamp CFL	0	0	3	0	7	0	80	0	0	0	0	3	0	7	0
Lamp inc.	0	0	3	0	10	0	0	83	0	0	0	0	0	3	0
Laptop	0	0	0	0	0	0	0	0	93	0	0	7	0	0	0
Microwave oven	10	0	0	0	0	0	0	0	0	90	0	0	0	0	0
Mobile phone	0	0	0	0	3	0	0	0	0	0	77	0	0	20	0
Monitor	0	0	0	0	7	0	0	3	7	0	0	63	0	0	20
Printer	10	0	0	0	0	0	7	0	0	3	3	0	**60**	17	0
Shaver	0	0	0	0	13	0	0	0	0	0	23	0	3	60	0
TV	0	17	0	0	3	0	0	0	10	0	0	10	3	0	**57**

(d) P2 – ACS-F2 – GMM – Accuracy: $76.7 \pm 3.9\,\%$

	Coffee machine	Computer station	Fridge / Freezer	Hi-Fi system	Lamp CFL	Laptop	Microwave oven	Mobile phone	Printer	TV
Coffee machine	80	10	0	0	0	0	0	0	10	0
Computer station	0	70	0	5	0	5	0	0	0	20
Fridge / Freezer	0	0	85	0	15	0	0	0	0	0
Hi-Fi system	0	0	0	85	10	0	0	5	0	0
Lamp CFL	0	10	5	5	**65**	0	0	10	0	5
Laptop	0	0	0	0	5	80	0	0	10	5
Microwave oven	10	10	0	0	0	0	80	0	0	0
Mobile phone	0	0	0	0	0	0	0	**100**	0	0
Printer	20	0	5	0	0	0	5	10	50	10
TV	0	0	0	0	0	15	0	0	5	80

(e) P2 – ACS-F1 – MLP – Accuracy: $77.5 \pm 5.8\,\%$

	Coffee machine	Computer station	Fan	Fridge / Freezer	Hi-Fi system	Kettle	Lamp CFL	Lamp inc.	Laptop	Microwave oven	Mobile phone	Monitor	Printer	Shaver	TV
Coffee machine	77	0	0	0	0	3	0	0	0	0	0	0	13	7	0
Computer station	0	**60**	0	3	7	0	0	0	20	0	0	0	0	0	10
Fan	0	0	70	7	0	0	10	10	0	0	0	3	0	0	0
Fridge / Freezer	0	0	7	90	0	0	0	3	0	0	0	0	0	0	0
Hi-Fi system	0	0	0	0	77	0	7	0	0	0	7	0	0	10	0
Kettle	0	0	0	0	0	100	0	0	0	0	0	0	0	0	0
Lamp CFL	0	0	3	0	0	0	87	7	0	0	0	0	0	3	0
Lamp inc.	0	0	3	0	17	0	0	73	0	0	3	0	0	3	0
Laptop	0	3	0	0	3	0	0	0	87	0	0	7	0	0	0
Microwave oven	3	10	10	0	0	0	0	0	0	73	0	0	3	0	0
Mobile phone	0	0	0	0	3	0	0	0	0	0	90	0	0	7	0
Monitor	0	0	3	0	0	0	0	0	0	0	0	87	3	0	7
Printer	13	3	0	0	0	0	3	0	0	3	13	3	**57**	3	0
Shaver	0	0	0	0	13	0	3	0	0	0	27	0	3	53	0
TV	0	7	0	0	0	0	0	0	0	7	0	27	3	0	**57**

(f) P2 – ACS-F2 – MLP – Accuracy: $75.8 \pm 4.0\,\%$

Table 3.8: Global approach: best classification confusion matrices (values are in %) obtained with protocol P2 *Unseen Instances* on both databases ACS-F1 and F2.

The lower the cluster, the lower consuming running states it will model and the more similar these running states will be amongst the appliance categories. Hence, the lowest cluster will model *idle* states which are often nearly null and quite the same for all recent appliance with common energy saving abilities. With test protocol P2, which is more complex than P1, global statistics features extracted from the 3rd cluster appeared to be helpful as well. For the hardest classification task, with test protocol P2 applied on the ACS-F2 database, more features were required and pertinent; 10 filter banks and local amplitude medians came completing the preceding ones.

The three classification algorithms tested provided similar performance scores, with a little advantage to SVM which seemed to be sensibly better and mostly faster in terms of training and tuning time. GMM performed nearly the same as SVM and needed less features than SVM with test protocol P1. However and surprisingly, the contrary was observed with test protocol P2, where SVM required less features than GMM. MLP were the worst classifiers, because they were the slowest ones in terms of training and tuning time. As the performance of their generated model depends a lot on the random initialization of their weights, several trainings were sometimes needed until a suitable model was obtained. Thus, MLP are particularly not adapted to test protocol P2 involving a k-fold CV with k training and test phases.

Table 3.9 shows the true positive rate (TPR) which was at least obtained for each appliance category, by each classifier with each protocol and database. With protocol P1, *kettles* of ACS-F2 database systematically obtained a TPR of 100%, while *printers* of ACS-F1 database and *TVs* of ACS-F2 database obtained a bad TPR. With protocol P2, *mobile phones* of ACS-F1 database systematically obtained a TPR of 100%, while *lamps CFL* of ACS-F1 database as well as *computer stations*, *printers* and *TVs* of ACS-F2 database obtained a bad TPR. These category names and TPR are shown in bold font in the confusion matrices.

Finally, we wanted to mention that multiple tests were done with SAX word frequencies as features. Although various size of alphabets and length of the SAX representation were tested, the results obtained were not convincing, probably because of the large number of SAX words (several hundreds, even after filtering) which yield big and sparse feature vectors. Such vectors probably lead to the well known *curse of dimensionality* problem. Even if we tried to perform some PCA to avoid that problem, the maximum accuracy obtained was about 54% with test protocol P1 on ACS-F2. Given the number of appliance categories, this result is not bad, but far from those obtained with the other features. Nevertheless, SAX words have a discriminating potential. Other tests were also performed with the raw frequency magnitudes obtained with FFT, and the Haar wavelet coefficients obtained with Fast Wavelet Transform (FWT), but were even less convincing, probably for the same reasons.

Protocol	Database	True Positive Rate (TPR)	Classifier	Coffee machine	Computer station	Fridge / freezer	Hi-Fi system	Lamp CFL	Laptop	Microwave oven	Mobile phone	Printer	TV	Fan	Kettle	Lamp inc.	Monitor	Shaver
1. Intersession	ACS-F1	> 90%	SVM	✓	✓		✓	✓	✓	✓	✓		✓	–	–	–	–	–
			GMM	✓		✓	✓	✓	✓	✓	✓		✓	–	–	–	–	–
			MLP	✓	✓	✓	✓	✓	✓	✓	✓			–	–	–	–	–
	ACS-F2	> 87%	SVM	✓	✓	✓	✓	✓	✓	✓	✓			✓	✓	✓	✓	
			GMM	✓				✓	✓	✓	✓	✓		✓	✓	✓	✓	✓
			MLP	✓			✓	✓	✓	✓	✓				✓	✓	✓	✓
2. Unseen Instances	ACS-F1	> 80%	SVM	✓	✓	✓	✓		✓	✓	✓			–	–	–	–	–
			GMM	✓	✓				✓	✓	✓	✓	✓	–	–	–	–	–
			MLP	✓		✓	✓		✓		✓		✓	–	–	–	–	–
	ACS-F2	> 73%	SVM	✓		✓	✓	✓	✓	✓	✓			✓	✓	✓	✓	✓
			GMM	✓		✓	✓	✓	✓	✓	✓				✓	✓		
			MLP	✓		✓	✓	✓	✓	✓	✓				✓	✓	✓	

N.B.: A blue ✓ ⇒ a TPR of 100%

Table 3.9: Global approach: minimal true positive rate (TPR) obtained for each appliance category by each classifier for each protocol and database.

Local Approach

With this local approach to time series classification, we built *local* feature vectors, in the sense that several feature vectors were extracted from different parts of each multivariate ACS time series. Such local feature vectors are built in a *vertical* (or *transversal*) way, by extracting one feature component after the other on each dimension (i.e. each univariate sub-series) of interest, and then simply concatenated them (see Figure 3.23).

Considering the relatively small length of the numerous parts of the time series, and the fact that each feature must have the same dimension, the quantity of potential local features to find out, extract and test, is smaller than for global vectors. Therefore with this local approach, the classifiers that we tested had to deal with a much larger quantity a feature vectors of relative small dimension (between 4 and 12), for each ACS time series. As with the global approach, the same three SVM, MLP, and GMM classifiers were tested for the task of appliance category recognition (for the sake of approach comparison).

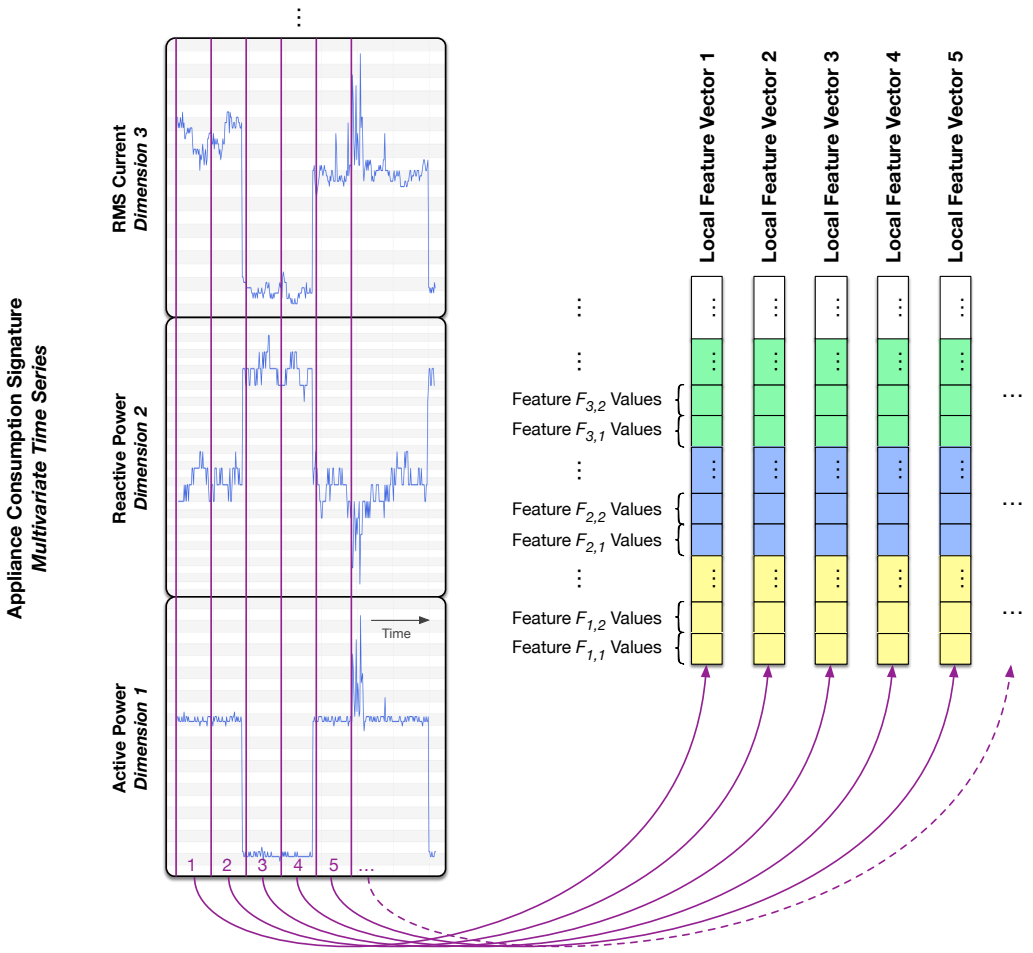


Figure 3.23: Local approach for ACS classification: several local feature vectors extracted per multivariate time series. For example, feature $F_{1,1}$ values can be local amplitudes extracted from the active power series, and $F_{2,2}$ values PAA segments values extracted from the reactive power series.

That said, once all feature vectors of a given ACS time series have been classified, their classification scores still have to be "merged" in order to return a final class decision for the time series. Two methods were first tested, namely the average of classification scores, and a majority voting on the local class decisions. However they were far less efficient than the third method that was finally retained. This method requires the classifier outputs to be probabilistic and computes a final global probabilistic score for each class by multiplying[12] all local probabilistic scores of each corresponding class. Therefore, all local feature vectors were assumed to be independent (which is however not true inside appliance states where they are temporally tied). This final score for each class model corresponds in fact to the

[12]Practically, for the sake of calculation precision, the logarithms of the probabilities are added and the overall probability is computed by taking the exponential of the sum obtained.

likelihood. According to the Bayes law (see Section 2.4.2), from these likelihoods and *a priori* probabilities of classes, *a posteriori* probabilities can be computed. In our case, we assume that all priors are equal because our appliances are uniformly distributed.

For our first tests, we wanted to use the most local features which are the closest to raw values themselves. This would then allow us to compare them, in terms of classification performance, with other lighter local features extracted from high level time series representations (see second tests). Hence, we took the raw time series points y_i and the dynamic coefficients of first and second order Δy_i and $\Delta\Delta y_i$ as well, in order to include information on the narrow neighborhood of the time series signal. These features have the advantage that no information is lost. Information on the narrow temporal context of the ACS is even added to the original time series points. However, they have the drawback to require lots of memory space and machine learning processing time, because they were not reduced to a more concise and discriminating form. The purpose of a good feature extraction is normally precisely to extract the smallest information (*features*) that will help the classifier to maximize the inter-class differences while minimizing the intra-class ones. Here we kept all the information and even added some more. We wanted to see how our classifiers would deal with that.

The results obtained for the task of appliance recognition (i.e. ACS classification) using the three above-mentioned classifiers on both databases ACS-F1 and F2 are presented in Table 3.10. The corresponding confusion matrices are not given because the results were not good enough as compared to the next tests with local amplitudes, PAA and SAX representations. The best classifier parameters and feature combinations were obtained as with the global approach. Results are given with their confidence interval ($CI_{95\%}$) (see Equation 2.55).

With both protocols P1 and P2, namely *Intersession* and *Unseen Instances*, and both databases ACS-F1 and F2, the raw time series values (points) of each ACS were always included in the best feature set after each feature selection. Both SVM and MLP found the dynamic coefficients of both orders also useful, but not GMM. In fact, the two discriminative algorithms provided almost similar results without dynamics coefficients, but the computation time was longer. In this experiment, they seem to have a bigger ease to deal with more features and therefore bigger feature vectors which help them to converge faster to a solution. GMM seem to deal with small feature sets easier, for all that these features are discriminant.

SVM provided better performance scores than GMM and MLP. With test protocol P1, GMM and MLP were quite equally efficient on both ACS-F1 and F2 databases, but with test protocol P2, GMM were better. In fact, as the number of instances increases, from ACS-F1 to ACS-F2, the number of local feature vectors explodes, and the training of a GMM or an MLP lasts much longer, as the computation depends on iterations over all these vectors.

Protocol	Database	Classifier	Best Classifier Parameters	Raw Values y_i	Dynamic Coefficients of 1st Order Δy_i	Dynamic Coefficients of 2nd Order $\Delta\Delta y_i$	Feature Vector Dimension	Classifier Performance: Accuracy [%] with $CI_{95\%}$
1. Intersession	ACS-F1	SVM	$C = 1e5$, $\gamma = 1e{-}1$	arcp	arcp	arcp	12	95.0 ± 4.3
		GMM	$m = 20$, $\nu = 1e{-}1$	arcp	—	—	4	81.0 ± 7.7
		MLP	$h/i = 1.5$, $\eta = 0.1$	arcp	arcp	arcp	12	77.0 ± 8.2
	ACS-F2	SVM	$C = 1e5$, $\gamma = 1e{-}1$	arcp	arcp	arcp	12	88.0 ± 4.2
		GMM	$m = 24$, $\nu = 1e{-}1$	arcp	—	—	4	62.7 ± 6.3
		MLP	$h/i = 3.0$, $\eta = 0.2$	arcp	arcp	arcp	12	38.4 ± 4.5
2. Unseen Instances	ACS-F1	SVM	$C = 1e6$, $\gamma = 1e{-}2$	arcp	arcp	arcp	12	66.0 ± 6.6
		GMM	$m = 7$, $\nu = 1e{-}1$	arcp	—	arcp	8	61.0 ± 6.8
		MLP	$h/i = 2.0$, $\eta = 0.1$	arcp	arcp	arcp	12	60.0 ± 6.8
	ACS-F2	SVM	$C = 1e5$, $\gamma = 1e{-}1$	arcp	arcp	arcp	12	65.3 ± 4.4
		GMM	$m = 10$, $\nu = 1e{-}1$	arcp	arcp	—	8	51.6 ± 4.6
		MLP	$h/i = 1.5$, $\eta = 0.2$	arcp	arcp	arcp	12	21.8 ± 3.8

N.B.: arcp ⇒ feature extracted on (a)ctive power, (r)eactive power, (c)urrent and (p)hase angle

Table 3.10: Local approach 1: appliance recognition (ACS classification) performances with points and dynamic coefficients as local features.

Again in this experiment, both SVM (though it provided the best results) and MLP seem to have difficulties to deal with the very small feature vectors.

For our second tests, we wanted to avoid the drawbacks of the preceding features, and perform a "real" feature extraction providing new features coming from lighter and better representations of the original information (i.e. the time series points) in the sense that they will get rid of the useless information and keep only general information on the time series shape. As explained in Section 2.2.4, efficient machine learning algorithms dealing with time series avoid working on the original raw data and consider some higher-level representation of the data. Thus, we split the ACS time series into parts and extracted some local features. We computed the local amplitudes, the PAA segment values, and the SAX letter values. A sliding window of length 9 was empirically tuned through the classification test process. No

overlap was set between the windows for the local amplitudes to be compatible with the length of PAA and SAX features. These three features were chosen because local amplitudes carry information on local amplitudes of the signal, PAA features mainly on its absolute position, and SAX features on its shape. Indeed, the SAX representation takes normalized time series as input, which is not the case for the PAA one.

The results obtained for the task of appliance recognition (i.e. ACS classification) using the three above-mentioned classifiers on both databases ACS-F1 and F2 are presented in Table 3.11 and the corresponding confusion matrices in Tables 3.12 and 3.13. The best classifier parameters and feature combinations were obtained as with the global approach. Results are given with their confidence interval ($CI_{95\%}$) (see Equation 2.55).

In a certain way, this second local approach is situated between the global and the first local approaches. The use of features based on state-of-the-art time series representations such as SAX helps a lot to perform a real and efficient feature extraction providing a lightweight representation of data and feature vectors. As shown by these last results, the accuracies of the SVM and GMM have become quite similar (about 95% and 92% for protocol P1 and P2), while the one of the MLP remains about 10% below.

With both protocols P1 and P2, and both databases ACS-F1 and F2, local amplitudes as well as PAA segment values were almost always included in the best feature set after each feature selection. However, SAX letter values were almost always useful for protocol P2 and useless for protocol P1 *Intersession*. This is not surprising because, as already explained, the SAX representation models the general shape of the time series, while the PAA one rather their absolute location. With protocol P1, one can easily imagine that, between the two sessions, the absolute locations of ACS from the same appliances are far more discriminant for the classifiers than their general shapes.

Table 3.14 shows the true positive rate (TPR) which was at least obtained for each appliance category, by each classifier with each protocol and database. With protocol P1, *mobile phones* of ACS-F1 database as well as *kettles* of ACS-F2 database systematically obtained a TPR of 100%, while *printers* and *shavers* of ACS-F2 database obtained a bad TPR. With protocol P2, *lamps CFL* and *shavers* of ACS-F2 database systematically obtained a bad TPR. These category names and TPR are shown in bold font in the confusion matrices.

3.5.5 Discussion

Further to the presented results, several considerations can be formulated.

Concerning the classification algorithms, it was interesting to observe that the three clas-

Protocol	Database	Classifier	Best Classifier Parameters	Local Amplitudes (Window length: 9, overlap: 0)	PAA Segment Values (Points per segment: 9)	SAX Letter Values (Points per letter: 9, alphabet size: 11)	Feature Vector Dimension	Classifier Performance Accuracy [%] with $CI_{95\%}$
1. Intersession	ACS-F1	SVM	$C = 1e5$, $\gamma = 1e{-}1$	(arcp)	arcp	—	(8) 4	95.0 ± 4.3
		GMM	$m = 17$, $\nu = 1e{-}5$	—	arcp	—	4	94.0 ± 4.7
		MLP	$h/i = 3.0$, $\eta = 0.15$	arcp	arcp	—	8	82.0 ± 7.5
	ACS-F2	SVM	$C = 1e4$, $\gamma = 1e1$	arcp	arcp	—	8	92.4 ± 3.5
		GMM	$m = 29$, $\nu = 1e{-}4$	—	arcp	—	4	93.3 ± 3.3
		MLP	$h/i = 3.5$, $\eta = 0.15$	arcp	arcp	—	8	82.2 ± 5.0
2. Unseen Instances	ACS-F1	SVM	$C = 1e6$, $\gamma = 1e{-}2$	arcp	—	arcp	8	76.0 ± 5.9
		GMM	$m = 7$, $\nu = 1e{-}2$	arcp	arcp	—	8	76.5 ± 5.9
		MLP	$h/i = 2.5$, $\eta = 0.1$	arcp	arcp	arcp	12	65.5 ± 6.6
	ACS-F2	SVM	$C = 1e6$, $\gamma = 1e{-}3$	arcp	arcp	arcp	12	73.6 ± 4.1
		GMM	$m = 9$, $\nu = 1e{-}3$	arcp	arcp	arcp	12	72.9 ± 4.1
		MLP	$h/i = 2.0$, $\eta = 0.1$	arcp	arcp	arcp	12	66.4 ± 4.4

N.B.: arcp ⇒ feature extracted on (a)ctive power, (r)eactive power, (c)urrent and (p)hase angle

Table 3.11: Local approach 2: appliance recognition (ACS classification) performances with local amplitudes, PAA segments and SAX letter values as local features.

sifiers don't have the same ability to recognize the appliances of each category, even if they yielded comparable classification accuracies. Moreover, discriminant classification algorithms like SVM seem to have a better propension to deal with big feature vectors and potential useless features. To a certain extent, they perform by themselves a kind of feature selection. Hence, the two discriminative algorithms provided almost similar results without dynamics coefficients, but the computation time was longer. They seem to have a bigger ease to deal with more features and therefore bigger feature vectors which help them to converge faster to a solution. GMM seem to deal with small feature sets more easily, for all that these features are discriminant. As already explained, the use of an MLP is not quite adapted to the 2$^{\text{nd}}$ protocol consisting of a k-Fold CV because its performance highly depends of a good random weight initialization, and for each CV step, the MLP would have normally required multiple training phases until a suitable random initialization would yield a good model.

	Coffee machine	Computer station	Fridge / Freezer	Hi-Fi system	Lamp CFL	Laptop	Microwave oven	Mobile phone	Printer	TV
Coffee machine	100	0	0	0	0	0	0	0	0	0
Computer station	0	100	0	0	0	0	0	0	0	0
Fridge / Freezer	0	0	100	0	0	0	0	0	0	0
Hi-Fi system	0	0	0	100	0	0	0	0	0	0
Lamp CFL	0	0	0	0	80	0	10	0	10	0
Laptop	0	10	0	0	0	90	0	0	0	0
Microwave oven	0	0	0	10	0	0	90	0	0	0
Mobile phone	0	0	0	0	0	0	0	**100**	0	0
Printer	0	0	0	0	0	0	0	0	100	0
TV	0	0	0	0	0	10	0	0	0	90

(a) P1 – ACS-F1 – SVM – Accuracy: $95.0 \pm 4.3\,\%$

	Coffee machine	Computer station	Fan	Fridge / Freezer	Hi-Fi system	Kettle	Lamp CFL	Lamp inc.	Laptop	Microwave oven	Mobile phone	Monitor	Printer	Shaver	TV
Coffee machine	87	7	0	0	0	7	0	0	0	0	0	0	0	0	0
Computer station	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0
Fan	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0
Fridge / Freezer	0	0	0	93	0	7	0	0	0	0	0	0	0	0	0
Hi-Fi system	0	0	0	0	100	0	0	0	0	0	0	0	0	0	0
Kettle	0	0	0	0	0	**100**	0	0	0	0	0	0	0	0	0
Lamp CFL	0	0	0	0	0	0	100	0	0	0	0	0	0	0	0
Lamp inc.	0	0	0	0	0	0	0	**100**	0	0	0	0	0	0	0
Laptop	0	7	0	0	0	0	0	0	93	0	0	0	0	0	0
Microwave oven	0	0	0	0	0	33	0	0	0	67	0	0	0	0	0
Mobile phone	0	0	0	0	0	7	0	0	0	0	93	0	0	0	0
Monitor	0	0	0	0	0	0	0	0	0	0	0	100	0	0	0
Printer	0	7	0	0	0	13	0	0	0	0	0	0	**80**	0	0
Shaver	0	0	0	0	0	7	0	0	0	0	13	0	0	**80**	0
TV	0	0	0	0	0	0	0	0	7	0	0	0	0	0	93

(b) P1 – ACS-F2 – SVM – Accuracy: $92.4 \pm 3.5\,\%$

	Coffee machine	Computer station	Fridge / Freezer	Hi-Fi system	Lamp CFL	Laptop	Microwave oven	Mobile phone	Printer	TV
Coffee machine	100	0	0	0	0	0	0	0	0	0
Computer station	0	100	0	0	0	0	0	0	0	0
Fridge / Freezer	0	0	100	0	0	0	0	0	0	0
Hi-Fi system	0	0	0	100	0	0	0	0	0	0
Lamp CFL	0	0	10	0	90	0	0	0	0	0
Laptop	0	10	0	0	0	60	0	0	30	0
Microwave oven	0	0	0	0	0	0	100	0	0	0
Mobile phone	0	0	0	0	0	0	0	**100**	0	0
Printer	0	0	0	0	0	0	0	0	100	0
TV	0	0	10	0	0	0	0	0	0	90

(c) P1 – ACS-F1 – GMM – Accuracy: $94.0 \pm 4.7\,\%$

	Coffee machine	Computer station	Fan	Fridge / Freezer	Hi-Fi system	Kettle	Lamp CFL	Lamp inc.	Laptop	Microwave oven	Mobile phone	Monitor	Printer	Shaver	TV
Coffee machine	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Computer station	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0
Fan	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0
Fridge / Freezer	0	0	0	100	0	0	0	0	0	0	0	0	0	0	0
Hi-Fi system	0	0	0	0	100	0	0	0	0	0	0	0	0	0	0
Kettle	0	0	0	0	0	**100**	0	0	0	0	0	0	0	0	0
Lamp CFL	0	0	0	0	0	0	100	0	0	0	0	0	0	0	0
Lamp inc.	0	0	0	0	0	0	0	**100**	0	0	0	0	0	0	0
Laptop	0	13	7	0	0	0	0	0	73	0	0	0	7	0	0
Microwave oven	0	0	0	0	0	0	0	0	0	100	0	0	0	0	0
Mobile phone	0	0	0	0	0	0	0	0	0	0	100	0	0	0	0
Monitor	0	0	0	0	0	0	0	0	0	0	0	93	7	0	0
Printer	0	7	0	0	0	0	0	0	7	0	0	0	**67**	13	7
Shaver	0	0	0	0	0	7	0	0	0	0	13	0	7	**73**	0
TV	0	0	0	0	0	0	0	0	0	0	0	0	7	0	93

(d) P1 – ACS-F2 – GMM – Accuracy: $93.3 \pm 3.3\,\%$

	Coffee machine	Computer station	Fridge / Freezer	Hi-Fi system	Lamp CFL	Laptop	Microwave oven	Mobile phone	Printer	TV
Coffee machine	70	0	10	0	0	0	0	20	0	0
Computer station	0	90	0	0	0	0	0	0	0	10
Fridge / Freezer	0	0	80	10	0	0	0	10	0	0
Hi-Fi system	0	0	0	80	10	0	0	10	0	0
Lamp CFL	0	0	10	0	70	10	0	10	0	0
Laptop	0	10	0	0	0	90	0	0	0	0
Microwave oven	0	0	10	10	0	0	80	0	0	0
Mobile phone	0	0	0	0	0	0	0	**100**	0	0
Printer	10	0	0	0	10	0	0	0	70	10
TV	0	0	0	0	10	0	0	0	0	90

(e) P1 – ACS-F1 – MLP – Accuracy: $82.0 \pm 7.5\,\%$

	Coffee machine	Computer station	Fan	Fridge / Freezer	Hi-Fi system	Kettle	Lamp CFL	Lamp inc.	Laptop	Microwave oven	Mobile phone	Monitor	Printer	Shaver	TV
Coffee machine	73	0	0	7	0	7	0	0	0	0	13	0	0	0	0
Computer station	0	87	0	0	0	0	0	0	7	0	0	0	0	0	7
Fan	0	0	87	0	0	0	0	0	0	0	0	13	0	0	0
Fridge / Freezer	0	0	0	93	0	7	0	0	0	0	0	0	0	0	0
Hi-Fi system	0	0	0	0	87	0	7	0	0	0	7	0	0	0	0
Kettle	0	0	0	0	0	**100**	0	0	0	0	0	0	0	0	0
Lamp CFL	0	0	7	0	0	20	67	0	0	0	0	0	7	0	0
Lamp inc.	0	0	0	0	0	0	0	**100**	0	0	0	0	0	0	0
Laptop	0	0	7	0	0	0	0	7	87	0	0	0	0	0	0
Microwave oven	0	0	0	0	0	7	0	0	0	93	0	0	0	0	0
Mobile phone	0	0	0	0	0	7	0	0	0	0	93	0	0	0	0
Monitor	0	0	0	0	0	0	7	0	7	0	0	87	0	0	0
Printer	0	0	0	7	0	13	7	0	7	0	0	7	**60**	0	0
Shaver	0	0	0	0	0	7	7	0	0	0	33	0	0	**53**	0
TV	0	0	0	0	0	0	7	0	13	13	0	0	0	0	67

(f) P1 – ACS-F2 – MLP – Accuracy: $82.2 \pm 5.0\,\%$

Table 3.12: Local approach 2: best classification confusion matrices (values are in %) obtained with protocol P1 *Intersession* on both databases ACS-F1 and F2.

	Coffee machine	Computer station	Fridge / Freezer	Hi-Fi system	Lamp CFL	Laptop	Microwave oven	Mobile phone	Printer	TV
Coffee machine	100	0	0	0	0	0	0	0	0	0
Computer station	0	75	0	0	0	15	0	0	10	0
Fridge / Freezer	0	0	80	5	5	0	10	0	0	0
Hi-Fi system	0	0	10	90	0	0	0	0	0	0
Lamp CFL	0	0	0	0	**70**	0	0	15	5	10
Laptop	0	10	0	0	5	80	0	0	0	5
Microwave oven	5	5	0	0	0	0	80	0	10	0
Mobile phone	0	0	0	0	10	0	5	75	10	0
Printer	10	0	5	5	0	0	15	0	60	5
TV	0	5	5	20	10	0	0	0	10	50

(a) P2 – ACS-F1 – SVM – Accuracy: $76.0 \pm 5.9\,\%$

	Coffee machine	Computer station	Fan	Fridge / Freezer	Hi-Fi system	Kettle	Lamp CFL	Lamp inc.	Laptop	Microwave oven	Mobile phone	Monitor	Printer	Shaver	TV
Coffee machine	93	0	0	0	3	0	0	3	0	0	0	0	0	0	0
Computer station	0	67	0	0	0	0	0	0	10	0	0	3	0	0	20
Fan	0	0	77	13	0	0	0	0	0	0	0	0	0	0	10
Fridge / Freezer	0	0	7	77	0	0	0	0	0	10	0	0	7	0	0
Hi-Fi system	0	0	0	3	70	0	0	3	0	0	7	3	3	10	0
Kettle	0	0	0	0	0	97	0	0	0	0	0	0	0	3	0
Lamp CFL	0	0	7	0	0	3	83	0	0	0	0	3	0	3	0
Lamp inc.	0	0	7	0	13	0	0	77	0	0	3	0	0	0	0
Laptop	0	3	10	0	0	0	3	0	73	0	0	10	0	0	0
Microwave oven	3	3	0	0	0	0	0	0	0	90	0	0	3	0	0
Mobile phone	0	0	0	0	0	0	0	0	0	0	83	0	0	17	0
Monitor	0	0	0	0	3	0	0	0	20	3	0	67	0	0	7
Printer	7	3	0	0	0	0	0	0	0	3	3	10	50	17	7
Shaver	0	0	0	0	13	7	10	0	0	0	0	17	0	0	**53**	0
TV	0	33	0	0	0	0	0	0	7	0	0	13	0	0	**47**

(b) P2 – ACS-F2 – SVM – Accuracy: $73.6 \pm 4.1\,\%$

	Coffee machine	Computer station	Fridge / Freezer	Hi-Fi system	Lamp CFL	Laptop	Microwave oven	Mobile phone	Printer	TV
Coffee machine	90	0	0	0	0	0	10	0	0	0
Computer station	0	50	0	0	20	0	0	0	0	30
Fridge / Freezer	0	0	70	5	10	10	5	0	0	0
Hi-Fi system	0	0	5	75	0	0	0	10	10	0
Lamp CFL	0	10	10	0	**60**	5	0	10	5	0
Laptop	0	0	0	0	10	85	0	0	5	0
Microwave oven	0	0	0	0	0	0	90	0	10	0
Mobile phone	0	0	10	0	0	0	0	90	0	0
Printer	0	0	5	0	0	0	5	10	80	0
TV	0	5	5	0	0	15	0	0	0	75

(c) P2 – ACS-F1 – GMM – Accuracy: $76.5 \pm 5.9\,\%$

	Coffee machine	Computer station	Fan	Fridge / Freezer	Hi-Fi system	Kettle	Lamp CFL	Lamp inc.	Laptop	Microwave oven	Mobile phone	Monitor	Printer	Shaver	TV
Coffee machine	73	0	0	0	0	0	0	0	0	13	0	0	13	0	0
Computer station	0	67	0	0	0	0	0	0	3	3	0	0	0	0	27
Fan	0	0	77	13	0	0	7	0	0	0	0	0	3	0	0
Fridge / Freezer	0	0	7	80	0	0	0	0	0	7	0	0	0	0	7
Hi-Fi system	0	7	3	0	77	0	0	0	0	0	7	3	3	0	0
Kettle	3	0	0	0	0	93	0	0	0	3	0	0	0	0	0
Lamp CFL	0	0	7	3	0	3	80	0	0	0	3	3	0	0	0
Lamp inc.	0	0	0	7	17	3	0	73	0	0	0	0	0	0	0
Laptop	0	3	3	0	0	0	0	0	80	0	0	10	0	0	3
Microwave oven	0	0	0	0	0	0	0	0	0	90	0	0	10	0	0
Mobile phone	0	0	0	3	3	3	0	0	0	0	73	0	0	17	0
Monitor	0	0	0	3	0	0	0	0	23	0	0	70	3	0	0
Printer	13	0	0	0	0	0	0	0	0	0	3	0	77	7	0
Shaver	0	0	0	0	13	0	7	13	0	0	30	0	3	**33**	0
TV	0	27	3	0	0	0	0	0	3	0	0	13	3	0	**50**

(d) P2 – ACS-F2 – GMM – Accuracy: $72.9 \pm 4.1\,\%$

	Coffee machine	Computer station	Fridge / Freezer	Hi-Fi system	Lamp CFL	Laptop	Microwave oven	Mobile phone	Printer	TV
Coffee machine	75	0	0	0	0	0	10	5	10	0
Computer station	0	45	5	0	0	20	5	0	5	20
Fridge / Freezer	0	0	65	5	10	10	0	0	10	0
Hi-Fi system	0	0	5	70	0	0	0	15	10	0
Lamp CFL	0	0	15	0	**40**	25	0	10	0	10
Laptop	0	0	0	0	5	90	0	0	5	0
Microwave oven	10	0	0	10	0	0	70	0	10	0
Mobile phone	0	0	0	0	10	0	5	85	0	0
Printer	0	0	0	0	0	25	0	10	60	5
TV	0	25	0	0	0	15	0	0	5	55

(e) P2 – ACS-F1 – MLP – Accuracy: $65.5 \pm 6.6\,\%$

	Coffee machine	Computer station	Fan	Fridge / Freezer	Hi-Fi system	Kettle	Lamp CFL	Lamp inc.	Laptop	Microwave oven	Mobile phone	Monitor	Printer	Shaver	TV
Coffee machine	73	0	0	0	3	0	0	0	0	23	0	0	0	0	0
Computer station	0	73	0	0	0	0	0	0	7	3	0	0	0	0	17
Fan	0	0	63	13	0	0	10	0	0	0	3	3	0	3	3
Fridge / Freezer	0	0	17	67	0	0	0	0	0	3	0	7	0	0	7
Hi-Fi system	0	0	0	3	80	0	0	3	0	0	7	3	3	0	0
Kettle	0	0	0	0	0	93	0	0	0	0	3	0	0	3	0
Lamp CFL	0	0	10	0	0	3	83	0	0	0	3	0	0	0	0
Lamp inc.	0	0	0	0	20	3	0	70	0	0	3	0	0	3	0
Laptop	0	17	3	0	0	0	3	0	50	0	0	13	3	0	10
Microwave oven	7	0	0	10	0	3	0	0	0	80	0	0	0	0	0
Mobile phone	0	0	0	3	0	3	0	0	0	0	83	0	0	10	0
Monitor	0	3	3	0	7	0	0	0	20	3	0	53	7	0	3
Printer	7	0	0	0	0	0	0	0	13	7	10	3	33	20	7
Shaver	0	0	7	0	13	7	7	3	0	0	17	0	3	**43**	0
TV	0	23	0	0	0	0	0	0	3	0	0	23	0	0	**50**

(f) P2 – ACS-F2 – MLP – Accuracy: $66.4 \pm 4.4\,\%$

Table 3.13: Local approach 2: best classification confusion matrices (values are in %) obtained with protocol P2 *Unseen Instances* on both databases ACS-F1 and F2.

Protocol	Database	True Positive Rate (TPR)	Classifier	Coffee machine	Computer station	Fridge / freezer	Hi-Fi system	Lamp CFL	Laptop	Microwave oven	Mobile phone	Printer	TV	Fan	Kettle	Lamp inc.	Monitor	Shaver
1. Intersession	ACS-F1	> 90%	SVM	✓	✓	✓	✓		✓	✓	✓	✓	✓	–	–	–	–	–
			GMM	✓	✓	✓	✓	✓		✓	✓	✓	✓	–	–	–	–	–
			MLP		✓				✓		✓		✓	–	–	–	–	–
	ACS-F2	> 87%	SVM	✓	✓	✓	✓	✓	✓		✓		✓	✓	✓	✓	✓	
			GMM	✓	✓	✓	✓	✓		✓	✓		✓	✓	✓	✓	✓	
			MLP		✓	✓	✓		✓	✓	✓			✓	✓	✓	✓	
2. Unseen Instances	ACS-F1	> 75%	SVM	✓	✓	✓	✓		✓	✓	✓			–	–	–	–	–
			GMM	✓			✓		✓	✓	✓	✓	✓	–	–	–	–	–
			MLP	✓			✓		✓		✓			–	–	–	–	–
	ACS-F2	> 70%	SVM	✓		✓	✓	✓	✓	✓	✓			✓	✓	✓		
			GMM	✓		✓	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓	
			MLP	✓	✓		✓	✓		✓	✓				✓	✓		

N.B.: A blue ✓ ⇒ a TPR of 100%

Table 3.14: Local approach 2: minimal true positive rate (TPR) obtained for each appliance category by each classifier for each protocol and database.

Concerning the extracted features, those which were extracted from the active power, the reactive power and the RMS current were always used and primordial for the classification task, while those from the phase angle did not always appear to be helpful depending on the classification algorithm and the approach used. As the number of extracted features was too large to perform a global grid search on all of them to find the best feature selection, we did the grid search on subsets of features of same nature, which were themselves individually selected with a grid search. Even if that method was rather efficient at the end, we plan to use a genetic algorithm, as we did in the next case study (see Chapter 4), in order to select a potential better subset as future work.

Concerning the two test protocols that we proposed, tested and published along with our ACS-F1 and F2 databases, we observed, as we predicted while defining them, that the classification task defined by the first protocol *intersession* is easier than the one defined by the second protocol *unseen instances*, and provides better results with an average accuracy

of 95% versus 75%, i.e. 20% more. Indeed, the appliances to recognize during the test phase have been already "seen" during the training phase.

Concerning the global and local approaches to time series classification, they may be not really *time series* oriented depending on the features used. If these features are only statistics computed on the raw points, the series can be considered as a set of time-independent points and not as a time-ordered sequence of points. A reason why these features worked is that they are *statistical* and thus efficient "by definition". It is not an easy task to find and use temporal features. However, some features that we used, such as the frequency-based features, the SAX or the PAA features, are strongly tied to the time. In the global approach, time series are considered as a whole "inseparable" entity. The more points the series will contain, the more precise the statistics will be. In the second local approach, the series are considered as a separable entity composed of more or less small parts. For the first experiments, raw points and dynamics coefficients can lead to good results but are more heavy to handle if the time series become long and numerous. For the second experiments, we observed that the AMP, PAA and SAX features are complementary because the first brings information on the amplitude of the signal, the second on its absolute position, and the third on its shape, relatively to other times series. With the SAX representation, the signal is normalized, which allows a better comparison between the time series [Ratanamahatana et al., 2010]. Our classification results have also confirmed the importance of approximating and simplifying the time series by using such representations. This results in a "real" feature extraction, providing discriminating features, decreasing the data space on hard disks, simplifying the data handling, and improving the scalability, the computation, the runtime, the results and the robustness of the classification algorithms. In the global approach, the statistical part starts at the level of the feature extraction, while in the local approach the whole work is left to the classifier. Moreover, while PAA was the best dominating feature in the 1st protocol *intersession*, SAX and AMP were the dominating ones in the 2nd protocol *unseen instances*, probably because again, they rather model the shape than the absolutes locations of the signal parts like PAA does. The simplifications engendered by these time series representations also take account of the transitions, for instance if a SAX or PAA segment falls astride two states of the signal, the transition will be represented by the mean segment. Finally, signal parts located in lower states are often not as well classified as those in upper parts because, lower states corresponds to lower electricity consuming states like "idle" states which tends to be similar between several appliance categories. The discriminating parts are logically the higher states corresponding to higher consuming states of running appliances (see Figure 3.24). This analysis done from the local approach results pushed us to elaborate the cluster feature consisting in automatically detecting states in time series with the k-Means++ clustering algorithm and then extracting various (rather statistical) features from the different clusters obtained.

