Automatic classification of natural signals for environmental monitoring

Marielle Malfante

Communauté UNIVERSITÉ Grenoble Alpes

THÈSE

pour obtenir le grade de

DOCTEUR DE LA COMMUNAUTÉ UNIVERSITÉ GRENOBLE ALPES

Spécialité : **Océan, Atmosphère, Hydrologie (CEOAH)**

Arrêté ministériel : 25 mai 2016

Présentée par

Marielle MALFANTE

Thèse dirigée par **Jérôme MARS**, Professeur,
et codirigée par **Mauro DALLA MURA**, Maître de Conférences,

préparée au sein du **Laboratoire Grenoble Images Parole Signal Automatique (GIPSA-LAB)**
dans l'école doctorale **Terre, Univers, Environnement (TUE)**

Automatic classification of natural signals for environmental monitoring

Thèse soutenue publiquement le **3 Octobre 2018**,
devant le jury composé de :

Monsieur Christian JUTTEN
Professeur, Université Grenoble Alpes, GIPSA-LAB, Président du jury
Monsieur Stéphane CANU
Professeur, INSA Rouen, LITIS, Rapporteur
Monsieur Ronan FABLET
Professeur, IMT Atlantique, LabSTICC, Rapporteur
Madame Carole NAHUM
Responsable du domaine scientifique environnement et géosciences DGA/DS/MRIS, Examinateur
Madame Eléonore STUTZMANN
Professeur, Institut de Physique du Globe de Paris, Examinateur
Monsieur Jérôme MARS
Professeur, Grenoble INP, GIPSA-LAB, Directeur de thèse
Monsieur Mauro DALLA MURA
Maître de Conférences, Grenoble INP, GIPSA-LAB, Co-directeur de thèse
Madame Odile GERARD
DGA Techniques navales/SDT/SCN/LSM, Invité
Monsieur Jean-Philippe METAXIAN
Chargé de Recherche, IRD, Institut de Physique du Globe de Paris, Invité

"All models are wrong, but some are useful."

— George E. P. Box,
Statistician & Professor at the University of Wisconsin

Remerciements

Cher lecteur,

Bienvenue dans ma thèse. Le présent (pavé) manuscrit relate le contexte, la démarche et les résultats associés à mes 3 ans de recherche effectués au GIPSA-Lab.

Ce que ce manuscrit ne relate pas concernant ma thèse pourrait emplir des volumes. Il y aurait là d'ailleurs matière à un bon roman. De la grande aventure de la thèse resteront donc quelques lignes de code, le présent manuscrit, et un sacré paquet de souvenirs. On ne parlera donc ni des alertes incendie à l'hôtel lors des conférences américaines, ni des hôteliers morts à Vienne, ni des hôpitaux indonésiens, ni de l'extrêmement grand nombre d'autres anecdotes ici censurées. N'ayant pas encore le privilège d'être extrêmement vieille, je m'abstiens d'utiliser ces remerciements pour m'épandre sur un peu plus de trois ans de souvenirs, et me contente de remercier en toute sobriété toutes les personnes et institutions qui m'ont permis de réaliser ma thèse dans d'excellentes conditions. On m'avait prévenu en commençant que la thèse était une aventure sacrément compliquée. Je fais partie des chanceux qui ne gardent que le souvenir de l'aventure.

Je commence donc par remercier mon jury de thèse, pour avoir pris le temps de lire et d'évaluer mes travaux. Merci donc à Christian Jutten, Stéphane Canu, Ronan Fablet, Carole Nahum, Eléonore Stutzmann, Odile Gérard, et Jean-Philippe Métaxian.

Je remercie également la DGA/MRIS Geosciences et le Labex OSUG@2020 (Investissements d'avenir - ANR10 LABX56) pour avoir financé ces trois ans de recherche. Merci également à l'école doctorale Terre Univers Environnement, l'IDEX de l'Université Grenoble Alpes, le Data Institute et au GIPSA-Lab qui ont permis de m'envoyer en Indonésie pour deux missions "longue durée" (la collaboration avec les observatoires indonésiens aurait été bien moins tangible et fructueuse sans les deux missions), mais aussi au Canada, aux Etats-Unis, en Espagne, en Autriche, en Grèce et au Japon (et à Brest aussi).

Dans le rôle des mentors qui ont su être toujours présents sans jamais m'étouffer, je tiens bien évidemment à remercier Jérôme et Mauro. Je suis particulièrement privilégiée quant à l'encadrement que j'ai reçu pour ma thèse, et je sais que c'est loin d'être une évidence. Merci de m'avoir mis sur les rails. Merci de d'avoir été disponibles pour d'innombrables discussions et conseils pendant ces trois ans, et merci de m'avoir laissé faire tout ce que

je voulais. La confiance gratuite (de mon point de vue en tout cas), c'est rare. Merci aussi d'avoir été ambitieux : on trouve beaucoup de gens qui sont ambitieux pour eux-mêmes, beaucoup mois qui sont ambitieux pour le groupe. Et occasionnellement, merci de ne pas m'avoir écorchée vive le jour où j'ai renversée une tasse de thé sur mon ordinateur. [1]

Un très grand merci également à Jean-Philippe pour la collaboration sur le Mérapi et Ubinas. J'ai énormément apprécié travailler sur le sujet, c'était une superbe opportunité et j'en suis ravie. Merci également à François Beauducel, Budi Santoso, Pak Wiwit and Robiah Al Wardah pour la collaboration avec le BPPTKG. Je n'avais pas envie de partir en fin de thèse en laissant simplement mes travaux sur un disque dur, c'est gratifiant de savoir que les algos tournent quelque part de l'autre côté de la planète. J'apprécie d'avoir pu contribuer à mon échelle. Merci également à Orlando Macedo et Adolfo Inza pour leur expertise sur Ubinas. Merci aussi à Omar Mohammed pour la collaboration orientée réseaux neuronaux.

Un énorme merci à l'étage administratif du GIPSA-Lab où je crois que j'ai acquis une réputation de phobique administrative. Merci donc à Lucia pour avoir facilité mon arrivée au laboratoire (et pour toutes les conversations et éclats de rire qui ont suivis pendant 3 ans), mais également aux ... huit gestionnaires successives de SIGMAPHY (Fanny et Sonia en tête) ainsi qu'à Akila pour avoir pris le temps de me sortir fréquemment de la panade. Promis, je m'améliorerai en ce qui concerne les procédures administratives (un jour).

Merci également à bon nombre de chercheurs du GIPSA pour les nombreuses discussions et conversations qui, pour la plupart, ont été formatrices, intéressantes, constructives, et bienveillantes. J'ai beaucoup appris du monde de la recherche et de l'enseignement à votre contact.

On en arrive à l'essentiel : merci les copains.

Les très vieux, les vieux, les copains de mon année, les désormais vieux nouveaux, et les bébés tous frais, en passant par les irréductibles, les copains de summer school, les fanfarons, et les ex-besties. Après un nouveau week-end passé en votre compagnie : mille mercis, vous êtes géniaux.

Marion, Fabrice, Raphaël, Kévin, Marc, Pedro, Miguel, Lucas, Victor, Quentin, Céline, Cindy, Taia, Tim de G, Tim G, Pascal, Guillaume, Pierre N, Alexandre, Quyen, Pierre M, Jeanne, Aziliz, Emmanuelle, Pedro, Jitendra, Paolo, Mélisandre, Alexis, Camille, Théo, Florent, Marine, Omar, Gaël, Fanny, la team Cadiz avec Ana et Alex, Ryan, Branko, Rémy, à l'ensemble des Pinks (des adultes mais pas trop non plus, vous êtes mes héros), Sarah, Flo, Pierre L., Magali, Julien, Chloe, Hugues, Pierre P, Youcef, Simon, Laura, Johan. Et bien sûr, Albin.

Dans le rôle des taverniers ayant accueillis cette charmante[2] troupe des compagnons d'infortune[3], merci au Family's d'avoir hébergé un très

[1] NB: la machine vit.
[2] notion subjective
[3] pour le style littéraire, la vérité étant nettement moins dramatique

grand nombre de soirées entre copains, et au Jardin du thé de m'avoir laissé squatter une table des journées (semaines) entières pendant la rédaction. J'espère ne pas avoir trop usé la chaise.

Et pour cette très jolie fin d'aventure, merci à tous ceux qui sont venus remplir la salle mont blanc le jour de ma soutenance. Je n'avais pas anticipé que ce serait aussi agréable de parler devant une salle pleine. Mention spéciale aux copains qui travaillaient mais qui se sont relayés pour passer pendant la présentation, les questions ou le pot (Cécilia, Quentin, Seb, Manon, Thomas), à la famille qui est venue avec l'assurance de faire beaucoup de route, d'attendre des heures et de ne pas comprendre grand chose, mais avec une fierté dans les yeux qui ne sera pas oubliée (Livio, Lucette, Maman, Papa, Gautier, et Claire par la pensée). Merci évidemment à tous les copains du labo (présents ou passés) d'avoir affrontés la session question en plus de la présentation. Et un énorme merci à Albin et Gautier sans qui le pot de thèse aurait été bien moins réussi (je note d'ailleurs pour la prochaine fois de me mettre une assiette de côté).