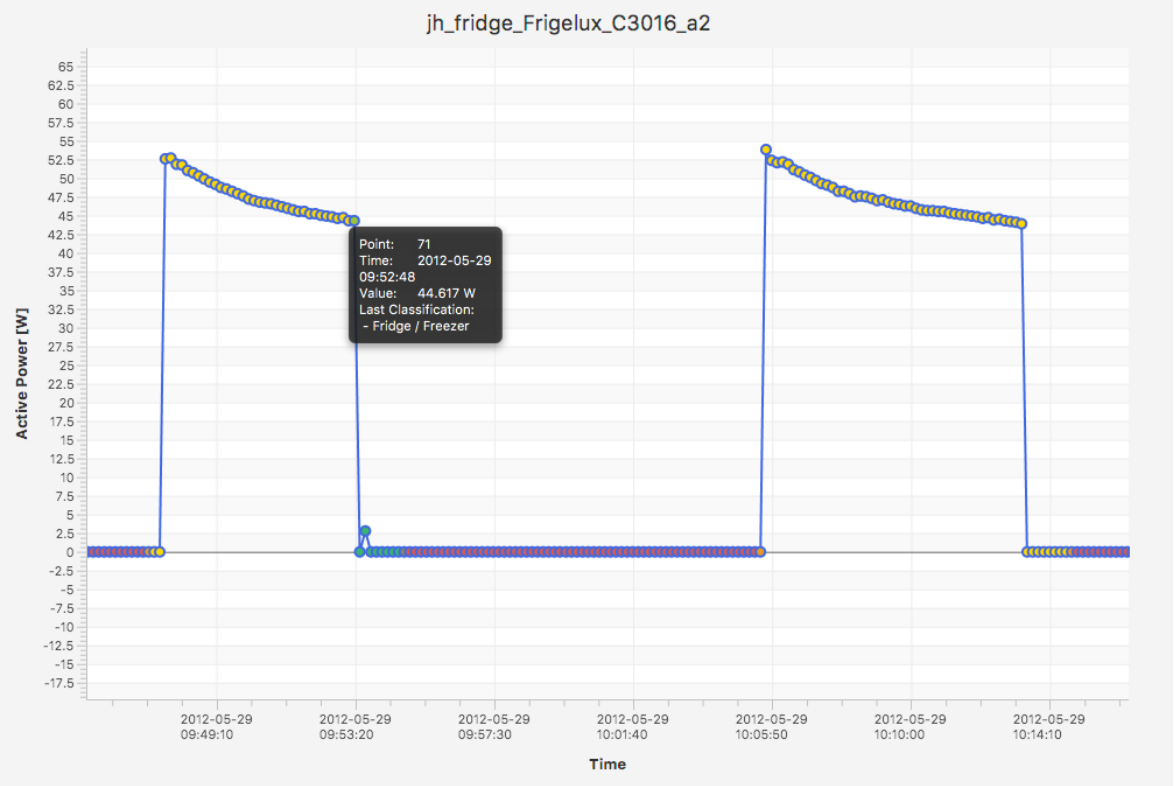


Figure 3.24: Local approach 2 with the annotated classified parts of an ACS: idle state parts are more often wrongly classified "higher" running state parts.

By definition, an ACS must contain the electrical consumption data of an appliance recorded in all its possible states, in order to have a so-called *appliance consumption signature*. In this case study, we developed appliance recognition systems trained on ACS from our ACS-F1 and F2 databases, and tested on other ACS of the same databases. Although it is convenient and efficient to train such classifiers on ACS databases, one can easily imagine that in real life applications, like in ILM systems, appliance recognition cannot be done on ACS too. Most of the time, ILM systems just continuously record appliance consumption signals along with the time, and one hour of signal does not correspond to an ACS. While the training phase will not change, the classification (test) phase will need to be adapted for both global and local approaches. The continuously recorded signals need to be split into parts, which will be generally smaller for a local approach than for a global one. Thus, we may first think that a classification decision can be returned earlier with a local approach, but the problem is not that simple. Even if with small time series parts, a decision can be computed earlier, it does not imply that the decision will be correct. Most appliances are often idle or in other low consumption mode, and their electrical consumption signal does not contain any specific useful information (so we could even need first a *running state detection* module). Hence, a small time series window will often just lead to a wrong classification, whatever

the approach used. What will make a decision more accurate is taking a larger window for the global approach, and several preceding windows (whose classification scores were already computed and can be combined with the one of the current window) for the local approach. Hence at last, we can say that both approaches are rather equivalent from this point of view.

As a final word, we warn that this discussion and some interpretations of the results should be taken carefully due the small size of the databases. However, in the scope of this research work, the results are valid and coherent. Different experiments with k-NN, GMM and HMM classifiers were made by my colleague Antonio Ridi. We worked together on this case study and mostly on the data acquisition and ACS-F databases creation. He went deeper into the particular local approach using raw points and dynamics coefficients as features. He obtained generally similar results, except those obtained with his GMM which were a bit better than mine with my GMM. His work was reported in his PhD thesis [Ridi, 2016, Chap. 3.6]. Our common published papers are listed at the end of this document.

3.6 Conclusion

In this case study, we reported on our research work done in the domain of intrusive load monitoring (ILM III). We acquired appliance consumption signatures (ACS) recorded with affordable and low-consuming "smart" outlets installed at the level of the electricity plugs in houses with the aim to automatically recognize electrical appliances from different categories. Such ACS are multivariate time series whose dimensions are constituted by six electrical features sampled at a low frequency (10^{-1} Hz), namely the active and reactive power, the current, the voltage, the phase angle, and the frequency.

The State of the Art revealed a large heterogeneity in the recorded appliance categories, the features extracted and the machine learning algorithms used by researchers, making all the classification results obtained difficult to compare. Furthermore, the lack of databases which are publicly available pushed us to create two consequent databases ideal for various machine learning tasks. We populated our databases called ACS-F1 and F2 with 200 and 450 consumption signatures recorded from 10 and 15 categories of home appliances during two sessions of 1 hour. For each appliance category, we set an acquisition protocol in order to record all possible running states. Our databases have two main advantages: (1) each ACS pair comes from a different appliance, allowing to build robust generic models, and (2) two test protocols are provided, allowing the research community to compare algorithm performances. The first protocol *intersession* consists in classifying appliances already seen in the training set, while the second *unseen appliances* consists in classifying unknown appliances. Making the so-called ACS-F1 and F2 databases available for free to the whole scientific community along with two specific test protocols constituted a first objective of this case study.

As second objective, we applied classification methodologies in order to build models of electrical appliance categories and recognize their consumption signatures, for instance when plugged in a power outlet in a room of a given house. The time series nature of ACS made this case study a typical case of time series classification where a generic data-driven approach could be successfully applied. We tested three classifiers, namely SVM, GMM and MLP, with two different approaches of the time series signal: a global and a local one. The first global approach consisted in building one unique feature vector per ACS, by extracting and concatenating different global features from each dimension of the time series. The second approach consisted in dividing the ACS in smaller segments, extracting features from the segments and building as many feature vectors as segments. Various features were extracted and both approaches provided similar results for each test protocol on each database, with a best accuracy around 91–95% $\pm$ 5% for the first protocol, and 73–81% $\pm$ 5% for the second protocol. Thus, our preference goes to the global approach, which provides a lighter representation of the data, as well as a better efficiency, notably in terms of computation time and scalability. The ability of each classifier to recognize different appliance categories will certainly lead us to perform a fusion of the classifier scores as a future work, in order to gain some percents of recognition accuracy.

In conclusion, the generic approaches to time series classification that we used worked quite well for this case study. From a *data mining process* point of view, we had the chance to carry this case study out from the start to the end, and mostly the data acquisition which was performed with a purpose of appliance recognition in mind. The quite good results obtained also depend on this. However, as explained in the general data mining process presented in Section 2.3, our results and conclusions still must be taken cautiously, as they might not be valid anymore when there will be more data, i.e. new ACS instances and appliance categories, which of course constitutes one of the perpectives of this research work. Other perspectives may push us to focus on running state recognition or domotics system improvement.

4

Case Study: Intraocular Pressure Profile Classification

When you open your eyes, I see your heart. When I close my eyes, I see the world.

Tibetan saying

Contents

4.1 Introduction

4.1.1 Context

According to the World Health Organization (WHO), glaucoma is the second most frequent cause of blindness in the world after cataract. Estimated to 60 millions in 2010, the number of people worldwide suffering from glaucoma is expected to reach the 80 millions by 2020

[Quigley and Broman, 2006]. This disease of the optic nerve is asymptomatic, painless, irreversible, and leads to blindness unless Intraocular Pressure (IOP) is controlled. For a long time, an Intraocular Pressure (IOP) measure greater than 21 mmHg has been considered by medical doctors as a direct indicator of the disease [Tielsch et al., 1991]. In addition to glaucomatous patients, studies have estimated that 4-10% of the population older than 40 years will have an IOP of 21 mmHg or higher without detectable signs of glaucoma damage. Over a 5-year period, people with such Ocular Hypertension (OHT) have a 10% overall risk of developing glaucoma [Berdahl, 2014].

IOP remains the only controllable risk factor to stabilize patients and various therapeutic options exist to reduce it [Chang and Hodapp, 2015]. However, IOP follows individual circadian[1] patterns and hence cannot be effectively monitored with current devices that lead to therapeutic failure and progressive vision loss by about one third of the glaucomatous patients. Furthermore, there is currently no mean to predict which OHT patients will convert to glaucoma, leaving most of them either overtreated or undertreated. The current way to monitor IOP and detect OHT is using tonometers whose main drawback is the punctual nature of their few IOP measurements which can be done during the eye examination routine. Hence, tonometry cannot be performed at night without disturbing sleep and inducing significant measurement artifacts.

A first breakthrough toward better understanding, management and prognosis of the disease has been the availability of a Contact Lens Sensor (CLS) called Triggerfish® and developed by the Swiss company Sensimed. The CLS allows to perform automated, practical, reliable and convenient ambulatory recording of continuous 24-hour IOP-related profiles, as opposed to repeated tonometry otherwise gathered in sleep laboratory. Since 2011, Sensimed has been collecting profiles from patients through clinical studies over the world, with the intention to do statistical analysis across populations, and discover new IOP patterns allowing a better understanding and detection of glaucoma.

A second breakthrough toward prognosis of the progression of the disease has been an independent and recent study reporting that the conversion to glaucoma might be predicted by looking at specific dynamic patterns of the 24-hour IOP profile (like the slope at awaking) in a population of OHT patients [Grippo et al., 2013].

4.1.2 Objectives

In collaboration with Sensimed, we exploited the 24-hour continuous IOP profiles acquired with the Triggerfish CLS and applied automatic machine learning methodologies in order to discriminate between healthy and glaucomatous patients. The final purpose of this is to

[1]The term *circadian* is used to qualify a biological cycle whose duration lasts around a day, i.e. 24 hours.

embed the generated models into utility softwares dedicated to help medical doctors to establish glaucoma diagnosis and take decisions concerning the treatment to adopt, for example medication or surgery.

More precisely, a first objective was to find out potentially discriminating features like specific patterns or biomarkers and to visualize them along with the 24-hour IOP profiles in an efficient way that will help medical doctor to interpret and analyse the profiles. The IOP profiles being time series, the state-of-the-art representations and features described in Chapter 2 could be used.

A second objective was to extract these features from the IOP profiles, and apply machine learning for the task of glaucoma detection by classifying the profiles into glaucomatous and healthy classes.

4.1.3 Outline

This chapter is organized as follows. First, in Section 4.2, we give the specific elements of theory related to the Intraocular Pressure (IOP) and the glaucoma disease. In Section 4.3, we present a State of the Art in the fields of (1) physiological signal monitoring with a focus on IOP continuous monitoring, and (2) machine learning for medical diagnosis with a focus on glaucoma diagnosis. Then, in Section 4.4, we describe the IOP-related profile database in terms of data acquisition, database contents and formats. In Section 4.5, we describe the experiments and discuss the results obtained in the task of glaucomatous profile detection on a valid profile subset, following our test protocol and both global and local generic approaches to time series classification using SVM and GMM as discriminative and generative classifiers. This section also presents the particular physiological features which were extracted. Finally, in Section 4.6, we conclude and give some research perspectives in the domain.

4.2 Specific Elements of Theory

Intraocular Pressure (IOP) is a *biomechanical physiological signal* which is strongly tied to the *glaucoma disease*. Thus, some specific elements of theory about physiological signals, IOP and glaucoma must be introduced before talking of IOP profile classification.

4.2.1 Physiological Signals

Physiological signals refer to physiological activities of living organisms (mostly humans), ranging from gene and protein sequences, to neural and cardiac rhythms, to tissue and organ images [Chang and Moura, 2010; Tripathi, 2011]. Such signals are often characterized by some pattern repetitions due to the cyclic nature of the functioning of certain (sub)systems in the organism. Strictly physiological cycles, such as the cardiac and the respiratory cycles, are

relatively short and semi-automatic, whereas more behavioural cycles, such as the circadian rhythm, are longer. Physiological signals that are recorded and thus sampled over the time can be seen as time series (see Chapter 2). Often also called biomedical signals, physiological signals are mainly characterized by their:

1. **Accessibilty**, expressing the way to access the signal which can be (1) *internal*, implying the use of capture devices (sensors) put on the body directly, (2) *emanated*, involving external sensors like infrared cameras, or (3) *derived*, when an analysis is required, for example a blood test to measure the glucose level.

2. **Physical quantity**, depending on the targeted biological (sub)system, and referring to one of the following categories: temperature, pressure, flow, dimensions, displacement, impedance, biopotential, chemical concentration and composition.

3. **Support**, expressing the nature of the signal and consequently the way to capture it, for example *bioelectrical signals*, produced by cells like neurons or myocytes (i.e. muscle cells) and measurable with an electroencephalogram (EEG), an electrocardiogram (ECG) or an electromyogram (EMG), or *biomechanical signals*, referring to movements, displacements, pressures or flows produced by an organism or some of its subsystems. Other types of signals include *biomagnetic*, *bioacoustic*, *biochemical*, *bioimpedance*, or *biooptical signals* [Bronzino, 2010].

Physiological signals are involved in four main types of applications: (1) *health monitoring*, e.g. to watch over elderly, ill or disabled people [Pantelopoulos and Bourbakis, 2010], (2) *biometry*, e.g. to recognize or verify the identity of a person, (3) *user activity recognition*, e.g. reading books or watching movies [Lara et al., 2012], and (4) *emotion recognition*, e.g. joy, anger, fear or disgust [Jerritta et al., 2011].

4.2.2 Intraocular Pressure (IOP)

Intraocular Pressure (IOP) is a dynamic biomechanical physiological signal characterized by a circadian rhythm and many spontaneous changes [Weinreb and Khaw, 2004]. These variations result from complex interactions between external stimuli, such as physical activities, food intake, stress, emotions, and intrinsic regulated biological processes, such as the sleep. IOP is usually measured in [mmHg] by means of a specific device called *tonometer*, which only allows to perform isolated and punctual measures. By capturing multiple IOP measures at different time points (mostly during clinic hours), ophthalmologists obtain what they call a diurnal tension curve (DTC).

IOP is mainly involved in health monitoring applications aiming at diagnosing the glaucoma disease. High IOP is known to be a strong indicator to the glaucoma disease and the only treatable symptom. Therefore, medical doctors notably base their diagnosis and the choice of treatment on the observed DTC.

4.2.3 Glaucoma Disease

Glaucoma refers to a group of related eye disorders that damage the optic nerve which transmit visual information from the eye to the brain. Glaucoma usually comes along with few or no initial symptoms, and half of the glaucomatous people don't know that they have it. That's why glaucoma is sometimes called the *silent thief of sight*. Most of the time, glaucoma is associated with an abnormally high IOP, a phenomenon called Ocular Hypertension (OHT), but can occur even if the IOP is normal. In the case it is not treated or controlled, glaucoma yields a peripheral vision loss in a first time, and can lead to blindness in a second time [Berdahl, 2014; Fundation, 2015; RNIB, 2015] (see Figure 4.1).

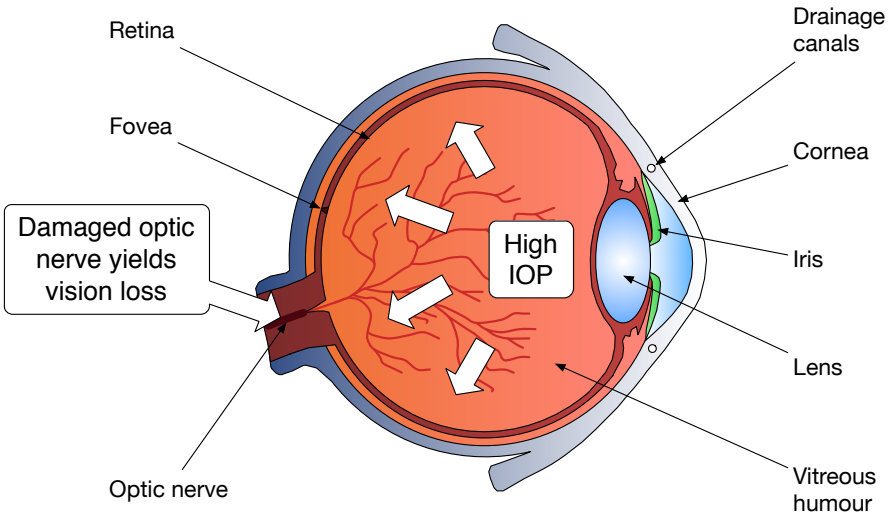


Figure 4.1: A high IOP damages the optic nerve, resulting in a loss of vision.

Types of Glaucoma

There are several types of glaucoma, the two main being the open-angle glaucoma (OAG) and narrow angle glaucoma. The terms *angle* refers to the drainage angle between the iris and the cornea that controls the outflow of the watery fluid (*aqueous*) continually produced inside the eye. When the aqueous reaches the drainage angle too easily, the glaucoma is an OAG, and when it doesn't because the angle is blocked, it is a narrow angle glaucoma.

Accounting for more than 90% of all glaucoma cases, OAG is the most common type of glaucoma. Main OAG varieties include:

- **Primary open-angle glaucoma (POAG)**, which gradually reduces the peripheral vision (*tunnel vision*) without other symptoms, until reaching blindness, unless the IOP is lowered.

- **Normal-tension glaucoma (NTG)**, also called low-tension or normal-pressure glaucoma, which causes visual field loss due to optic nerve damage like POAG, with an IOP remaining in its normal range. Thus there are often no symptoms until a tunnel vision appears. Its cause remains unknown, but many doctors believe that it is related to poor blood flow to the optic nerve.
- **Pigmentary glaucoma**, which is a rare OAG form caused by the clogging of the drainage angle due to pigments that have broken loose from the iris, reducing the aqueous outflow from the eye. The rare symptoms are pain and blurry vision after exercise.
- **Exfoliation glaucoma (XFG)**, also called pseudoexfoliation glaucoma, which is caused by an abnormal accumulation of proteins in the drainage system.
- **Secondary glaucoma**, which is a chronic form of OAG that may occur after an eye injury, infection, inflammation, a tumor or an enlargement of the lens due to a cataract.
- **Congenital glaucoma**, which is an inherited form of glaucoma present at birth. Eighty percents of the cases are diagnosed by the age of one by children who are born with narrow angles or other defects in the drainage system of the eye.

Main narrow angle glaucoma varieties include:

- **Acute angle closure glaucoma (ACG)**, which causes sudden symptoms like eye pain, headaches, light halos, dilated pupils, vision loss, red eyes, nausea and vomiting. Each of these attacks can yield progressive vision loss.
- **Neovascular glaucoma**, which is caused by an abnormal formation of blood vessels on the iris and over the drainage channels of the eye. This glaucoma is always associated with other pathologies, like diabetes.

Prevention and Risk Factors

As for many health problems, healthy and varied diet, active lifestyle with exercise, no smoking tend to reduce the risk of glaucoma by people. However, several factors seem to increase the risk of developing a glaucoma, notably the:

- **Ocular Hypertension (OHT)** – People with high IOP are at higher risk.
- **Age** – The risk increases as people are getting older.
- **Race/Ethnicity** – African-americans, east-asians or eskimos seem to be at higher risk.
- **Family** – People having a glaucomatous family member are at higher risk.
- **Others**, like high myopia, diabetes, or a central corneal thickness less than 0.5 mm.

Diagnosis

In order to establish a diagnosis, a comprehensive glaucoma exam usually involves the five following tests:

1. **Tonometry**, for measuring the Intraocular Pressure (IOP). A device called *tonometer* is used to measure the IOP by applying a small amount of pressure on the eye either with a tiny device or a warm puff of air. A normal IOP value is in the range 12-21 mmHg (millimeters of mercury).
2. **Ophthalmoscopy** (or dilated pupil test), for measuring the shape and color of the optic nerve.
3. **Perimetry** (or visual field test), for measuring the field of vision.
4. **Gonioscopy**, for measuring the angle between the iris and the cornea. A special lens with a mirror is used to see if the angle between the iris and cornea is closed and blocked, or wide and open.
5. **Pachymetry**, for measuring thickness of the cornea. A probe called *pachymeter* is put on the cornea to measure its thickness because it may influence IOP readings.

Treatment

The various treatments of glaucoma involve surgery (conventional or with lasers) and/or medication, depending on the seriousness. Medication with eye drops aimed at lowering the IOP are usually tried at first in order to control the disease.

4.3 State of the Art

Considering this case study, we focused this State of the Art on the general topics of visual health monitoring and machine learning for medical diagnosis, with a specific focus on IOP monitoring and glaucoma detection.

4.3.1 Visual Health Monitoring

Visual health monitoring is a branch of health monitoring which consists in observing physiological signals of the eye, such as the IOP. As it requires using non-invasive techniques and consequently intensive work and resources, only few researchers and companies have tackled the problem until now. The two main approaches are based either on *smart glasses* or *smart lens*. In order to perform visual health monitoring, smart glasses generally use cameras to capture limited physiological signals [Le et al., 2011], while smart lenses which are directly in contact with the eye can record more varied signals. Various types of sensors that have been embedded in a smart contact lens, notably to record:

- **Eye movements** – Movements of the eye were tracked using a wired contact lens sensor recording alternating magnetic fields in an former work [Robinson, 1963], and using a wireless contact lens embedding a magnet and circuit board with two magnetoresistive sensors in a recent work [Kim et al., 2004].

- **Tear glucose concentration** – Several contact lens have been developed to measure the glucose level in tears, the most famous being probably the one presented by Google and Novartis in 2014 [Senior, 2014]. A contact lens using boronic acid-containing fluorophores (BAFs) allows to monitor the glucose concentration through pH and polarity variations [Badugu et al., 2004]. A fluorescent contact lens equipped with a glucose sensor was successfully tested on five patients during a clinical trial. A glucose-sensing material called *photonic crystal* that can be embedded in a contact lens or directly implanted in the eye was proposed in [Alexeev et al., 2004]. Its color diffraction changes across the entire visible spectral region along with the glucose concentration, making glucose level readings easy. Some other recent works include [Chu et al., 2011; Yao et al., 2012, 2011].

- **Corneal temperature** – Phenomenons like tearing, inflammations, carotid artery diseases have effects on the temperature of the cornea [Mapstone, 1968]. Hence, a thermal-sensitive contact lens encapsulating liquid crystals that change color along with the temperature of the eye was reported in [Kinn and Tell, 1973].

- **Blood oxygen** – A contact lens probe measuring the pulse oximetry from the blood in the eye fundus is described in [Scott et al., 1996]. The light of an infrared and a red LED is emitted from the lens and focused toward the retina. Then a photo diode measures the response of this light stimulation and the pulse-oximetry signal is computed.

- **Intraocular Pressure (IOP)** – As explained in Section 4.2.2, IOP is usually measured *punctually* with a tonometer device. State-of-the-art ways to monitor and measure IOP *continuously* are depicted in the following Section 4.3.2.

4.3.2 Intraocular Pressure (IOP) Monitoring

Intraocular Pressure (IOP) monitoring is a branch of visual health monitoring which is often tied to the glaucoma disease. Considering dynamic character of IOP as well as the drawbacks of the tonometer based measure methods currently used by medical doctors, researchers have now turned to continuous IOP monitoring methods. Further information on current standard tonometer technologies can be found in [Resende et al., 2015].

IOP Monitoring Related to Glaucoma

As already explained, for years, an IOP value greater than 21 mmHg was considered as a glaucoma symptom by medical doctors [Tielsch et al., 1991]. However, several patients developed

the disease without showing a high IOP, and others never developed it although they had a high IOP. Nowadays, variations of IOP are commonly noticed by medical doctors through the standard DTC method (see Section 4.2.2), but they are still not well characterized and often neglected in the management of glaucoma patients [Singh and Sit, 2011].

IOP fluctuations of 4.5 mmHg in healthy subjects and substantially higher in some glaucoma patients have been reported [Liu et al., 1999, 2003]. Therefore a single IOP measure in the sitting position during normal office hours does not reflect the true range of a person's IOP, the IOP peak, nor the variation throughout the day [Barkana et al., 2006; Mosaed et al., 2005]. Studies in which IOP was measured several times over 24 hours have shown that about 2/3 of glaucoma patients had their IOP peak outside regular office hours, most frequently during sleep periods [Liu et al., 2003; Barkana et al., 2006; Hughes et al., 2003]. Other studies reported that IOP has a decreasing trend during the daytime with peaks occur during the morning [De Vivero et al., 1994; David et al., 1992]. One study reported that these findings led to an immediate change of therapy in 36% of glaucoma patients [Barkana et al., 2006]. It has been suggested that a sub-optimal approach to IOP assessment may account for nearly 1/3 of treated glaucoma patients showing progressive vision loss [Hattenhauer et al., 1998].

Another big limitation of the standard method is that IOP values are not captured during nighttime. Even if they can be obtained by hospitalisation or within a sleep laboratory, they are cumbersome, costly, and require awakening of patients during the sleep period, introducing potential stress-related artifacts [Liu and Weinreb, 2011]. Studies performed under the controlled environment of sleep laboratories have shown that the IOP of most glaucoma patients is highest during the nocturnal/sleep period with the patient in the supine body position (i.e. lying on his back) [Liu et al., 2003, 2010]. However, due to the unavailability of IOP-monitoring techniques that measure IOP during physiological sleep, little is known about the contribution of nocturnal IOP rhythms to the pathophysiology of glaucomatous damage [Weinreb and Liu, 2006].

IOP monitoring data has also been used by ophthalmologists for treating glaucomatous patients. A study showed how this information was important for the decision of treatment of 15 glaucomatous patients by comparing standard IOP measures with continuous monitoring for 24 hours using Sensimed Triggerfish® [Mansouri and Shaarawy, 2011]. 69% of the patients had their IOP peak during their sleep period and 80% had prolonged peaks. The usefulness of the continuous IOP monitoring data has been shown in practice, because for 73% of the patients the ophthalmologists changed the treatment, for example by changing the timing of a treatment, adding anti-glaucomatous drops or proposing alternative treatments as laser and surgery.

Regarding diagnosis potentials, a recent study has indicated that the conversion to glaucoma might be predicted by looking at specific dynamic patterns of the 24h circadian IOP profile in a population of OHT patients, notably at the awaking time [Grippo et al., 2013]. This could announce a breakthrough toward prognosis of the progression of the disease.

IOP Continous Monitoring

There are two categories of continuous IOP monitoring devices: (1) wired and (2) wireless [Katuri et al., 2008]. Wireless devices generally use an active or passive inductively powered sensor coupled with telemetry. Active devices are generally more robust, mostly in terms of communication, but more complex and intrusive. The fact that they often need to be implanted in the eye by a surgical operation constitutes their main drawback. Therefore this solution is often preferred for patients needing to replace their native lens, such as those suffering from cataract. Passive devices require a external antenna to power the sensor and communicate with it. Their major drawback is that the position of the sensor relatively to the antenna may have non negligible repercussions on the IOP measures.

Continuous monitoring of IOP can be performed either: (1) permanently, by means of an implantable sensor, or (2) temporarily, with a contact lens sensor [Collins, 1967; Resende et al., 2015]. As we used a wireless and non-intrusive device in this case study, we focused on those kinds of devices. Over the last years, several models of CLS able to measure the IOP continuously have seen the light:

- The **CLS Triggerfish®**, developed by the company Sensimed SA in Lausanne (Switzerland). This nonintrusive device capture the IOP-related variations by measuring the spontaneous circumferential changes at the corneoscleral area [Sensimed, 2014a,b]. More details are given in Section 4.4.1, as this device was used for this case study.

- The **DCT Lens**, developed by Ziemer Ophthalmology. This wired lens directly measures the IOP for a period of 24 hours [Ophthalmology, 2015b,a; Twa et al., 2010]. The wire constitutes the main

- The **thin film transistors**, developed at the ETH of Zurich (Switzerland). These thin-film transistors can be embedded in various objects to measure various physiological signals. In a recent project, they were embedded together with strain gauges in a contact lens to monitor the IOP, and a prototype has been successfully tested on an artificial eye. However, problems with energy supply and other electronic components still need to be fixed before commercialization [Salvatore et al., 2013].

- A **microfluidic pressure CLS** composed of polydimethylsiloxane (PDMS), a silicon based organic polymer used as structural element for the lens, and a dyed glycerol contained in a circular sensing chamber network within the lens. The fluid displaces

proportionally to the variations of IOP which can be determined by observation. The fact that this observation can only be done with eyelids open, and therefore not be during sleep periods, constitutes the main drawback of this device [Yan, 2011].

- A CLS embedding an LC resonator composed of an inductive coil (L) and a capacity (C). Variations of IOP lead to changes in the corneal curvature and in the resonance frequency of the LC circuit. These frequency variations are then recorded wirelessly. A prototype has been successfully tested on a silicon eye [Chen et al., 2013, 2014].

- A CLS embedding miniaturized capacity pressure sensors made of PDMS. These sensors are arranged in array within the lens in order to capture the whole pressure distribution, and thermal compression makes them take the desired lens form. That said, the read-out circuitry is not presented. A prototype has been tested [Cong and Pan, 2009].

4.3.3 Disease Diagnosis with Machine Learning

Machine learning applied to biomedical data for disease diagnosis[2] is not a new topic. Various techniques were applied for the detection and diagnosis of numerous diseases, such as:

- **Alzheimer's disease** (AD) – A survey reports on various machine learning methods, such as ANN/MLP, Bagging, Decision Tree (DT), CANFIS and genetic algorithms, used to classify different stages of AD. Based on the obtained accuracies, different treatment strategies for mild, moderate and severe AD were elucidated [Sandhya et al., 2010].

- **Parkinson's disease** (PD) – A survey presenting a comparative analysis of various machine learning methods, such as SVM, ANN/MLP, and Bayesian Networks, in the task of PD diagnosis [Bhande and Raut, 2013]. Another recent survey reviewed several research works appealing to various machine learning classification methods, including mainly SVM and ANN, and Random Forest (RF), for the task of PD prediction [Bind et al., 2015].

- **Heart/cardiovascular diseases** (CVD) – A survey reports on machine learning methods including Naive Bayes, k-NN, SVM, ANN, DT, and genetic algorithms, as well as their applications in cancer prognosis [Cruz and Wishart, 2006]. Another survey of 47 articles published between 2005 and 2013 refers to machine learning approaches applied for the diagnosis of many categories of heart diseases [Gayathri and Jaisankar, 2013]. In particular, a consequent survey was done on machine learning approaches to electrocardiogram (ECG) classification for the diagnosis of CVD, evaluating them in terms of classifiers used (SVM, HMM, HMM, fuzzy and hybrid approaches), features extracted,

[2]The terms *diagnosis* and *detection* are often equally used in the literature, while the former rather apply for medical doctors and the latter for computer.

types of CVD and classification accuracy [Salem et al., 2009]. Other similar surveys include [Gnanasoundhari et al., 2014; Karpagachelvi, 2015]. Handling ECG signals and IOP profiles appeal to quite analog processing and machine learning techniques due to the time series nature of the data (see Chapter 2).

- **Skin diseases** – An evaluation of three machine learning algorithms, namely Naive Bayes, MLP and J48 DT, in the task of diagnosing Erythemato-Squamous Disease (ESD) on a data set of 366 instances was done in [Danjuma and Osofisan, 2015], reporting that the former performed the best, followed by the second. Another comparison with other classifiers, such as SVM and hybrid methods, was done on the same database in [Badrinath and Gopinath, 2014], reporting an accuracy within 90-99% for all methods. Other more general surveys include [Rambhajani et al., 2011; Barati et al., 2011].

- **Cancer** – A survey on breast cancer diagnosis presents various machine learning methods, such as ANN/MLP, SVM, or RVM (Relevance Vector Machine), successfully used to improve the accuracy of predicting breast cancer [Gayathri et al., 2013].

- **Diabetes** – Two surveys report on the various machine learning techniques used for the diagnosis of diabetes, with an emphasis on ANN and SVM algorithms [Agrawal and Dewangan, 2015; Bromuri et al., 2014; Shankaracharya et al., 2010].

- **Eye diseases** – A survey on computer aided diagnosis for ocular diseases reviews the databases and the traditional features used to automatically detect different ocular diseases including glaucoma [Zhang et al., 2014]. Concerning glaucoma, standard features are extracted from photographies, such as the vertical cup to disc ratio (CDR), the disc diameter, the peripapillary atrophy (PPA) or the notching. In the next section, we focus on glaucoma diagnosis with machine learning.

All these surveys show that the most frequent and efficient classification algorithms in the domain of disease diagnosis are SVM and ANN/MLP, although each disease has its proper features and indicators to be used as input for the algorithms. Beside this, a more general review focusing on several research advances done on detection, diagnosis and therapeutic monitoring of diseases, states that supervised discriminative models explicitly dealing with the bias-variance trade-off like SVM have shown to be efficient for the diagnosis of diseases in computational biology, where data are disparate and of high dimension [Sajda, 2006]. It also states that generative models, such as Bayesian networks, are generally well suited for the analysis of biomedical images and signals by allowing to directly model the uncertainty and variability inherent to biomedical data, in addition to provide classification, segmentation and compression methods. Other similar general surveys on disease diagnosis include [Sahu and Kumar Khare, 2014; Satyanandam et al., 2012; **?**].

4.3.4 Glaucoma Diagnosis with Machine Learning

Focusing on glaucoma diagnosis/detection with machine learning techniques, several works can be mentioned. In [Goldbaum et al., 2002], three classifiers, namely ANN/MLP, SVM and GMM, were trained and tested by cross-validation on the task of diagnosing glaucoma from standard automated perimetry[3] of 189 normal eyes and 156 glaucomatous eyes. GMM gave the best ROC AUC of 92%. In [Sample et al., 2002], both SVM and GMM classifiers were compared in the task of predicting development of abnormal fields at follow-up in ocular hypertensive eyes that had normal visual fields in baseline examination. A specificity of 96% was obtained for both classifiers on data from 94 eyes. In [Goldbaum et al., 2009], unsupervised (i.e. clustering) machine learning techniques were applied to identify patterns of glaucomatous visual field loss in sita fields automatically identified using variational bayesian-independent component analysis. The best model yielded three clusters corresponding to classes *normal field*, *mildly abnormal field* and *severely abnormal field*, with an average sensitivity of 88.9% for the two last. These results were quite good because the classification was based on the visual field whose loss is a consequence of the glaucoma, unlike the IOP whose rise can be a direct cause. Thus, glaucoma detection from IOP changes might lead to glaucoma prediction.

4.4 IOP-related Profile Database

This case study is part of a bigger project done in collaboration with the Swiss company Sensimed, which has developed Triggerfish®, a new type of contact lens sensor for IOP continuous monitoring (see Figure 4.2). Considering the preceding State of the Art and experts' opinion in the medical domain, this new CLS technology may fundamentally change the ways glaucoma is treated. In this Section, we give details on the Triggerfish device, the so-called IOP-related profiles obtained from the recorded data, and the acquisition campaigns.

4.4.1 Data Acquisition

We present here the acquisition device which was used to perform the data acquisition, the resulting CLS recordings, as well as the acquisition campaigns that were conducted over the world to obtain them.

Acquisition Device

Unlike a standard tonometer that only permits to get a few isolated measures of patients' IOP at the doctor's surgery, the CLS Triggerfish® allows a practical, reliable and convenient ambulatory continuous monitoring of IOP over 24 hours. The Triggerfish system is sensitive,

[3] *Perimetry* is obtained by measuring the visual field and allows to define the extent and progression of the glaucoma.

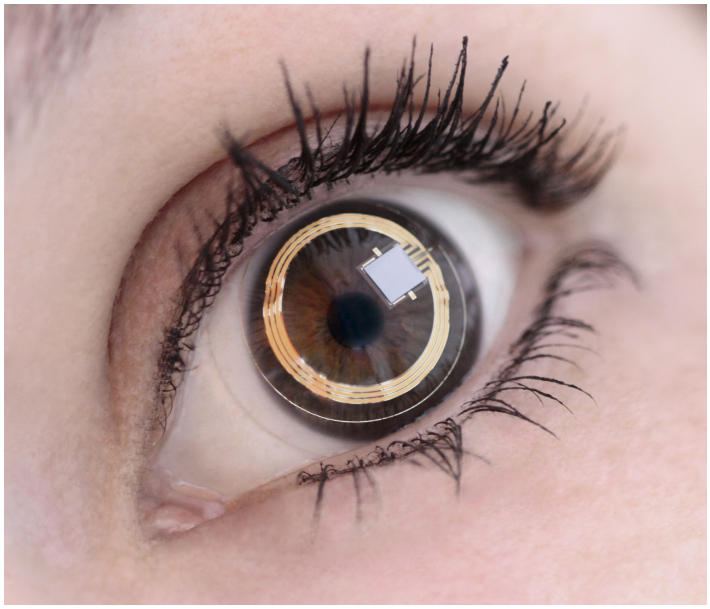


Figure 4.2: The contact lens sensor Triggerfish® set up in the eye [image used with permission of Sensimed SA].

safe and non-invasive, and works as follows. The patient wears the medical device up to 24 hours during which he keeps doing usual activities, including sleeping. Its installation and removal on the patient is simple and executed by a healthcare professional. The soft disposable silicone lens is highly oxygen-permeable soft and embeds a micro-sensor able to measure via two active sensing-resistive strain gauges the spontaneous circumferential changes in the area at the corneoscleral junction of the eye. Placed around the eye, an adhesive antenna receives the measure data from the CLS over the air and transmits them to a wearable recorder that stores them until the end of recording session (see Figure 4.3). After the session, the practitioner uses a dedicated software to retrieve the data from the recorder to his computer via Bluetooth. Finally, the data is anonymized, completed with various metadata, and sent to Sensimed via Internet for storage, analysis and modeling.

CLS Recordings

For technical reasons, the Triggerfish device performs the measures during continuous periods of 30 seconds every 5 minutes. These short acquisition periods are called *bursts* (see Figure 4.6). The CLS has a sampling frequency of 10 Hz, yielding 300 points per burst [M. et al., 2009; Mansouri and Weinreb, 2012]. The CLS recordings consist of multiple consecutive data entries containing different raw values, including the IOP-related measure, the timestamp, the burst ID, the ambiant temperature, the transmission quality and the battery level.

Focusing on the IOP-related measures, they are expressed in millivolts [mV] and are proportional to the circumferential deformation of the strain gauges (see Figure 4.4). A strong

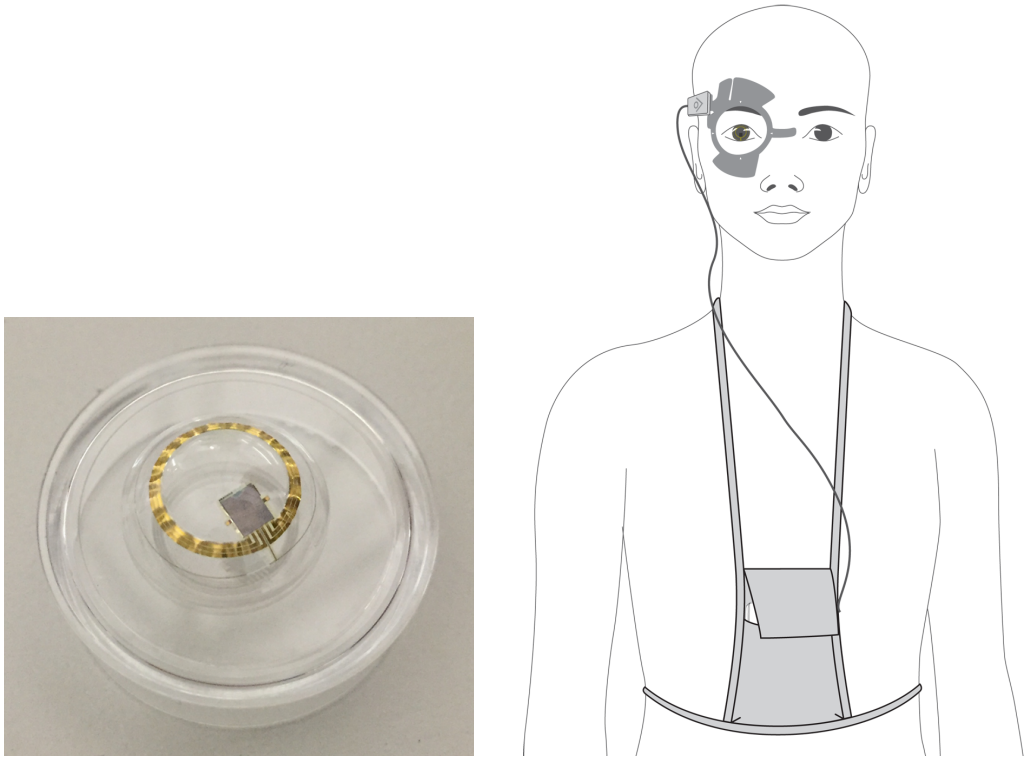


Figure 4.3: The Triggerfish® full equipment is composed of the contact lens sensor, an adhesive antenna and the wearable recorder [image used with permission of Sensimed SA].