Et pour finir, merci au lecteur qui parcourt ce qui finalement, est mon premier livre.

Je ferme désormais cette page de vie qu'était « la grande aventure de ma thèse » pour partir vers d'autres horizons[4], j'espère que la lecture de ce qu'il en reste t'apportera ce que tu y cherches !

[4]l'autre côté de Grenoble quoi, c'est ce que cette ville est bigrement attachante

Abstract

This manuscript summarizes a three years work addressing the use of machine learning for the automatic analysis of natural signals. The main goal of this PhD is to produce efficient and operative frameworks for the analysis of environmental signals, in order to gather knowledge and better understand the considered environment. Particularly, we focus on the automatic tasks of detection and classification of natural events.

This thesis proposes two tools based on supervised machine learning (Support Vector Machine, Random Forest) for (i) the automatic classification of events and (ii) the automatic detection and classification of events. The success of the proposed approaches lies in the feature space used to represent the signals. This relies on a detailed description of the raw acquisitions in various domains: temporal, spectral and cepstral. A comparison with features extracted using convolutional neural networks (deep learning) is also made, and favours the physical features to the use of deep learning methods to represent transient signals.

The proposed tools are tested and validated on real world acquisitions from different environments: (i) underwater and (ii) volcanic areas. The first application considered in this thesis is devoted to the monitoring of coastal underwater areas using acoustic signals: continuous recordings are analyzed to automatically detect and classify fish sounds. A day to day pattern in the fish behavior is revealed. The second application targets volcanoes monitoring: the proposed system classifies seismic events into categories, which can be associated to different phases of the internal activity of volcanoes. The study is conducted on six years of volcano-seismic data recorded on Ubinas volcano (Peru). In particular, the outcomes of the proposed automatic classification system helped in the discovery of misclassifications in the manual annotation of the recordings. In addition, the proposed automatic classification framework of volcano-seismic signals has been deployed and tested in Indonesia for the monitoring of Mount Merapi. The software implementation of the framework developed in this thesis has been collected in the Automatic Analysis Architecture (AAA) package and is freely available.

Résumé

Ce manuscrit de thèse résume trois ans de travaux sur l'utilisation des méthodes d'apprentissage statistique pour l'analyse automatique de signaux naturels. L'objectif principal est de présenter des outils efficaces et opérationnels pour l'analyse de signaux environnementaux, en vue de mieux connaitre et comprendre l'environnement considéré. On se concentre en particulier sur les tâches de détection et de classification automatique d'événements naturels.

Dans cette thèse, deux outils basés sur l'apprentissage supervisé (Support Vector Machine et Random Forest) sont présentés pour (i) la classification automatique d'événements, et (ii) pour la détection et classification automatique d'événements. La robustesse des approches proposées résulte de l'espace des descripteurs dans lequel sont représentés les signaux. Les enregistrements y sont en effet décrits dans plusieurs espaces: temporel, fréquentiel et quéfrentiel. Une comparaison avec des descripteurs issus de réseaux de neurones convolutionnels (Deep Learning) est également proposée, et favorise les descripteurs issus de la physique au détriment des approches basées sur l'apprentissage profond.

Les outils proposés au cours de cette thèse sont testés et validés sur des enregistrements in situ de deux environnements différents : (i) milieux marins et (ii) zones volcaniques. La première application s'intéresse aux signaux acoustiques pour la surveillance des zones sous-marines côtières : les enregistrements continus sont automatiquement analysés pour détecter et classifier les différents sons de poissons. Une périodicité quotidienne est mise en évidence. La seconde application vise la surveillance volcanique : l'architecture proposée classifie automatiquement les événements sismiques en plusieurs catégories, associées à diverses activités du volcan. L'étude est menée sur 6 ans de données volcano-sismiques enregistrées sur le volcan Ubinas (Pérou). L'analyse automatique a en particulier permis d'identifier des erreurs de classification faites dans l'analyse manuelle originale. L'architecture pour la classification automatique d'événements volcano-sismiques a également été déployée et testée en observatoire en Indonésie pour la surveillance du volcan Mérapi. Les outils développés au cours de cette thèse sont rassemblés dans le module Architecture d'Analyse Automatique (AAA), disponible en libre accès.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Contents

Introduction

The main topic of this PhD and of this manuscript is the automatic processing of environmental data. In this introduction chapter, I hope to provide the reader with general pieces of information regarding the context and content of this thesis. In particular, we will address the following questions:

What is behind the title and what is this thesis about?

Why should we investigate this particular topic?

Upon which scientific fields is this work built?

And finally, what answers and contributions do we provide to the scientific community?

We will also give keys to reading the manuscript and detail on its general organization. The contributions of this work are detailed at the end of this introduction chapter, along with a glossary of most important terms used in this manuscript. This manuscript is written with a will to be as clear, easy to read and understandable as possible, and we hope it will be of use to the greatest number. Enjoy the reading!

1.1 "Automatic Classification of Natural Signals for Environmental Monitoring": an Attempt at Demystifying the Problem.

Let us consider an environment. It could be an ants terrarium, your backyard garden, the Amazonian forest, the bottom of the sea, a volcano, your favorite mountain or even the Moon. Any environment. As curious living beings, humans have always – and probably always will – asked questions and wonder about everything and anything. So let us have a look at a considered environment (Figure 1). *What is happening inside? Is it evolving? Can we guess how it will look like on tomorrow?* To answer those questions, we look at our environment, we take measurements, and we study them. From a pluviometry measurement in our backyard garden, we know whether the night was rainy or not. And from pluviometry measurements over a few years, we know how dry the land is compared to the previous years, and if the plants need watering. In other words, we gather knowledge on our environment from a simple parameter. From this simple parameter evolution, we infer more knowledge on our environment. This information is non-trivial since it then helps to predict the evolution of the environment. For more complex environments and more tangible problematics, it can even influence our decision making process. An underlying and key point behind this purposefully extremely simple example, is that studying some parameters of a given environment allows us a retrieve information and knowledge on the environment itself. The monitoring of a environment is done through the study of one or several parameters. This PhD is built around this idea.

If this example regarding a backyard garden and plants growth is rather trivial, the situation, issues and more importantly consequences, remain similar when considering more complex environments. And as it stands, two main points separate the context of this PhD from this example.

The first one is the concerned environment: in this manuscript we focus on areas that are difficult to access (i.e., under-water and volcano areas). Such environments are not only hard to observe, but also difficult to measure. If an ant terrarium is small and easy to observe, if your backward

Figure 1
Considered and unknown environment.

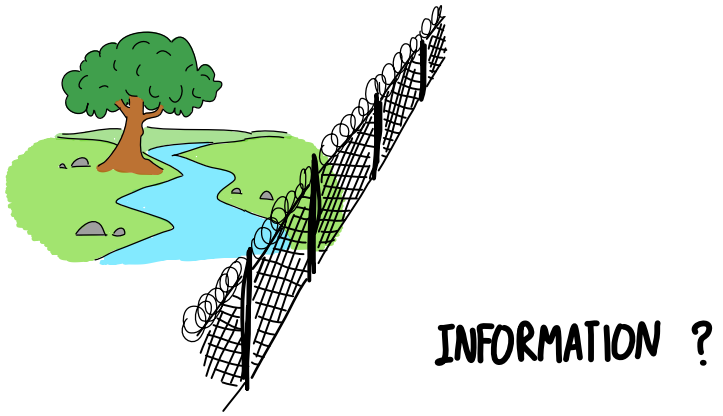


Figure 2
Considered and unknown environment, separated from us.

garden is easily accessible, conditions are completely different for the heart of the Amazonian forest, the Moon, or oceans. The environment is physically separated from us, and the question of its access is real (Figure 2). Recording measurements becomes a challenging task. A consequence and solution to this issue is to make use of technological developments by working with sensors that collect the wanted data. Once set up, those sensors can run continuously to send measurements and signals in real time. If the use of continuous and remote sensors is not exactly new, autonomous recording systems are quite recent and such systems have open a window on even more remote environment, such as underwater areas for example. A direct consequence of this point is the frequency rate at which data are sampled. If manual measurements are relatively sparse, the use of a sensor lead to a continuous flow of measures. At some point, the manual analysis is no longer possible.