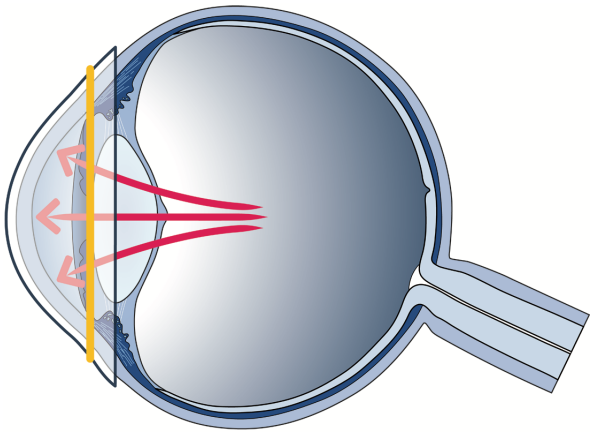


Figure 4.4: As IOP fluctuates, circumferential dimensional changes in the area of the corneoscleral junction of the eye are captured by the highly sensitive strain gauges of the CLS Triggerfish® [image used with permission of Sensimed SA].

correlation has been shown between direct IOP measures in common mmHg and measures in mV returned by the Triggerfish [Leonardi et al., 2004]. However, each CLS would need a particular calibration to be converted to mmHg. Each burst includes important and varied information, such as the eye blinks or the ocular pulse (see Figures 4.5 and 4.6). Patient's blinks produce brief peaks of IOP which are visible in bursts during wake periods. Similarly, the heart beating causes periodic fluctuations of IOP which are mostly visible in bursts during sleep periods, when there are no blinks. We developed algorithms to detect both blinks and ocular pulse (see Section 4.5.2). By taking the median of all points contained in each burst, we obtain a series of points uniformly spread on time (i.e. 5 minutes between each couple of points) and forming by definition a univariate time series. By plotting the curve of the so-called *IOP-related profile*[4], we can observe the general evolution of the varying IOP throughout the 24 hours (see Figure 4.7). By extracting other useful features from the raw measures of each burst, like frequencies and amplitudes of blinks and ocular pulse, we can add some new dimensions to the time series which becomes multivariate (see Section 4.5.2).

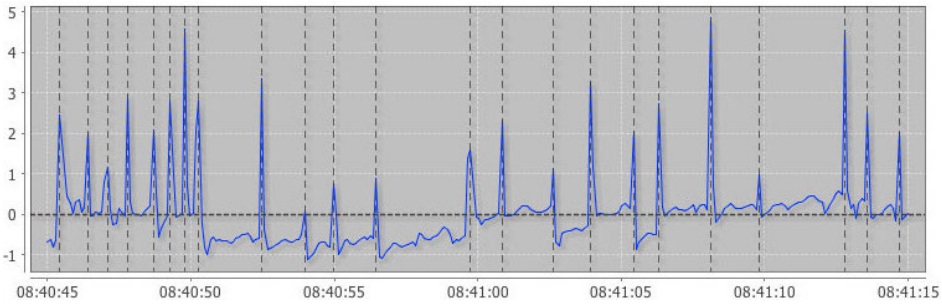


Figure 4.5: Eye blinks are visible during wake periods. In this burst, they were detected (see vertical dashed lines) by using our blink detection algorithm.

Acquisition Campaigns

The acquisition of the CLS recordings was supervised by Sensimed, the company producing the CLS used for this case study. Since 2011, medical doctors have been collecting CLS recordings from glaucomatous and healthy patients through clinical studies over the world. The anonymized recordings have been collected by Sensimed with the intention to do statistical analysis across populations, e.g to identify cohorts of profiles (patients sharing similar evolutions), and discover new glaucoma indicators, like IOP patterns, that will permit to perform a better detection of the disease. As the patented technology and the type of data are quite new, the quantity of exploitable data has been sufficient for statistical analysis and data mining experiments only recently. This has been notably emphasized by the fact that

[4]Rigorously speaking, we should always use the denomination of IOP-*related* profile as the CLS does not directly measure the IOP, but related values in mV. However, for the sake of conciseness of the prose, we used IOP profile indifferently.

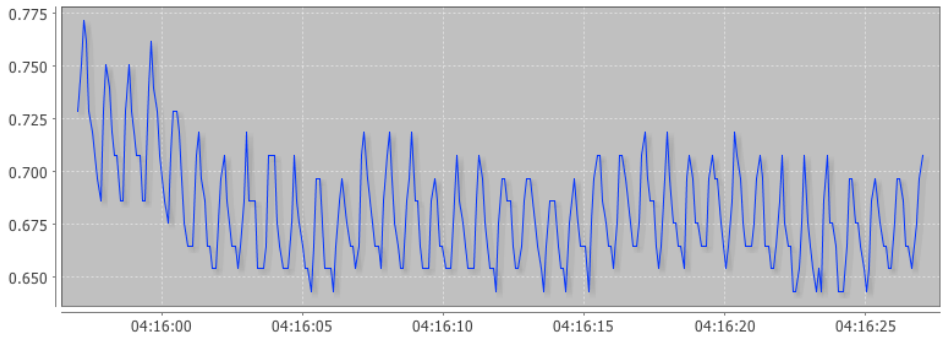


Figure 4.6: The ocular pulse, caused by the heart beating, is mostly visible during sleep periods in the IOP profiles (here a burst).

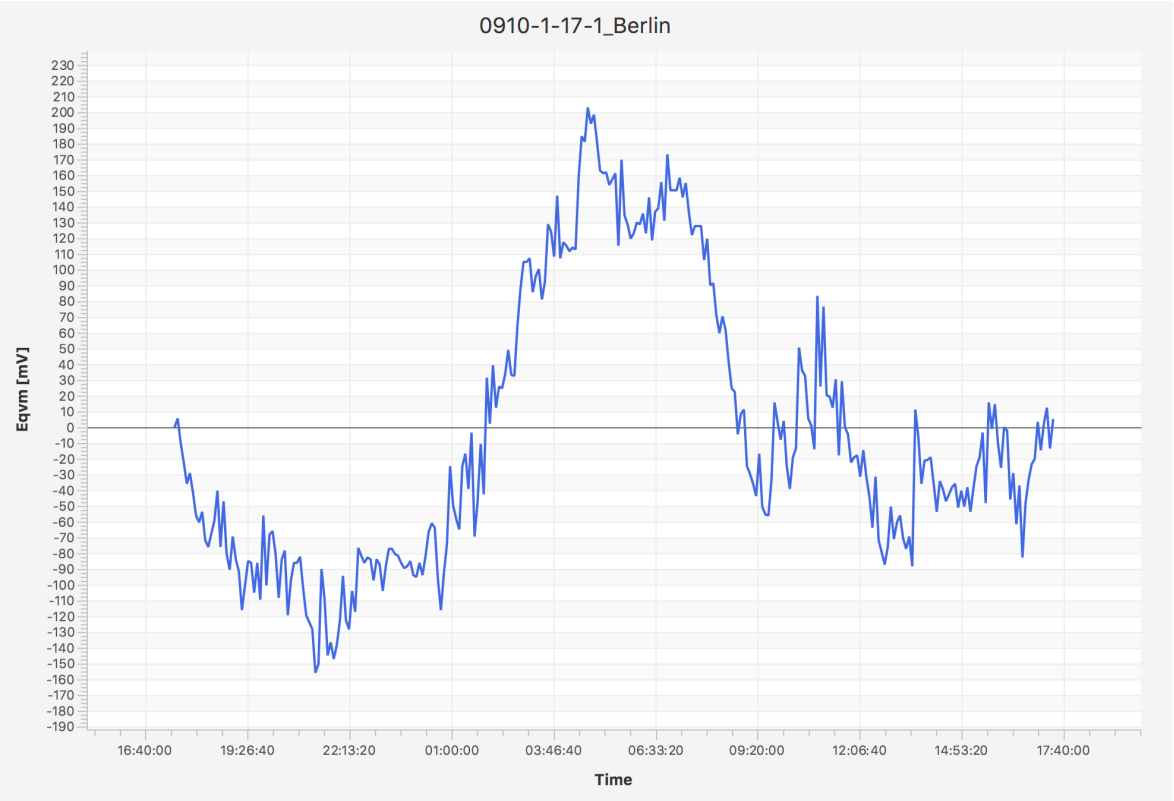


Figure 4.7: IOP-related profile of a POAG glaucomatous patient (here the Eqvm dimension). Such a profile is a time series whose points are the medians of all recorded 30-second bursts over 24 hours.

data from healthy persons are needed and primordial for the investigations and development of detection algorithms. Finding healthy persons consenting to participate in acquisition campaigns is not an easy task, even if we pay them.

4.4.2 Database Description

As a result of the worldwide acquisition campaigns, a large database of CLS recordings has been built by Sensimed. Each recording comes with meta-information on the patient, such as the health condition/diagnosis, the age, tonometer measures of IOP at the start and the ends of the recording session, and many others. All data are anonymized for privacy protection.

Database Contents

The part of the database that has been put at disposal for this case study contains 2568 CLS recordings acquired during 53 clinical studies from 2013 persons. Table 4.1 gives some statistics on the recorded persons.

Sex	**#**
Female (F)	1077
Male (M)	927
Unknown (UN)	9

Age (Range)	**#**
01 – 10 years old	7
11 – 20 years old	13
21 – 30 years old	83
31 – 40 years old	92
41 – 50 years old	195
51 – 60 years old	386
61 – 70 years old	586
71 – 80 years old	512
81 – 90 years old	124
91 – 93 years old	3

Diagnosis / Health Condition	**#**
Healthy (H)	334
Ocular hypertensive (OHT)	103
Primary open-angle glaucoma (POAG)	917
Normal-tension glaucoma (NTG)	241
Exfoliation glaucoma (XFG)	133
Other	285

Race (Code)	**#**
White (2106-3)	1598
Asian (2028-9)	263
Black or African American (2054-5)	110
Other (2131-1)	42

Table 4.1: Contents of the CLS recording database – Statistics on the recorded persons.

Data Formats

The database content is structured into three types of files, namely *lab*, *burst* and *json* files, which are stored in three corresponding directories.

A **CSV *lab* file** contains the raw data recorded by the CLS Triggerfish for one patient, i.e. all multivariate points of all bursts, including the IOP-related measures. It is the main file which the *burst* file was generated from. It was notably used to extract some useful features from the bursts, like the eye blinks and the ocular pulse. The principal features which are measured and stored in a *lab* file are described in Table 4.2.

Feature	Unit / Range	Description
Time	Second [s]	Time of measure (the session starts at 0)
Eqvm	Millivolt [mV]	IOP-related measure value (i.e. what really matters)
Temperature	Celsius Degree [°]	Ambiant temperature measured by the CLS
Pwm	[0, 127]	CLS power control value (maybe tied to eye movements)
Status	[0, 128]	If 0 → OK, else → error ID (point should be discarded)
BurstId	[1, 288*]	ID of the burst which the point belongs to

* There are at most 288 bursts in an exact 24-hour CLS recording

Table 4.2: List of features measured by the Triggerfish and stored in a *lab* file.

All other features are related to the recording device itself (battery level, current consumption, transmission/reception quality) and are not relevant for our work. The following listing extract shows how the format of a *lab* file looks like.

```
1 Time,Eqvm,Temperature,Type,Modok,Srect,Vrect,MibattRf,VbattRf,Vpwr,MibattIdle,
    VbattIdle,Pwm,Status,BurstId
2 0.016,-18.26262,23.065,1.0,8.0,0.115,9.926,0.0,3.955,1.621,0.0,4.043,38.0,0.0,1
3 0.117,-18.2459,23.065,1.0,7.0,0.115,9.926,0.0,3.955,1.621,0.0,4.043,37.0,0.0,1
4 0.216,-18.22721,23.065,1.0,8.0,0.115,9.926,0.0,3.955,1.621,0.0,4.043,38.0,0.0,1
5 0.316,-18.19279,23.065,1.0,6.0,0.115,9.926,0.0,3.955,1.621,0.0,4.043,37.0,0.0,1
6 0.416,-18.1859,23.065,1.0,8.0,0.115,9.926,0.0,3.955,1.621,0.0,4.043,38.0,0.0,1
7 0.515,-18.14557,23.065,1.0,7.0,0.115,9.926,0.0,3.955,1.621,0.0,4.043,37.0,0.0,1
8 ...
```

A **CSV *burst* file** contains the precomputed medians of the bursts and the related timestamps constituting an IOP-related profile. All other features also extracted from the bursts are included to burst medians entries (see Section 4.5.2). Together with the burst medians they form the multivariate time series that were primary used for our machine learning experiments. The principal features which were extracted and stored in such a file

are described in Table 4.3.

Feature	Unit / Range	Description
Time	Second [s]	Median time inside the given burst
Eqvm	Millivolt [mV]	Median IOP-related measure in the burst
BurstId	$[0, +\infty[$	ID of the input burst
Temperature	Celsius Degree [°]	Mean ambiant temperature in the burst
Min, Max, Amplitude Mean, Std Skewness, Kurtosis	Millivolt [mV]	Other statistical values computed from the IOP-related measures in the burst
PwmStd	[0, 127]	Standard deviation of the CLS power values
OPA	Millivolt [mV]	Ocular pulse amplitude in the burst*
OPF	Herz [Hz]	Ocular pulse frequency*
BlinkCount	[0, 100]	Number of blinks detected in the burst*
BlinkMeanAmplitude	Millivolt [mV]	Mean amplitude of the blinks in the burst*
BlinkMeanDuration	Second [s]	Mean duration of the blinks in the burst*
InterblinkMeanDuration	Second [s]	Mean duration between 2 successive blinks*

*More information in Section 4.5.2

Table 4.3: List of features extracted from the Triggerfish measures and stored in a *burst* file.

The following listing shows how the format of a *burst* file looks like.

```
1 Time,Eqvm,BurstId,Mean,Min,Max,Amplitude,Std,Skewness,Kurtosis,PwmStd,Temperature,
    OPA,OPF,BlinkCount,BlinkMeanAmplitude,BlinkMeanDuration,BlinkMeanInterDuration
2 0.3892131481,0.0,1,0.067606442953,5.640485,...
3 0.3926876852,31.13115,2,0.369735212014,...
4 0.3961575926,32.06557,3,0.349117058824,...
5 0.3996298148,40.72131,4,0.4116248,...
6 0.4031020486,31.52459,5,0.3754354947,...
7 ...
```

A **JSON metadata file** contains all the metadata, namely the information about the patient (ID, age, race, gender, medical diagnosis and history,...), the recording session (study, center, visit, start and end time, recorded eye, start and end tonometer IOP values in mmHg,...), the CLS (size and type), the recorder (version, firmware,...) and other features not relevant for our work. This file was also used to store all global features extracted from the whole IOP profile (i.e. the time series), such as all global statistics, the frequency filter banks or the sleep periods (see Section 4.5.2). The following listing extract shows how the format of a *json* file looks like.

```
{
  "Patient": "1500-CH001-01-L",
  "PatientAge": 76,
  "PatientBirthdate": "1939-01-01T00:00:00",
  "PatientCode": "M001",
  "PatientId": "1",
  "PatientGender": "F",
  "PatientRace": "2106-3",
  "Study": "1500",
  "Center": "CH001",
  "Visit": "1",
  "Diagnosis": "xfg",
  "Eye": "L",
  "LensSize": "m",
  "EqvmOffset": -18.247374999999998,
  "StartIOP": 23.0,
  "EndIOP": 0.0,
  "StartTime": 0.3892131481,
  "EndTime": 1.3753242708000002,
  "Duration": 23.666666944800003,
  ...
}
```


All files were denominated according to the following rule (see example below): (a) ID of the clinical study, (b) ID of the center, (c) patient's ID, (d) ID of the visit, i.e. the recording number, and (e) acquisition place. All parts are separated by a dash ('-').

$$\overbrace{1500}^{\text{Clinical study}}\text{-}\underbrace{\text{CH001}}_{\text{Center}}\text{-}\overbrace{01}^{\text{Patient}}\text{-}\underbrace{1}_{\text{Visit}}\_\overbrace{\text{Switzerland}}^{\text{Acquisition place}}\text{.lab}$$

4.5 IOP-related Profile Classification

In the literature, the term *classification* is often mixed up with *recognition*, *identification*, *detection* or *discrimination*. In fact, one must simply remember not to mix the final objective of the task with its object. In this case study, our final objective was to *recognise/identify/detect* the glaucoma disease by patients, and in order to perform this task, we *classified/discriminated* IOP profiles recorded from a CLS.

In this section, we present the experiments and results obtained in the task of classifying the IOP profiles of the Sensimed dedicated database, by adopting two generic approaches to time series data, namely a global (or wholistic) and a local approach. Various features were tested and tuned, including statistical features, or other SAX based features. Two classifiers

were tested, namely an SVM and a GMM, and two feature selection methods were applied, based on genetic and sequential selection algorithms. Parts of this research work and results were published in [Gisler et al., 2014] and [Gisler et al., 2015].

4.5.1 Data Preparation

according to the standard data mining process described in Section 2.3.3, a data preparation phase was required in order to exploit the raw data of the CLS recordings.

Data Selection

From an *attribute* point of view, the data selection was done at the level of the *lab* files, which contain the raw information recorded by the CLS, and from which the *burst* files used for our classification experiments were constructed. As already explained in the previous case study, data attributes (or features) should be relevant and not redundant for a machine learning task, such as classification. A feature is considered as relevant if it is correlated with the classes and redundant if it is correlated with the other features [Yu and Liu, 2003]. Thus, not all features recorded by the CLS were relevant for our classification tasks. In fact, the main attribute to keep was the Eqvm containing the IOP-related information of every burst. As already said in the previous section, other features related to the recording device and not to IOP, such as the battery level or the current consumption, are not relevant and were discarded.

From an *instance* point of view, the data selection was performed at the level of *burst* files (i.e. the profiles) mainly based on the information contained in the metadata *json* files. This selection depended directly on the classification task, that is the classification of healthy and glaucomatous profiles. Several selection criteria were discussed and defined with the company Sensimed. First, we chose to target the population of patients suffering from the most frequent variety of glaucoma (in both database and world), namely the POAG. This population is far bigger than the healthy one at our disposal in the database. Then, a minimum profile (i.e. time series) duration criterion of 22 hours was set. This criterion ensures that the selected time series contain nearly a whole day of signal and both awake and sleep periods that may contain valuable discriminating information. Another selection criterion was to discard profiles containing the so-called *jumps*. Jumps consist of abrupt shifts in a profile and may occur for example when a patient rubs his eye or for whatever reason causing a displacement of the CLS (see Figure 4.8). Hence, we developed a jump detection algorithm that works as follows: in a given profile, a jump is detected when the absolute difference between a point and each of its $m = 5$ consecutive points is greater than an empirically found threshold $T = 1$ [mV]. Finally, we added criteria to discard supernumerary profiles, that is for the few patients who provided more than one profile (in order to make our experiments independent of specific subject), and profiles whose patients already had surgery (as it might create bias).

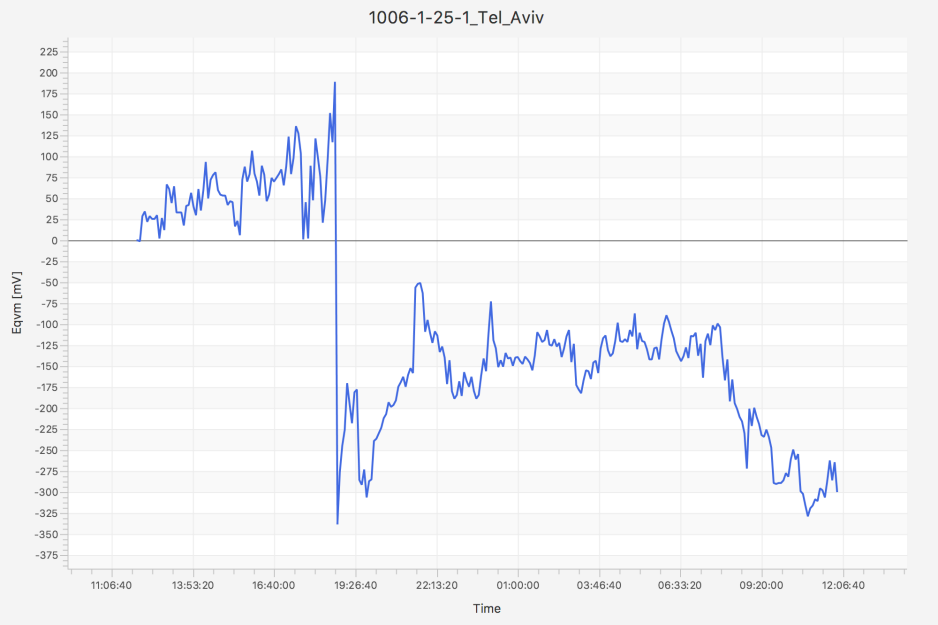


Figure 4.8: IOP-related profile containing a so-called *jump*, which consists of an abrupt shift in the signal caused a displacement of the CLS, occuring e.g. when a patient rubs his eye.

At the end of this data selection, a data set of 632 IOP-related profiles containing 270 healthy (H) and 362 glaucomatous (POAG) profiles was obtained. In order to deal with the fact that the two created sets are not balanced, we elaborated a specific test protocol (see next Section 4.5.4).

Data Construction

From an *operability* point of view, we had to shift every first Eqvm value to 0 and every following values accordingly, in order to deal with the fact that every CLS may have varying conception properties and perform different absolute measures depending on its positioning on the eye and the effective contact with the surface.

From an *efficiency* point of view, we first manually added some meta-information consisting of sleep period annotations to the profiles. These annotations were made by looking inside every burst of every *lab* file. During the nighttime, the IOP-related information is less likely to be biased by artefacts such as patients' activities or eye blinking, and specific information like the ocular pulse is more easily detectable. However, such manual annotations may not be trustable as they were provided by two persons and thus prone to errors. Then, the *burst* files constituting the *IOP profiles* were created by computing new features (attributes) from the raw values inside the bursts of the *lab* files, and adding them as new multivariate times series points (see Table 4.3 in preceding section). At this point, additional features, mostly

statistical ones, were computed from the different time series dimensions and used for the classification task (see next Section 4.5.2).

From an *impartiality* point of view, once all features were extracted and gathered in feature vectors, the latter were normalized according to the *Z-score standardization* formula (see Equation 2.2). This normalization process ensures most of the feature values to be centred around 0 and within $[-1, 1]$, as required by the classifiers that we used.

Data Cleaning

For various reasons, such as a technical problem or displacement caused by a patient rubbing his eye, the CLS may have some acquisition problems and parts of the recorded signal might be corrupted or missing in the primary *lab* files. Therefore, the created IOP profiles had to be carefully cleaned for the sake of our classification task.

First, we applied Hampel's outlier filter in all dimensions of the time series (i.e. the IOP profiles) in order to correct outliers (see Section 2.2.2), and then, we performed a linear interpolation in order to replace the missing values. A value may be missing because there were no or not enough values present in the corresponding burst in the original *lab* file.

4.5.2 Feature Extraction

As already explained, the glaucoma disease is still quite a mystery for medical doctors and IOP-related continuous data is quite new and has been never used before. Without having any a priori knowledge on the information potentially contained in the IOP profiles, we decided to adopt a generic data-driven approach to time series classification. This case study is a bit particular because there were two feature extractions, at two different levels:

1. In the *lab* files, from which we notably extracted some new specific physiological features (as a part of the previous step of the time series data construction);

2. In the *burst* files containing the created multivariate time series, from which we applied our generic approach and extracted time series generic features.

Features from Lab Files

The raw data recorded in a *lab* file do not correspond to a time series which we can directly work with. A *lab* file contains a succession of so-called *bursts*. Each burst corresponds to the data of 30 seconds of CLS signal, i.e. nearly 300 timestamped values, recorded each 5 minutes. Hence, the values of the time series formed by all bursts in a *lab* are not uniformly distributed along the time (there are "holes" of about 450 seconds between bursts). Therefore, from the *lab* files, we built the *burst* files containing multivariate time series which we could

work with. From each burst of a given *lab* file, we extracted some features constituting the multivariate points of these time series, and these points were uniformly distributed every 5 minutes (see Figure 4.7). Thus, at this level, this first feature extraction was only local, reducing every 30-seconds burst acquisition of a *lab* file to one multivariate time series point of a *burst* file. These local time series features (attributes) were of two types:

1. **Statistical features** on the Eqvm values (i.e. the IOP-related measures) of each burst, such as the Eqvm minimum, maximum, amplitude, mean, standard deviation (SD), skewness, and kurtosis (see Table 4.3 in preceding section);

2. **Physiological features**, which are certainly very interesting and important for medical doctors because they could provide them tangible and interpretable causes of the glaucoma. By analyzing the Eqvm signal of each burst, two main physiological phenomena can be identified and measured (see next section):

   (a) The *eye blinks*, which appear during wake periods. The blink density (or frequency), the blink mean amplitude, the blink mean duration, and the interblink mean duration were extracted;

   (b) The *ocular pulse*, which is particularly visible during sleep periods (when there are no blinks) and is the manifestation of the heart beating in the blood vessels of the eye. The ocular pulse amplitude (OPA) and frequency (OPF) were extracted.

3. Specific **device-based features**, like the standard deviation of the power of the CLS device (PwmStd). The Pwmstd was used because it was expected to provide information on eye movements.

Global and Local Features from Burst Files

At the level of the multivariate time series stored in the *burst* files, the second feature extraction was performed to build either global or local features, i.e. at the profile or burst level, depending on the approach chosen for the classification task. As presented in Section 3.5.2 for the previous case study, various local and global features could be extracted.

The following global features were extracted from IOP profiles for the global approach to time series classification:

- The mean, the median, the minimum, the maximum, the amplitude, the standard deviation (SD), the skewness, and the kurtosis of the Eqvm (median) series;

- The frequency filter banks computed with FFT on the Eqvm series (best found number of banks: 10, and overlap ratio: 0.1);

- The coefficients of two polynomial curves fitted on the Eqvm series (best found polynom degree: 4 and 20). Polynomial fitted curves allow to model the shape of each profile, and also provide indirect information on the slopes of the profiles;
- The cosinor rhythmometry of the Eqvm series. The cosinor is a linear regression method which allows to estimate the circadian rhythm of an IOP profile mathematically using a function of the form $y = a + b\ cos(w\ x) + c\ sin(w\ x)$ with parameters a, b and c and where $w = 24$ hours [Cornelissen, 2014; Resende et al., 2015].
- The SAX word frequencies of the Eqvm series, as described in the preceding case study (see Section 3.5.2).
- The mean of all skewness and kurtosis series;
- The mean of all blink density, blink amplitude, blink duration, and interblink duration series;
- The mean of all ocular pulse amplitude (OPA) and frequency (OPF) series;
- The mean of all CLS power standard deviation (PwmStd) series.

For the local approach to time series classification, we took the raw values y_i as well as the dynamic coefficients of 1^{st} order Δy_i and 2^{nd} order $\Delta\Delta y_i$ (as defined in Section 3.5.2 of previous case study) of the following series: Eqvm mean, minimum, maximum, amplitude, standard deviation (SD), skewness, kurtosis, PWM standard deviation, ocular pulse amplitude (OPA), and the blink density (frequency).

4.5.3 Specific Algorithms Developed

Two specific algorithms were elaborated in order to detect eye blinks and ocular pulse in the burst series of the *lab* files, and then compute the physiological features mentioned above.

Eye Blink Detection

Eye blinks cause brusque IOP elevations [Johnstone et al., 2011], and together with eyes movements, they are suspected to be related to glaucoma by potentially damaging the optic nerve [An et al., 2013]. The blink frequency varies and notably depends on patients' activities during wake periods. In this case, blinks are independent of the disease and can be considered as noise in the recorded IOP-related signal. Activities like reading or watching television generally yield a lower blink frequency. During sleep periods, blinks are obviously almost inexistent. The only blinks detected are in fact false positives coming from brusque signal variations which are suspected to be rapid eye movements (REM).

We proposed and used the following blink detection algorithm. As illustrated in Figure 4.9, a blink can be characterized as a sequence of three successive and continuous local extrema in the signal: first a local minimum (the start point), then a local maximum (the peak) and finally a local minimum again (the end point). All burst points are chronologically parsed. Each point may potentially be the peak of a new blink. A given candidate point can be the peak of a blink only if the sequences of nearby points toward the left and the right are strictly monotonically decreasing until reaching a local minimum, namely the blink start and end points. When such points have been found, the difference between the peak and the start as well as the end point is computed. When these differences are significantly greater than the mean amplitude of all bursts of the profile, a blink is detected. An example of result of our blink detection algorithm applied on a burst is given in Figure 4.5.

As soon as we detected blinks with this algorithm, we were able to perform various measures and statistics over the recorded patient profiles. The *blink density* (or *count* per burst) corresponds to the number of blinks occurring during a burst and is measured in blinks/burst. The *blink duration* is measured in seconds and corresponds to time difference between the start point and the end point of a blink. The so-called *interblink duration*, also measured in seconds, corresponds to the time difference between the end point of a blink and the start point of the following blink. Finally, the *blink amplitude* is defined as the difference between the Eqvm values of the peak and the mean of the start and end points of the blink in the signal. Blinks provide useful features but can also be considered as noise in the IOP signal variation during the wake time.

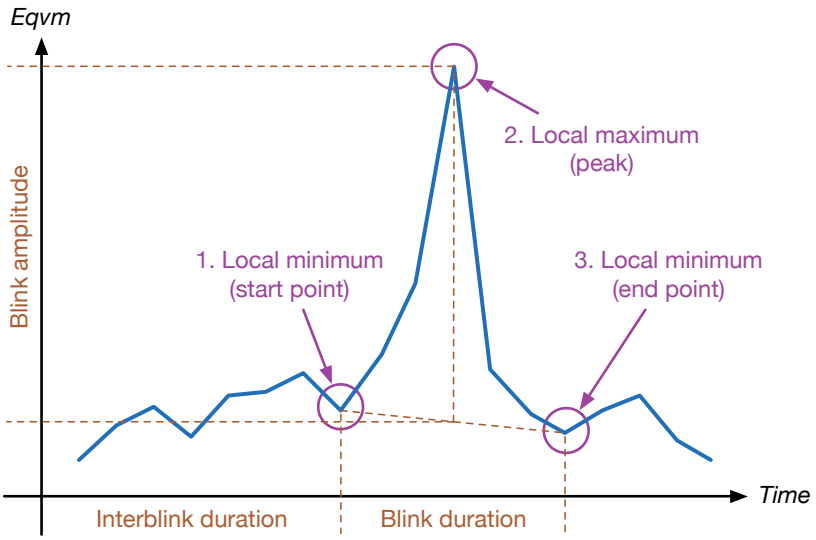


Figure 4.9: In a burst, a blink is defined by a first local minimum (start point), followed by a local maximum (peak) and a final local minimum (end point).

Ocular Pulse Detection

The ocular pulse *is generated by the pulsatile ocular blood flow in the choroid and depends on the dynamics of the cardiovascular system, the rigidity of the ocular vessels on one side and the biomechanical properties of the eye on the other side* [Galin et al., 1969]. Characterised by its amplitude and frequency, the ocular pulse can typically be observed in bursts during sleep periods (see example in Figure 4.6). The Ocular Pulse Amplitude (OPA), which corresponds to *the difference between the diastolic and the systolic intraocular pressure*, has been reported to be related to IOP and potentially some varieties of glaucoma [Schmidt et al., 1998; Schwenn et al., 2002; Punjabi et al., 2006; Stalmans et al., 2008].

We proposed and used the following ocular pulse detection algorithm. In a given burst of a *lab* file, a sliding window is shifted over each burst in order to identify IOP signal parts which are suitable enough to detect the ocular pulse. Such parts must contain no eye blinks and no holes (i.e. no missing values), which hide or degrade the ocular pulse in the signal. A sliding window length of 5 seconds and an overlap of 4.5 seconds result from a compromise between the quantity of suitable windows and their reliability for the ocular pulse computation. The longer the window is, the better the computation precision in both time and frequency domains will be, all while increasing the probability of finding noise. In every analysis window, the continuous component of the signal is removed to avoid side effects in the frequency analysis. Moreover, in order to increase the resolution in the frequency domain and be able to perform FFT, each window is zero-padded at the end to reach a greater power-of-two length [Instruments, 2006]. At this point, the Ocular Pulse Frequency (OPF) is searched by looking for a maximum f_{peak} in the frequency band of the heart rate, that is between $f_{min} = 0.7$ Hz (42 BPM) and $f_{max} = 1.85$ Hz (111 BPM). This procedure is similar to the one used, for instance, to find the heart rate frequency in photoplethysmographic signals [Kamshilin et al., 2011]. The f_{peak} found in the window corresponds to the ocular pulse whose amplitude is then retrieved (see Figure 4.10). With a CLS sampling frequency $f_s = 10$ Hz, the frequency domain spreads over 0 to $f_s/2 = 5$ Hz according to Shannon's theorem.

However, the signal may contain noise leading to unwanted frequencies within the searched frequency band. In that case, the maximum found is too weak relatively to its neighbourhood to ensure that the computed ocular pulse is reliable and significant enough. Thus, a quality factor QF was introduced to quantify the prominence strength of that maximum. Typically, QF should be high when the ocular pulse is clearly detectable and low otherwise. Let $\sum_{f=f_{peak}-\Delta f}^{f_{peak}+\Delta f} f$ be the sum of all frequency values within a $2\Delta f$ wide neighbourhood of the peak found, and $\sum_{f=0}^{f_{peak}-\Delta f} f + \sum_{f=f_{peak}+\Delta f}^{f_s/2} f$ the sum of all remaining frequency values within the frequency domain. A $\Delta f = 0.2$ Hz was empirically determined. Then, the QF

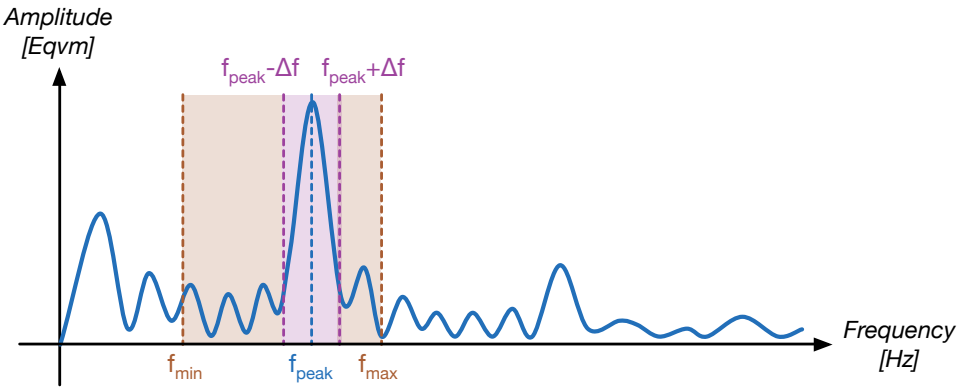


Figure 4.10: Detection of the ocular pulse in the frequency domain: a significant peak is searched in the heart beat frequency band.

value has been defined as:

$$QF = \frac{\sum_{f=f_{peak}-\Delta f}^{f_{peak}+\Delta f} f}{\left(\sum_{f=0}^{f_{peak}-\Delta f} f + \sum_{f=f_{peak}+\Delta f}^{f_s/2} f\right)^2} \tag{4.1}$$

If QF is greater than a given threshold T, the ocular pulse computed in the current window is retained. T was empirically found and set to 0.2, a relative small value allowing to compute the ocular pulse in more than 90% of the bursts. A 30 seconds burst yields $n = 55$ windows and ocular pulse values. If at least three of them have a suitable QF, the Ocular Pulse Frequency (OPF) of a given burst is computed as the weighted mean of the $f_{peak,i}$ values with their QF_i values ($0 < i < n$):

$$OPF = \frac{\sum_{i=1}^{n} f_{peak,i}\ QF_i}{\sum_{i=1}^{n} QF_i} \tag{4.2}$$

The OPF in Herz [Hz] can be easily converted in Beats Per Minute [BPM] by multiplying it by 60. In order to compute the Ocular Pulse Amplitude (OPA), a simple model is used to fit the data and represent the ocular pulse curve (see Figure 4.11). The following model takes the f_{peak} as input and depends on a constant term, a linear drift, and the two first f_{peak} harmonics whose parameters a, b, c, d, e and f must be estimated:

$$y = a + b\ x + c\ sin(2\pi\ f_{peak}\ x + d) + e\ sin(4\pi\ f_{peak}\ x + f) \tag{4.3}$$

For each valid QF, the model parameters are fitted and the model equation without drift is deduced as follows:

$$y' = \hat{c}\ sin(2\pi\ f_{peak}\ x + \hat{d}) + \hat{e}\ sin(4\pi\ f_{peak}\ x + \hat{f}) \tag{4.4}$$

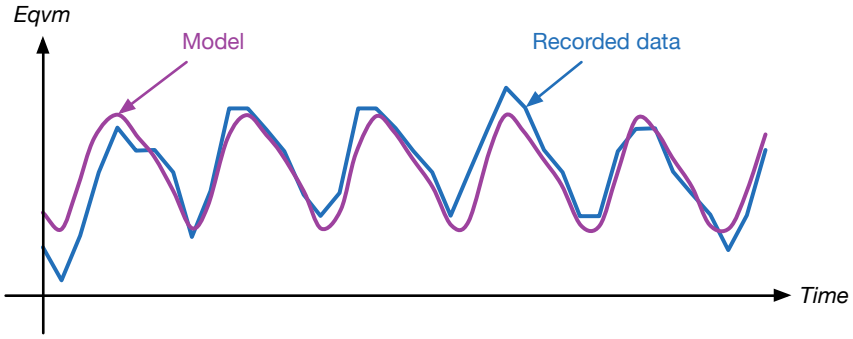


Figure 4.11: Detection of the ocular pulse in the time domain: the recored data is fitted by the estimated model used to compute the OPA.

Then, the ocular pulse amplitude opa_i of a given window is given by the difference between the maximum and the minimum of y' Finally and similarly to the OPF, the OPA of a given burst is computed as the weighted mean of the opa_i values with the QF values:

$$OPA = \frac{\sum_{i=1}^{n} opa_i \, QF_i}{\sum_{i=1}^{n} QF_i} \tag{4.5}$$

4.5.4 Machine Learning Test Protocol

In accordance with the model validation procedure described in Section 2.3.4, we had to determine a test protocol in order to validate and compare the different models obtained during our classification experiments. We decided to use the ROC AUC as performance score, as it is the most common metric used by both machine learning and medical communities in case of a binary statistical test (here a classification outcome). Because the two healthy (H) and glaucomatous (POAG) data sets are rather small and unbalanced (269 H and 363 POAG), we decided to perform to a repeated random subsampling cross-validation (see Section 2.3.4) with balanced training and test sets. This protocol, which has been discussed and validated by the company Sensimed, can be divided into 5 steps and is depicted in Figure 4.12:

1. **Initial situation** – Class 1 contains n_1 samples and class 2 contains n_2 samples (potentially $n_2 \neq n_1$);
2. **Training / test set splitting** – Let $f = 0.25$ be the wanted ratio for the test set size and $t = f \times minimum(n1, n2)$, then t samples are randomly picked from each class;
3. **Train set completion** – $n_2 - n_1$ samples are repeated within class 1 in order to balance the training set;
4. **Model fitting and evaluation** – A model is fitted on the training set and evaluated on the test set to provide the ROC AUC value which is stored.

5. **Process repetition** – The whole process is repeated $k = 100$ times to provide the distribution and average of the k ROC AUC values.

Contrary to a k-fold cross validation, this protocol has the advantage to maintain constant the number of samples in the train and test sets which are not dependent on the number of iterations.

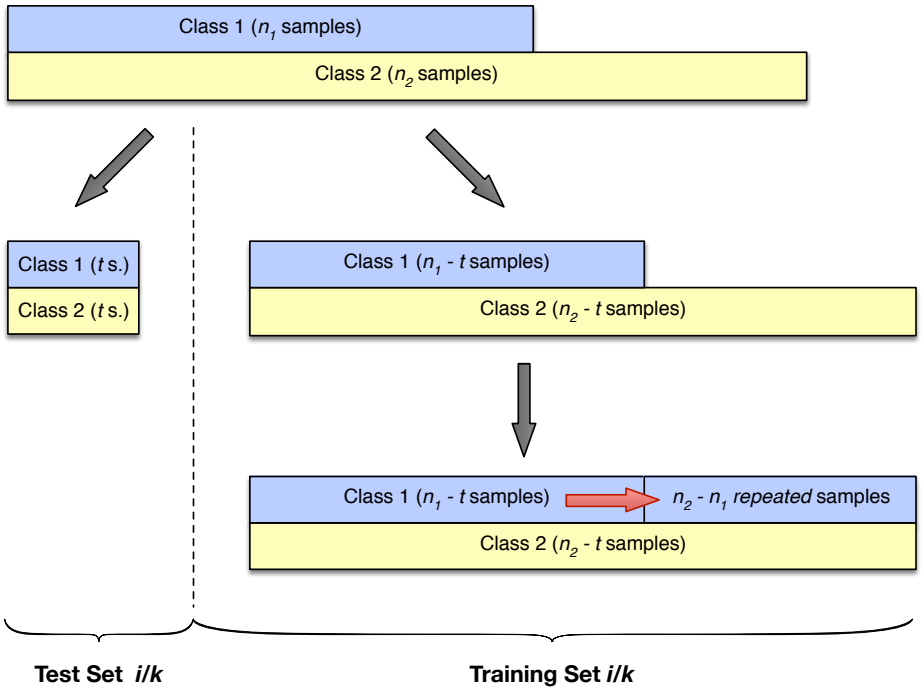


Figure 4.12: Training and test set splitting in one iteration of the test protocol with *repeated random subsampling cross-validation*.

4.5.5 Experiments and Results

As for the preceding case study, two types of experiments were realized following two different approaches to time series classification, namely a global and a local approach. In this Section, we present the experiments and results obtained in the task of glaucoma detection through IOP profile classification. More precisely, the task consisted in discriminating glaucomatous (POAG) versus healthy (H) profiles.

Global Approach with Genetic Feature Selection

As presented in Section 3.5.4, this global (or holistic) approach to time series classification, we built *global* feature vectors, meaning that one unique feature vector was extracted from

each multivariate time series. Such global feature vectors are built in a *horizontal* (or *longitudinal*) way, by extracting feature components one after the other on each dimension (i.e. univariate series) of interest, and then simply concatenating them (see Figure 4.13).