The second point comes from the environment size. If small and simple environments can easily be studied by a person, the task is more complex when considering a growth and change of scale. We can either consider the evolution of geographic and/or temporal scales: considering large and extended environments and/or studying them continuously for years or decades significantly complicates the task. Instead of considering a pluviometry measure once in a while, we consider continuous flows of measurements for several years or decades, possibly on various locations. This accumulation of data is a well known phenomenon often referred as big data, and has consequences well beyond the scientific community. Typically: From storing a few photo albums, we now store gigabytes of numerical photos. From having a nice library of albums, we now access any artist discography in a few "clicks" and for a few euros. The general trend goes toward a data accumulation, and if the broader consequences on society – privacy issues for instance – are well beyond the scope of this work, the issue of environmental monitoring has clearly been impacted by the big data phenomenon. A straightforward consequence to this data accumulation is the cost[1] needed to store, analyze and process them (Figure 3). In particular, the information retrieval question is raised: *What information are we looking for among the data? How to proceed to extract such a piece of*

[1] energy, human effort, which eventually lead to an economic cost

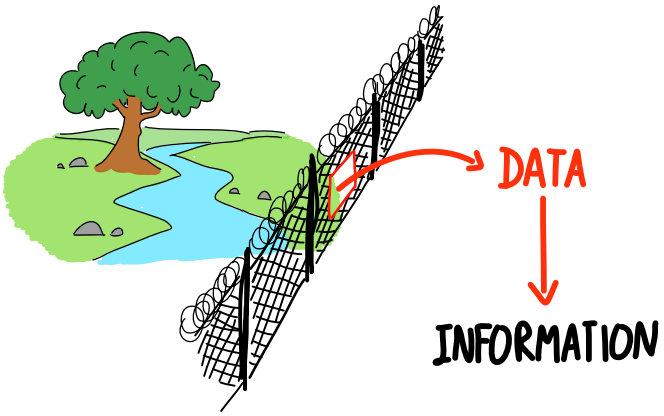


Figure 3
Use of on field data to gather knowledge about a considered and unknown environment.

information? Those questions will be addressed in this manuscript. They can almost be considered universal to the scientific community: they are shared by many communities, and are relevant regardless the applicative field. Research around those questions is of importance, and is therefore funded. Typically, this PhD has been funded by the DGA/MRIS, and by the OSUG@2020, and received travel grants from the doctoral school "*Terre Univers Environnement*" (TUE), the IDEX, and the Data Institute.

Exposing those two points naturally conducts us to an elucidation of this manuscript title, and at the same time to state the goal of this PhD. This is also the opportunity to give the reasons behind and justifications for this study.

Research in machine learning is today relatively advanced. The original theory, ideas and methods emerged a few decades ago, even if more and more complex methods are still developed [HTF09, GBC16]. Machine learning has been successful in different fields: human versus computer games [Sam59, Tes95, Hsu99, SHM+16], computer vision [SZ14, KFF15, SKB13, GEB15] and speech processing [CCC+17, Wan03]. Machine learning is those fields that has been widely investigated, is still being improved and leads to innovative and operational technological achievements. The reason why those specific tasks were chosen to illustrate machine learning algorithms is likely related to the availability of datasets, which is today extremely easy for image and speech data for instance. For applicative fields where shaped dataset are not common, and/or publicly available, machine learning is not yet widespread, and operative tools are yet to be developed. Environmental data register in those applicative fields where machine learning tools are needed, but not common yet. In this work, we contribute to the state of the art, by proposing general and robust schemes for the automatic analysis of natural signals.

Choosing a title of a few words to summarize a three years long project can be a challenging task. Once settled upon however, it carries quite a lot of meaning and for this reason, we take the time to detail and expose the underlying concepts carried by each word of the title. Many concepts and words will be introduced. Some of them are easily understandable, but to avoid any confusion and misunderstanding related to the chosen terminology, a section is dedicated to the concepts and words definitions

used through this manuscript will follow (Section 1.4).

> **"Automatic classification of natural signals for environmental monitoring"**

- **Automatic** — As previously stated, this study registers within the big data phenomenon, where we face too many signals to process them all manually. We are therefore aiming at automatic tools, that will answer this issue. The use of the word *automatic* also implies that the tools we are looking for are somewhere around a field named machine learning [HTF09].

- **Classification** — Up to this point, we purposefully remained vague on the kind of analysis we are aiming at, simply saying that we want to process natural signals. By performing an automatic analysis on a set of signals, we face many various possibilities. This analysis of automatic processing methods is general to many types of data, whether natural signals or not. We can distinguish two main types of analysis [HTF09]. Strictly speaking, all methods do not fit within this binary view but it is helpful to have a general view of machine learning approaches. The first type of analysis is to replicate a human analysis in which typically, a person is looking for something in the set of signals. This something is referred as elements of a class, and several classes can be considered. In image processing for example, a typical classification task would consist in describing an image. Is it a `cat`? Is it a `dog`? Is it `something else`? Technically speaking, `cat` and `dog` are refer as classes, and more specifically as positive classes. In this manuscript, classes are written with `this specific font`. Elements of a class can be referred as examples, data, elements, or as observations in the case of natural signals. `Something else` can be referred as the negative class, since it does not contain what we are looking for - in this case `cats` and `dogs`. In many applicative fields, the classification task is very easily done by hand[2]. But as previously explained, the task can be time consuming in a context of big data and methods referred to as supervised machine learning were therefore developed. Those methods will be more thoroughly described in Chapter 2 but the overall idea is to build a prediction model to replicate a given task. Look for cats and dogs in those images, Is there a piano playing in this audio recording? Automatic processing or analysis are very general terms, full of many different possibilities. Classification is one of them, and is the focus of this work. Those classification methods (supervised machine learning) [HTF09] can be summarized as using a large dataset of already classified observations to teach a machine to classify new observations. We will come

[2] In computer vision for instance, the human brain very easily describes and summarizes the information. In other fields the data are too complex and machine learning methods are also used to understand the data, not only to automate a task (remote sensing for instance)

back to this point when exposing the objectives of our study. The second type of analysis is out of the scope of this work and consists in exploring a set of data. The objective is then to extract information from a set of data, without knowing what information we are looking for. Those methods are referred as unsupervised machine learning methods [HTF09], and analysis models are built upon a set of data, without any prior human[3] knowledge on those data. Chapter 2 gives a more thorough background on machine learning methods.

- **Natural** — The context of this PhD is to work with signals that have a natural cause. This point however, is more to contextualize the focus of the PhD, since the theory and proposed solutions would be left unchanged if we were to deal with other types of data. The word observations is used for to describe the natural data input in the analysis model, while elements or examples refer to more general data.

- **Signals** — Up to this point, we have used different words to name the physical quantity that we record and work on (data, signals, observations, examples, etc.). We here introduce more concepts that will be used throughout the manuscript, and explain the differences between all data-related vocabulary. As a precaution, we remind that terminology related to data in the literature can vary significantly from one domain to another. A glossary of chosen words used in this manuscript can be found in Section 1.4. We call recordings the continuous physical value that is measured on the environment. A recording is therefore a signal of natural origin, we can for instance consider acoustic recordings. We refer to an events as something of interest happening inside the environment and that is relevant for its monitoring. The events imprint are visible on the recordings, and are referred as signatures. Events signatures are sorted into the classes. If the environment is a nearby street and the signals of interest are the acoustic waves recorded at one point of the street. A passing car for instance, can be considered as an event of one class, and a barking dog as an event of another class. The sound of the passing car therefore is the signature of the class `presence of car` and the noise of a barking dog the signature of the class `presence of dog`. We refer to observation as a small portion of the recording that we want to analyze and classify (typically, recordings can be analyzed with a sliding window, leading to a large number of observations). The term data is used in a more generic way. We also clarify that during this thesis, signals are always digital, and consideration regarding their digitization will not be discussed. But for more information on the topic, the reader is welcome to read to [RG75].

- **Environmental Monitoring** — This part of the title is easily understandable, but remains purposefully vague. More details on the

[3] This point is particularly important to understand the philosophy behind machine learning methods, and in particular to understand their limitations. More details will be given in Chapter 2.

kind of monitoring targeted will be given in the chapters dedicated to the various applications (Underwater acoustic in Chapter 4 and volcano-seismic in Chapter 5). Purposefully, we remain generic on the applications considered for this study, for the methods and proposed tools are designed for transient signals, and not for a given application. For now, let us simply keep in mind that the overall goal of the tools build in this PhD is for the environmental monitoring – even if the same theory and methods could easily be applied on other types of data and for goals beyond environmental monitoring. Environmental monitoring of a given environment can be done in many different way. Typically, the scale at which the environment is studied is of importance, and sensors are chosen and positioned according to the targeted goal. An ocean for example can be studied as a whole (hundred of squared kilometers), regionally (10x10km^2), locally (1x1km^2), or at even smaller scales (100x100m^2).

Once explained, the title summarizes reasonably well what is the goal of this work: build tools that automatically classify signals recorded from an environment, in order to monitor this environment. The main goal is therefore to build a model that automatically classify the data that are recorded on the environment into categories that – roughly speaking – mean something. The main question is how to. This manuscript provides answers and showcases to this question.

1.2 Objectives, Constraints and Contributions

The main objective of this manuscript is first to give an answer to the automatic classification issue in the case of environmental data. Secondly, to give sufficient theoretical materials on those tools to understand their mechanisms, strengths and limitations. Thirdly, this manuscript is also written as a tutorial that can be used to deploy the proposed tools for new applications. Fourthly, tools proposed in this manuscript are used to analyze various environmental. The analysis of those results is of great importance to understand and illustrate the use of the proposed tools. Finally, we took great care to develop tools that can be shared, easily used for other applications, and useful in operative context. All tools developed within the frame of this PhD are available on GitHub [Mal18], and contributions to the code are welcomed.