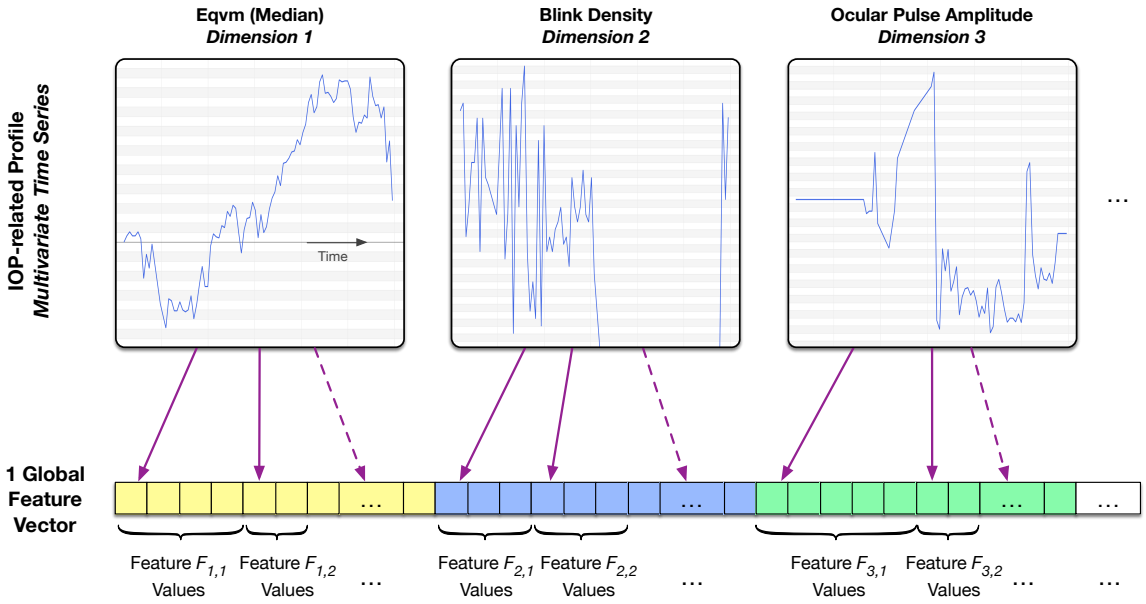


Figure 4.13: Global approach for IOP profile classification: one global feature vector extracted per multivariate time series. For example, feature $F_{1,1}$ can be filter banks extracted from the Eqvm series, and $F_{2,2}$ profile statistics (mean, max, min,...) extracted from the burst blink density series.

The features which were used are the global ones described in the preceding Section 4.5.2. Depending on the subsets of features which are tested during the feature selection, the classifier had to deal with a small quantity (one per multivariate time series) of feature vectors of medium dimension (between 2 and 51 components).

For this global approach, we decided to use SVM as classification algorithm (see Section 2.4.3)[5]. The nature of the classification and relative small quantity of data available led us to use SVM which are one the most recent and efficient discriminative modeling algorithm (see Section 2.4).

Taking into account the large number of features which were extracted, we decided to implement a Genetic Algorithm (GA) in order to perform the feature selection. Indeed, with $f = 17$ features, the number of feature combinations to test is equal to $2^f - 1 = 131'071$. If each feature combination took 4 minutes to be tested (and it takes much more time in reality,

[5]MLP are not adapted to the chosen test protocol with repeated random subsampling, because they much depend on the random initialization of their weights which may require several trainings per iteration.

given that the classifier specific parameters have to be tuned), it would take 1 year of computation to find the best solution, and this duration would be doubled each time we add a new feature. Therefore with the GA, we wanted to find a good, if not the best, subset of features that returns the best classification score (AUC in our case) in an acceptable computation time.

The idea of GA was first proposed in 1975 by J. H. Holland in the field of cellular automatons [Holland, 1975]. Since then, GA were applied in various fields of artificial intelligence including machine learning, and notably for the feature selection [Pei et al., 1995; Yang and Honavar, 1998; Mohamad et al., 2004]. GA are search (meta)heuristic belonging to the class of evolutionary algorithms which use mechanisms inspired by the natural selection of living organisms, such as selection, crossover, and mutation, in order to generate solutions to an optimization problem. Candidate solutions to the problem act as individuals in a population, and a fitness function is used to assess the quality of the generated solutions. Starting with an initial often randomly chosen population, the population evolves then by repeating the application of the above operators until a stop criterion is met, generally when the solutions are good enough or a maximal number of generations is reached.

The principle of our GA which is a *wrapper method* for feature selection (see 2.3.3) is the following. A candidate solution, that is a selection of features, is represented by a *chromosome* in the form of a binary array whose length is equal to the number of available features. When a given feature is included in the selection subset of features, the corresponding *gene* in the chromosome, i.e. the bit in the array, is coded by 1 and otherwise by 0 (see Figure 4.14).

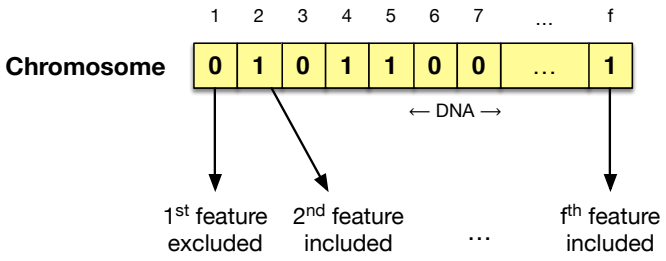


Figure 4.14: Genetic Algorithm for feature selection: a feature selection (chromosome) is represented with an array of bits.

First, we create the initial population by generating candidates according to the following procedure: select all features, then 1 feature over 2 with a shift from 0 to 1, then 1 feature over 3 with a shift from 0 to 2, then 1 feature over 4 with a shift from 0 to 3. Thanks to this procedure, the size of the initial population remains constant (here 10), whatever the number of available features is. Then the fitness function used to evaluate the solution consists of the

AUC score function applied on the classifier outputs. From this point, the next generation of solutions to test is created by repeating the three following steps:

1. **Selection**, which consists in selecting the k best candidates, that are the solutions whose AUC scores were the best. We compute the value of k as the square root of the number of features f, so that the size of the population does not explode along with the time whatever the number of available features is. The k selected candidates are called the *parents* in the next step.

2. **Crossover**, which consists in performing a crossover of the chromosomes of the parents taken 2 by 2 to generate two new children for each possible pair of parents. The most common types of crossover are the *single point crossover* and the *uniform crossover* that we used. The single point crossover proceeds by randomly choosing one crossover index and by swapping all genes (bits) of the two parents beyond that index to produce the two new children. Unlike the single point crossover, the uniform crossover uses a fixed mixing ratio and enables the swapping of the genes of the parents to occur at the level of each gene rather than the level of one segment of genes (see Figure 4.15).

3. **Mutation**, which consists in swapping one gene (bit) according to a low mutation probability p (generally $p \leq 0.05$) (see Figure 4.15). The mutation allows to introduce and preserve the diversity in populations. Moreover, it may avoid the GA to fall into local minima and stop the evolution by preventing the candidate solutions from becoming too similar.

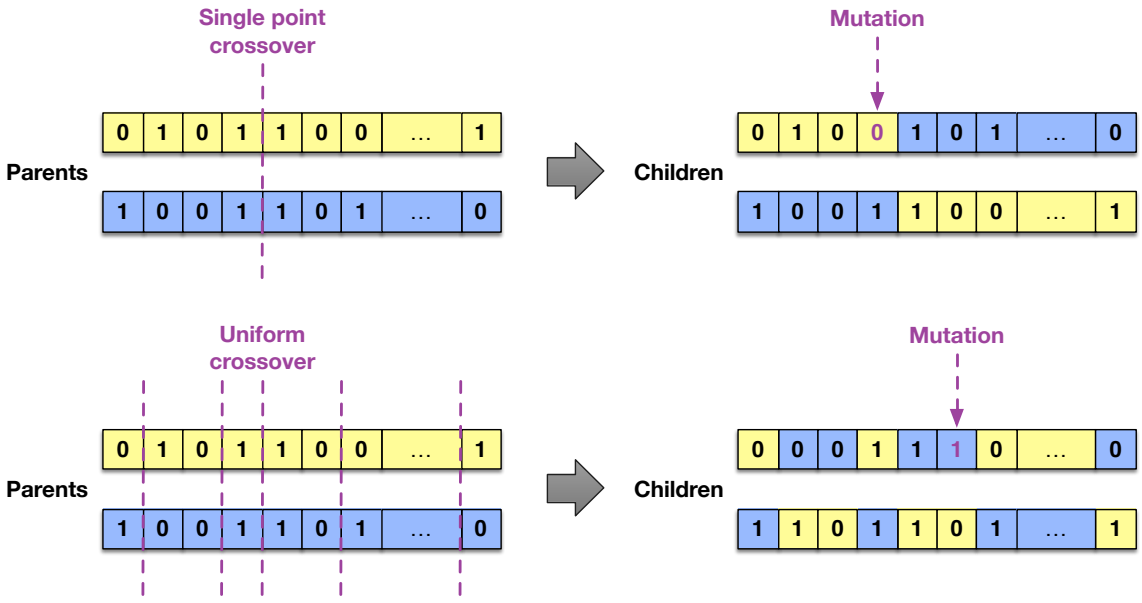


Figure 4.15: Genetic Algorithm for feature selection: principles of crossover and mutation.

Finally, we stop our GA when a maximal number of generations is reached and retain the best selection of features found.

We performed the feature selection by computing 7 generations of feature sets with our GA. For each feature set, we performed a tuning of the SVM parameters by doing a grid search, i.e. by varying the cost C from 1e−1 to 1e7 and the RBF kernel parameter γ from 1e−7 to 1e−1, by a factor of 10. The performance scores obtained by the 20 best feature sets for the task of glaucoma detection (or IOP profile classification) on the profiles selected from the Sensimed database (i.e. 270 H and 362 POAG profiles) is presented in Table 4.4. The results are given in terms of ROC AUC with their confidence interval ($CI_{95\%}$) computed according to Equation 2.55 of Chapter 2.

Hence, during the feature selection, the best ROC AUC score of $74.3 \pm 0.7\%$ was obtained with 13 features. The corresponding ROC curve is given in Figure 4.16. We observe that some features are systematically included in the feature sets providing the best results, namely the Eqvm mean, the Eqvm maximum, the Eqvm cosinor, the Eqvm FFT filter banks, the Eqvm fitted polynom coefficients of order 5, the blink density mean, the blink amplitude mean, the kurtosis mean, and the PwmStd mean. For each best feature set, we also observed that the best SVM parameter values stay rather constant and range from 1e4 to 1e6 for the cost C and from 1e−5 to 1e−4 for the RBF kernel γ. Moreover, it is interesting to see that, depending on the feature set, the runtime can be much faster, as shown e.g. by the two first results.

Finally, we wanted to mention that multiple tests were done with SAX word frequencies as features. Although various size of alphabets and length of the SAX representation were tested, the results obtained were not convincing, probably because of the large number of SAX words (more than 200 after filtering with a minimal frequency threshold of 25) which yield big and sparse feature vectors. Such vectors probably lead to the well known *curse of dimensionality* problem. Even if we tried to perform some PCA to avoid that problem, the best ROC of AUC $55.9 \pm 0.8\%$ was obtained without PCA, which is just a bit better than a random decision score. Other tests were also performed with the raw frequency magnitudes obtained with FFT (without computing filter banks) and with Haar wavelet coefficients obtained with FWT. The best ROC AUC of $61.1 \pm 0.8\%$ in the first case, and $56.6 \pm 0.8\%$ in the second case (with a PCA dimension reduction of 25%) were obtained.

Local Approach with Sequential Feature Selection

With this local approach to time series classification, we built *local* feature vectors, in the sense that multiple feature vectors were extracted from each multivariate time series. Such local feature vectors are built in a *vertical* (or *transversal*) way, by extracting one feature component after the other on all dimensions of interest in parallel, and then concatenating

Rank 20 first results obtained with genetic feature selection	**Classification Performance** ROC AUC [%] with $CI_{95\%}$	**Best SVM Parameter** Cost parameter C	**Best SVM Parameter** RBF kernel parameter γ	**Runtime [s]**	**Feature Vector Dimension**	**Number of Features**	**Eqvm Amplitude**	**Eqvm Cosinor**	**Eqvm FFT Filter Banks** Bank number: 10, overlap ratio: 0.1	**Eqvm Fitted Polynom** 4^{th} order coefficients	**Eqvm Fitted Polynom** 20^{th} order coefficients	**Eqvm Maximum**	**Blink Amplitude Mean**	**Blink Density Mean**	**Eqvm Mean**	**Kurtosis Mean**	**OPA Mean**	**PwmStd Mean**	**Skewness Mean**	**Eqvm Median**	**Eqvm Minimum**	**Eqvm Stand. Deviation**
1	**74.3 ± 0.7**	**1e5**	**1e-4**	**1462**	**28**	**13**	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓		✓
2	73.8 ± 0.7	1e5	1e-4	390	27	12	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓		
3	73.7 ± 0.7	1e5	1e-4	1848	25	10	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓				
4	73.4 ± 0.7	1e5	1e-4	340	26	11	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓		
5	73.2 ± 0.7	1e5	1e-4	265	25	10		✓	✓	✓		✓	✓	✓	✓	✓	✓		✓		
6	73.0 ± 0.7	1e5	1e-4	1779	25	10		✓	✓	✓		✓	✓	✓	✓	✓	✓	✓			
7	72.9 ± 0.7	1e6	1e-5	1419	26	11		✓	✓	✓		✓	✓	✓	✓	✓	✓			✓	
8	72.7 ± 0.7	1e5	1e-4	776	21	10	✓	✓	✓			✓	✓	✓	✓	✓	✓	✓	✓		
9	72.5 ± 0.8	1e5	1e-4	352	26	11		✓	✓	✓		✓	✓	✓	✓	✓	✓	✓			
10	72.4 ± 0.8	1e5	1e-4	523	28	13		✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓		
11	72.2 ± 0.8	1e5	1e-4	371	27	12		✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓	✓	
12	72.2 ± 0.8	1e1	1e-2	29	22	11	✓	✓	✓			✓	✓	✓	✓	✓	✓	✓	✓		
13	71.8 ± 0.8	1e4	1e-4	371	35	13	✓	✓			✓	✓	✓	✓	✓	✓	✓	✓	✓		✓
14	**70.7 ± 0.8**	**1e4**	**1e-4**	**720**	**51**	**16**	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
15	70.7 ± 0.8	1e6	1e-5	1783	49	14	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
16	70.3 ± 0.8	1e5	1e-4	444	46	11	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
17	70.2 ± 0.8	1e5	1e-4	268	25	10	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
18	70.0 ± 0.8	1e5	1e-4	456	22	7	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
19	69.9 ± 0.8	1e5	1e-4	238	24	9	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
20	69.7 ± 0.8	1e6	1e-4	1925	10	6	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Table 4.4: Global approach: 20 first results obtained for glaucoma detection (IOP profile classification) using an SVM classifier and a genetic algorithm for feature selection. Results with best found feature selection (rank 1) and all features (rank 14) are highlighted in blue.

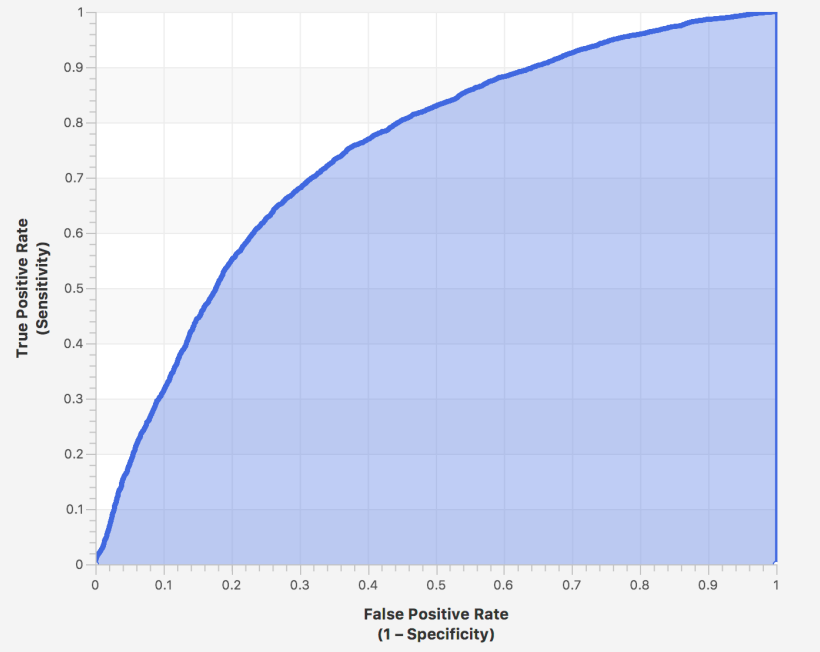


Figure 4.16: Global approach: ROC curve obtained for glaucoma detection (IOP profile classification).

them (see Figure 4.17). With this local approach, classification algorithm have to deal with a large quantity of feature vectors of relative small dimension for each multivariate time series.

The features which were used are the local ones described in the preceding Section 4.5.2. Depending on the subsets of features which are tested during the feature selection, the classifier had to deal with a small quantity (one per multivariate time series) of feature vectors of medium dimension (between 2 and 51 components).

We decided to use a GMM as classification algorithm (see Section 2.4.3)[6]. As we have seen in the preceding case study, this generative classification algorithm which returns probabilistic outputs may be the best suited for this local approach to time series classification. Moreover, its ability to deal with small feature vectors is well adapted to the feature selection methods which were chosen. Moreover, using such a generative modeling GMM as classifier allows us to simplify and improve our test protocol by suppressing the 3$^{\text{rd}}$ step consisting in completing the train set by repeating samples of smallest class in order to balance the

[6]MLP are not adapted to the test protocol, as explained in the previous experiment. SVM may have difficulty to deal with lots of small features vectors when a problem is hard like our, and the training algorithm may take a lot of time to converge to a solution (we observed that after trying) [Hsu et al., 2010, Section C]. In addition, as the local approach needs the classifier to return probabilistic outputs, what SVM were originally not conceived for, the training shall take even more time. SVM need an additional layer to convert the scores of all couples of classes into probabilities of classes [Chih-Chung and Chih-Jen, 2005, FAQ - Q06].

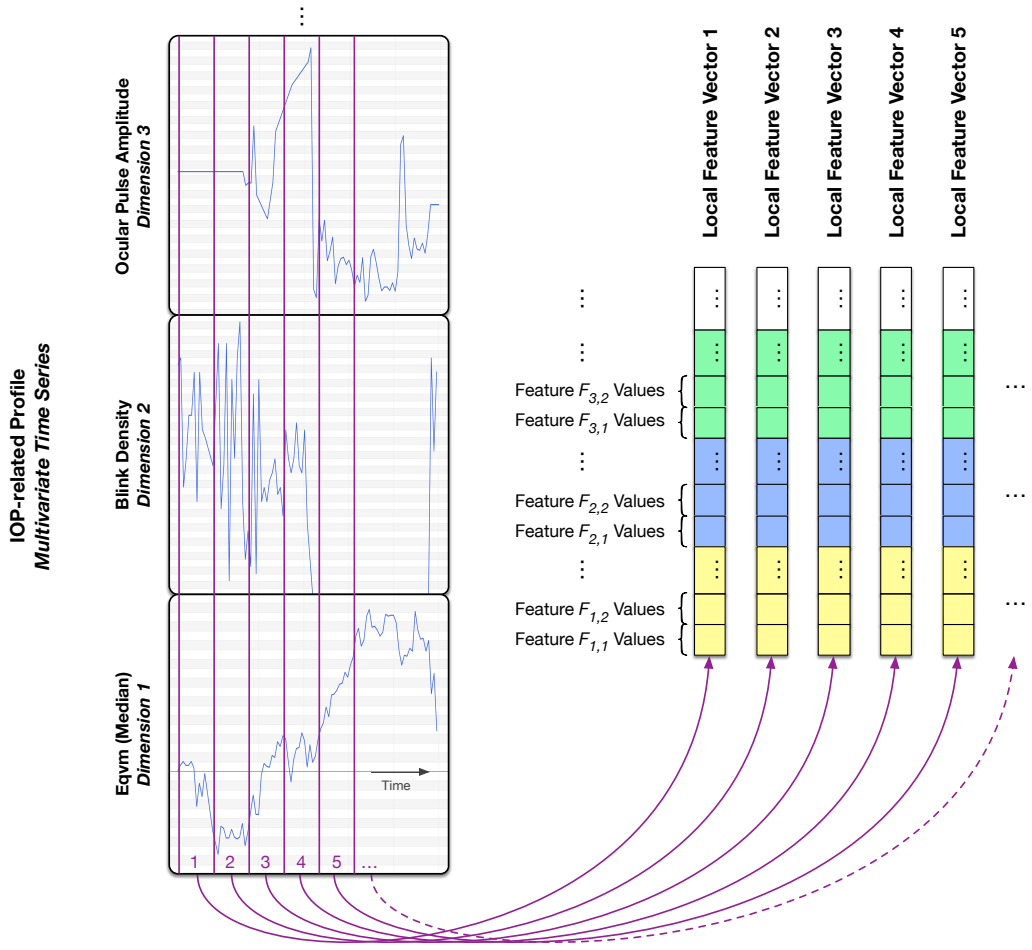


Figure 4.17: Local approach for IOP profile classification: several local feature vectors extracted per multivariate time series. For example, features $F_{1,1}$ to $F_{1,4}$ and $F_{2,1}$ to $F_{2,4}$ values can be local statistics (mean, min, max, standard deviation) extracted from the Eqvm and blink density series.

set. Indeed, this step is required by discriminative classification algorithms such as SVM, but not by GMM. Since H and POAG classes are not balanced and the original test protocol requires the test set to be balanced, the distributions of the two classes is modified in both training and test sets. Priors are equal in the test set and greater for the POAG class in the training set. Thus, the H class may have more difficulties to be well classified. Discriminative classifiers like SVM require the repetition of samples in the smallest class or the deletion of samples in the biggest class, which potentially changes the structure of the underlying information. Both operations have drawbacks: the first may repeat noisy samples, and the second may delete useful ones. Hence, the first operation is preferred to the second, mostly if the database is small. Generative algorithms such as GMM do not bias the outputs in favor of the biggest class, since the priors are taken in consideration by the model.

As for the preceding case study, once all feature vectors of a given time series have been classified, their classification scores still have to be "merged" in order to return a final class decision for the time series. This final global probabilistic score is computed for each class by multiplying[7] all local probabilistic scores of each corresponding class. Therefore, all local feature vectors are assumed to be independent (which is in the reality not true given that they are temporally tied). This final score for each class model corresponds in fact to the likelihood. According to the Bayes law (see Section 2.4.2), from these likelihoods and *a priori* probabilities of classes, *a posteriori* probabilities can be computed.

As for the preceding global approach, we performed a feature selection in order to increase the classification score. However, in this local approach, a single training iteration with some set of features could last several hours and using a GA would have taken too much runtime. Thus, we chose to perform a feature ranking to assess the classification potential of each feature. We opted for two sequential selection algorithms commonly used by machine learning scientists [Gutierrez-Osuna, 2015]. These algorithms which are *wrapper methods* for feature selection (see 2.3.3) work as follows. Let F the set of all available features, X the current set of features which have been already selected, and $J(X)$ an objective function, which in our case is the classification score (ROC AUC) obtained at the end of the test protocol, then:

1. **Sequential Backward Selection (SBS)** is a greedy search algorithm that begins with the whole set of features $X = F$ and then sequentially removes from X the feature x that yields the smallest decrease (if not the biggest increase) in the value returned by the objective function $J(X - x)$, where $-$ indicates the exclusion of x from X;

2. **Sequential Forward Selection (SFS)** is a greedy search algorithm that begins with an empty set of features $X = \emptyset$ and then sequentially adds to X the feature $x \in F - X$ that yields the biggest increase (if not the smallest decrease) in the value returned by the objective function $J(X + x)$, where $-$ indicates the inclusion of x in X.

Both Sequential Backward Selection (SBS) and Sequential Forward Selection (SFS) were applied along with our test protocol and a feature ranking was set by ordering the features according to their importance. As the obtained ranks might vary depending on the sample sets used in each repetition of the repeated random subsampling (RRSS) of the evaluation protocol, scores were given to the features according to their rank. With f features, f points were given to the best, $f - 1$ to the second, until 1 point to the last. At the end of the RRSS, the points obtained by each feature were added and a final rank was obtained. Finally, the feature selection was made by retaining the features which yielded the best mean ROC AUC.

[7] Practically, for the sake of calculation precision, the logarithms of the probabilities are added and the overall probability is computed by taking the exponential of the sum obtained.

First, we used the whole feature set and performed our classification tests by taking their (1) raw values y_i, (2) raw values and dynamic coefficients of 1st order Δy_i, and (3) raw values and dynamic coefficients of 1st order Δy_i and 2nd order $\Delta\Delta y_i$. For each test, we performed a tuning of the GMM parameters by varying the number of gaussians m from 0 to 40 by step of 4 (with 1 instead of 0 as first value), and the variance floor value ν from 1e−7 to 1e−1 by a factor of 10. The performance scores obtained by these three feature set configurations for the task of glaucoma detection (or IOP profile classification) on the profiles selected from the Sensimed database (i.e. 270 H and 362 POAG profiles) is presented in Table 4.5. The results are given with their confidence interval ($CI_{95\%}$) computed according to Equation 2.55.

Features: – Eqvm Minimum – Eqvm Maximum – Eqvm Amplitude – Eqvm Mean – Eqvm Standard Deviation – Eqvm Skewness – Eqvm Kurtosis – Blink Density – Ocular Pulse Amplitude – Pwm Standard Deviation	**Feature Vector Dimension**	**Best GMM Parameter** Number of gaussians m	**Best GMM Parameter** Variance floor value ν	**Classification Performance** ROC AUC [%] with $CI_{95\%}$
Raw values y_i	10	12	1e−6	70.3 ± 0.9
Raw values y_i Dynamic coefficients of 1st order Δy_i	20	36	1e−6	72.6 ± 0.8
Raw values y_i Dynamic coefficients of 1st order Δy_i Dynamic coefficients of 2nd order $\Delta\Delta y_i$	30	24	1e−6	72.7 ± 0.8

Table 4.5: Local approach: results obtained for glaucoma detection (IOP profile classification) by taking raw values y_i, dynamic coefficients of 1st order Δy_i or 2nd order $\Delta\Delta y_i$.

The best ROC AUC score of $72.7 \pm 0.8\%$ was obtained with all raw values y_i and dynamic coefficients of 1st order Δy_i and 2nd order $\Delta\Delta y_i$, a variance floor value $\nu = 1e{-}6$ and a number of gaussians $m = 24$. We kept these best GMM parameter values for the feature selection. We observed that adding the Δy_i made significantly increase the ROC AUC score, while adding the $\Delta\Delta y_i$ not, but it made decrease the needed number of gaussians. Hence, when varying the number of gaussians with the fixed $\nu = 1e{-}6$, the ROC AUC begins to stabilize from $m = 20$, reaching its maximum sooner by using the $\Delta\Delta y_i$ too.

Then, we performed the feature selection with the SBS and SFS methods. The feature

ranks obtained with each method separated and both methods combined, as well as the ROC AUC score obtained by using (1) all features and (2) the best features selected for the task of glaucoma detection on the selected profiles are presented in Table 4.6.

IOP profile Subset	270 H, 362 POAG				
Feature Selection	None	SBS+SFS Selection	SBS Rank	SFS Rank	SBS+SFS Rank
Features:	10	6			
– Eqvm Minimum	✓	✓	9	10	9
– Eqvm Maximum	✓		10	9	9
– Eqvm Amplitude	✓		2	2	2
– Eqvm Mean	✓	✓	8	8	8
– Eqvm Standard Deviation	✓		3	3	3
– Eqvm Skewness	✓	✓	5	4	4
– Eqvm Kurtosis	✓		6	6	6
– Burst Density	✓	✓	7	7	7
– Ocular Pulse Amplitude	✓	✓	4	5	4
– Pwm Standard Deviation	✓	✓	1	1	1
Feature Vector Dimension	10	6			
Best GMM Parameters:					
– Number of gaussians m	24	24			
– Variance floor value ν	1e−6	1e−6			
Classification Performance:					
– ROC AUC [%] with $CI_{95\%}$	72.7 ± 0.8	74.3 ± 0.8			

Table 4.6: Local approach: results obtained for glaucoma detection (IOP profile classification) before and after the feature selection with the Sequential Backward Selection and Sequential Forward Selection methods.

Hence, during the feature selection, the best ROC AUC score of $74.3 \pm 0.8\%$ was obtained with 6 features. Moreover, we observed that both SBS and SFS returned the same rank for 6 features and the 3 first features occupied the same rank, namely the Pwm standard deviation, the Eqvm amplitude and the Eqvm standard deviation. However, the difference between the classification scores obtained before and after the feature selection is not great enough to be considered as significant.

4.5.6 Discussion

Further to the presented results and the considerations already formulated in the specific sections, some more general considerations can be added.

Concerning the extracted features, it was interesting to see that the physiological ones provide useful discriminative information. Unlike other more generic statistical features, such physiological features may be better comprehensible by medical experts, in terms of which factors might explain the disease. Concerning the dataset and the test protocol, it would be better to have an independent test set in order to assess the performance of the models. The tuning of the classifier parameters and the feature selection being performed through the test protocol, the results are certainly a bit optimistic and the obtained models a bit overfitted. Concerning the global and local approaches to time series classification, it was also interesting to observe that, even if the approaches, features and classifiers used were different, similar classification performances were obtained in terms of ROC AUC. However, we would recommend the global approach which extracts and processes much less data (features vectors) and is much lighter to apply and faster at getting classification results, even if the time series become long and numerous.

In this case study, contrary to the preceding one, the data mining process was not conducted from the beginning to the end. Initially, the IOP profiles were not specifically acquired for the purpose of doing machine learning tasks, such as IOP profile classification for glaucoma detection. As explained in the Section 2.3, this often happens and it is part of the game to deal with this. Furthermore, at the end of the experiments, it is still difficult to answer the general and fundamental questions stated in Sections 2.3.1 and 2.3.2, namely:

1. Are the data quality and quantity sufficient? The data is rather noisy, contains jumps, and should be calibrated in order to convert the values from [mV] to [mmHg]. As we have seen, after the data selection phase, the obtained data sets are in all likelihood still too small to actually provide significant and reliable results. Therefore, we kept ourselves from interpreting the results too much. That said, maybe when more data will be available, new recent deep learning approaches shall be considered.

2. Are we really sure that the problem has chances to be solved? Is the data relevant to the problem? Is it valid? Can it be trusted? For instance, is the labelling (H, POAG,...) of the profiles correct? As already said, medical doctors and specialists make significant errors in their diagnosis and still have a lot of questions on the disease. How can we be sure that the IOP profiles contains sufficient information to detect the disease? After our experiments, one thing is sure: given the ROC AUC score of $74.3 \pm 0.8\%$ obtained, the profiles *do* contain valuable information for the task of glaucoma detection.

Hence, although the results are not tremendous, to a certain extent the developed models are able to detect the glaucoma disease in patients' IOP profiles. Thus, according to the company Sensimed, they might be already a significant indicator and help for medical doctors to make their diagnosis. As a final word, we warn that this discussion and some interpretations of the results should be taken carefully due the small size of the database (as we have seen, lots of profiles must be excluded for quality reasons). However, in the scope of this research work, the results are valid and coherent. I worked my colleague Antonio Ridi on this case study and notably on the feature extraction part. I performed the global approach experiments with SVM. A. Ridi performed the local approach experiments with GMM and also used HMM for which he obtained no significantly better ROC AUC score ($73.0 \pm 0.8\%$) than for GMM. His results as well as the common parts were reported in his PhD thesis [Ridi, 2016, Sections 4.4 and 4.5]. Our common published papers are listed at the end of this document.

4.6 Conclusion

In this case study, we reported on our research work done in the domain of the glaucoma detection, from the parent fields of visual health monitoring and disease diagnosis with machine learning. We exploited a set of IOP profiles acquired with the CLS Triggerfish® developed by the Swiss company Sensimed. This CLS allows to perform automated, practical, reliable and convenient ambulatory recording of continuous 24-hour IOP-related profiles, contrary to repeated tonometry commonly gathered in sleep laboratory. The IOP is the only controllable risk factor to stabilize patients and various therapeutic options exist to reduce it. Such IOP-related profiles being time series, the state-of-the-art features and representations described in Chapter 2 could be used. Specific features allowing a better detection of glaucoma were extracted, and machine learning approaches to time series data were applied to discriminate between healthy and glaucomatous patients, the final purpose being to embed the generated models into softwares dedicated to help medical doctors to establish glaucoma diagnosis and decide which treatment to give.

More precisely, as a first objective, we found out and extracted some specific physiological features besides common generic statistical features and we provided a way to visualize them (see Chapter 5). Statistical features consisted of the minimum, maximum, amplitude, mean, standard deviation, skewness, and kurtosis of IOP-related measures (Eqvm) of each burst, as well as the standard deviation of a device-based feature (PwmStd). Physiological features, which are more prone to provide tangible and explicable causes of the glaucoma to medicals, consisted of the measures of two physiological phenomena appearing on the IOP signal, namely the eye blinks, appearing during wake periods, and the ocular pulse, appearing particularly well during sleep periods (when there are no blinks) and being the manifestation of the heart beating in the blood vessel of the eye. Thus, two specific feature extraction algo-

rithms were elaborated in order to detect and measure these phenomena in the burst series. In the first case, the blink density, blink mean amplitude, blink mean duration, and interblink mean duration were extracted. In the second case, the ocular pulse amplitude (OPA) and frequency (OPF) were extracted.

As a second objective, we applied classification methodologies in order to build models of healthy (H) and glaucomatous (POAG) patients and detect the disease in their IOP profiles when measured by the CLS Triggerfish®. The time series nature of the IOP profiles made this case study a typical case of time series classification where a generic data-driven approach could be applied. The two profile classes being unbalanced, a test protocol based on a repeated random subsampling (RRSS) with 100 iterations was established in order to provide stable and reliable classification results. Two classification algorithms, namely an SVM and a GMM, were tested with two different approaches to time series classification: a global and a local one. The first global approach consisted in building one unique feature vector per IOP profile, by concatenating the global features extracted from several dimensions of the time series during the first step. The second approach consisted in taking the IOP profile points (consisting of features extracted from the so-called *bursts*) separately, and building as many feature vectors as points. A feature selection was performed, based on a genetic algorithm in the global approach, and two sequential selection algorithms in the local one. Both approaches provided similar results on the eligible selection of 270 H and 362 POAG profiles with a ROC AUC score of $74.3 \pm 0.8\%$. Thus, our preference goes to the global one which provides a lighter representation of the data, as well as a better efficiency in terms of computation time and scalability.

In conclusion, the generic approach to time series classification provided acceptable results in this case study. From a *data mining process* point of view, we didn't carry this case study out from the start to the end, the hardware and data acquisition parts being managed by the company Sensimed. With the final objective to build reliable models able to detect the glaucoma disease by patients efficiently, the CLS and the data acquisition should certainly still be improved to record raw data of better quality. The classification results firstly depend on this. Perpectives of this research work will push Sensimed to focus on the detection of others glaucoma varieties or the prediction of the disease evolution, while improving the obtained models as more data are recorded and available.

This research was partially funded by the Swiss Commission for Technology and Innovation CTI under the grants 13406.1 PFFLE-LS (*Sensimed Diagnosis – Advanced signal analysis technologies for the early detection of glaucoma*) and 17325.1 PFLS-LS (*Glaucoma Prognostication Platform*).

5

A Tool for Generic Time Series Classification

A clever person solves a problem. A wise person avoids it.

Albert Einstein

Contents

5.1 Introduction

5.1.1 Context

The two preceding case studies got us to realize that even if the domains of application were totally different, the underlying problems were the same: a classification of time series. Time series differ to other data from several perspectives. They can be very numerous, long and have a high dimensionality. Common data mining and machine learning approaches don't work well on time series because of their unique structure. Practical problems of memory

and computation can occur, as well as others more abstract, like the *curse of dimensionality*. During the data mining process of our case studies, all inherent problems, small or big, were just the same, from the data comprehension to the evaluation of the generated models. The need to easily manipulate time series, visualize them, annotate them, extract specific high-level representations and features, visualize and tune these features, test different classification approaches, like the global and local ones, and visualize the classify outputs, pushed us to unify the different processing scheme and develop the Generic Time Series Classification Tool (GTSCT).

The desire of visualizing data graphically is natural and fundamental for humans, and particularly for data scientists (see Section 2.3.2). "There is no excuse for failing to plot and look" [Tukey, 1977]. Even if some programming languages, like Python, propose more and more useful tools and libraries, like *Scikit-learn*, to handle various data types and mining problems, coupled with the difficulties to get to know these languages, all their functionalities and options, the difficulties to get to know a specific data type and mining problem, imply that it is hard to know what to do, search, see, test, or tune at the right moment and correctly. Moreover, the multitude of experiments and results obtained often come with numerous programming, setting and data files, difficult to read, compare or run again after a certain time.

Although time series are present and collected everywhere, although researchers needing to perform data mining and machine learning tasks on time series are getting more and more numerous, there is currently no specific tool for generic time series classification and covering the whole time series machine learning process chain. More specific to the feature extraction step, we also do not see any unified tool for time series visualization and classification based on state-of-the-art time series representations, such as SAX and PAA, and other more common features, such as sliding window based statistics.

5.1.2 Objectives

In this chapter, we present our development called *Generic Time Series Classification Tool (GTSCT)*. This tool aims at performing generic time series classification tasks respecting the data mining process described in Section 2.3. We used the Generic Time Series Classification Tool (GTSCT) tool to make all the experiments of the two case studies presented in Chapters 3 and 4.

More precisely, our first objective was the creation of the GTSCT, a JavaFX interactive and easy to use application which will notably able to import time series in various file formats, visualize and manipulate them, handle outliers and missing values, perform sample selection, create and visualize high-level representations like SAX, create and visualize various

features, perform time series classification with various settings, algorithms and approaches (e.g global and local), perform feature selection, generate formatted results for reporting, export models, reload experiments and data, reproduce results, avoid user's methodological errors, and respect the data mining process.

Our second objective was the use of the GTSCT tool in the real contexts of our two case studies in order to show that two different problems of time series classification could be solved with a generic approach.

5.1.3 Outline

This chapter is organized as follows. First, in Section 5.3, we explain what is a generic time series data mining approach. Then, in Section 5.3, we present a brief State of the Art of (1) general data mining and machine learning tools, and (2) specific time series visualization and mining tools. In Section 5.4, we describe the main functionalities and give an overview of the Generic Time Series Classification Tool (GTSCT). Finally, in Section 5.5, we conclude.

5.2 Toward a Generic Time Series Data Mining Approach

During the last years, many research works were done in the domain of time series data mining, rather in the field of prediction (forecasting) than the one of classification. Various algorithms appeared in the domains of machine learning, signal processing, information retrieval, and statistics. The purpose is here not to make their survey, which would be too consequent. However, there are differences between common data and time series mining problems which deserve to be detailed.

As explained in [Ratanamahatana et al., 2010, Chapters 3 and 4], time series may be very big and their databases have huge sizes. All inherent data mining tasks and algorithms have to deal with storage, memory, computation, and scalability problems. Furthermore, most common mining and machine learning algorithms do not work well on time series because of their unique data structure. Time series may have a high dimensionality, a high feature correlation, and a large amount of noise [Chakrabarti et al., 2002]. Problems with high dimensional data go beyond computation time considerations. The meanings of expressions like "similar to" or "cluster forming" which are usually intuitive become confusing in high dimensional spaces. As the dimensionality increases, the objects become sparse and equidistant to each other, and classification (or clustering) loses its sense. This surprising phenomenon called the *curse of dimensionality* has led to intensive research [Aggarwal et al., 2001].

In order to perform efficient time series data mining, the key idea to understand is that, although the actual time series dimensionality may be high, the intrinsic and significant di-

mensionality is usually much lower. Hence, most machine learning algorithms dedicated to time series avoid working on the original raw data and consider some high-level representation of the data. It may seem contradictory that, after all the work to acquire and store all these precise time series values, we abandon them for some high-level approximation. There are two main reasons to do this. First, we are not interested in exact point values of time series, but in the trends, shapes and patterns that they contain and which may best be captured by some high-level representation. Then, the size of the time series and database may be too big to deal with. The passage to a lower dimensionality representation may permit more efficient storage, visualization, processing, and computation. Thus, there has been a big interest in finding such efficient approximating representations of time series.

The questions of preparation, representation, transformation and reduction of time series commonly used, notably for their classification, were already presented in Section 2.2[1]. The numerous research works on time series data mining have shown that no particular type of representation is the best for all problems. The efficiency of the time series data mining tasks (including classification) will depend on the suitable choice of representation (approximation) of the time series, as well as the quality of the representation generated.

Hence, from these observations we converged to a strategy for generic time series data mining composed of the three following steps:

1. Create a suitable representation of the original time series, which fits in memory (because time series may long and high dimensional), and extract features of interest;

2. Solve the given problem in main memory;

3. Make as less as possible accesses to the original time series on disk in order to validate or modify the solution obtained.

5.3 State of the Art

To the best of our knowledge, there currently exists no dedicated, open and scientific tool for time series visualization and classification, which notably integrates state-of-the-art high-level time series representations, such as SAX and PAA, and handles the entire time series data mining process. In this section, we make a brief survey of the existing general data mining and machine learning tools, as well as the more specific time series visualization and mining tools.

[1] We decided to include this parts in the Chapter 2 because most of the time series theory which is described was used for the two case studies in Chapters 3 and 4.

5.3.1 General Data Mining and Machine Learning Tools

The purpose here is not to describe in details, rather to give an overview of existing machine learning and data mining tools, frameworks and platforms, which are numerous and varied. Such tools are, for most, general and very complete. However, even if some include generic visualization modules and even some plugins to manipulate time series, they are not dedicated to time series. Due to a "too large" generic approach to data mining problems, they cannot entirely handle the (pre)processing steps and machine learning problems specific to the nature of time series. Even if they allow to handle the whole data mining process, they can not, at least not easily, take advantage of the links which are specific to time series and which can be made between all the phases of the data mining process. For example, they do not provide visual ways to tune the extracted features and link the classification outcomes to the raw imported time series. Moreover, some of them are so complete and complex that their modular graphical user interfaces (GUI) is more similar to a programming language. Such tools also expose users to risks of forgetting to do the right thing at the right time during data mining process. Below are given the most relevant general tools for data mining and machine learning (see Figures 5.1, 5.2 and 5.3):

- **Knime** (Konstanz Information Miner), which is a modular data exploration platform allowing to visually create data flows (*pipelines*), selectively execute some or all analysis steps, and analyze the results through interactive views on data and models. Written in Java, it offers additional plugins integrating methods for text, image, and time series mining [AG, 2011; Berthold et al., 2010].

- **RapidMiner** (previously YALE), which is an integrated environment for machine learning, data mining, and business analytics. It supports all steps of the data mining process including data preparation, results visualization, validation and optimization. Written in Java, the basic edition is free and the professional one starts at $2000. It has some functionalities for time series analysis and prediction [Klinkenberg et al., 2016].

- **Orange**, which is an open source data visualization and analysis tool for novices and experts, providing interactive workflows, and a large toolbox. Developed in Python and C, C++ at the University of Ljubljana, if also offers add-ons, but none for time series mining [Demsar et al., 2013].

- **Tanagra**, which is a free machine learning tool supporting several data mining tasks like visualization, descriptive statistics, sample selection, feature selection and construction, regression, factor analysis, clustering, and classification. It was developed for Windows only at the Lumière University Lyon 2 in France, for research and academic purposes. It currently has no functionalities for time series [Rakotomalala, 2013].

- **Weka** (Waikato Environment for Knowledge Analysis), which is a free software with a graphical user interface offering a set of visualization tools and algorithms for data

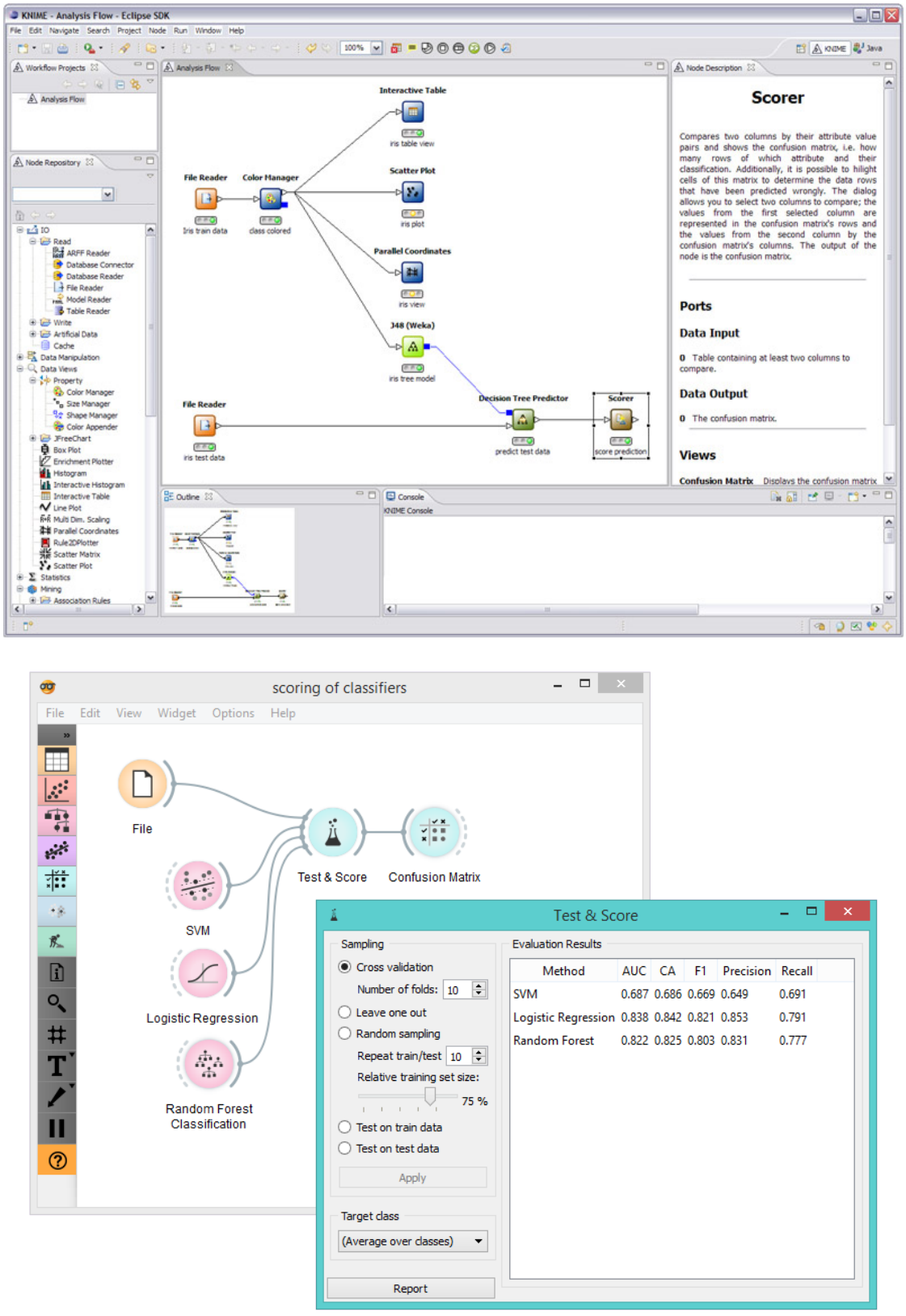


Figure 5.1: General tools for data mining and machine learning (part 1): *Knime* (top) and *Orange* (bottom).

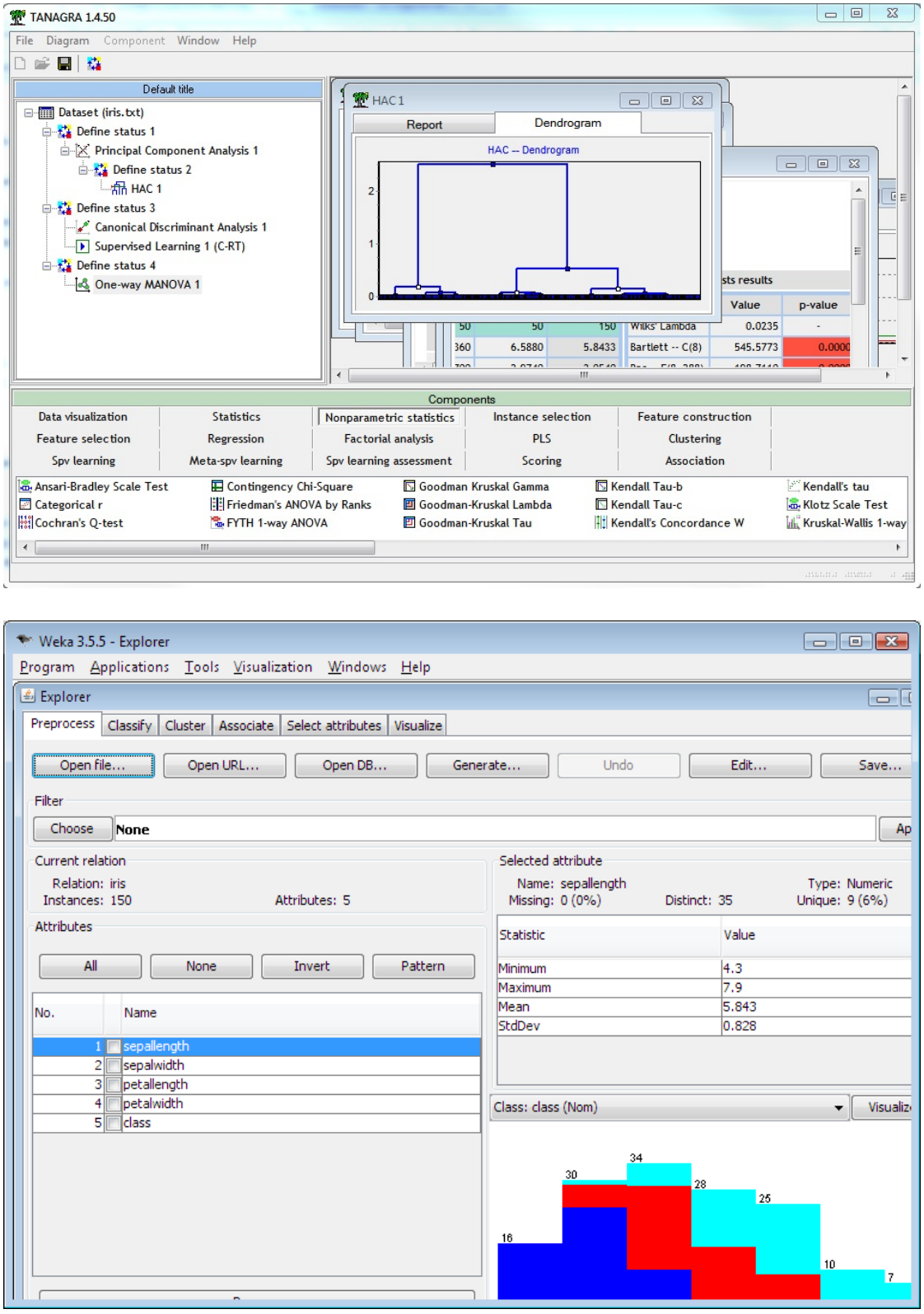


Figure 5.2: General tools for data mining and machine learning (part 2): *Tanagra* (top) and *Weka* (bottom).

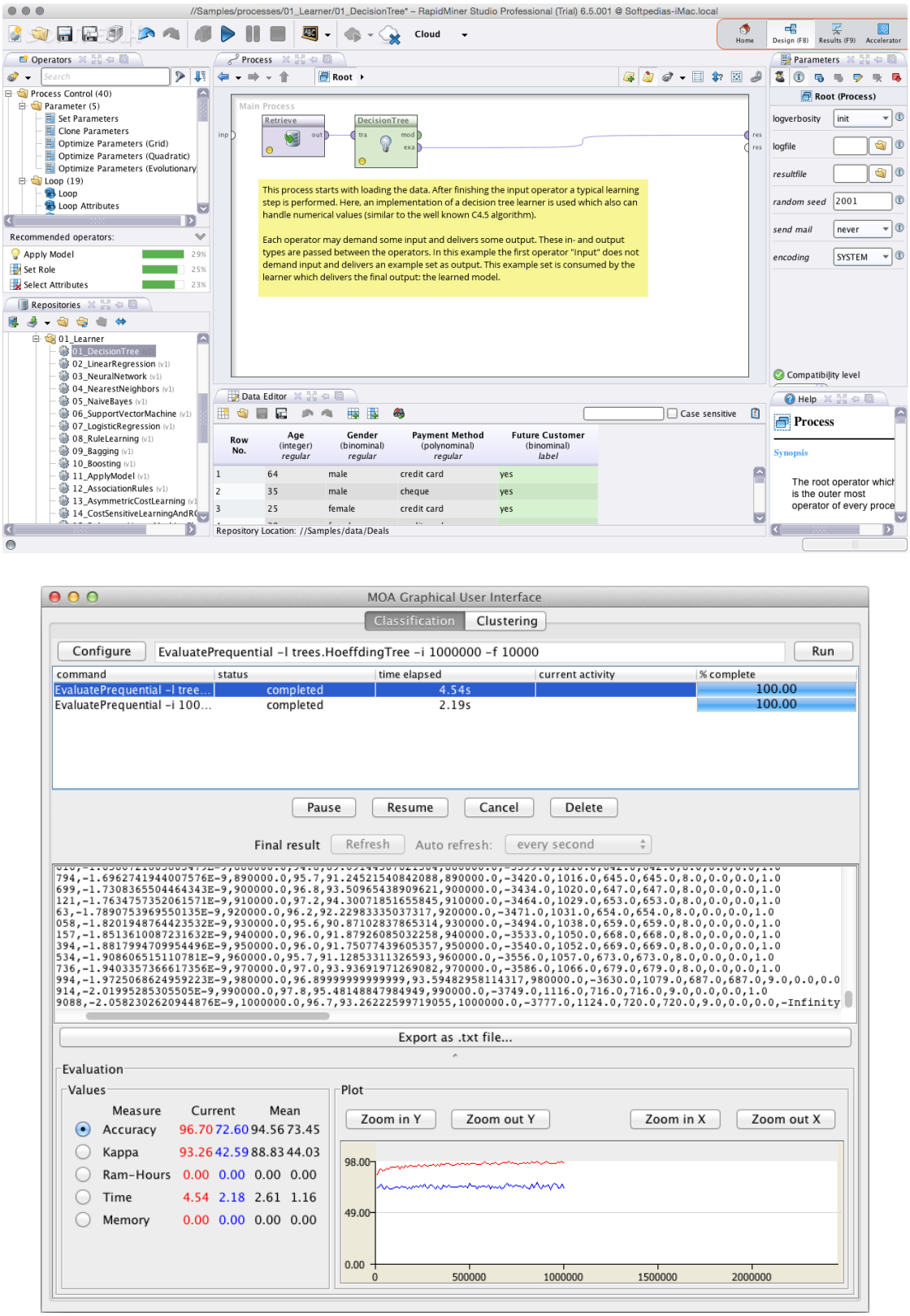


Figure 5.3: General tools for data mining and machine learning (part 3): *RapidMiner* (top) and *MOA* (bottom).

analysis and predictive modeling. Written in Java at the University of Waikato in New Zealand, it has also some functionalities for time series forecasting (prediction) [Witten et al., 2014].

- **MOA** (Massive Online Analysis), which is a open source software for machine learning and data mining. Developed in Java at the University of Waikato in New Zealand, it includes several learners and stream generators which can be used via a graphical user interface, a command-line, or an API. It currently has no specific functionalities for time series [Bifet et al., 2014].

There also exist some dedicated machine learning or time series libraries or module for several programming languages, such as Python (*Sci-kit* [P. and Knox, 2009] and *Pandas* [McKinney, 20xx]), Matlab [MathWorks, 2016], Mathematica [Research, 2014], or R [Hyndman, 2017], but as they are programming languages and not tools, we will not detail them here.

5.3.2 Specific Time Series Visualization and Mining Tools

There exist more than 100 methods for time-oriented data visualization which are all described in [Aigner et al., 2011]. The *TimeViz Browser* actually available at `http://survey.timeviz.net` consists of a visual survey of 115 visualization techniques and tools (with brief descriptions and references) for time-oriented data. Visualization and mining tools for time series represents a subset of them. However, most of them are only visualisation and browsing tool for time series, and more and more for massive databases of big time series, which is currently a big trend and a challenge. As the main purpose of our tool was more the generic approach to classification of time series than their visualization, we only mention here the most relevant time series mining tools, which have more functionalities than just visualization and navigation (see Figures 5.4, 5.5 and 5.6). Half of them are also described in [Maimon and Rokach, 2010] and the other in [Baggio, 2014]. No specific tool was found for generic time series classification.

- **TimeSearcher** is a query-by-example time series exploratory and visualization tool allowing to retrieve time series by creating *TimeBoxes* queries and to identify and find patterns in the dataset. Users may need some knowledge about the time series set in order to have a general idea of what is interesting to look for [Hochheiser and Shneiderman, 2004; Buono et al., 2005].

- **Cluster and Calendar-Based Visualization** is a visualization tool that cuts time series into sequences of day patterns which are clustered with a bottom-up algorithm. Found patterns are represented by cluster average, together with a calendar whose days are colored according to the clusters they belong to [Van Wijk and Van Selow, 1999].

- **Spiral** is a visualization tool that maps periodic parts of time series onto rings whose

attributes like color and thickness are used to characterize the time series values. Its main purpose to find periodic structures in the time series [Weber et al., 2001].

- **VizTree** is a time series pattern discovery and visualization tool based on augmenting suffix trees. The time series are approximated using a SAX representation (developed by the same authors) and codified into a suffix tree, where pattern frequencies and other characteristics are mapped using visual features like colors [Lin et al., 2005, 2007].

- **KronoMiner**, is a multipurpose time series exploration tool providing rich navigation and analysis abilities. The visualization is based on a hierarchical radial layout, allowing to drill into details by focusing on different pieces which can be manipulated for various tasks of time series exploration and analysis. It offers two analysis methods: *MagicAnalytics Lens*, which shows correlations between two parts of overlapped time series, and *Best Match Mode*, which displays an arch shape indicating the matching parts of two time series relatively to a given similarity measure [Zhao et al., 2011].

- **VizTool** is an interactive and explorative time series visualization tool offering functionalities to import/export data, share experiments, and perform data mining tasks such as motif discovery, correlation matrix computation and similarity computations. It exploits a hierarchy-based visualization in order to reduce the computation. The motif discovery algorithm can return the best n matches found at anytime [Baggio, 2014].

5.4 Generic Time Series Classification Tool (GTSCT)

Our contribution called Generic Time Series Classification Tool (GTSCT) is a JavaFX interactive and easy to use application for time series visualization, handling and classification. It notably integrates high-level time series representations and handles the whole data ming process (see Section 2.3). The tool was primarily dedicated to facilitate the experiments of the two case studies, by offering the possibilities to get, visualize, and reproduce results rapidly, without having to program or assembling numerous processing *nodes*. We also wanted to show that a generic approach to time series classification could be applied for different problems.

5.4.1 Main Functionalities

More precisely, GTSCT is a application whose main functionalities allow to:

- Import time series data and metadata in various file formats (XML, YAML, and CSV);

- Visualize time series (zoom in/out, and rescaling);

- Annotate time series (manually or automatically);

- Handle, detect and correct outliers and missing values in time series;

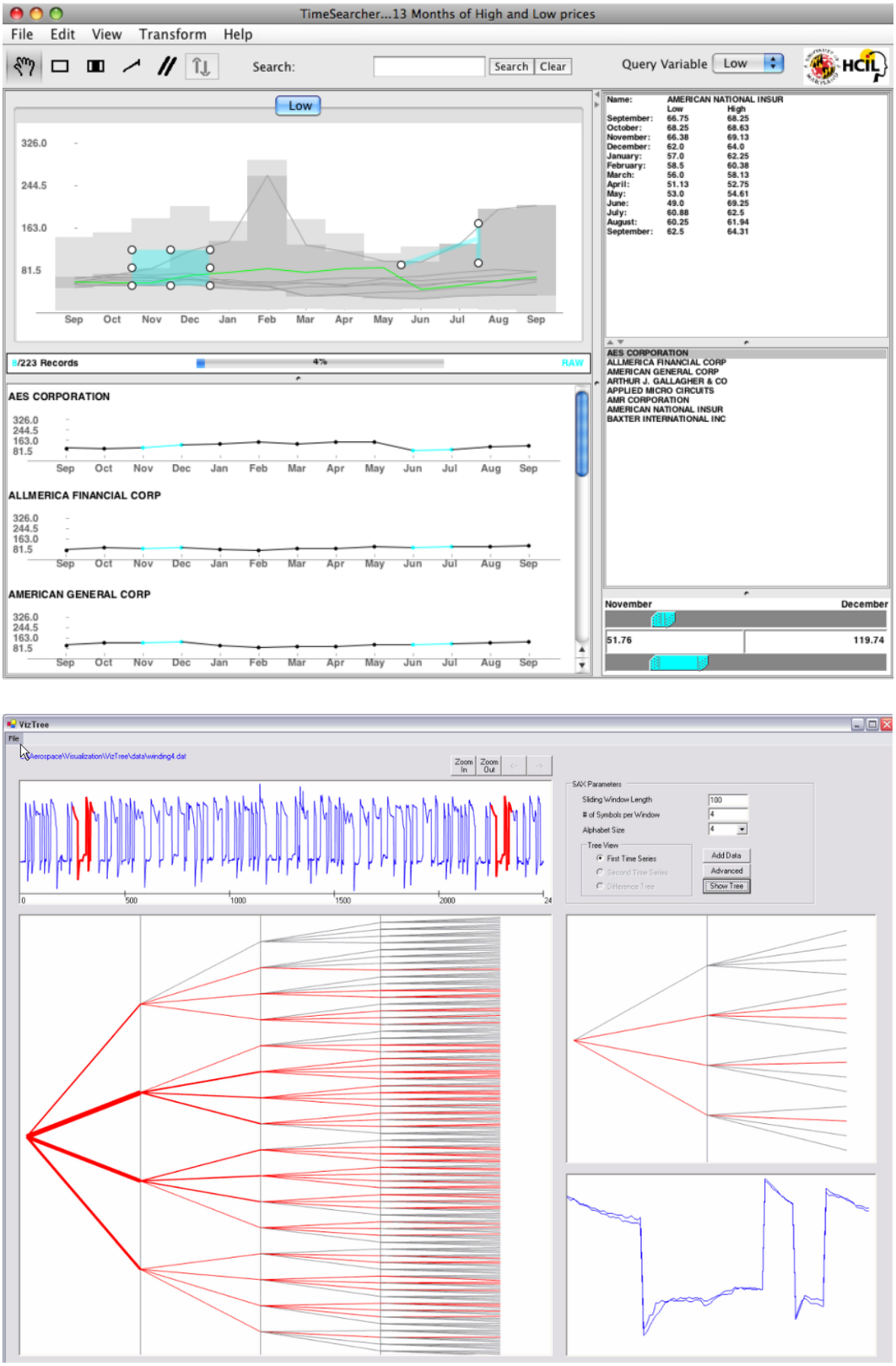


Figure 5.4: Tools for time series visualization and mining (part 1): *TimeSearcher* (top) and *VizTree* (bottom).

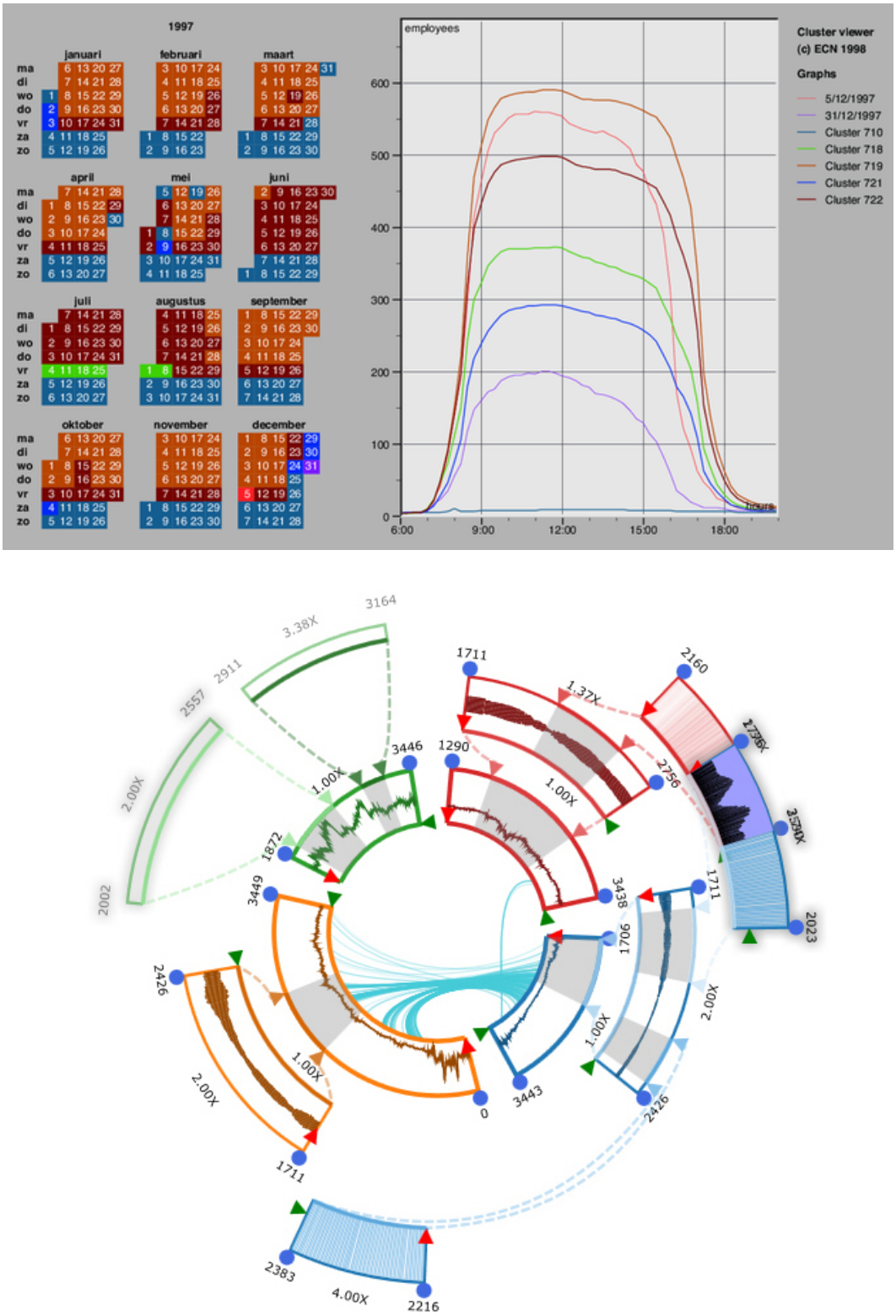


Figure 5.5: Tools for time series visualization and mining (part 2): *Cluster and Calendar-Based Visualization* (top) and *KronoMiner* (bottom).

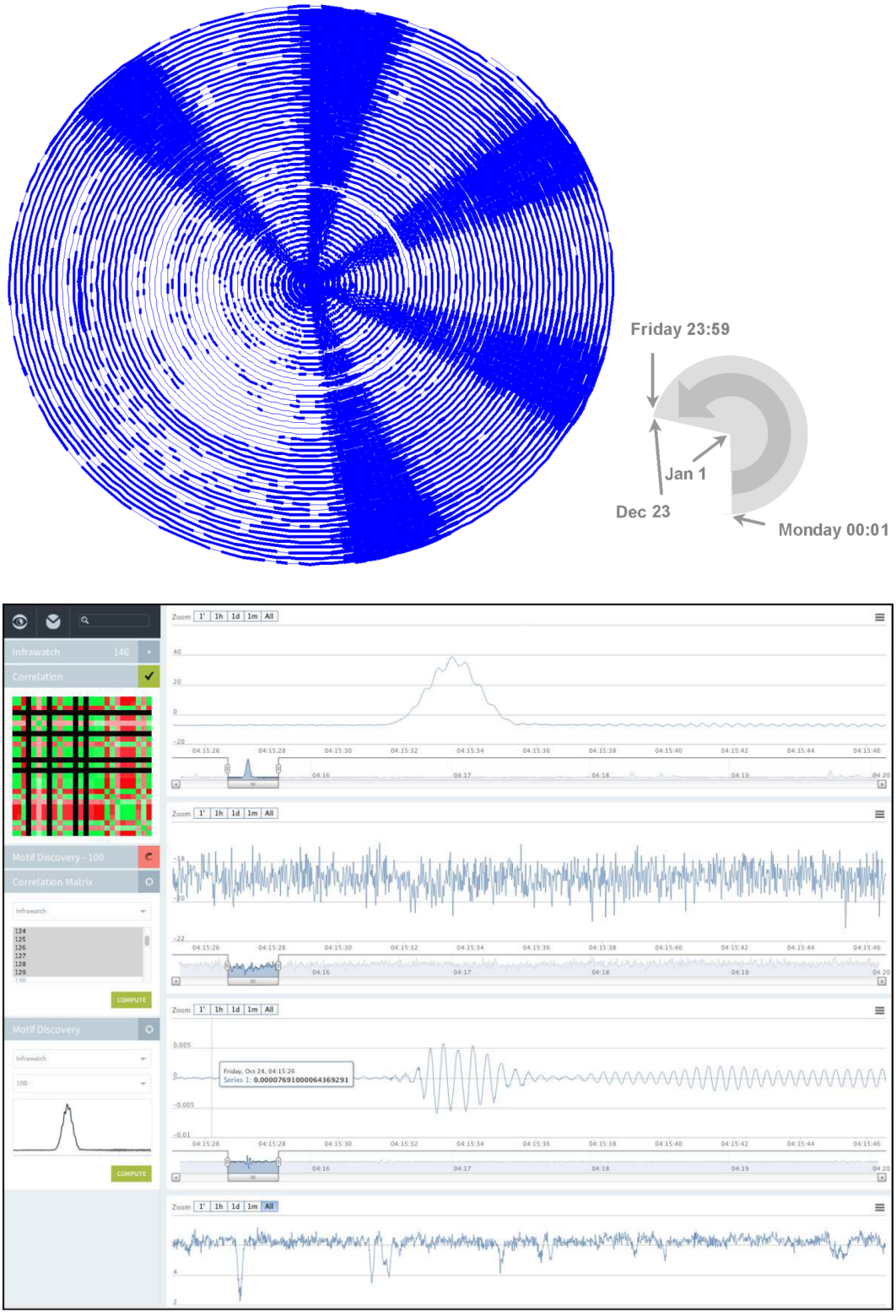


Figure 5.6: Tools for time series visualization and mining (part 3): *Spiral* (top) and *VizTool* (bottom).

- Perform sample selection based on the time series data and metadata;
- Cope with time series quality problem, such as noise and holes;
- Create and visualize high-level representations/transformations (PAA, SAX, FFT, FWT,...);
- Create, tune, extract and export various features (for the classification), and visualize them side by side with the time series;
- Perform time series classification with various classification algorithms (discriminative and generative), options, training/test set composition, and cross-validation;
- Use global and local generic approaches to time series classification (like in the case studies);
- Perform various methods of feature selection (from grid search to genetic algorithm);
- Generate formatted training, CV and test results automatically saved to disk with all experiment settings for reporting;
- Export reusable and "deployable" trained models, for instance to be embedded in other tools;
- Provide reporting abilities and experiment/results reproducibility;
- Rerun previous experiments and reproduce results easily;
- Avoid methodological errors in the time series classification workflow and with the algorithms, thanks to a step-by-step process;
- Provide intuitive use and simple process for time series classification (e.g. avoid "spaghetti schemes" of some data mining tools);
- Provide machine learning learning tool dedicated to time series visualization, handling and classification, usable for scientific, didactic and academic purposes.

5.4.2 Application Overview

Our GTSCT has been thought to be intuitive and straightforward to use. The main graphical user interface (GUI) is composed of a unique *File* menu mainly for time series data and metadata importation, and four tab panes for time series visualization, selection, feature extraction, and classification (see Figures 5.7 and 5.10). This section proposes an overview of these different components with the included functionalities. The related theory of every implemented feature has been explained in Chapter 2 and is not presented here.

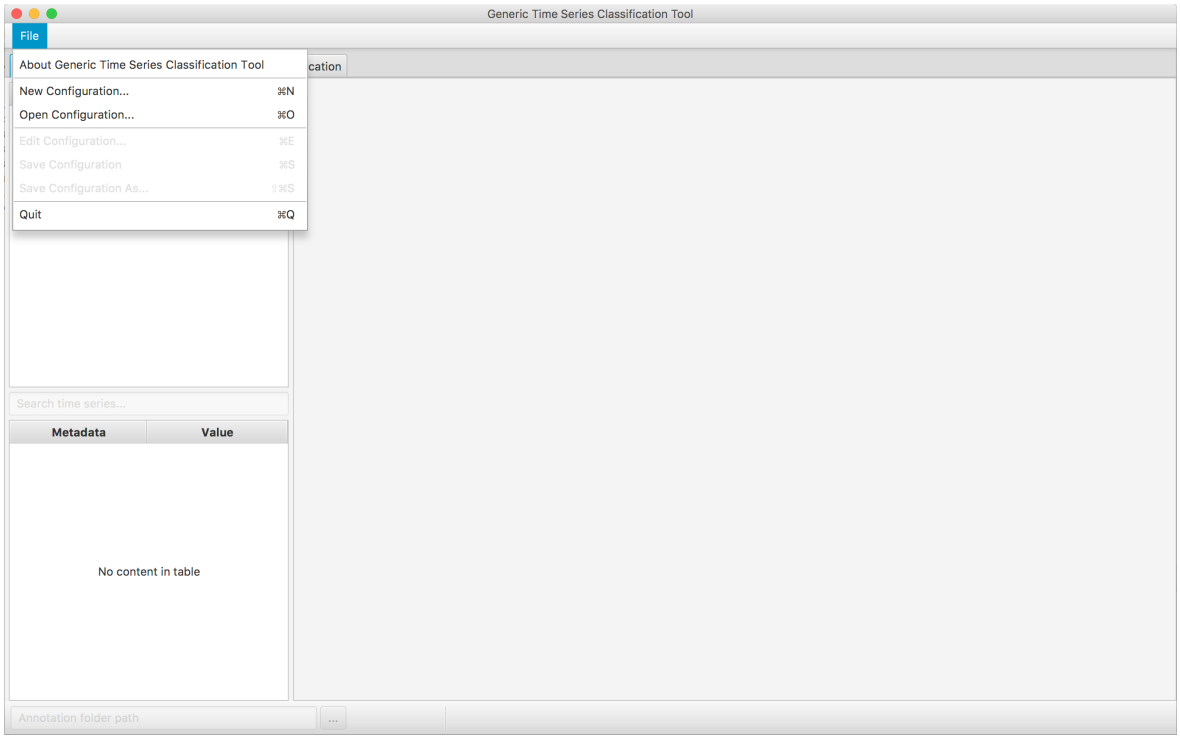


Figure 5.7: Initial window of GTSCT application. The user can open/create a new configuration file to load a database, select, extract features and classify time series by choosing the corresponding option in the *File* menu.

The options *Open Configuration*, *Save Save Configuration* and *Save Save Configuration As* of the *File* menu permit to easily open or save (as) a configuration file containing all the settings for the time series data and metadata importation, selection, feature extraction and classification, that is for a given experiment. The configuration file is in the YAML format which is an human-readable data format, much simpler than XML to handle (see Figure 5.8).

Time Series Data and Metadata Importation

The options *New Configuration* and *Edit Configuration* of the *File* menu open a specific window dedicated to easily create or edit the settings for the importation of the time series data and metadata in the application. GTSCT was thought to have few information to enter in order to import various time series datasets (see Figure 5.9). The different fields are checked before validation and tooltips help the user to enter right information.

The supported file formats for the time series data and metadata are XML, CSV, and JSON. The data and metadata of a given time series can be in same file (as in first case study, see 3.4) or separated files and formats (as in second case study, see 4.4). The time series database, that is all the data and eventual metadata files can be compressed into a ZIP

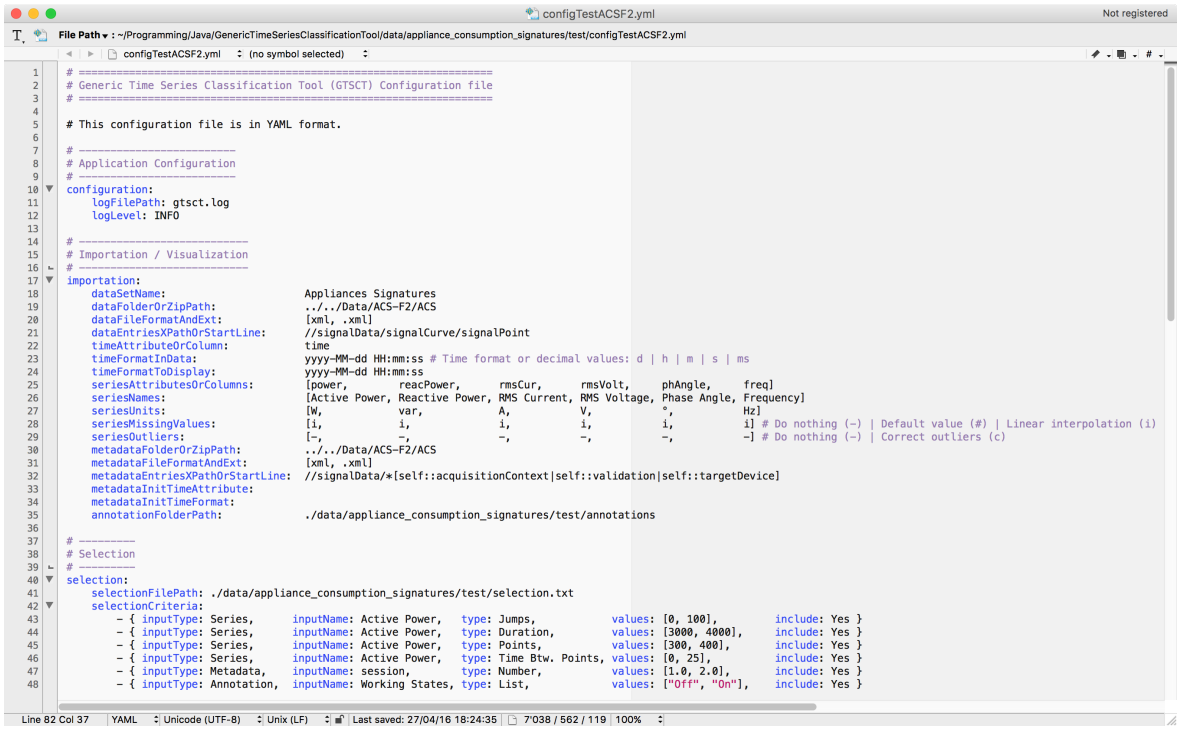


Figure 5.8: A GTSCT configuration file in human-readable YAML data format.

archive in order to reduce the size of the memory used on the hard drive. This also increase the access to time series data on the disk.

Time Series Visualization

The first tab of the application is dedicated to visualize and manipulate the imported time series, which includes standard operations, such as zooming in and out, or rescaling the axes in real time. If time series are stored in *category* folders on the disk, they are listed in the corresponding categories in the tree view. A search field allows to research and display only the time series whose name contains the input string searched. The related metadata of the current time series is also displayed in a table which can be browsed (see Figure 5.10).

Time Series Manual or Automatic Annotation

This first visualization tab also allows to annotate time series, which consists in giving one or several labels to the sequences of points of a given time series. The annotation process can be done manually by the user, or automatically by the computer through the k-Means algorithm (see Figures 5.11 and 5.12), as it was done in our two case studies (see 3.5.3 and 4.5.1). Such annotations may serve to select/filter time series in the second tab *Selection*, or

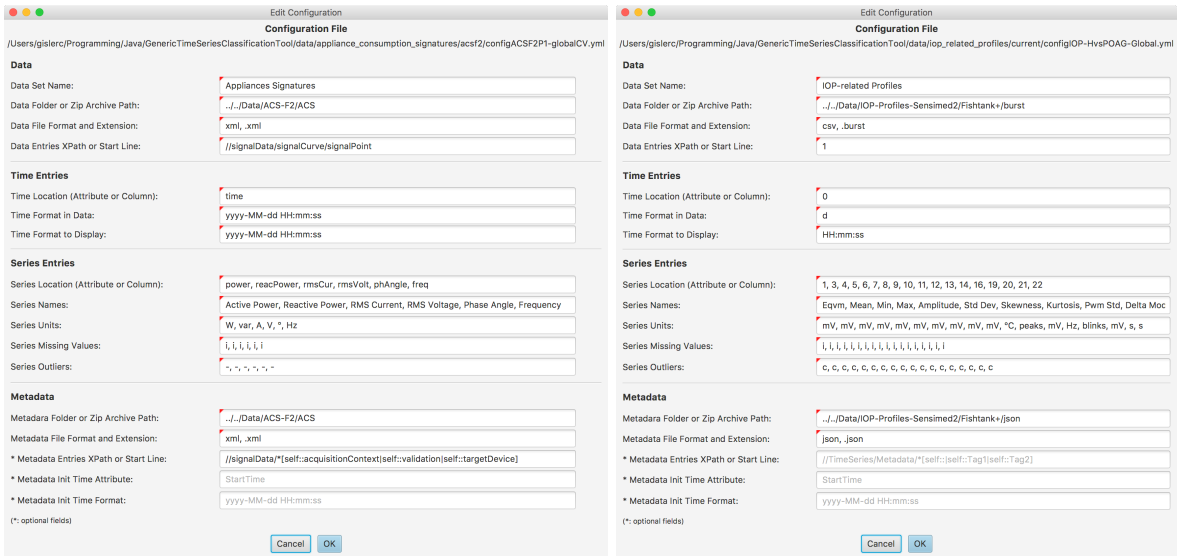


Figure 5.9: Settings for the importation of the time series data and metadata of the first (left) and second (right) case study. Few information are required to import and visualize various time series.

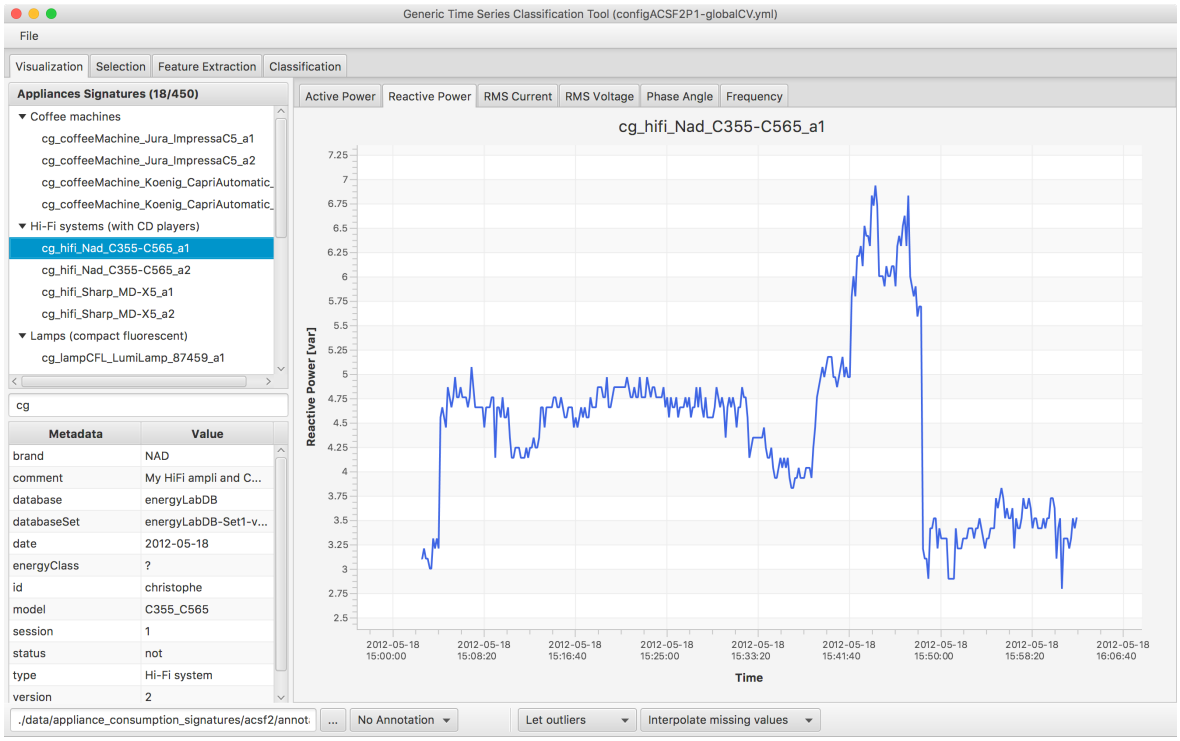


Figure 5.10: Visualization of time series in the GTSCT application.

to extract certain features only on specific sequences[2] of time series points in the third tab *Feature Extraction*, or to be used as class labels for the classification task in the fourth tab *Classification*.