One of the main constraints on this work is to keep the tools as general as possible: an incredible amount of studies use machine learning methods to classify signals of a certain type, and for a given application. Typically, we chose not to develop the applications context and constraints in this introduction. We here aim at building tools that can be used regardless of the application. In order to underline the fact that the proposed tools can be used for other applications, we therefore present this work by dissociating theory and applications. The major goal is not to build a model that has once successfully classified a dataset but to propose a general framework of tools that can be used to analyze data. However, we precise that

we would not recommend using the tools without taking time to understand the data, the applicative field, and its constraints. Machine learning is here a tool to understand the data, and to extract relevant information from a dataset, but knowledge of the applicative expert has always been present. One of the success of this work is the collaborations it led to, and we take this opportunity to stress their value (more detail on the application can be found Chapter 4 and Chapter 5). Another main constraint is to put a great importance on building operative tools that can be used in environmental observatories. A real will and effort was put to develop tools that can be shared, but also deployed in operational contexts. This work is difficult and require time: the use of new tools in environmental observatories is always a tricky stage, including training of the local teams, and maintenance issues among others.

The main contributions of this work that are described in this manuscript are:

- The work on the observations representations or feature vectors. The final feature vectors are extracted from the observations represented in several domains, thereby leading to a precise and specific description of the data.
- The framework for the automatic classification of transient signals.
- The framework for the automatic detection and classification of transient signals.
- The application to the underwater monitoring of coastal areas, with:
  - a manual analysis of the dataset content,
  - a validation of the proposed models in various recording places,
  - a study of the influence of the various parameters used in the proposed architecture,
  - and the automatic analysis of several days of recording, registered in different underwater areas.
- The application to volcanic monitoring, with:
  - a validation of the proposed models,
  - a study of the influence of the various parameters used in the proposed architecture,
  - the automatic analysis of six years of recording of Ubinas volcano, which showed that in crisis period the automatic model performed better than the human operators, and was able to detect changes in the observations that were missed by the human eye.,
  - a simulation of the operative monitoring of Ubinas,
  - the deployment of the Automatic Analysis Architecture in Indonesia for the monitoring of the volcano Merapi [Mal18], with the use of trust index.

- The code of all those applications, which is available for real time or a posteriori analysis, available under CeCILL license and open to contributions [Mal18].
- An investigation and first approach on the use of neural networks for this issue of environmental monitoring.
- Publications and communications related to this PhD project are listed in Appendix A.

1.3 General Organization & Keys to Reading the Manuscript

The work presented in this manuscript is build on many different fields: signal processing, information retrieval, machine learning, computer science, acoustics, seismic, biology and geology. It is my hope that this manuscript can be read by people with or without a background any of those fields. To do so, each chapter will propose a very general, simple and accessible introduction which will give the necessary keys to have a general understanding of the chapter And each chapter finishes by an easy to read summary entitled *Highlights and Summary*. Technical concepts exposed in this manuscript are first introduced in a popularized way before being more strictly explained. Depending on his background and his interest, the reader is welcome to chose which view he prefers to spend reading time on, and eventually to skip some sections. Introduction and end of chapter summaries are easily noticeable sections, and are written over a gray background.

This manuscript can be read under different angles. Depending on the reader's interest, not all chapters must be read. If skipping some chapters however, we would recommend the reader to read the popularized parts of the skipped chapters: the *Introduction* and the *Highlights and Summary* sections. For readers interested in machine learning, Chapter 2 can be read independently, as an introduction and tutorial on the subject. For readers interested in under-water acoustic or in volcano-seismic, Chapters 4 and 5 can be read independently. Some of the technical material exposed in these chapters is reminded if needed (but for a better comprehension, we would advice to at least read the popularized *Introduction* and the *Highlights and Summary* section of the Chapter 2 and Chapter 3). For readers interested in the use of convolutional neural networks to represent environmental signals, Chapter 4 is advised and can be read independently, with more information about the data in Chapter 4. Finally, for the eventual reader who would read this manuscript top to bottom, you might find some repetition, which are the natural consequence of the relatively independent chapters. For this, we apologize, do not hesitate to skip the material with which you are familiar.

The general organization of the manuscript is as follow:

- **Introduction** — The present introduction chapter exposes the general context of this PhD and give insights on what to expect in the

following pages. Some key concepts are also introduced and detail on the vocabulary used in this manuscript are given.