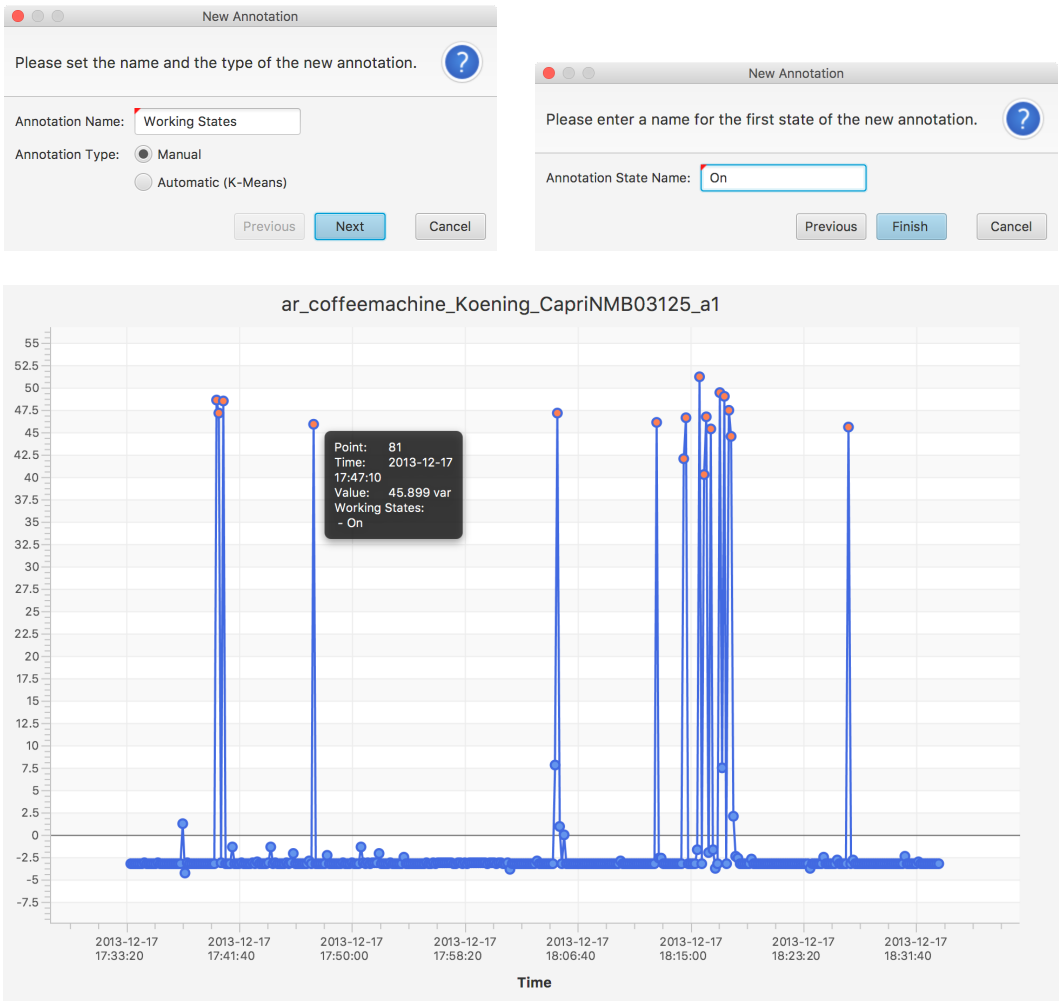


Figure 5.11: Manual annotation of time series (by the user) in the GTSCT application. Each annotation is automatically saved to a file into the folder specified by the user.

Time Series Outliers and Missing Values Handling

This first visualization tab also permits to handle outliers and missing values. Outliers can be either ignored, i.e. let as they are, or corrected by using **Hampel's outlier filter** (see Figure 5.13 and 2.2.2). Missing values can be either ignored, i.e. let as they are, replaced by a default value to enter, or interpolated (see Section 2.2.2).

[2] A sub-sequence of contiguous points of a given time series is still a time series and all available features can be extracted.

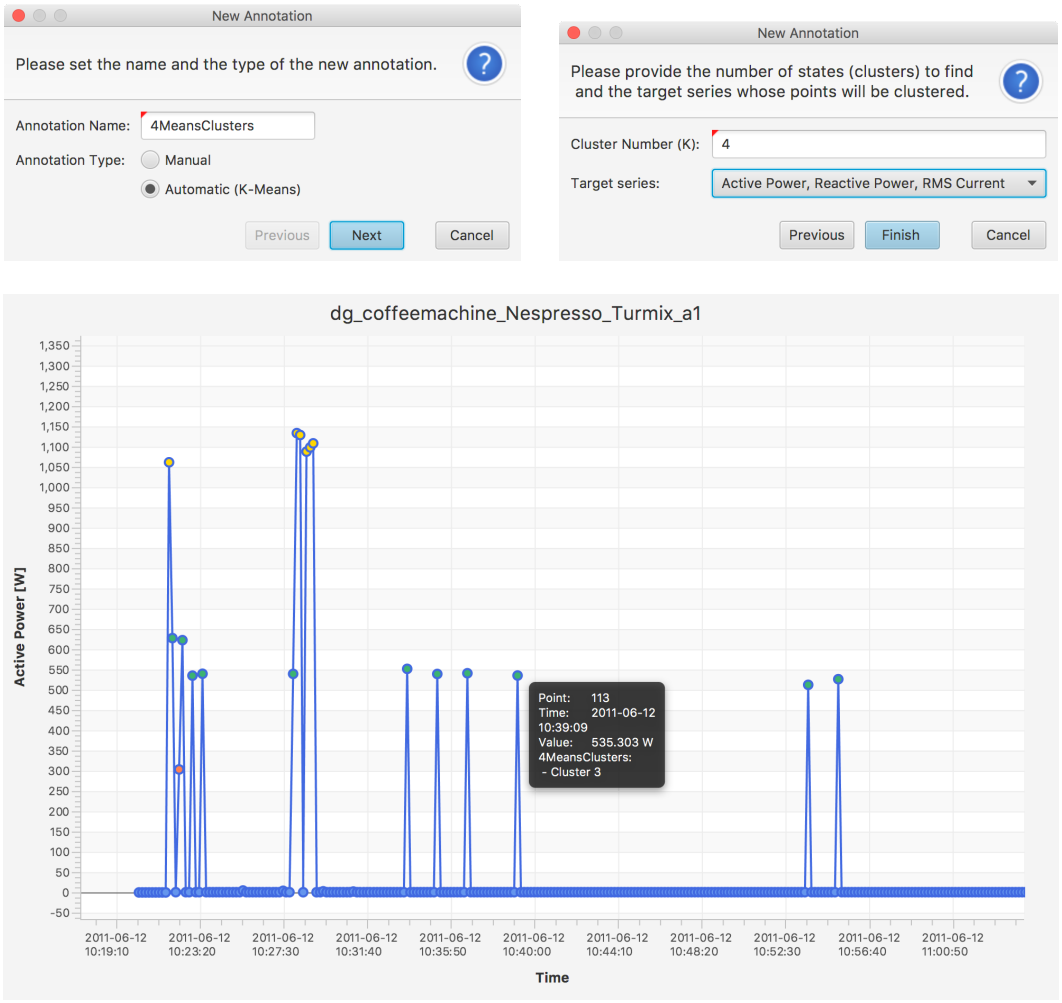


Figure 5.12: Automatic annotation of time series (using the k-Means algorithm) in the GTSCT application. Each annotation is automatically saved to a file into the folder specified by the user.

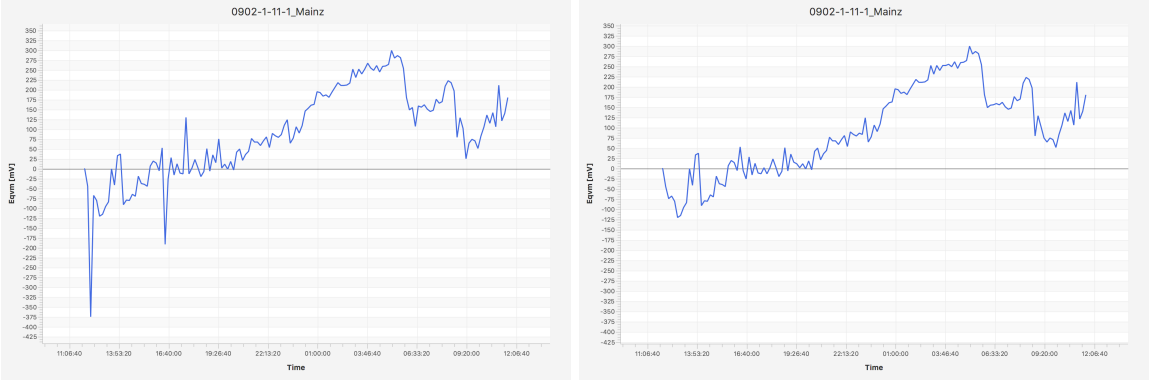


Figure 5.13: Time series outlier correction in the GTSCT application: example without (left) and with (right) outlier correction.

Time Series Selection

The second tab of the application is dedicated to select/filter the imported time series, mainly for the next steps of feature extraction and classification (see Figure 5.14). The time series can be selected through four types of filters allowing to select time series under all aspects:

1. **Time series** names or categories, in order to include or exclude time series by name or category;

2. **Series** data, in order to include or exclude time series whose duration, time between consecutive points, number of points, jumps or missing values is in or out a given interval of values;

3. **Metadata**, in order to include or exclude time series containing or not the given metadata values, which can be *categorical* or *numerical* (see 2.3.2);

4. **Annotation**, in order to include or exclude time series containing or not the selected annotations.

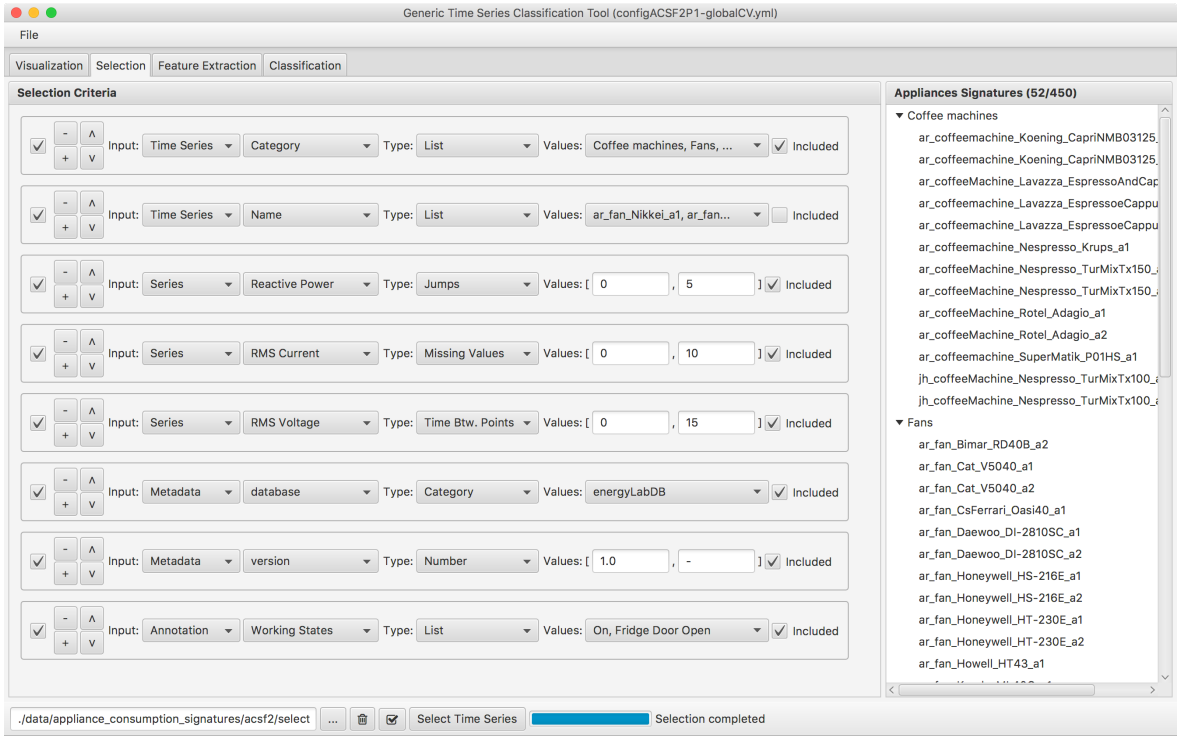


Figure 5.14: Time series selection in the GTSCT application. The list containing the selected time series is automatically saved to a file specified by the user (bottom left of the window).

Time Series Generic Feature Extraction

The third tab of the application is dedicated to extract various generic features from the selected time series, mainly for the next step of classification (see Figure 5.15). Most of them are detailed in Section 2.2.4, 3.5.2 and 4.5.2. The following features can be extracted:

1. **Global operation** values, where the operation can be chosen from a list of about 50 statistical or other mathematical operations;
2. **Local global operation** values, where both local and global operations can be chosen from a list of about 50 statistical or other mathematical operations;
3. **Local operation histogram** values, where the operation can be chosen from a list of about 50 statistical or other mathematical operations;
4. **Fitted polynom** coefficients of the variables per degree;
5. **Fitted polynom** curve values, for a polynom of given degree;
6. **Loess fitted curve** values, by using the eponym curve fitting algorithm;
7. **Dynamic coefficients**, i.e. delta of delta-delta coefficients, which can be coefficient raw values, or coefficient presences, counts, or percentages (for which a coefficient histogram is built before);
8. **FFT filter banks**, with given number of banks and overlap ratio;
9. **FFT frequency magnitudes**
10. **FFT filtered values**, with a given low or high bandpass frequency filter;
11. **FWT Haar wavelet coefficients**
12. **FWT Haar filtered values**, with a given low or high bandpass coefficient filter;
13. **PAA representation values**, which can be segment raw values, or segment presences, counts, or percentages (for which a segment histogram is built before);
14. **SAX representation values**, which can be letter values (letters mapped to numbers), or letter presences, counts, or percentages (for which a letter histogram is built before)
15. **SAX word frequencies**, whose frequency values can be raw (the raw count of each retained word), or binary (the presence or not of each retained word);
16. k**-Means cluster** statistical values, which can also be obtained by using the automatic annotation with the k-Means algorithm, and global operation features extracted on the resulting annotated parts (i.e. the clusters);

17. **Annotation state values**, which can be values indicating the presence (in majority or not) of a given state, the count or the percentage of points which are in this state;

18. **Metadata values**, which can be a numeric value, a category value, or a category (boolean) vector.

Except for the last two features of the list, all features can be extracted from selected annotated parts of the times series if necessary. In certain situations, depending on the time series classification approach used (global or local), when the time series can have a varying length, a *length fitting/adjusting* may be required before extracting certain features in order to have feature vectors of equal dimension, which is required by the classification algorithms. Therefore, the tool proposes to automatically adjust the length of the selected time series if necessary, by either cropping or extending (by extrapolation) them in order to have the minimal, maximal or median length computed over the whole time series set.

Lastly, every feature can be visualized and tuned side by side with the time series in the visualization tab, before performing the extraction, for example as shown in Figures 3.15, 3.17, 3.18, 3.19, or 3.21 of first case study.

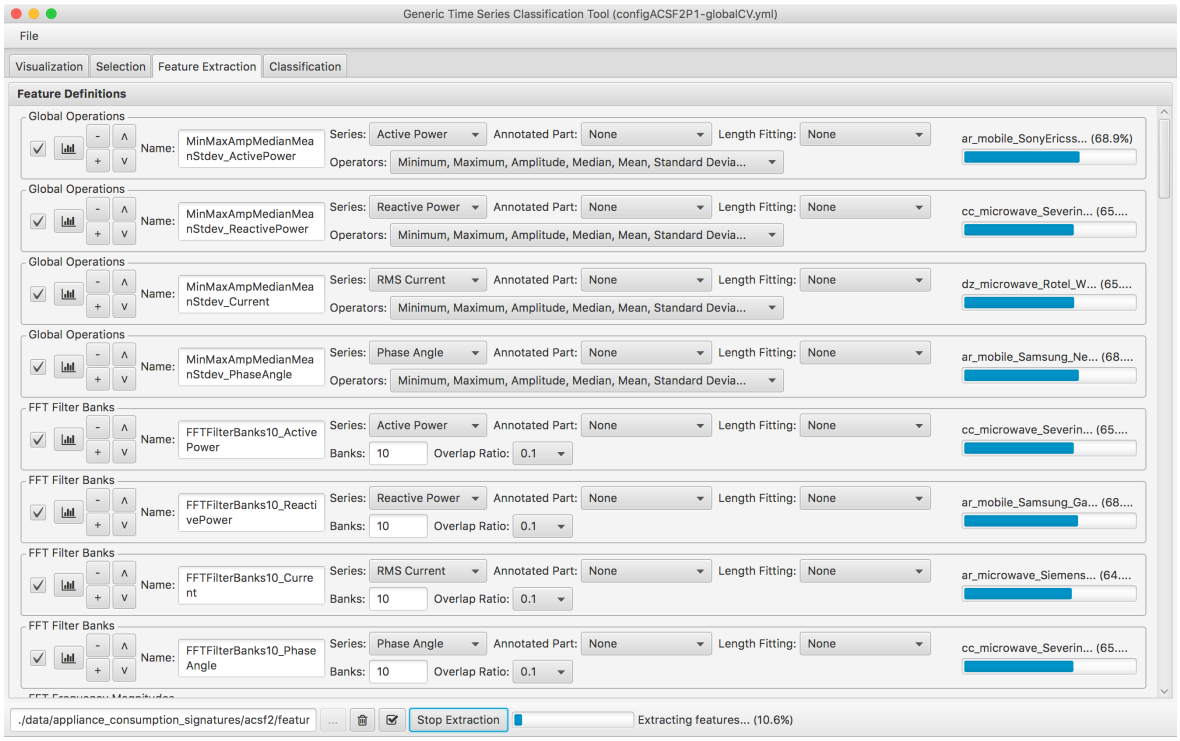


Figure 5.15: Time series generic feature extraction in the GTSCT application. The extracted features are saved to separated files into the folder specified by the user (bottom left of the window).

Time Series Classification with Generic Approaches

The fourth and last tab of the application is dedicated to classify time series by using a global or local generic approach and the extracted features previously. It is composed of two main panels (*titled panes*): the classification settings and the classification console.

The panel for the classification settings is composed of five sections (see Figure 5.16):

1. **Class labels**, allowing to choose two or more labels to use for the classes, either from the time series metadata, annotation, or category (i.e. name of folder containing a subset of the time series on the disk). All possible values available are dynamically displayed and proposed to the user.

2. **Sample sets**, allowing to set a list (from a file) of samples to use for the training set and eventually the test set. When a unique sample list is available, it can be split into training and test sets with respect to certain sample percentage for the test set. Various set options can be selected: none, stratify, balance, or balance by repeating samples, and shuffle samples before splitting. Hence, we can notably deal with the problem of unbalanced data sets.

3. **Features**, allowing to select the extracted features to use, by setting the path of the folder containing the feature files. The global or local approach to time series classification must here be set by choosing the way of building the feature vectors. The method for feature normalization (none, min-max normalization, or standard score) as well as the percentage of feature dimensionality reduction with PCA can also be set. PCA can sometimes help to deal with the *curse of dimensionality* problem.

4. **Parameter tuning / Performance estimate**, allowing to chose a CV method (none, Leave-One-Out, k-Fold, or Repeated Random Subsampling) to use during the training. Depending on the method, various options can be set, such as none, stratify, balance, or balance by repeating samples, and shuffle samples before splitting for the internal CV sets. The input features can be tested either by trying all the features together, each feature separately, all feature combinations, or by using a sequential forward/backward selection method, or a genetic algorithm. The features can also be grouped by name prefix or suffix (when an underscore char is present in the feature names) and the resulting groups of features can be tested either separately or combined. At last, all classifier parameter values within the given ranges can be tested to find the best values during the model fitting.

5. **Classifier**, allowing to choose the classification algorithm to use among SVM, MLP, and GMM. Each classifier owns specific parameters whose values can be directly set to be used during the test phase, or value ranges can be given to be tested during the

training phase. Finally, the performance score to use must be selected in the available measure list, which varies depending on the classification type: binary or multi-class.

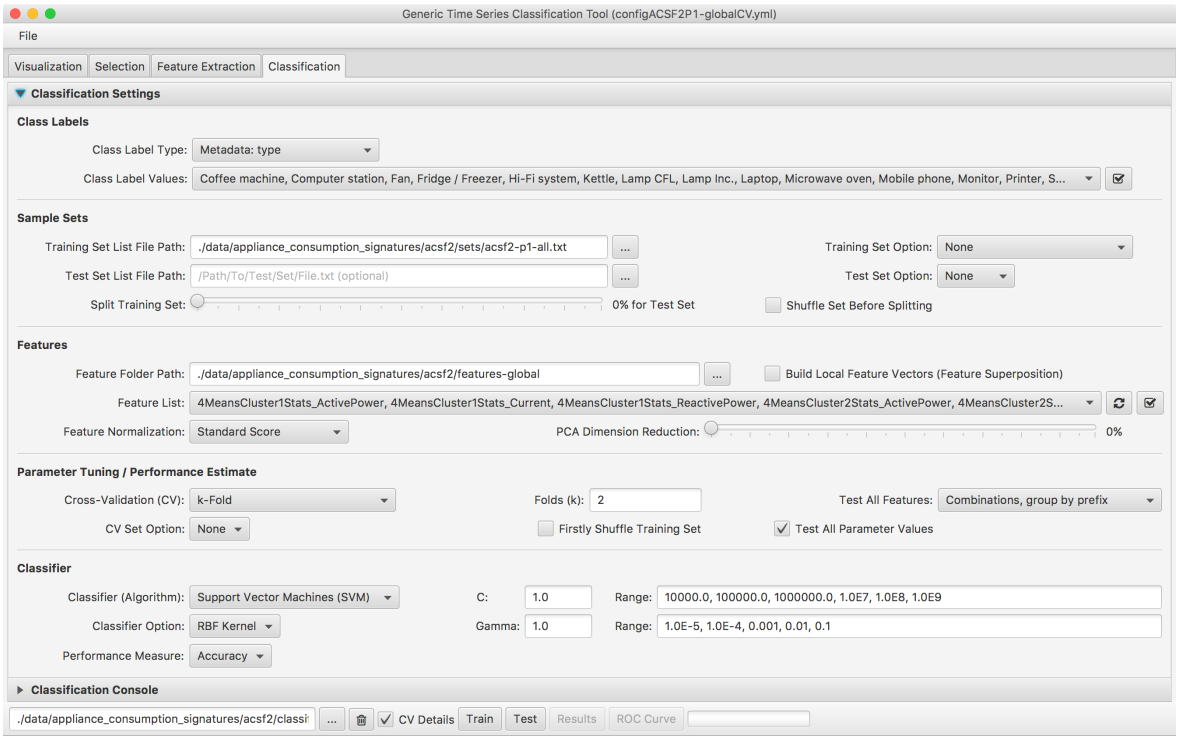


Figure 5.16: Classification settings panel in the GTSCT application. Each option has a tooltip to help and guide the user.

The panel with the classification console is dedicated to display all pertinent information to the user in real time, notably concerning the selected settings, the composition of training and test sets, and fitting of the model, and the classification outputs. Hence, the console allows to display the current best classifier parameters along with the evolution of the confusion matrix on the training set in real time during the training phase (see Figure 5.17).

At the bottom the tab, a button allows to visualize the classified time series decision labels side by side with the corresponding true labels. In case of a binary classification task, another button allows to visualize to classification ROC curve with AUC (see Figure 5.18).

The formatted classification results, which are displayed in the console and automatically saved to disk for reporting, contain all information resulting from the training and/or the test procedures, namely the composition of training/test sets per classes, the input features, the class labels, the classification settings (local/global approach, training/test protocol, cross-validation, feature normalization, feature selection, PCA dimensionality reduction, classifier options, ranges of classifier parameters, and performance measure/score to use), the classifier

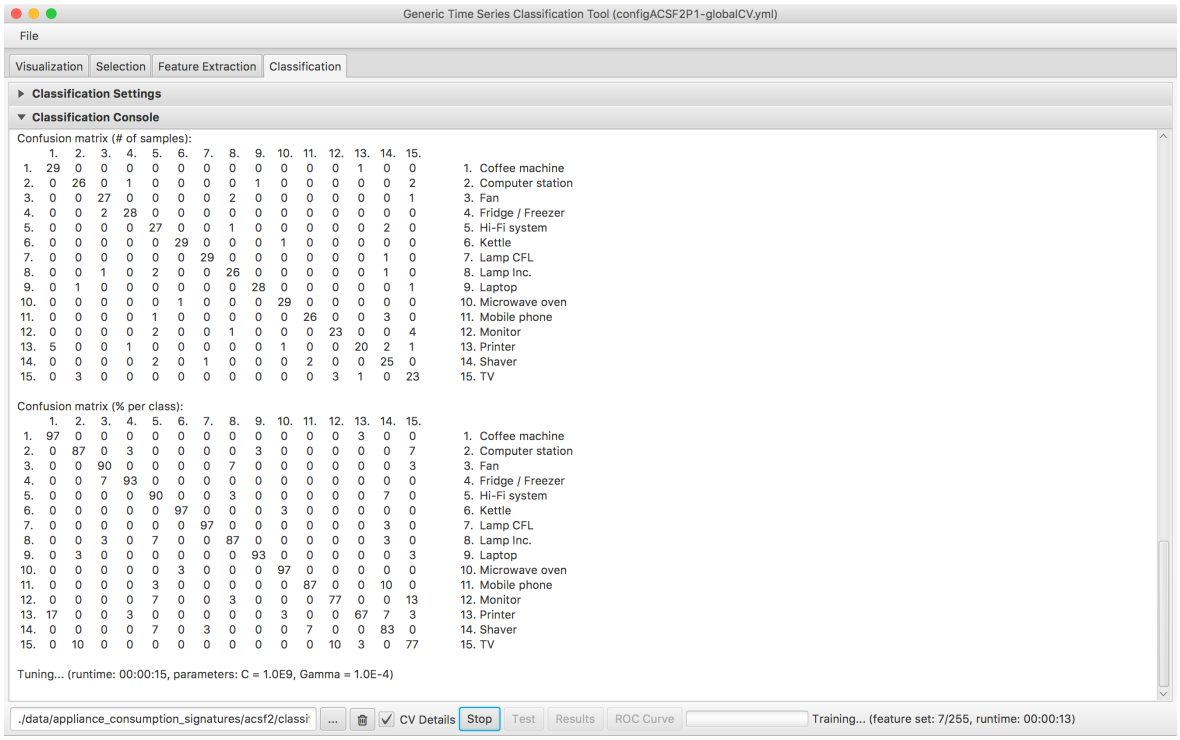


Figure 5.17: Classification console in the GTSCT application. The evolution of the confusion matrix on the training set can be seen in real time during the training phase.

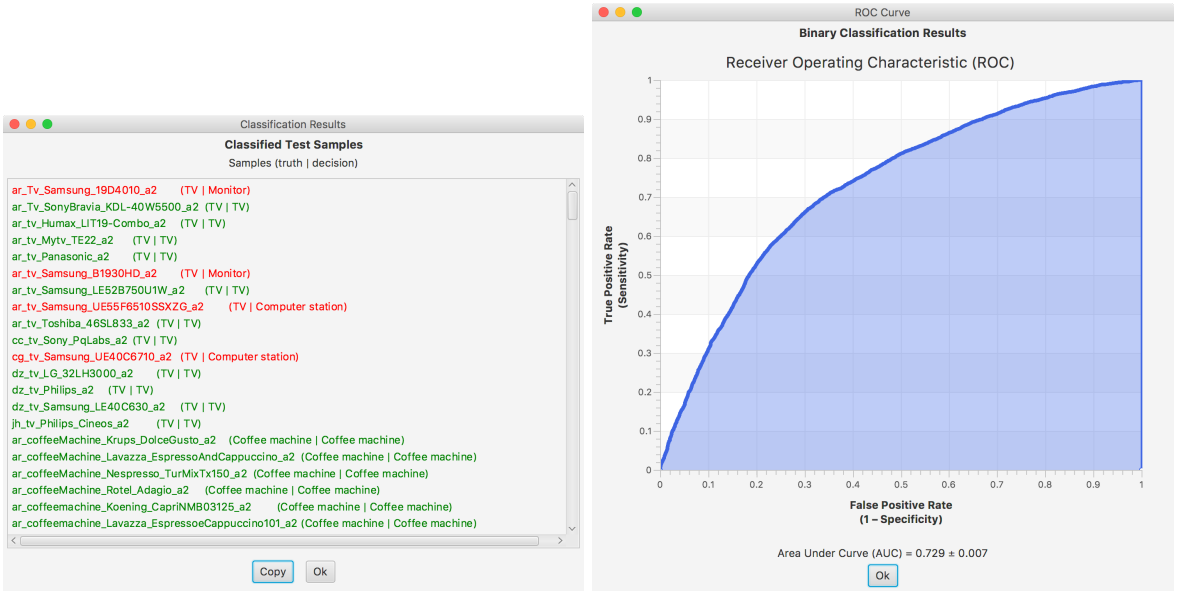


Figure 5.18: Classification results in the GTSCT application: list of classified samples with decision and truth labels (left) and ROC curve with AUC in case of binary classification (right).

parameters yielding the best classification score, other available classification performance measures, a confusion matrix with the raw counts and another one with the percentages per class, and the listing of classified test samples. All training and test sample set lists are also saved to independent files along with results. In case of cross-validation, a ranked list with the best classifier parameters and scores obtained for each selection of features is automatically generated and saved to a special CSV file on disk (see Figure 5.19).

# Cross-Validation Results									
Rank	Accuracy (95% confidence interval)	C	Gamma	Computation time [ms]	Feature vector dimension	Feature number	4MeansCluster1Stats_ActivePower	4MeansCluster1Stats_Current	4MeansCluster1Stats_ReactivePower
1	0.884 ± 0.030	1000000.0	0.001	58	110	14	X	X	X
2	0.882 ± 0.030	1000000.0	0.001	49	96	12	X	X	X
3	0.882 ± 0.030	1.0E7	0.001	55	129	18	X	X	X
4	0.882 ± 0.030	1000000.0	0.001	67	157	21	X	X	X
5	0.880 ± 0.030	1000000.0	0.001	60	100	13	X	X	X
6	0.878 ± 0.030	1000000.0	0.001	47	93	12	X	X	X
7	0.878 ± 0.030	1000000.0	0.001	48	93	12	X	X	X
8	0.878 ± 0.030	100000.0	0.001	44	86	14	X	X	-
9	0.878 ± 0.030	100000.0	0.001	59	128	17	X	X	X
10	0.878 ± 0.030	1000000.0	0.001	65	143	19	X	X	X

Figure 5.19: Extract of a CSV result file. Cross-validation results are automatically saved to a CSV file and consist of a ranked list of classifier settings, scores and feature selection.

In case of a local approach to time series classification, a so-called *last classification* annotation is automatically generated (and saved to disk) for every times series of the test set. This annotation is directly viewable on the tested time series in first visualization tab of the application (see Figure 5.20). This is a also a very useful feature to have in order to understand the classification outputs and tune the features to extract eventually.

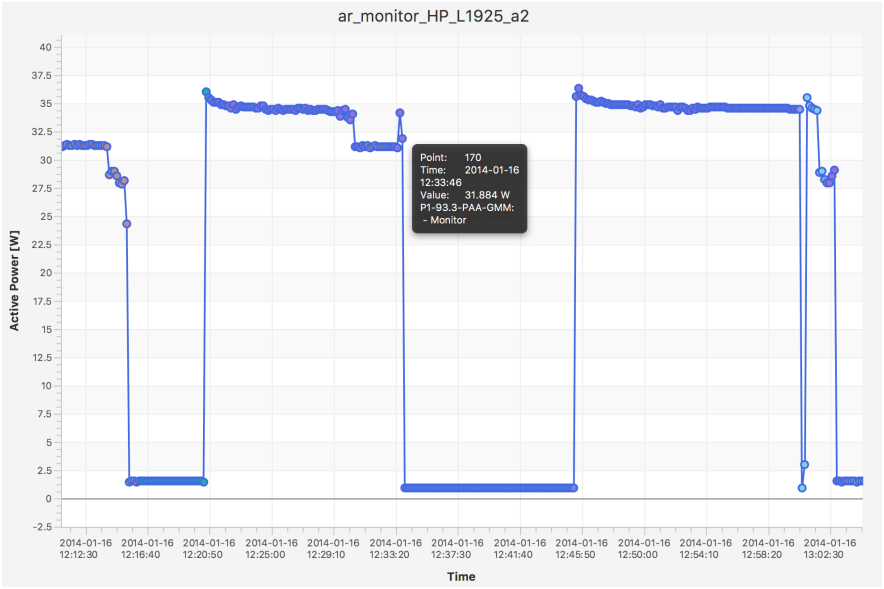


Figure 5.20: With local approach, the classified times series parts are annotated with the resulting class labels.

Input/Output File Importation and Exportation

On the one hand, after each step of the data mining process, in our case a machine learning classification process, all output are saved as TXT or CSV files to the disk at the paths given by the user at the bottom of each tab of the application. Then, these output files can be (re)used either by the GTSCT application for the next step of the machine learning process, or eventually by another external application. On the other hand, before each step of the data mining process, related input files can be loaded from the disk. These input files may either come from the GTSCT (output of a previous step), or from another external application.

These input and output files can be time series annotation, selection, extracted feature or classification data files, which consist of training/test set sample lists, normalization or PCA data, and classifier model. Together with the main unique setting file, they allow to save and reload all settings of an experiment and reproduce its results in two clicks. The formatted classification output files are directly reusable for reporting the experiment results. Moreover, these files are automatically generated with a meaningful name containing the classification score, the number of features and the name of the classifier used, in order to help the user to get his bearings when he is doing a lot of experiments.

At last, it was also fundamental for the tool to be able to export reusable and "deployable" trained models, for example to be embedded in another tool. All required files for normalization or PCA reduction are also automatically provided.

Parallel Computation

Time series may be long and of high dimensionality, as well as their data set can be big. The application GTSCT has been thought to do the less disk accesses as possible and to load into memory only the required data at the right time, as strongly recommended in numerous paper and books. For example, the data of a given time series is loaded into memory only when it has to be displayed and the memory is released when another one is displayed. The feature extraction is done in parallel for all features and for a given feature, the application checks if the needed time series data is already in memory and used for another feature. Like the feature extraction, the classification (particularly the cross-validation during the training) also takes advantage of the multicore CPU computation.

5.4.3 Selection Criteria to Choose between Global and Local Approaches

As mentioned above, the application GTSCT implements both generic global and local approaches to classify time-series as presented in the case studies of chapters 3 and 4. Table 5.1 gives some guidelines to help choose between the global and local approach. That said, the case studies showed that using both global and local approaches may be complementary and

increase the classification performance in certain problems.

Selection Criterion	Global Approach	Local Approach
Feature vector (FV) dimension	Bigger	Smaller
Feature vectors number	1 per time series (TS)	Between 2 and l per TS (l = length of TS)
Feature extraction time	Faster	Slower
Training time (with same classifier)	Faster	Slower
Classification time	Faster	Slower (if more FV per TS)
Classification ouptut number	1 per TS	1 per TS 1 per TS point
Classification ouptut nature	Probabilistic or not (depending on the classifier)	Probabilistic only (imperatively)

* Some indications for a given approach depend on the problem and are given relatively to the other approach.

Table 5.1: Selection criteria to choose between global and local approaches to classify time series.

5.4.4 Classification Algorithms Libraries

The classification algorithms which used in the case studies were implemented from scratch to be embedded in the GTSCT. As results and secondary contributions of this thesis, three Java libraries have been made available. All these libraries support both binary and multi-class classification tasks.

SVM4J Library

The SVM4J library has been primarily developed to help to understand the functioning of a Support Vector Machine (SVM), which was detailed in Section 2.4.4. To the best of our knowledge, the only existing SVM library for Java is LibSVM [Chang and Lin, 2013]. This professional library has been developed for more than 10 years and implements all possible tricks to faster the computation. Hence, its purpose is efficiency to the detriment of simplicity. First developed in C++, it has later been translated to Java. The resulting Java code is nearly not commented and very difficult to understand. Our SVM4J library is a bit slower in some cases depending on the data and the problem to solve, but its code is very simple, short, and much approachable, also from an academic and didactical point of view. Implemented on the basis on the theory detailed in [Chang and Lin, 2013] and the pseudo code of the Sequential Minimal Optimization (SMO) algorithm to solve the quadratic problem in [Platt, 1998], SVM4J has been tested and verified. Classification outputs obtained on common open

data sets were successfully compared with those of *LibSVM* and were absolutely equal. At last, in addition to the API, SVM4J is also directly runnable in batch mode in a terminal.

MLP4J Library

The MLP4J library has been also primarily developed to help to understand the functioning of a Multilayer Perceptron (MLP), which was detailed in Section 2.4.5. To the best of our knowledge, there currently exists no independent and open source Java MLP library. Our MLP4J library has been implemented on the basis of the theory detailed in [Duda et al., 2001, Chapter 6]. Its code is very simple, short, and much approachable, also from a academic and didactical point of view.

GMM4J Library

The GMM4J library has been also primarily developed to help to understand the functioning of a Gaussian Mixture Model (GMM), which was detailed in Section 2.4.6. To the best of our knowledge, there exists no independent and open source Java GMM library. Our GMM4J library has been implemented on the basis of the theory detailed in [Duda et al., 2001; Bishop and Nasrabadi, 2007]. Its Java classes code have been adapted from some classes of the open source project *Apache Commons Maths* and under the Apache License [, ASF]. The principal adaptation consisted in adding variance flooring functionality to the existing class `MultivariateNormalMixtureExpectationMaximization`, which has been itself implemented on the basis of the the theory described in [Chen and Gupta, 2010].

5.4.5 Evaluation

Strictly speaking, GTSCT has not been fully evaluated against other toolkits regarding usability and performance, e.g. in terms of CPU, memory usage and classification. However, we strongly and successfully used it to perform all experiments of the two case studies presented in Chapters 3 and 4. Hence, the first purpose of GTSCT was to demonstrate the feasibility and applicability of the global and local generic approaches to time series classification. Portions of the code and libraries were individually tested and compared with other implementations, such as the SVM4J library.

5.4.6 Discussion

We believe that GTSCT is very useful, easy-to-use and efficient application for generic time series classification. Its visualization abilities, generic time series representations and features, as well as the local and global approaches to time series classification tend to demonstrate it. Our GTSCT embeds all tested generic time series global/local approaches and features, as well as three classification algorithms whose libraries SVM4J, MLP4J, and GMM4J are

available which constitutes also contributions of this thesis.

Some specific time series features, like SAX word frequencies and dynamics coefficients are specific achievements as, to the best of our knowledge, they have never been integrated as such in other tools. The same can be said about the feature selection methods based on Genetic Algorithm (GA), Sequential Backward Selection (SBS), and Sequential Forward Selection (SFS).

The possibility to combine and use global time series representations like PAA and SAX in a local approach to time series classification, by decomposing them into small local parts constitutes one of the contribution and novelty of this thesis. Hence, the global knowledge captured by the holistic representation is then used to classify time series parts (feature vectors) locally. Such local segment-based features have the advantage to tolerate time series misalignments and are computationally much more efficient, as compared with local point-based features. The possibility to extract features from clusters of time series points previously found by means of the k-Means algorithm, and use them in a global approach to time series classification also constitutes an interesting and novel approach.

In order to improve the GTSCT, some new functionalities might deserve to be implemented in the future, such as the possibility to:

- Align time series relatively to a given point in time. Depending on the problem to solve, the features and the approach (e.g. global) used, time series sometimes need to be aligned and/or sometimes trimmed relatively to the time of certain events.
- Display the correlation between the extracted features.
- Use an assistant to automatically tune a feature of a certain type, by generating, extracting and testing a subset of features of this type with varying parameter values.
- Perform a PCA per feature in order to decrease the dimensionality, and thus the weight of big sparse features relatively the other smaller but powerful features, and hence maybe deal better with the curse of dimensionality;
- Perform the fusion of classifier outputs in order to improve the overall classification score and performance;
- Use other classifiers, such as RF, k-NN, HMM, or other deep learning algorithms;
- Find and display the best discriminative time series subsequences found with the local approach, and maybe use them for event detection.

5.5 Conclusion

In this chapter, we presented the so-called Generic Time Series Classification Tool (GTSCT), which is a tool developed to perform generic time series classification tasks respecting the data mining process described in Section 2.3. We successfully used the GTSCT to make all the experiments of the two case studies presented in Chapters 3 and 4. Through these cases, the developed tool has notably shown that the same generic approach to time series classification could be applied to different classification problems in machine learning.

We believe that it is only by creating such a complete tool that one develops an *A-Z* expertise in such domain. Indeed, many mistakes and traps are not detailed in theory papers and books. To comprehend and address from the biggest fundamental and general question of the data mining process to the most little, apparently insignificant, but tricky implementation detail of a classification algorithm like SVM or a feature extraction on times series with varying lengths, it takes several case studies and the duration of a PhD thesis. Referring to Einstein's saying at the beginning of this chapter, perhaps the most important purpose of this tool is not just to *solve*, but furthermore to *avoid* most problems that may occur when somebody wants to tackle a time series classification problem. At last, the desire of making a whole complex task the most transparent and simple to understand and solve for other persons is certainly one of the most fundamental and respectable reasons why informatics and computers were invented: to help and serve humans. Computers scientists often forget that is not the Human who has to adapt himself to the Computer, but the contrary. Hence, such a tool may also constitute a didactic tool that can contribute to make accessible the domain of machine learning to a greater part of computer scientists.

6

Conclusion

You can't connect the dots looking forward; you can only connect them looking backwards. So you have to trust that the dots will somehow connect in your future.

Steve Jobs

Contents

6.1 Summary

Time series data represent a large part of the data supply in the world. Many data mining tasks, such as classification, regression and prediction, which are mostly solved by machine learning methods have to deal with time series data. Existing machine learning algorithms and their performance do not only depend on the problem to solve, but also on the type of data. In every data mining process, the preprocessing phase and the representation of the data, especially time series, is fundamental. Most machine learning algorithms do not work well on time series because of their unique data structure which requires a particular approach to the related problems. State-of-the-art research works have recommended to work with high level representations of time series in order to avoid problems, such as the curse of

dimensionality. Numerous time series representations and features have been proposed and used, notably for the feature extraction phase of the traditional machine learning process, in various applications and problems. Research works on time series data mining have shown that no particular type of representation is the best for all problems. The efficiency of the time series data mining tasks (including classification) depends on a suitable representation of the time series, as well as the quality of the representation and the generated features. Moreover, time series data often come with meaningful metadata of various types, such as numbers, words, lists, or enumerations. These metadata are sometimes left out from the machine learning process because they cannot be easily or automatically handled, that is without human manual intervention. Hence, there is a necessity to adopt a specific approach to handle them generically during the machine learning process, and combine them automatically into efficient feature vectors for a particular classification problem.

In this thesis, we have reported on our research in the field of generic approaches to time series classification. First, we gave all the fundamentals required for our work and generic data-driven approaches to time series classification, which includes the data mining process, the classification task and the used algorithms (SVM, MLP, and GMM), as well as the specific handling, representations, features, and methods commonly used for mining time series data. Then, through two different case studies, we presented, used, tested and evaluated two generic approaches to multivariate time series classification: a global and a local one. The global approach consisted in building one global unique feature vector per multivariate time series, by extracting and concatenating different global features from each dimension of the time series. The second approach consisted in dividing the same time series in smaller segments, extracting features from these segments and building as many feature vectors as segments. The first case study concerned the task of automatic detection/recognition of categories of home appliances, based on their electric consumption signatures recorded with a low-end smart outlet sensor. Faced with the lack of data in the domain, we built two new databases of appliance consumption signatures (ACS), namely ACS-F1 and F2, which have been made freely available for the scientific community, and proposed two appliance recognition test protocols. The second case study concerned the task of automatic detection/recognition of the glaucoma disease by patients, based on their Intraocular Pressure (IOP) profiles recorded with a new type of contact lens sensor during 24 hours. In both case studies, the databases at disposal were relatively small (< 1000 instances). For each approach and classification problem, various generic features, some based on time series representations like PAA and SAX, were extracted, and the three classification algorithms used to provide and compare the results. For each problem, each database and test protocol, both global and local approaches provided rather similar results. However, after the analysis, the preference was given to the global approach whose related feature vectors provide a better and lighter representation of the data, as well as a more efficient handling by the various classifiers in general, notably

in terms of computation time and scalability. Finally, we presented the so-called Generic Time Series Classification Tool (GTSCT) elaborated during the case studies. This JavaFX application implements both global and local generic approaches to time series classification. It is intuitive, easy-to-use, and notably provides functionalities for importing, visualization, annotation, selection, representation/summarization, and classification of time series with respect to the data mining process. We also presented the three SVM, MLP and GMM Java libraries created for the tool (but not only). We successfully used our software to perform all the experiments of the two case studies and showed that the same generic approaches to time series classification could be applied to different classification problems in machine learning. Hence, the main contribution of this research thesis is the proposition of a unified tool and a generic methodology for time series classification. This methodology involves a systematic and semi-automated way to discover, thanks to the GTSCT application, an efficient, if not the best configuration of extracted features and classifier parameters.

6.2 Contributions

The main contributions of this thesis are:

- The elaboration of two generic data-driven approaches to time series classification, namely a *global* and a *local* approach, either considering each time series as a global entity, or a set of local parts, taken then for generic feature extraction and classification;
- The elaboration of specific time series features, such as SAX word frequencies;
- The decomposition of some holistic time series representations, such as PAA and SAX, into parts to be used in the local approach to time series classification;
- The application, evaluation, and comparison of these generic approaches through two classification problems (*case studies*), namely for appliance detection and glaucoma detection;
- The specific research work, results and knowledge brought in the specific parent research fields of the two case studies, namely Appliance Load Monitoring / Intrusive Load Monitoring and Visual Health Monitoring / Disease Diagnosis with machine learning, namely:
  - The creation of two consequent databases of appliance consumption signatures (ACS), ACS-F1 and ACS-F2, with dedicated test protocols. Now being publicly available and containing 200 and 450 consumption signatures recorded from 10 and 15 categories of home appliances during two sessions of 1 hour, the databases have already been demanded and downloaded about 130 times worldwide;

  - The elaboration and integration of some specific algorithms (e.g. for eye blink and ocular pulse detection) and models in dedicated medical assistant softwares for intraocular pressure (IOP) analysis and glaucoma diagnosis provided by the company Sensimed to health specialists.

- The creation of a useful and easy-to-use Generic Time Series Classification Tool (GTSCT) allowing to perform any task of time series classification with respect to the *data mining process*. More specifically, the tool allows to apply our generic approaches, use the state-of-the-art time series representations and features, and provide advanced visualization, tuning, and reporting functionalities. Until now, we believe that there existed no equivalent machine learning tool for handling time series specific data and solve related classification problems in a generic manner;

- The elaboration of a generic methodology for time series classification, which involves a systematic and semi-automated process to discover an efficient, if not the best configuration of extracted features and classifier parameters, thanks to the developed GTSCT tool;

- The creation of three simple Java libraries for SVM, MLP, and GMM, namely *SVM4J*, *MLP4J*, and *GMM4J*;

The publications related to this thesis are listed at the end of this document.

6.3 Perspectives

Several general perspectives to our research work can be mentioned. First, the presented generic global and local approaches could be tested and evaluated in other case studies, with different time series databases, small and big. Then, we saw that both approaches have advantages and the used classifiers have different discrimination abilities. It would certainly be interesting to merge the approaches as well as the classifier outputs in order to obtain better results and more robust classification systems. In addition, with the rise of big data, storage capacities and computing performances, new deep learning (DL) approaches have emerged and thus could be tested. They have advantages and disadvantages that traditional machine learning approaches don't have and *vice versa*. For instance, DL methods can learn features automatically, that is in an unsupervised way, but they rather work well with structured data available in big quantity. Hence, they have mostly been applied for the classification of images, audio and video data collected through the web, rather than time series data sparingly collected via physical sensors. Moreover, unsupervised features learnt by current DL methods may be difficult to represent and interpret, e.g in our specific use cases. On the one hand, it still remains that in many applications and problems, like disease diagnosis, data which is generally measured by sensors on small groups of patients is limited in quantity and quality,

and thus DL approaches will not work. On the other hand, it is also clear that an increasing number of problems have a huge quantity of data available, and thus DL techniques will be a good choice. Convolution-based approaches may automatically realize the fusion between the global and local approaches. Hence, one major perspective will be to prospect, use and include DL approaches to time series classification in our GTSCT. Finally, the GTSCT can be improved with new and better functionalities, and its development can be carried on.

Some other perspectives, more specific to the two case studies can be mentioned. First, in the case of appliance detection, the classification tests can be carried on in order to detect more appliance categories, and even their running states, for which the use of some state-based machine learning algorithms, such as HMM, may be judicious. However some research works have already reported that the random aspect of the changing of states by appliances, is difficult to model. Then, we would like to continue to populate and increase our ACS-F databases, as more and more demands are coming from the scientific community. Finally, in the case of glaucoma detection, as more patients' IOP profiles are currently recorded and made available by the company Sensimed, the classification tests can be carried on in order to detect more glaucoma varieties on ill patients, as well as to detect fast/slow evolutive ill patients, or even healthy patients at risk.

6.4 Afterword

Most importantly, this research work was an opportunity to learn a lot of interesting things in all its related fields. As my my thesis director Jean Hennebert said: *The most fundamental thing to do in research is to open doors...* Maybe one day, as the few other lucky ones, you will find something new, wonderful, and even revolutionary. For the remaining ones, their small contributions will somehow help the whole scientific community to progress, which is fundamental and honorable.

An expert is a man who has made all the mistakes, which can be made, in a very narrow field., said Niels Bohr. We believe that it is also that, the purpose of a PhD thesis: to become an expert. In machine learning, you cannot become an expert, if you have only used all the existing tools, frameworks and libraries as black boxes. You must have implemented them by yourself, used them, and made all possible mistakes with them.

Appendices

A

Appliance Consumption Signature Fribourg (ACS-F) Databases

A.1 Availability and Download

The Appliance Consumption Signature – Fribourg database is freely available on the Watt-ICT project (`http://www.wattict.com`):

- ACS-F1: `http://www.wattict.com/web/index.php/databases/acs-f1`
- ACS-F2: `http://www.wattict.com/web/index.php/databases/acs-f2`

A.2 References

Ridi, A., Gisler, C. and Hennebert, J. (2014). *ACS-F2 – A New Database of Appliance Consumption Signatures.* In 6th International Conference of Soft Computing and Pattern Recognition (SoCPaR 2014), pages 145–150. Tunis, Tunisie. IEEE Computer Society.

Gisler, C., Ridi, A., Zujferey, D., Khaled, O. A. and Hennebert, J. (2013). *Appliance Consumption Signature Database and Recognition Test Protocols.* In 8th International Workshop on Systems, Signal Processing and Their Applications (WoSSPA 2013), pages 336–341. Alger, Algeria. IEEE Computer Society.

A.3 Acknowledgement

The elaboration of this database was supported by the grant *Smart Living Green-Mod* from the *Hasler Foundation* (`http://www.haslerstiftung.ch`) in Switzerland.

A.4 Licensing Form

WATT-ICT - License

Please save this document that contains the terms of the license of use of the ACS-F2 database.
We will send you the credentials per e-mail in few minutes

License between the OWNER of the ACS-F2 database and the LICENSEE.

The OWNER of the ACS-F2 database is iCoSys Institute, University of Applied Sciences HES-SO//Fribourg, Engineering and Architecture Faculty, Pérolles 80, CP 32, CH-1705 Fribourg, Switzerland, Phone: +41 79 900 08 62, Fax: +41 26 429 66 00,
contact person: Jean Hennebert, jean.hennebert@hefr.ch

The LICENSEE of the ACS-F2 database is :

First name :	Gisler
Last name :	Christophe
Institution :	University of Fribourg
Email :	christophe.gisler@unifr.ch
City :	Fribourg, Switzerland
Address :	Boulevard de Pérolles 90, 1700

The LICENSEE understands that the data is supplied with no guarantee of accuracy or usability. Therefore, the OWNERS can not be liable of any loss or damage resulting of the use of the database. The OWNERS can not guarantee to maintain the ACS-F2 database.

The LICENSEE will use the ACS-F2 database for non-commercial research only.

When publishing work using the ACS-F2 database, the LICENSEE will cite the publication:
A. Ridi, C. Gisler and J. Hennebert. ACS-F2 - A New Database of Appliance Consumption Analysis, to appear in Proceedings of the International Conference on Soft Computing and Pattern Recognition (SocPar 2014), 2014

The LICENSEE Institution will be listed among those using the ACS-F2 Database

Figure A.1: Licensing form of the ACS-F2 database

Acronyms

ACS Appliance Consumption Signature

ALM Appliance Load Monitoring

ANN Artificial Neural Network

APCA Adaptive Piecewise Constant Approximation

AUC Area Under Curve

CLS Contact Lens Sensor

CRISP-DM CRoss Industry Standard Process for Data Mining

CV Cross-Validation

DFT Discrete Fourier Transform

DT Decision Tree

DTW Dynamic Time Warping

DWT Discrete Wavelet Transform

EM Expectation Maximisation

ERM Empirical Risk Minimization

FFT Fast Fourier Transform

FWT Fast Wavelet Transform

GA Genetic Algorithm

GMM Gaussian Mixture Model

GTSCT Generic Time Series Classification Tool

HMM Hidden Markov Model

ILM Intrusive Load Monitoring

IOP Intraocular Pressure

k-NN k-Nearest Neighbours

MLP Multilayer Perceptron

NILM Non-Intrusive Load Monitoring

OHT Ocular Hypertension

OPA Ocular Pulse Amplitude

OPF Ocular Pulse Frequency

PAA Piecewise Aggregate Approximation

PCA Principal Component Analysis

PLA Piecewise Linear Approximation

RF Random Forest

ROC Receiver Operating Characteristic

SAX Symbolic Aggregate Approximation

SBS Sequential Backward Selection

SFS Sequential Forward Selection

SRM Structural Risk Minimization

SVD Singular Value Decomposition

SVM Support Vector Machine

V-I Voltage-Current

List of Figures

List of Tables

Bibliography

AG, KNIME.COM (2011). *KNIME, the Konstanz Information Miner*. `http://www.knime.org`.

Aggarwal, Charu C. (2015). *Data Mining: The Textbook*. Springer. ISBN 978-3-319-14141-1, 729 pages. doi:10.1007/978-3-319-14142-8.

Aggarwal, Charu C., Alexander Hinneburg, and Daniel A. Keim (2001). *On the Surprising Behavior of Distance Metrics in High Dimensional Space*. In *Proceedings of the 8th International Conference on Database Theory (ICDT 2001)*, pages 420–434. 1, London, UK. ISBN 978-3-540-41456-8. ISSN 0956-7925. doi:10.1007/3-540-44503-X_27.

Agrawal, Pragati and Amit Kumar Dewangan (2015). *A BRIEF SURVEY ON THE TECHNIQUES USED FOR THE DIAGNOSIS OF DIABETES-MELLITUS. International Research Journal of Engineering and Technology (IRJET)*, 2(3), pages 1039–1043.

Aigner, Wolfgang, Silvia Miksch, Heidrun Schumann, and Christian Tominski (2011). *Visualization of Time-Oriented Data*. 1st Edition. Springer Publishing Company, Incorporated. ISBN 0857290789, 9780857290786.

Alexeev, V.L., S. Das, D. N. Finegold, and S. A. Asher (2004). *Photonic crystal glucose-sensing material for noninvasive monitoring of glucose in tear fluid. Chimical Chemistry*, 50(12).

An, L., J. Chao, M. Johnstone, and R. K. Wang (2013). *Noninvasive imaging of pulsatile movements of the optic nerve head in normal human subjects using phase-sensitive spectral domain optical coherence tomography. Optics Letters*, 38(9), pages 1512–1514.

Anderson, Kyle, Adrian Filip Ocneanu, Diego Benitez, Derrick Carlson, Anthony Rowe, and Mario Bergés (2012). *BLUED : A Fully Labeled Public Dataset for Event-Based Non-Intrusive Load Monitoring Research*. In *Proceedings of the 2nd KDD Workshop on Data Mining Applications in Sustainability (SustKDD)*, pages 1 – 5. October 2011. ISBN 9781450315586.

(ASF), Apache Software Foundation (2016). *Commons Math: The Apache Commons Mathematics Library*. http://commons.apache.org/math.

Badrinath, N. and G. Gopinath (2014). *A Survey on Utilization of the Machine Learning Algorithms for the Prediction of Erythemato Squamous Diseases. World Applied Sciences Journal*, 29(6), pages 752–757. ISSN 20407467. doi:10.5829/idosi.wasj.2014.29.06.1836.

Badugu, R., J. R. Lakowicz, and C. D. Geddes (2004). *Noninvasive continuous monitoring of physiological glucose using a monosaccharide-sensing contact lens. Analytical chemistry*, 76(3), pages 610–618.

Baggio, Alberto (2014). *Interactive Visualization and Mining of Massive Time Series*. Master thesis, Universiteit Leiden, Netherlands.

Baranski, M. and J. Voss (2003). *Non-intrusive appliance load monitoring based on an optical sensor*. In *Proceedings of the IEEE Bologna Power Tech Conference*, volume 4, pages 267–274.

Barati, E., M. Saraee, A. Mohammadi, N. Adibi, and M. R. Ahamadzadeh (2011). *A Survey on Utilization of Data Mining Approaches for Dermatological (Skin) Diseases Prediction. Journal of Selected Areas in Health Informatics (JSHI)*, 2(3), pages 1–11. ISSN 1735-143X.

Barkana, Y., S. Anis, and J. et al. Liebmann (2006). *Clinical utility of intraocular pressure monitoring outside normal office hours in patients with glaucoma. Archives of Ophthalmology*, 124(6), pages 793–797.

Barker, Sean, Aditya Mishra, David Irwin, and Emmanuel Cecchet (2012). *Smart*: An open data set and tools for enabling research in sustainable homes*. In *ACM SustKDD 2012*. ISBN 9781450315586. http://wan.poly.edu/KDD2012/forms/workshop/SustKDD12/doc/SustKDD12{_}3.pdf.

Basu, K., V. Debusschere, and S. Bacha (2012). *Appliance Usage Prediction Using a Time Series Based Classification Approach*. In *Proceedings of the 38th Annual Conference on IEEE Industrial Electronics Society (IECON)*, pages 1217–1222.

Batra, N., M. Gulati, A. Singh, and M. B. Srivastava (2013). *It's Different: Insights into Home Energy Consumption in India*. In *Proceedings of the 5th ACM Workshop on Embedded Systems For Energy-Efficient Buildings*, pages 3:1–3:8. BuildSys'13, ACM, New York, NY, USA. ISBN 978-1-4503-2431-1. doi:10.1145/2528282.2528293. http://doi.acm.org/10.1145/2528282.2528293.

Ben-Hur, Asa and Jason Weston (2010). *A user's guide to support vector machines. Methods in molecular biology (Clifton, N.J.)*, 609, pages 223–239. ISSN 19406029. doi:10.1007/978-1-60327-241-4_13. http://link.springer.com/protocol/10.1007/978-1-60327-241-4{_}13.

Bengio, Yoshua, Aaron Courville, and Pascal Vincent (2013). *Representation Learning: A Review and New Perspectives. IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1993), pages 1798–1828. ISSN 01628828. doi:10.1109/TPAMI.2013.50.

Berdahl, John (2014). *Glaucoma: Types, Symptoms, Diagnosis and Treatment.* `http://www.allaboutvision.com/conditions/glaucoma.htm`.

Berges, M., E. Goldman, H. Matthews, and L. Soibelman (2008). *Training Load Monitoring Algorithms on Highly Sub-Metered Home Electricity Consumption Data. Tsinghua Science & Technology*, 13(S1), pages 406–411.

Berthold, Michael, Christian Borgelt, Frank Höppner, and Frank Klawonn (2010). *Guide to Intelligent Data Analysis.* March, Springer verlag.

Bhande, Sanjivani and Ranjan Raut (2013). *Parkinson Diagnosis using Neural Network: a Survey. International Journal of Innovative Research in Science, Engineering and Technology (IJIRSET)*, 2(9), pages 4843–4846.

Bifet, A., G. Holmes, R. Kirkby, and B. Pfahringer (2014). *Massive Online Analysis (MOA).* `http://moa.cms.waikato.ac.nz`.

Bind, Shubham, Arvind Kumar Tiwari, and Anil Kumar Sahani (2015). *A Survey of Machine Learning Based Approaches for Parkinson Disease Prediction. International Journal of Computer Science and Information Technologies (IJCSIT)*, 6(2), pages 1648–1655.

Bishop, Cm and Nm Nasrabadi (2007). *Pattern Recognition and Machine Learning*, volume 16. ISBN 9780387310732, 049901 pages. doi:10.1117/1.2819119. `http://medcontent.metapress.com/index/A65RM03P4874243N.pdf{%}5Cnhttp://www.library.wisc.edu/selectedtocs/bg0137.pdf{%}5Cnhttp://electronicimaging.spiedigitallibrary.org/article.aspx?doi=10.1117/1.2819119`.

Blume, Steven W. (2007). *Electric Power System Basics For the Nonelectrical Professional.* IEEE Press.

Bromuri, S., D. Zufferey, J. Hennebert, and M. Schumacher (2014). *Multi-label classification of chronically ill patients with bag of words and supervised dimensionality reduction algorithms. Journal of Biomedical Informatics*, 51, pages 165–175.

Bronzino, J.D. (2010). *The Biomedical Engineering HandBook, Second Edition.* Ed. Joseph D. Bronzino, Boca Raton: CRC Press LLC, 2000.

Buono, Paolo, Aleks Aris, Catherine Plaisant, Amir Khella, and Ben Shneiderman (2005). *Interactive Pattern Search in Time Series.* In *Proceedings of the Conference on Visualization and Data Analysis (VDA 2005)*, volume 5669, pages 175–186. SPIE. ISBN 1301405272. ISSN 0277786X. doi:10.1117/12.587537.

Burges, Christopher J C (1998). *. ' ' 1-43 () A Tutorial on Support Vector Machines for Pattern Recognition*. Technical Report.

Cease, T.W. and P. Johnston (1990). *A magneto-optic current transducer. Power Delivery, IEEE Transactions on*, 5(2), pages 548–555.

Chakrabarti, K., E. Keogh, M. Pazzani, and S. Mehrotra (2002). *Locally Adaptive Dimensionality Reduction for Indexing Large Time Series Databases. ACM Transactions on Database Systems*, 27, pages 188–228.

Challapalli, Kiran (2014). *The Internet of Things : A time series data challenge*. `http://www.ibmbigdatahub.com/blog/internet-things-time-series-data-challenge`.

Chan, Kin-pong and Ada Wai-chee Fu (1999). *Efficient Time Series Matching byWavelets*. Technical Report, The Chinese University of Hong Kong, Shatin, Hong Kong.

Chan, W.L., a.T.P. So, and L.L. Lai (2000). *Harmonics load signature recognition by wavelets transforms*. In *Proceeding of the International Conference on Electric Utility Deregulation and Restructuring and Power Technologies.*, pages 4–7.

Chang, Chih-Chung and Chih-Jen Lin (2013). *LIBSVM: A Library for Support Vector Machines*. Technical Report, Department of Computer Science, National Taiwan University, Taipei, Taiwan.

Chang, H. H. and J. M. F. Moura (2010). *Biomedical Engineering and Design Handbook*, 1, pages 559–579.

Chang, T. C. P. and E. A. Hodapp (2015). *Risk Factors in Glaucomatous Progression. Glaucoma Today*, pages 24–25.

Chapman, Pete, Julian Clinton, Randy Kerber, Thomas Khabaza, Thomas Reinartz, Colin Shearer, and Wirth Rudiger (2000). *CRISP-DM 1.0 - Step-by-Step data mining guide. CRISP-DM Consortium*, page 76.

Chen, C, B Das, and D J Cook (2010). *Energy Prediction Based on Resident ' s Activity. Proceedings of the International workshop on Knowledge Discovery from Sensor Data*.

Chen, G. Z., I. S. Chan, and D. C. C. Lam (2013). *Capacitive contact lens sensor for continuous non-invasive intraocular pressure monitoring. Sensors and Actuators A: Physical*, 203, pages 112–118.

Chen, G.-Z., I. S. Chan, L. K. K. Leung, and D. C. C. Lam (2014). *Soft wearable contact lens sensor for continuous intraocular pressure monitoring. Medical Engineering & Physics*, 36(9), pages 1134–1139.

Chen, Yihua and Maya R. Gupta (2010). *EM Demystified: An Expectation-Maximization Tutorial*. Technical Report 2, University ofWashington, Seattle, Washington, USA. http://scholar.google.com/scholar?hl=en{&}btnG=Search{&}q=intitle:EM+Demystified:+An+Expectation-Maximization+Tutorial{#}0.

Chih-Chung, C. and L. Chih-Jen (2005). *LIBSVM - A Library for Support Vector Machines*. http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Cho, H.S. and M. Hahn (2009). *Simple and Robust Method for Detecting the Electrical Appliances Using Markers and Programmable Logic Devices*. In *Proceedings of the 13th IEEE International Symposium on Consumer Electronics (ISCE)*, pages 334–338.

Cho, H.S., T. Yamazaki, and M. Hahn (2009). *Determining Location of Appliances from Multi-hop Tree Structures of Power Strip Type Smart Meters. IEEE Transactions on Consumer Electronics*, 55(4), pages 2314–2322.

Chrominski, Kornel and Magdalena Tkacz (2010). *Comparison of Outlier Detection Methods in Biomedical Data. journal of Medical Informatics & Technologies*, 16, pages 89–94.

Chu, M. X., K. Miyajima, D. Takahashi, T. Arakawa, K. Sano, S. Sawada, H. Kudo, Y. Iwasaki, K. Akiyoshi, M. Mochizuki, and Mitsubayashi K. (2011). *Soft contact lens biosensor for in situ monitoring of tear glucose as non-invasive blood sugar assessment. Talanta*, 83(3), pages 960–965.

Collins, C. C. (1967). *Miniature Passive Pressure Transensor for Implanting in the Eye. Biomedical Engineering, IEEE Transactions on*, BME-14(2), pages 74–83.

Cong, H. and T. Pan (2009). *Microfabrication of conductive PDMS on flexible substrates for biomedical applications*. In *Nano/Micro Engineered and Molecular Systems, 2009. NEMS 2009. 4th IEEE International Conference on*, pages 731–734.

Cornelissen, Germaine (2014). *Cosinor-based rhythmometry. Theoretical Biology and Medical Modelling*, 11(1), page 24. ISSN 1742-4682. doi:10.1186/1742-4682-11-16.

Cruz, Joseph A. and David S. Wishart (2006). *Applications of Machine Learning in Cancer Prediction and Prognosis. Cancer Informatics*, 2, pages 59–78. ISSN 11769351.

Cybenko, G. (1989). *Approximation by superpositions of a sigmoidal function. Mathematics of Control, Signals, and Systems (MCSS)*, 2(4), pages 303–314.

Danjuma, Kwetishe and Adenike Osofisan (2015). *Evaluation of Predictive Data Mining Algorithms in Erythemato-Squamous Disease Diagnosis. CoRR*, abs/1501.0, page 10. http://arxiv.org/abs/1501.00607.

Darby, Sarah (2006). *The Effectiveness of Feedback on Energy Consumption a Review for Defra of the Literature on Metering , Billing and Direct Displays. Environmental Change Institute University of Oxford*, 22(April), pages 1–21. ISSN 2169-2637. doi:10.4236/ojee.2013.21002. http://www.citeulike.org/user/cblock/article/4123593.

David, Robert, Linda Zangwill, Daniel Briscoe, Michael Dagan, Ronit Yagev, and Yuval Yassur (1992). *Diurnal intraocular pressure variations: an analysis of 690 diurnal curves. British Journal of Ophthalmology*, 76(5), pages 280–283. ISSN 0007-1161. doi:10.1136/bjo.76.5.280.

De Almeida, A. and P. Fonseca (2006). *Residential Monitoring to Decrease Energy Use and Carbon Emissions in Europe.* In *Proceedings of the 4th International Energy Efficiency in Domestic Appliances and Lighting Conference (EEDAL).*

De Vivero, C., C. O'Brien, L. Lanigan, and R. Hitchings (1994). *Diurnal intraocular pressure variation in low-tension glaucoma. Eye*, 8, pages 521–523.

Demsar, J., T. Curk, A. Erjavec, C. Gorup, T. Hocevar, M. Milutinovic, M. Mozina, M. Polajnar, M. Toplak, A Staric, M. Stajdohar, L. Umek, L. Zagar, J. Zbontar, M. Zitnik, and Zupan B. (2013). *Orange Data Mining Toolbox.* http://orange.biolab.s.

Dormehl, Luke (2014). *Why Google Is Investing In Deep Learning.* http://www.fastcolabs.com/3026423/why-google-is-investing-in-deep-learning.

Duarte, L.F.C., E.C. Ferreira, and J.A.S. Dias (2012). *Energy Measurement Techniques for Energy Efficiency Programs.* InTech, Chapters published.

Duda, Richard O., Peter E. Hart, and David G. Stork (2001). *Pattern Classification.* Second Edition. Wiley-Interscience. ISBN 0-471-05669-3, 654 pages.

Durand, Jean-Baptiste, Laurent Bozzi, Gilles Celeux, and Christian Derquenne (2004). *Analyse de courbes de consommation eÌĄlectrique par chaiÌĆnes de Markov cacheÌĄes. Revue de statistique appliquée*, 52(4), pages 71–91.

Elkan, C. (2001). *Magical Thinking in Data Mining: Lessons from Coil Challenge 2000. Proceedings of the 7th International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 426–431.

Energie-environnement.ch (2015). *Plateforme d'information Services de l'Energie et de l'Environnement des cantons de Berne, Fribourg, Geneve, Jura, Neuchatel, Valais et Vaud.* http://www.energie-environnement.ch.

Englert, F., T. Schmitt, S. Kößler, A. Reinhardt, and R. Steinmetz (2013). *How to Auto-Configure Your Smart Home? High-Resolution Power Measurements to the Rescue.* In

Proceedings of the 4th International Conference on Future energy systems (e-Energy), pages 215–224.

Esling, Philippe and Carlos Agon (2012). *Time-Series Data Mining. ACM Computing Surveys (CSUR)*, 45(1), pages 1–34. ISSN 03600300. doi:10.1145/2379776.2379788. `http://dl.acm.org/citation.cfm?doid=2379776.2379788{%}5Cnhttp://dl.acm.org/citation.cfm?id=2379788`.

Faloutsos, C., M. Ranganathan, and Y Manolopoulos (1994). *Fast subsequence matching in time-series databases.* In *ACM SIGMOD International Conference on Management of Data*, pages 419–429. ACM, Minneapolis, MN, USA.

Figueiredo, M., A. De Almeida, and B. Ribeiro (2012). *Home electrical signal disaggregation for non-intrusive load monitoring (NILM) systems. Neurocomputing*, 96, pages 66–73.

Figueiredo, Marisa B., Ana De Almeida, and Bernardete Ribeiro (2011). *An experimental study on electrical signature identification of non-intrusive load monitoring (NILM) systems. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6594 LNCS, pages 31–40.

Fitta, M. T. (2010). *Load Classification and Appliance Fingerprinting for Residential Load Monitoring System.* Master's Thesis, Aalto University, Faculty of Electronics Communications and Automation, Espoo, Finland.

Fundation, Glaucoma Research (2015). *Types of Glaucoma.* `http://www.glaucoma.org/glaucoma/types-of-glaucoma.php`.

Galin, M. A., I. Baras, and M. Best (1969). *The Nature of the Ocular Pulse. The Journal of Head and Face Pain*, 9, pages 112–128.

Gao, J., S. Giri, and E.C. Kara (2014). *Demo Abstract: PLAID - A Public Dataset of High-Resolution Electrical Appliance Measurements for Load Identification.* In *1st ACM International Conference on Embedded Systems For Energy-Efficient Buildings.*

Gayathri, B. M., C. P. Sumathi, and T. Santhanam (2013). *Breast Cancer Diagnosis Using Machine Learning Algorithms - A Survey. International Journal of Distributed and Parallel Systems (IJDPS)*, 4(3), pages 105–112.

Gayathri, P. and N. Jaisankar (2013). *Comprehensive Study of Heart Disease Diagnosis Using Data Mining and Soft Computing Techniques. International Journal of Engineering and Technology (IJET)*, 5(3), pages 2947–2958. ISSN 09754024.

Gisler, Christophe and Grazia Barchi (2012). *Demonstration Of A Monitoring Lamp To Visualize The Energy Consumption In Houses.* In *10th International Conference on Pervasive Computing (Pervasive 2012)*, pages 1–4. Newcastle, UK.

Gisler, Christophe, Antonio Ridi, Dominique Genoud, and Jean Hennebert (2014). *Towards Glaucoma Detection Using Intraocular Pressure Monitoring.* In *6th International Conference of Soft Computing and Pattern Recognition (SoCPaR 2014)*, pages 255–260. IEEE Computer Society, Tunis, Tunisie. ISBN 9781479959341.

Gisler, Christophe, Antonio Ridi, Jean Hennebert, Robert N. Weinreb, and Kaweh Mansouri (2015). *Automated Detection and Quantification of Circadian Eye Blinks Using a Contact Lens Sensor. Translational Vision Science and Technology (TVST)*, 4(1), page 10. doi:10.1167/tvst.4.1.4. `http://tvstjournal.org/doi/full/10.1167/tvst.4.1.4`.

Gisler, Christophe, Antonio Ridi, Damien Zufferey, Omar Abou Khaled, and Jean Hennebert (2013). *Appliance Consumption Signature Database and Recognition Test Protocols.* In *8th International Workshop on Systems, Signal Processing and Their Applications (WoSSPA 2013)*, pages 336–341. IEEE Computer Society, Alger, Algeria. ISBN 9781467355407. doi:10.1109/WoSSPA.2013.6602387.

Gnanasoundhari, S. J., G. Visalatchi, and M. Balamurugan (2014). *A Survey on Heart Disease Prediction System Using Data Mining Techniques. International Journal of Computer Science and Mobile Applications*, 2(2), pages 72–77.

Goldbaum, Michael H., Gil-Jin Jang, Christopher Bowd, Jiucang Hao, Linda M. Zangwill, Jeffrey Liebmann, Christopher A. Girkin, Tzyy-Ping Jung, Robert N. Weinreb, and Pamela A. Sample (2009). *Patterns of Glaucomatous Visual Field Loss in Sita Fields Automatically Identified Using Independent Component Analysis. Transactions of the American Ophthalmological Society*, 107, pages 136–144. ISSN 1545-6110.

Goldbaum, Michael H., Pamela A. Sample, Kwokleung Chan, Julia Williams, Te-Won Lee, Eytan Blumenthal, Christopher A. Girkin, Linda M. Zangwill, Christopher Bowd, Terrence Sejnowski, and Robert N. Weinreb (2002). *Comparing Machine Learning Classifiers for Diagnosing Glaucoma from Standard Automated Perimetry. Investigative Ophthalmology and Visual Science*, 43(1), pages 162–169. ISSN 01460404.

Gonçalves, Hugo, Adrian Ocneanu, and Mario Bergés (2011). *Unsupervised disaggregation of appliances using aggregated consumption data.* In *SustKDD Workshop on Data Mining Applications in Sustainability.*

Grippo, Tomas M., John H. K. Liu, Nazlee Zebardast, Taylor B. Arnold, Grant H. Moore, and Robert N. Weinreb (2013). *Twenty-four-hour pattern of intraocular pressure in untreated patients with ocular hypertension. Investigative ophthalmology & visual science*, 54(1), pages 512–517. ISSN 1552-5783. doi:10.1167/iovs.12-10709.

Guinard, Dominique, Markus Weiss, and Vlad Trifa (2009). *Are you energy-efficient? sense it on the web.* In *Adjunct Proc. Pervasive09, Nara, Japan*, pages

3–6. Citeseer. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.154.9516{&}rep=rep1{&}type=pdf.

Gutierrez-Osuna, Ricardo (2015). *Lecture 11: Sequential Feature Selection*. http://www.facweb.iitkgp.ernet.in/~sudeshna/courses/ML06/featsel.pdf.

Hampel, Frank R. (1974). *The Influence Curve and its Role in Robust Estimation*. *Journal of the American Statistical Association*, 69(346), pages 383–393. ISSN 0162-1459. doi:10.1080/01621459.1974.10482962.

Hannani, A. El and J. Hennebert (2008). *A Review of the Benefits and Issues of Speaker Verification Evaluation Campaigns*. In *Proceedings of the ELRA Workshop on Evaluation at LREC 08, Marrakech, Morocco*, pages 29–34.

Harrington, L., J. Brown, S. Holt, A. Meier, B. Nordman, and M. Ellis (2007). *Standby Energy - Building a coherent international policy framework moving to the next level. Summer Study Proceedings, European Council for an Energy Efficient Economy*, 3, pages 1285–1296.

Hart, G.W. (1992). *Nonintrusive Appliance Load Monitoring*. *Proceedings of the IEEE*, 80(12), pages 1870–1891.

Hattenhauer, M. G., D. H. Johnson, and H. H. et al. Ing (1998). *The probability of blindness from open-angle glaucoma*. *Ophthalmology*, 105(11), pages 2099–2104.

Hearst, Marti a., Susan T. Dumais, Edgar Osuna, John Platt, and Bernhard Schölkopf (1998). *Support vector machines*. *IEEE Intelligent Systems and their Applications*, 13, pages 18–28. ISSN 1094-7167. doi:10.1109/5254.708428. http://ieeexplore.ieee.org/xpls/abs{_}all.jsp?arnumber=708428.

Hennebert, Jean (1998). *HIDDEN MARKOV MODELS AND ARTIFICIAL NEURAL*. Phd thesis, Ecole Polytechnique Fédérale de Lausanne (EPFL).

Hochheiser, Harry and Ben Shneiderman (2004). *Dynamic query tools for time series data sets: Timebox widgets for interactive exploration*. *Information Visualization*, 3(1), pages 1–18. ISSN 1473-8716. doi:10.1057/palgrave.ivs.9500061.

Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI, USA.

Hsu, Chih-Wei, Chih-Chung Chang, and Chih-Jen Lin (2010). *A Practical Guide to Support Vector Classification*. Technical Report, Department of Computer Science, National Taiwan University, Taipei 106, Taiwan. http://www.csie.ntu.edu.tw/{~}cjlin/papers/guide/guide.pdf.

Huang, Y.M. and Y.X. Lai (2013). *Distributed Energy Management System within Residential Sensor-Based Heterogeneous Network Structure. Wireless Sensor Networks and Ecological Monitoring*, 3, pages 35–60.

Hughes, E., P. Spry, and J. Diamond (2003). *24-hour monitoring of intraocular pressure in glaucoma management: a retrospective review. Journal of Glaucoma*, 12(3), pages 232–236.

Hyndman, R. J. (2017). *CRAN Time Series Library for R.* `https://cran.r-project.org/web/views/TimeSeries.html`.

Instruments, National (2006). *Zero padding does not buy spectral resolution.* `http://www.ni.com/tutorial/4880/en`.

Ito, M., R. Uda, S. Ichimura, K. Tago, T. Hoshi, Y. Matsushita, and K. Hachioji (2004). *A Method of Appliance Detection Based on Features of Power Waveform.* In *Proceedings of the 4th IEEE/IPSJ International Symposium on Applications and the Internet (SAINT)*, pages 291–294.

Jerritta, S., M. Murugappan, R. Nagarajan, and K. Wan (2011). *Physiological signals based human emotion Recognition: a review.* In *Proceeding of the 7th International Colloquium on Signal Processing and its Applications (CSPA 2011)*, pages 410–415. IEEE.

Jiang, L., S. Luo, and J. Li (2012). *An Approach of Household Power Appliance Monitoring Based on Machine Learning.* In *Proceedings of the 5th IEEE International Conference on Intelligent Computation Technology and Automation (ICICTA)*, pages 577–580.

Jiang, Xiaofan, Minh Van Ly, Jay Taneja, Prabal Dutta, and David Culler (2009). *Experiences with a high-fidelity wireless building energy auditing network. Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems SenSys 09*, page 113. doi:10.1145/1644038.1644050. `http://portal.acm.org/citation.cfm?doid=1644038.1644050`.

Johnstone, M., E. Martin, and A. Jamil (2011). *Pulsatile flow into the aqueous veins: manifestations in normal and glaucomatous eyes. Experimental Eye Research*, 92(5), pages 318–327.

Kahveci, Tamer and Ambuj Singh (2001). *Variable Length Queries for Time Series Data.* In *Proceedings of 17th International Conference on Data Engineering 2001*, pages 273–282. Heidelberg, Germany.

Kamshilin, A. A., S. Miridonov, V. Teplov, R. Saarenheimo, and E. Nippolainen (2011). *Photoplethysmographic imaging of high spatial resolution. Biomedical Optics Express*, 2, pages 996–1006.

Kang, S.J., K. Shin, R. Yang, S.Y. Lee, and J. Sohn (2013). *Cost effective disaggregation mechanism for the NIALM system. International Journal of Smart Grid and Clean Energy*, 2(3), pages 364–370.

Karpagachelvi, S (2015). *Classification of ECG Signals Using Extreme Learning Machine: A Survey.* In *Proceedings of the International Conference on Advances in Computer Engineering and Applications (ICACEA 2015)*, pages 42–52. IEEE Computer Society, Ghaziabad, India. ISBN 9781467369114.

Kato, Takekazu, Hyun Sang Cho, Dongwook Lee, Tetsuo Toyomura, and Tatsuya Yamazaki (2009). *Appliance recognition from electric current signals for information-energy integrated network in home environments. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 5597 LNCS, pages 150–157. ISSN 03029743. doi:10.1007/978-3-642-02868-7_19. `http://www.springerlink.com/index/61W89262388NU301.pdf`.

Katuri, K. C., S. Asrani, and M. K. Ramasubramanian (2008). *Intraocular Pressure Monitoring Sensors. Sensors Journal, IEEE*, 8(1), pages 12–19.

Kelly, J. and W. J. Knottenbelt (2014). *UK-DALE: A dataset recording UK Domestic Appliance-Level Electricity demand and whole-house demand. CoRR*, abs/1404.0284, pages 1–13. `http://arxiv.org/abs/1404.0284`.

Keogh, E. and S. Kasetty (2002). *On the Need for Time Series Data Mining Bench- marks: A Survey and Empirical Demonstration.* In *The 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 102–111. ACM, Edmonton, Alberta, Canada.

Keogh, Eamonn, Kaushik Chakrabarti, Sharad Mehrotra, and Michael Pazzani (2001a). *Locally Adaptive Dimensionality Reduction for Indexing Large Time Series Databases.* In *ACM SIGMOD International Conference*, pages 151–162. ACM.

Keogh, Eamonn, Kaushik Chakrabarti, Michael Pazzani, and Sharad Mehrotra (2001b). *Dimensionality Reduction for Fast Similarity Search in Large Time Series Databases. Knowledge and Information Systems*, 3(3), pages 263–286.

Kim, K.Y., S.Y. Lee, and H.C. Kim (2004). *A wireless measurement system for three-dimensional ocular movement using the magnetic contact lens sensing technique.* In *Proceedings of the 26th International Conference of the IEEE Engineering in Medicine and Biology Society (IEMBS 2004)*, pages 2287–2289. IEEE.

Kinn, J.B. and R.A. Tell (1973). *A Liquid-Crystal Contact-Lens Device for Measurement of Corneal Temperature. IEEE Transactions on Biomedical Engineering*, BME-20(5), pages 387–388.

Klinkenberg, R., I Mierswa, and S. Fischer (2016). *RapidMiner*. http://rapidminer.com.

Kolter, J. Zico and Tommi Jaakkola (2012). *Approximate Inference in Additive Factorial HMMs with Application to Energy Disaggregation*. In *Proceedings of the 15th International Conference on Artificial Intelligence and Statistics (AISTATS 2012)*, volume XX, pages 1472–1482. La Palma, Canary Islands.

Kolter, J Zico and Matthew J Johnson (2011). *REDD : A Public Data Set for Energy Disaggregation Research*. In *SustKDD workshop*, volume xxxxx, pages 1–6. ISBN 9781450308403. http://users.cis.fiu.edu/{~}lzhen001/activities/KDD2011Program/workshops/WKS10/doc/SustKDD3.pdf.

Lam, H.Y., G.S.K. Fung, and W.K. Lee (2007). *A Novel Method to Construct Taxonomy of Electrical Appliances Based on Load Signatures. IEEE Transactions on Consumer Electronics*, 53(2), pages 653–660.

Lara, O. D., A. J. Pérez, M. A. Labrador, and J. D. Posada (2012). *Centinela: a human activity recognition system based on acceleration and vital sign data. Pervasive and Mobile Computing*, 8(5), pages 717–729.

Le, Quoc V., Marc'Aurelio Aurelio Ranzato, Rajat Monga, Matthieu Devin, Kai Chen, Greg S. Corrado, Jeff Dean, and Andrew Y. Ng (2011). *Building High-level Features Using Large Scale Unsupervised Learning. International Conference in Machine Learning*, page 38115. ISSN 10535888. doi:10.1109/MSP.2011.940881.

Lee, S., G. Ryu, Y. Chon, R. Ha, and H. Cha (2013). *Automatic Standby Power Management Using Usage Profiling and Prediction. IEEE Transactions on Human-Machine Systems*, 43(6), pages 535–546.

Lee, Shih Chiang, Gu Yuan Lin, Wan Rong Jih, Chi Chia Huang, and Jane Yung Jen Hsu (2012). *Energy-aware agents for detecting nonessential appliances*. doi:10.1007/978-3-642-25920-3_34.

Lee, W. K., G. S. K. Fung, H. Y. Lam, F. H. Y. Chan, and Mark Lucente (2004). *Exploration on Load Signatures*. In *Electrical Engineering*, page 5. 725, Japan.

Leeb, Steven B, James L Kirtley, Michael S Levan, and Joseph P Sweeney (1993). *Development and Validation of a Transient Event Detector. AMP Journal of Technology*, 3, pages 69–74.

Leeb, Steven B., Steven R. Shaw, and James L. Kirtley (1995). *Transient event detection in spectral envelope estimates for nonintrusive load monitoring. IEEE Transactions on Power Delivery*, 10(3), pages 1200–1210. ISSN 08858977. doi:10.1109/61.400897. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=400897.

Leonardi, M., P. Leuenberger, and D. et al. Bertrand (2004). *First steps toward noninvasive intraocular pressure monitoring with a sensing contact lens. Investigative Ophthalmology & Visual Science (IOVS)*, 45(9), pages 3113–3117.

Leys, Christophe, Christophe Ley, Olivier Klein, Philippe Bernard, and Laurent Licata (2013). *Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median. Journal of Experimental Social Psychology*, 49(4), pages 764–766. ISSN 00221031. doi:10.1016/j.jesp.2013.03.013.

Liang, Jian, Simon K. K. Ng, Gail Kendall, and John W. M. Cheng (2010). *Load Signature Study - Part I: Basic Concept, Structure, and Methodology. IEEE Transactions on Power Delivery*, 25(2), pages 551–560. ISSN 08858977. doi:10.1109/TPWRD.2009.2033799.

Lifton, J., M. Feldmeier, Y. Ono, C. Lewis, and J.A. Paradiso (2007). *A Platform for Ubiquitous Sensor Deployment in Occupational and Domestic Environments.* In *Proceedings of the 6th International Symposium on Information Processing in Sensor Networks (IPSN)*, pages 119–127.

Lin, Gu-Yuan, Shih-Chiang Lee, and Jane Yung-Jen Hsu (2009). *Sensing from the panel: Applying the Power Meters for Appliance Recognition. Environment*, 2011, pages 1–10. ISSN 10958606. doi:10.1016/j.jhevol.2011.11.006.

Lin, Jessica, Eamonn Keogh, and Stefano Lonardi (2005). *Visualizing and Discovering Non-Trivial Patterns In Large Time Series Databases. Information Visualization*, 4(2), pages 61–82. ISSN 14738716. doi:10.1057/palgrave.ivs.9500089.

Lin, Jessica, Eamonn Keogh, Stefano Lonardi, and Bill Chiu (2003). *A Symbolic Representation of Time Series, with Implications for Streaming Algorithms. Proceedings of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, pages 2–11. doi:10.1145/882082.882086.

Lin, Jessica, Eamonn Keogh, Li Wei, and Stefano Lonardi (2007). *Experiencing SAX: a Novel Symbolic Representation of Time Series. Data Mining and Knowledge Discovery*, 15(2), pages 107–144. ISSN 1573-756X.

Lipton, Zachary Chase, John Berkowitz, and Charles Elkan (2015). *A Critical Review of Recurrent Neural Networks for Sequence Learning. CoRR*, abs/1506.0, pages 1–38. ISSN 9781450330633. doi:10.1145/2647868.2654889. `http://arxiv.org/abs/1506.00019`.

Liu, J. H., D. F. Kripke, and M. D. et al. Twa (1999). *Twenty-four-hour pattern of intraocular pressure in the aging population. Investigative Ophthalmology & Visual Science (IOVS)*, 40(1), pages 2912–2917.

Liu, J. H., F. A. Medeiros, J. R. Slight, and R. N. Weinreb (2010). *Diurnal and nocturnal effects of brimonidine monotherapy on intraocular pressure. Ophthalmology*, 117(11), pages 1075–1079.

Liu, J. H. and R. N. Weinreb (2011). *Monitoring intraocular pressure for 24 h. The British Journal of Ophthalmology*, 65(5), pages 599– 600.

Liu, J. H., X Zhang, D. F. Kripke, and R. N. Weinreb (2003). *Twenty-four-hour intraocular pressure pattern associated with early glaucomatous changes. Investigative Ophthalmology & Visual Science (IOVS)*, 44(4), pages 1586–1596.

M., Leonardi, E. M. Pitchon, and A. et al. Bertsch (2009). *Wireless contact lens sensor for intraocular pressure monitoring: assessment on enucleated pig eyes. Acta Ophthalmologica*, 87(4), pages 433–437.

Maimon, Oded and Lior Rokach (2010). *Data Mining and Knowledge Discovery Handbook*. Second Edition. Springer. ISBN 978-0-387-09822-7, 1306 pages. doi:10.1007/978-0-387-09823-4.

Mak, Ginny (2000). *The Implementation of Support Vector Machines using the Sequential Minimal Optimization Algorithm*. Master thesis, McGill University, Montreal, Canada.

Makonin, S., F. Popowich, L. Bartram, B. Gill, and I.V. Bajic (2013). *AMPds : A Public Dataset for Load Disaggregation and Eco-Feedback Research*. In *IEEE Electrical Power and Energy Conference (EPEC)*.

Mansouri, K. and T. Shaarawy (2011). *Continuous intraocular pressure monitoring with a wireless ocular telemetry sensor: initial clinical experience in patients with open angle glaucoma. The British Journal of Ophthalmology*, 95(5), pages 627–629.

Mansouri, Kaweh and Robert N. Weinreb (2012). *Continuous 24-hour intraocular pressure monitoring for glaucoma - time for a paradigm change. Swiss medical weekly*, 142(March 28), pages 1–8. ISSN 14243997. doi:10.4414/smw.2012.13545.

Mapstone, R. (1968). *Determinants of corneal temperature. The British Journal of Ophthalmology*, 52(10), pages 729–741.

Marceau, M. L. and R. Zmeureanu (2000). *Nonintrusive load disaggregation computer program to estimate the energy consumption of major end uses in residential buildings. Energy Conversion and Management*, 41(13), pages 1389–1403. ISSN 01968904. doi:10.1016/S0196-8904(99)00173-9. `http://linkinghub.elsevier.com/retrieve/pii/S0196890499001739`.

Marchiori, A., D. Hakkarinen, Q. Han, and L. Earle (2011). *Circuit-Level Load Monitoring for Household Energy Management. IEEE Pervasive Computing*, 10(1), pages 40–48.

Marcu, M. and A. Stancovici (2011). *Systems classification based on power signatures.* In *Proceedings of the 2nd IEEE PES International Conference and Exhibition on Innovative Smart Grid Technologies (ISGT)*, pages 1–6.

MathWorks (2016). *MATLAB Time Series Module.* `https://ch.mathworks.com/help/matlab/time-series.html`.

McKinney, W. (20xx). *Pandas Time Series Module for Python.* `http://pandas.pydata.org/pandas-docs/stable/timeseries.html`.

Melin, H. and J. Lindberg (1999). *Variance flooring, scaling and tying for text-dependent speaker verification.* In *Proceedings of Eurospeech '99*, pages 1975–1979. Budapest, Hungary.

Mohamad, Mohd Saberi, Safaai Deris, Safie Mat Yatim, and Muhammad Razib Othman (2004). *Feature Selection Method Using Genetic Algorithm for the Classification of Small and High Dimension Data.* In *First International Symposium on Information and Communications Technologies.*, pages 1–4. Putrajaya, Malaysia.

Monacchi, A., D. Egarter, W. Elmenreich, S. D'Alessandro, and A.M. Tonello (2014). *GREEND: An Energy Consumption Dataset of Households in Italy and Austria. CoRR*, abs/1405.3100, pages 1–16. `http://arxiv.org/abs/1405.3100`.

Morel, Nicolas (2008). *Bio-Inspired Control Systems for Building Services to Save Energy Consumption and Increase Indoor Comfort.* `https://documents.epfl.ch/groups/e/en/enac-unit/www/CESS_Seminars/LesoControlCessSeminarsOct08.pdf`.

Mosaed, S., J. H. Liu, and R. N. Weinreb (2005). *Correlation between office and peak nocturnal intraocular pressures in healthy subjects and glaucoma patients. American Journal of Ophthalmology*, 139(2), pages 320–324.

Najmeddine, Hala, Khalil El Khamlichi Drissi, Christophe Pasquier, Claire Faure, Kamal Kerroum, Alioune Diop, Thierry Jouannet, and Michel Michou (2008). *State of art on load monitoring methods.* In *Proceedings of the 2nd IEEE International Conference on Power and Energy (PECon 2008)*, pages 1256–1258. IEEE Computer Society, Johor Baharu, Malaysia. ISBN 978-1-4244-2404-7. doi:10.1109/PECON.2008.4762669.

Neenan, B. (2009). *Residential Electricity Use Feedback: A Research Synthesis and Economic Framework.* Technical Report February, Electric Power Research Institute.

NIST/SEMATECH (2013). *e-Handbook of Statistical Methods.* `http://www.itl.nist.gov/div898/handbook`.

Ophthalmology, Ziemer (2015a). *DCT Lens.* `http://www.ziemertonometry.com/dct-lens.html`.

Ophthalmology, Ziemer (2015b). *Ziemer Ophthalmic Systems*. `http://www.ziemergroup.com`.

P., Gerard-Marchant and M. Knox (2009). *Scikit Time Series Module for Python*. `http://pytseries.sourceforge.net`.

Pantelopoulos, A. and N.G. Bourbakis (2010). *A Survey on Wearable Sensor-Based Systems for Health Monitoring and Prognosis. Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 40(1), pages 1–12.

Paradiso, Francesca, Federica Paganelli, Antonio Luchetta, Dino Giuli, and Pino Castrogiovanni (2013). *ANN-based Appliance Recognition from Low-frequency Energy Monitoring Data*. In *Proceedings of the 14th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM 2013)*, pages 1–6. IEEE Computer Society, Madrid, Spain. ISBN 9781467358286. doi:10.1109/WoWMoM.2013.6583496.

Patel, Shwetak N, Thomas Robertson, Julie a Kientz, Matthew S Reynolds, and Gregory D Abowd (2007). *At the Flick of a Switch : Detecting and Classifying Unique Electrical Events on the Residential Power Line*. In *UbiComp*, pages 271–288. ISBN 9783540748526. ISSN 0302-9743. doi:10.1007/978-3-540-74853-3_16.

Pavlidis, Theodosios and Steven L. Horowitz (1974). *Segmentation of Plane Curves. IEEE Transactions on Computers*, c-23(8), pages 860–870.

Pearson, Ronald K. (2002). *Outliers in Process Modeling and Identification. IEEE Transactions on Control Systems Technology*, 10(1), pages 55–63.

Pei, Min, Erik D. Goodman, William F. Punch III, and Ying Ding (1995). *Genetic Algorithms For Classification and Feature Extraction. Classification Society Conference*, pages 1–28.

Phelan, S., C. McArdle, S. Daniels, and P. Meehan (2012). *Temporal and frequency analysis of power signatures for common household appliances*. In *Procedings of the Symposium on ICT and Energy Efficiency and Workshop on Information Theory and Security (CIICT)*, pages 22–27.

Piatetsky, Gregory (2014). *KDnuggets Exclusive: Interview with Yann LeCun, Deep Learning Expert, Director of Facebook AI Lab*. `http://www.kdnuggets.com/2014/02/exclusive-yann-lecun-deep-learning-facebook-ai-lab.html`.

Platt, John C. (1998). *Fast Training of Support Vector Machines Using Sequential Minimal Optimization. Advances in kernel methods*, pages 185 – 208. doi:10.1109/ISKE.2008.4731075.

PLOGG (2014). *PLOGG Wireless Energy Management*. `http://www.plogg.co.uk`.

Plugwise (2015). *Plugwise - Energy management systems of the 21st century.* `http://www.plugwise.com`.

Popivanov, I. and R. J. Miller (2002). *Similarity Search Over Time-Series Data Using Wavelets.* In *Proceedings 18th International Conference on Data Engineering (ICDE)*, pages 212–221. IEEE Computer Society, San Jose, CA. ISBN 0-7695-1531-2. ISSN 1063-6382.

Prasad, R.S. and S. Semwal (2013). *A Simplified New Procedure for Identification of Appliances in Smart Meter Applications.* In *Proceedings of the 7th IEEE International Systems Conference (SysCon)*, pages 339–344.

Prudenzi, A. (2002). *A Neuron Nets Based Procedure for Identifying Domestic Appliances Pattern-of-Use from Energy Recordings at Meter Panel.* In *Proceedings of the 2002 IEEE Power Engineering Society Winter Meeting*, pages 941–946. IEEE Computer Society. ISBN 0-7803-7322-7. doi:10.1109/PESW.2002.985144.

Punjabi, O. S., H. K. V. Ho, and C. Kniestedta (2006). *Intraocular Pressure and Ocular Pulse Amplitude Comparisons in Different Types of Glaucoma Using Dynamic Contour Tonometry. Current Eye Research*, 31, pages 851–862.

Quigley, H. A. and A. T. Broman (2006). *The number of people with glaucoma worldwide in 2010 and 2020. The British Journal of Ophthalmology*, 90(3), pages 262–267.

Rafiei, D. and A. Mendelzon (1998). *Efficient Retrieval of Similar Time Sequences Using DFT.* In *Proceedings of the 5th International Conference on Foundations of Data Organization and Algorithms.* Kobe, Japan. `http://arxiv.org/abs/cs/9809033`.

Rakotomalala, Ricco (2013). *Tanagra.* `http://eric.univ-lyon2.fr/~ricco/tanagra/en/tanagra.html`.

Rambhajani, Madhura, Wyomesh Deepanker, and Neelam Pathak (2011). *A Survey on Implementation of Machine Learning Techniques for Dermatology Diseases Classification. International Journal of Advances in Engineering & Technology (IJAET)*, 8(2), pages 194–202.

Ratanamahatana, Chotirat Ann, Jessica Lin, Dimitrios Gunopulos, Eamonn Keogh, Michail Vlachos, and Gautam Das (2010). *Mining Time Series Data. Data Mining and Knowledge Discovery Handbook*, pages 1049–1077. doi:10.1007/978-0-387-09823-4_56.

Reinhardt, Andreas, Paul Bauman, Daniel Burgstahler, Matthias Hollick, Hristo Chonov, Marc Werner, and Ralf Steinmetz (2012a). *On the Accuracy of Appliance Identification Based on Distributed Load Metering Data. Proceedings of the 2nd IFIP Conference on Sustainable Internet and ICT for Sustainability*, (October), pages 1–9.

Reinhardt, Andreas, Dominic Burkhardt, Manzil Zaheer, and Ralf Steinmetz (2012b). *Electric Appliance Classification Based on Distributed High Resolution Current Sensing.* In *Proceedings of the 7th IEEE International Workshop on Practical Issues in Building Sensor Network Applications (SenseApp 2012)*, pages 999–1005. October, IEEE Computer Society. ISBN 978-1-4673-2128-0.

Reinhardt, Andreas, Delphine Christin, and Salil S. Kanhere (2013). *Predicting the Power Consumption of Electric Appliances through Time Series Pattern Matching.* In *Proceedings of the 5th ACM Workshop on Embedded Systems For Energy-Efficient Buildings (BuildSys)*, pages 1–2. November, ACM.

Research, Wolfram (2014). *Mathematica Time Series Module.* `https://www.wolfram.com/mathematica/new-in-10/time-series`.

Resende, Arthur Fernandes, Edward Stephen Yung, Michael Waisbourd, Arthur Fernandes, Edward Stephen Yung, Michael Waisbourd, Arthur Fernandes, and Stephen Yung (2015). *Monitoring intra ocular pressure in glaucoma : current recommendations and emerging cutting-edge technologies. Expert Review of Ophthalmology*, 9899(October), page 15. doi:10.1586/17469899.2015.1100539.

Ridi, Antonio (2016). *Generative Models for Time Series in the Context of In-Home Monitoring.* Phd thesis, University of Fribourg, Switzerland.

Ridi, Antonio, Christophe Gisler, and Jean Hennebert (2013). *Unseen Appliances Identification.* In *18th Iberoamerican Congress on Pattern Recognition (CIARP 2013)*, volume 8259 LNCS, pages 75–82. Springer-Verlag Berlin Heidelberg, La Habana, Cuba. ISBN 9783642418266. ISSN 03029743. doi:10.1007/978-3-642-41827-3_10.

Ridi, Antonio, Christophe Gisler, and Jean Hennebert (2014a). *A Survey on Intrusive Load Monitoring for Appliance Recognition.* In *22nd International Conference on Pattern Recognition (ICPR 2014)*, pages 3702–3707. IEEE Computer Society, Stockholm, Sweden. doi:10.1109/ICPR.2014.636.

Ridi, Antonio, Christophe Gisler, and Jean Hennebert (2014b). *ACS-F2 - A New Database of Appliance Consumption Signatures.* In *6th International Conference of Soft Computing and Pattern Recognition (SoCPaR 2014)*, pages 145–150. IEEE Computer Society, Tunis, Tunisie. ISBN 9781479959341.

Ridi, Antonio, Christophe Gisler, and Jean Hennebert (2015). *User Interaction Event Detection in the Context of Appliance Monitoring.* In *Proceedings of the 2015 IEEE International Conference on Pervasive Computing and Communications (PerCom 2015)*, pages 323–328. IEEE Computer Society, St. Louis, USA. ISBN 978-1-4799-8425-1. doi:10.1109/PERCOMW.2015.7134056. `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7134056`.

RNIB, RCOphth (2015). *Glaucoma.* `http://www.aberdeeneyes.com/index.php/eye-conditions/glaucoma`.

Robert, L. and K. Liszewski (2013). *Methods of Electrical Appliances Identification in Systems Monitoring Electrical Energy Consumption.* In *Proceedings of the 7th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS)*, pages 10–14.

Robinson, D.A. (1963). *A Method of Measuring Eye Movemnent Using a Scleral Search Coil in a Magnetic Field. IEEE Transactions on Bio-medical Electronics*, 10(4), pages 137–145.

Ross, J.P. and L. Berkeley (2000). *Whole-House Measurements of Standby Power Consumption.* In *Proceedings of the 2nd International Conference on Energy Efficiency in Household Appliances (EEDAL).*

Ruzzelli, a. G., C. Nicolas, a. Schoofs, and G. M P O'Hare (2010). *Real-time recognition and profiling of appliances through a single electricity sensor.* Technical Report, School of Computer Science and Informatics, University College Dublin, Ireland, Dublin, Ireland. doi:10.1109/SECON.2010.5508244.

Sahu, Garima and Rakesh Kumar Khare (2014). *A Survey on classification & feature selection technique based ensemble models in health care domain. International Journal on Computer Technology & Applications (IJCTA)*, 5(3), pages 957–962. `http://www.ijcta.com`.

Saitoh, Takeshi, Yuuki Aota, Tomoyuki Osaki, Ryosuke Konishi, and Kazunori Sugahara (2008). *Current Sensor based Non-intrusive Appliance Recognition for Intelligent Outlet. Signals*, 37, pages 349–352. doi:10.1306/74D717A3-2B21-11D7-8648000102C1865D.

Sajda, Paul (2006). *Machine Learning for Detection and Diagnosis of Disease. Annual Review of Biomedical Engineering*, 8(April), pages 537–565. ISSN 1523-9829. doi:10.1146/annurev.bioeng.8.061505.095802.

Salem, Abdel-Badeeh M., Kenneth Revett, and El-Sayed A. El-Dahshan (2009). *Machine Learning in Electrocardiogram Diagnosis.* In *International Multiconference on Computer Science and Information Technology (IMCSIT)*, pages 429–433. ISBN 9788360810224.

Salvatore, G. A., N. MÃijnzenrieder, T. Kinkeldei, L. Petti, C. Zysset, I. Strebel, L. BÃijthe, and G. TrÃűster (2013). *Wafer-scale design of lightweight and transparent electronics that wraps around hairs. Nature Communication*, 5.

Sample, Pamela A., Michael H. Goldbaum, Kwokleung Chan, Catherine Boden, Te-Won Lee, Christiana Vasile, Andreas G. Boehm, Terrence Sejnowski, Chris A. Johnson, and Robert N. Weinreb (2002). *Using Machine Learning Classifiers to Identify Glaucomatous Change Earlier in Standard Visual Fields. Investigative Ophthalmology and Visual Science*, 43(8), pages 2660–2665. ISSN 01460404.

Sandhya, Joshi, Simha G. G. Vibhudendra, Shenoy P. Deepa, K. R. Venugopal, and L. M. Patnaik (2010). *Classification and Treatment of Different Stages of Alzheimer's Disease using Various Machine Learning Methods. International Journal of Bioinformatics Research*, 2(1), pages 44–52.

Satyanandam, N., Ch. Satyanarayana, Md. Riyazuddin, and Amjan Shaik (2012). *Data Mining Machine Learning Approaches and Medical Diagnose Systems : A Survey. International Journal of Computer & Organization Trends*, 2(3), pages 53–60.

Schmidt, K. G., A. von RÃijckmann, and T. W. Mittag (1998). *Ocular pulse amplitude in ocular hypertension and open-angle glaucoma. International Journal of Ophthalmology*, 212, pages 5–10.

Schoofs, A., A. Guerrieri, D.T. Delaney, G.M.P. O'Hare, and A.G. Ruzzelli (2010a). *ANNOT - Automated Electricity Data Annotation Using Wireless Sensor Networks.* In *Proceedings of the 7th Annual IEEE Conference on Sensor Mesh and Ad Hoc Communications and Networks (SECON)*, pages 1–9.

Schoofs, A., A.G. Ruzzelli, and G.M.P. O'Hare (2010b). *Appliance Activity Monitoring Using Wireless Sensors.* In *Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, pages 434–435.

Schwenn, O., R. Troost, A. Vogel, F. Grus, S. Beck, and N. Pfeiffer (2002). *Ocular pulse amplitude in patients with open angle glaucoma, normal tension glaucoma, and ocular hypertension. The British Journal of Ophthalmology*, 86, pages 981–985.

Scott, V.A., L. Tarassenko, C.J. Glynn, and A.R. Hill (1996). *Contact lens oximetry: a valid concept?* In *IEE Colloquium on Pulse Oximetry: A Critical Appraisal*, pages 1–3.

Senin, Pavel and Sergey Malinchik (2013). *SAX-VSM: Interpretable Time Series Classification Using SAX and Vector Space Model. Proceedings of the 13th International Conference on Data Mining, ICDM*, pages 1175–1180. ISSN 15504786. doi:10.1109/ICDM.2013.52.

Senior, M. (2014). *Novartis signs up for Google smart lens. Nature Biotechnology*, 32(856).

Sensimed (2014a). *Sensimed AG, Lausanne, Switzerland.* `http://www.sensimed.ch`.

Sensimed (2014b). *Triggerfish Contact Lens Sensor.* `http://www.sensimed.ch/en/sensimed-triggerfish/sensimed-triggerfish.html`.

Serra, Henrique, João Correia, António J. Gano, António M. De Campos, and Isabel Teixeira (2005). *Domestic Power Consumption Measurement and Automatic Home appliance detection. Proceedings of the 4th IEEE International Workshop on Intelligent Signal Processing (WISP 2005)*, pages 128–132. doi:10.1109/WISP.2005.1531645.

Shahabi, C., X. Tian, and W. Zhao (2000). *TSA-Tree: A Wavelet Based Approach to Improve the Efficiency of Multi-Level Surprise and Trend Queries.* In *In proceedings of the 12th International Conference on Scientific and Statistical Database Management*, pages 55–68. Berlin, Germany.

Shankaracharya, Devang Odedra, Subir Samanta, and Ambarish S. Vidyarthi (2010). *Computational Intelligence in Early Diabetes Diagnosis: A Review. Review of Diabetic Studies*, 7(4), pages 252–261. ISSN 16136071. doi:10.1900/RDS.2010.7.252. `www.The-RDS.org`.

Singh, K. and A. J. Sit (2011). *Intraocular pressure variability and glaucoma risk: Complex and controversial. Archives of Ophthalmology*, 129(8), pages 1080–1081.

Smola, Alexander J., Peter Bartlett, Bernhard Schölkopf, and Dale Schuurmans (2000). *Advances in Large Margin Classifiers.* MIT Press, Cambridge, MA, USA. ISBN 0262194481, 61 pages.

Stalmans, I., A. Harris, V. Vanbellinghen, T. Zeyen, and B. Siesky (2008). *Ocular Pulse Amplitude in Normal Tension and Primary Open Angle Glaucoma. Journal of Glaucoma*, 17, pages 403–407.

Sultanem, F. (1991). *Using Appliance Signatures for Monitoring Residential Loads ot Meter Panel Level. IEEE Transactions on Power Delivery*, 6(4), pages 1380–1385.

Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le (2014). *Sequence to Sequence Learning with Neural Networks.* pages 1–9. `http://arxiv.org/abs/1409.3215`.

Tapia, E.M., S.S. Intille, L. Lopez, and K. Larson (2006). *The Design of a Portable Kit of Wireless Sensors for Naturalistic Data Collection.* In *Proceedings of the 4th International Conference on Pervasive Computing (PerCom).*

Telegesis (2015). *Telegesis ETRX357USB ZigBee Transceiver.* `http://www.telegesis.com/products/etrx3-based-products/etrx3-usb-sticks`.

Tielsch, J. M., J. Katz, K. Singh, H. A. Quigley, J. D. Gottsch, J. Javitt, and A. Sommer (1991). *A population-based evaluation of glaucoma screening: the Baltimore Eye Survey. American Journal of Epidemiology*, 134(10).

Tripathi, K. (2011). *Important Physiological Signals in the body.* `http://biomedikal.in/2011/05/important-physiological-signals-in-the-body`.

Tufte, Edward R. (1986). *The Visual Display of Quantitative Information.* Second Edition Edition. Graphics Press, Cheshire, CT, USA.

Tukey, J. W. (1977). *Exploratory Data Analysis.* Addison-Wesley, Menlo Park, Cambridge, MA, USA. ISBN 0201076160, 688 pages.

Twa, M. D., C. J. Roberts, H. J. Karol, A. M. Mahmoud, P. A. Weber, and R. H. Small (2010). *Evaluation of a Contact Lens-Embedded Sensor for Intraocular Pressure Measurement. Journal of Glaucoma*, 19(6), pages 382–390.

Van Wijk, Jarke J. and Edward R. Van Selow (1999). *Cluster and Calendar based Visualization of Time Series Data.* In *Proceedings 1999 IEEE Symposium on Information Visualization (InfoVis '99)*, page 6. IEEE, San Francisco, CA, USA. ISBN 0-7695-0431-0. ISSN 1522-404X. doi:10.1109/INFVIS.1999.801851.

Vapnik, Vladimir N. (1999). *An Overview of Statistical Learning Theory. IEEE transactions on neural networks / a publication of the IEEE Neural Networks Council*, 10(5), pages 988–999. ISSN 1045-9227. doi:10.1109/72.788640.

Watt-ICT (2013-2016). *The Watt-ICT Project.* `http://www.wattict.com`.

Weber, Marc, Marc Alexa, and Wolfgang Müller (2001). *Visualizing Time-Series on Spirals.* In *Proceedings of the IEEE Symposium on Information Visualization (InfoVis)*, pages 7–14. IEEE Computer Society, San Diego, CA, USA. ISBN 0-7695-7342-5. ISSN 1522-404X. doi:10.1109/INFVIS.2001.963273.

Weinreb, R. N. and P. T. Khaw (2004). *Primary open-angle glaucoma. Lancet*, 363(9422), pages 1711–1720.

Weinreb, R. N. and J. H. Liu (2006). *Nocturnal rhythms of intraocular pressure. Archives of Ophthalmology*, 124(2), pages 269–270.

Weiss, Markus, Adrian Helfenstein, Friedemann Mattern, and Thorsten Staake (2012). *Leveraging smart meter data to recognize home appliances.* In *Proceedings of the 2012 IEEE International Conference on Pervasive Computing and Communications*, pages 190–197. March, IEEE Computer Society, Lugano, Switzerland. ISBN 978-1-4673-0258-6. doi:10.1109/PerCom.2012.6199866. `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6199866`.

Weiss, Markus, Thorsten Staake, Dominique Guinard, and Wolf Roediger (2009). *eMeter : An interactive energy monitor. UbiComp*, pages 3–4. `http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.154.9440`.

Witten, Ian H., Eibe Frank, and Mark A. Hall (2014). *Waikato Environment for Knowledge Analysis (Weka).* `http://www.cs.waikato.ac.nz/~ml/weka`.

Wu, Yi-Leh, Divyakant Agrawal, and Amr El Abbadi (2000). *A Comparison of DFT and DWT Based Similarity Search in Time-series Databases. Proceedings of the 9th International Conference on Information and Knowledge Management (CIKM)*, 35(2000-08), pages 488–495. doi:10.1145/354756.354857. `http://portal.acm.org/citation.cfm?doid=354756.354857`.

Xue, Jing H. and D. Michael Titterington (2008). *Comment on "On Discriminative vs. Generative Classifiers: A comparison of logistic regression and naive Bayes". Neural Processing Letters*, 28(3), pages 169–187. ISSN 13704621. doi:10.1007/s11063-008-9088-7.

Yan, J. (2011). *An unpowered, wireless contact lens pressure sensor for point-of-care glaucoma diagnosis.* In *Engineering in Medicine and Biology Society, EMBC, 2011 Annual International Conference of the IEEE*, pages 2522–2525.

Yang, Jihoon and Vasant Honavar (1998). *Feature Subset Selection Using a Genetic Algorithm. IEEE Intelligent Systems*, 13(1), pages 44–49.

Yao, H., A. Afanasiev, I. Lahdesmaki, and B.A. Parviz (2011). *A dual microscale glucose sensor on a contact lens, tested in conditions mimicking the eye.* In *Micro Electro Mechanical Systems (MEMS), 2011 IEEE 24th International Conference on*, pages 25–28.

Yao, H., C. Marcheselli, A. Afanasiev, I. Lahdesmaki, and B.A. Parviz (2012). *A soft hydrogel contact lens with an encapsulated sensor for tear glucose monitoring.* In *Micro Electro Mechanical Systems (MEMS), 2012 IEEE 25th International Conference on*, pages 769–772.

Yi, Byoung-Kee and Christos Faloutsos (2000). *Fast Time Sequence Indexing for Arbitrary Lp norms. Proceedings of the 26st International Conference on VLDB*, pages 385–394. ISSN 1553-3514. doi:10.1016/j.cppeds.2011.05.001.

Yu, Lei and Huan Liu (2003). *Feature Selection for High-Dimensional Data: A Fast Correlation-Based Filter Solution.* In *Proceedings of the 20th International Conference on Machine Learning (ICML 2003)*, pages 1–8. AAAI Press, Washington DC, USA. ISBN 1-57735-189-4. ISSN 01469592. doi:citeulike-article-id:3398512. `http://www.aaai.org/Papers/ICML/2003/ICML03-111.pdf`.

Zaidi, Adeel Abbas, Friederich Kupzog, Tehseen Zia, and Peter Palensky (2010). *Load recognition for automated demand response in microgrids. IECON Proceedings (Industrial Electronics Conference)*, pages 2442–2447. ISSN 1553-572X. doi:10.1109/IECON.2010.5675022. `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5675022`.

Zeifman, M. (2012). *Disaggregation of Home Energy Display Data Using Probabilistic Approach. IEEE Transaction on Consumer Electronics*, 58(1), pages 23–31.

Zhang, Zhuo, Ruchir Srivastava, Huiying Liu, Xiangyu Chen, Lixin Duan, Damon Wing Kee Wong, Chee Keong Kwoh, Tien Yin Wong, and Jiang Liu (2014). *A survey on computer aided diagnosis for ocular diseases. BMC Medical Informatics and Decision Making*, 14(80), pages 1–29. ISSN 1472-6947. doi:10.1186/1472-6947-14-80. `http://www.biomedcentral.com/1472-6947/14/80`.

Zhao, Jian, Fanny Chevalier, and Ravin Balakrishnan (2011). *KronoMiner: Using Multi-Foci Navigation for the Visual Exploration of Time-Series Data.* In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI 2011)*, pages 1737–1746. ACM, Vancouver, BC, Canada. ISBN 9781450302678. doi:10.1145/1978942.1979195. `http://dl.acm.org/citation.cfm?id=1979195`.

Zia, Tehseen, Dietmar Bruckner, and Adeel Zaidi (2011). *A Hidden Markov Model Based Procedure for Identifying Household Electric Loads.* In *IECON 2011 - 37th Annual Conference on IEEE Industrial Electronics Society*, pages 3218–3223. ISBN 9781612849720.

Zimmermann, J.P. (2009). *End-use metering campaign in 400 households In Sweden Assessment of the Potential Electricity Savings.* Technical Report, Enertech.

Zoha, Ahmed, Alexander Gluhak, Muhammad Ali Imran, and Sutharshan Rajasegarar (2012). *Non-Intrusive Load Monitoring Approaches for Disaggregated Energy Sensing: A Survey. Sensors (Switzerland)*, 12(12), pages 16838–16866. ISSN 14248220. doi:10.3390/s121216838.

Zufferey, Damien, Christophe Gisler, Omar Abou Khaled, and Jean Hennebert (2012). *Machine Learning Approaches for Electric Appliance Classification.* In *11th International Conference on Information Sciences, Signal Processing and their Applications (ISSPA 2012)*, pages 757–762. IEEE Computer Society, Montreal, Canada. ISBN 9781467303828.

Curriculum Vitæ

Personal Data

First name	Christophe
Last name	Gisler
Date of birth	January 28, 1982
Nationality	Swiss
Languages	French (native), English (fluent), German (basic knowledge)

Education & Diploma

2011 – 2017	University of Fribourg, Switzerland PhD of Science in Computer Science (Doctor Scientiarum Informaticarum)
2011 – 2013	University of Fribourg, Switzerland Diploma in Higher Teaching and Education Technology (Diploma of Advanced Studies)
2006 – 2008	University of Fribourg, Switzerland Master of Science in Computer Science
2003 – 2006	University of Fribourg, Switzerland Bachelor of Science in Informatics with Minor in Mathematics
2002 – 2003	Swiss School of Osteopathy, Belmont-sur-Lausanne, Switzerland First year exam (passed) in Osteopathic Medicine
2001 – 2002	Swiss Federal Institute of Technology (EPFL), Lausanne First year exam (passed) in Computer Science
1997 – 2001	St-Michel High School, Fribourg, Switzerland Swiss Federal Maturity in Science (Type C)

Academical & Work Experience

10/2009 – Present	High School of Engineering and Architecture of Fribourg, member of the University of Applied Sciences Western Switzerland (HES-SO) Scientific collaborator in the Institute of Complex Systems (iCoSys) Work on various projects in machine learning, biomedical informatics (modeling of intraocular pressure curves for glaucoma diagnosis) and Green IT (modeling of electrical appliance consumption signals in buildings). Setting up of projects as the CTI Sensimed (*Glaucoma Prognostication Platform*) Teaching and supervision of BSc and MSc students
01/2011 – 04/2017	University of Fribourg, Department of Informatics Ph.D. student in the DIVA research group (Document Image and Voice Analysis) in collaboration with the High School of Engineering and Architecture
09/2013 – 09/2014	University of Fribourg, Department of Biology. Scientific collaborator Main work on a MATLAB tool for the tracking of drosophila larvae and a Java tool for the automation of biological experiments
08/2008 – 07/2009	Swisscom Strategy and Innovation, Bern, external engineer Design and Java development for a mobile web advertising service
08/2008 – 07/2009	University of Fribourg, Department of Informatics Research and teaching assistant in the DIVA research group
10/2007 – 04/2008	University of Fribourg and Swisscom Strategy and Innovation. Master Thesis: *SymbiosArt - An Interactive Painting Based on Image Selection and Voice Input*
01/2006 – 06/2006	University of Fribourg, Bachelor Thesis: *DET4J - Une Application Java de visualisation de courbes de performance biométriques*

Technical Skills

Fields	Machine learning, pattern recognition, statistics, image and signal processing, health/biomedical informatics, sustainable IT, big data
Programming	Java, Python, Swift, C++, PHP, HTML, JavaScript, Shell scripts, XML, SQL
Applications	Eclipse, Xcode, Matlab, Mathematica, Scikit-learn, LaTeX, GIT, Adobe CS, Microsoft Office
Platforms	Mac OS, iOS, UNIX/Linux, Windows
Methodologies	Scrum, Extreme Programming (XP)

Centers of Interest & Hobbies

Music	Jazz musician (drums and piano) in several bands with professionals and amateurs
Sport & Nature	Taekwondo (red belt), diving (CMAS P2 license), hiking, fishing

List of Publications

2015

Gisler, C., Ridi, A., Hennebert, J., Weinreb, R. N. and Mansouri K. (2015). *Automated Detection and Quantification of Circadian Eye Blinks Using a Contact Lens Sensor.* Translational Vision Science and Technology (TVST), vol. 4, no. 1, article 4.

Ridi, A., Zarkadis, N., Gisler, C. and Hennebert, J. (2015). *Duration Models for Activity Recognition and Prediction in Buildings using Hidden Markov Models.* Proceedings of the 2015 International Conference on Data Science and Advanced Analytics (DSAA 2015), 10 pages. Paris, France. IEEE Computer Society.

Ridi, A., Gisler, C. and Hennebert, J. (2015). *Processing Smart Plug Signals Using Machine Learning.* In Proceedings of the 2015 IEEE Wireless Communications and Networking Conference (WCNC 2015), pages 75–80. New Orleans, USA. IEEE Computer Society.

Ridi, A., Gisler, C. and Hennebert, J. (2015). *User Interaction Event Detection in the Context of Appliance Monitoring.* In Proceedings of the 2015 IEEE International Conference on Pervasive Computing and Communications (PerCom 2015), pages 323–328. St. Louis, USA. IEEE Computer Society.

2014

Gisler, C., Ridi, A., Genoud, D. and Hennebert, J. (2014). *Towards Glaucoma Detection Using Intraocular Pressure Monitoring.* In 6th International Conference of Soft Computing and Pattern Recognition (SoCPaR 2014), pages 255–260. Tunis, Tunisie. IEEE Computer Society.

Ridi, A., Gisler, C. and Hennebert, J. (2014). *ACS-F2 – A New Database of Appliance Consumption Signatures.* In 6th International Conference of Soft Computing and Pattern Recognition (SoCPaR 2014), pages 145–150. Tunis, Tunisie. IEEE Computer Society.

Ridi, A., Gisler, C. and Hennebert, J. (2014). *A Survey on Intrusive Load Monitoring for Appliance Recognition.* In 22nd International Conference on Pattern Recognition (ICPR 2014), pages 3702–3707. Stockholm, Sweden. IEEE Computer Society.

Ridi, A., Gisler, C. and Hennebert, J. (2014). *Appliance and State Recognition using Hidden Markov Models.* In Proceedings of the 2014 International Conference on Data Science and Advanced Analytics (DSAA 2014), pages 270–276. Shanghai, China. IEEE Computer Society.

2013

Gisler, C., Ridi, A., Zufferey, D., Khaled, O. A. and Hennebert, J. (2013). *Appliance Consumption Signature Database and Recognition Test Protocols.* In 8th International Workshop on Systems, Signal Processing and Their Applications (WoSSPA 2013), pages 336–341. Alger, Algeria. IEEE Computer Society.

Ridi, A., Gisler, C. and Hennebert, J. (2013). *Unseen Appliances Identification.* In 18th Iberoamerican Congress on Pattern Recognition (CIARP 2013), LNCS 8259, pages 75–82. La Habana, Cuba. Springer-Verlag Berlin Heidelberg.

Ridi, A., Gisler, C. and Hennebert, J. (2013). *Automatic Identification of Electrical Appliances Using Smart Plugs.* In 8th International Workshop on Systems, Signal Processing and Their Applications (WoSSPA 2013), pages 301–305. Alger, Algeria. IEEE Computer Society.

Ridi, A., Gisler, C. and Hennebert, J. (2013). *Le machine learning : un atout pour une meilleure efficacité.* Electro-Suisse (CISBAT), no. 10s, 4 pages. `http://www.electrosuisse.ch`.

2012

Gisler, C. and Barchi, G. (2012). *Demonstration Of A Monitoring Lamp To Visualize The Energy Consumption In Houses.* In 10th International Conference on Pervasive Computing (Pervasive 2012), pages 1–4. Newcastle, UK.

Zufferey, D., Gisler, C., Abou Khaled, O. and Hennebert, J. (2012). *Machine Learning Approaches for Electric Appliance Classification.* In 11th International Conference on Information Sciences, Signal Processing and their Applications (ISSPA 2012), pages 757–762. Montreal, Canada. IEEE Computer Society.