- **Chapter 2** — This chapter gives all the theoretical material needed to understand this manuscript. In particular, we focus on the signals representation issue and on machine learning. Purposefully, this chapter is not related to the context of this PhD in the sense that it can be read separately, as a tutorial and article or a course. Some sections of this chapter are also popularized, with an emphasis on high level explanation rather than on the understanding of the mathematical concepts. And while both views are equally important, we hope to propose a balance between the overall comprehension and the scientific rigor in this chapter. Many documents of the literature provide many details on the mathematical concepts, but we judged important to also provide easily understandable explanations of those concepts. This chapter is therefore destined for readers with or without a background on machine learning or signal representation. Readers with a strong background can skip this chapter – we would in this case advice to carefully read the glossary of Section 1.4 to be sure of the terminology used in the manuscript, especially regarding the data. This chapter is mainly based on the literature. For readers without a background on machine learning, we would advice spending some time on this chapter, at least on the concepts popularized explanations if not on the more technical parts. The introduction and easy to read summary propose a condensed version of the concepts if you are in a hurry.

- **Chapter 3** — This chapters details the proposed approaches for the automatic processing of environmental signals. It gathers and explain the major contributions of this PhD: from the proposed architectures to the feature extraction scheme and its consequences. In particular, the general approach for the analysis of environmental signals and its consequences is detailed. Both analysis schemes, namely dedicated to the automatic classification and to the automatic detection and classification of environmental data are presented. A comparison between both approaches is also proposed. Finally, a feature extraction scheme for time series is proposed and explained. This chapter is necessary for the comprehension of this manuscript. By nature this chapter is bound to be rather technical, but detailed explanations are given in order to avoid a too heavy result. All major notions needed to follow through this chapter are given in the introduction and in Chapter 2.

- **Chapters 4 and 5** — Both chapters are applicative chapters and use the analysis scheme proposed in Chapter 3 on real world data. Chapter 4 is dedicated to the underwater acoustic and fish sounds while Chapter 5 focuses on volcano-seismic signals and on volcanic monitoring. Both chapter introduce the general applicative context, and in particular related constraints. A state of the art concerning the automatic analysis methods in those fields is presented. Precisions are given on the data used for this study, on the analysis that is

performed. Results are then detailed, along with an analysis of the considered environment. Both chapters can be read with or without any knowledge on the applicative field since the needed general material is given in their introduction. Both chapters can also be read independently, even if Chapter 3 is a recommended prerequisite.

- **Chapter 6** — This last chapter is rather different from the previous ones, since it does not offer a straightforward answer or application to the the automatic classification of natural signals issue. It answers some prospects on this work. Namely, it deals with the choice of signals representation, and question the choice to design features (chapters 2 and 3). In particular, deep learning models are used to extract features, and results are compared to the proposed approach. The study is based upon the use of technical tools that are first introduced, and challenges the work hypothesis of this PhD. The data used are the one of Chapter 4 This chapter can be considered as going beyond the strict topic of this PhD and its reading is not necessary for the comprehension of this work. However working hypothesis and some of the many questions raised by the previous chapters are questioned and answered in this chapter. This chapter is interesting for the research point of view and for future developments. Sections exposing the involved concepts and tools are technical, but are also popularized in order to be as understandable as possible. This chapter is build upon all of the previous chapters and would be better understood if they had been read or reviewed first. However it is also presented as independent studies that can be reasonably well understood independently from the rest of this manuscript.

1.4 Glossary

Machine learning and signal processing methods are used in for wide applicative fields, each applicative domain using a different terminology. We here propose a glossary of the terminology used in this manuscript. The glossary is set into three parts, dedicated to the main terminology used for the data, for the data representations, and to machine learning in general. We hope that it will improve the overall comprehension and prevent any misunderstanding related to the word choice. In the manuscript, words of the glossary are displayed in blue each time they are used in a new paragraph.

1.4.1 Data

class – Category in which data are gathered. A class is usually associated to a physical interpretation. Typically in computer vision, a model trained to recognize object has learnt hundreds of classes, including `cats` or `dogs` for example. In this manuscript classes are written using `this font`, and are generically noted as c_i, with $1 \leq i \leq C$ and C the number of considered classes.. xv, xvii, xix,

an ensemble of recordings, usually gathered under the same experimental conditions, and at least describing the same phenomenon (i.e. set of recording of the seismic signals around a volcano). In this manuscript the set of recordings all come from the same data gathering campaign. xvi, xvii, xix, 8, 10, 14, 15, 25, 49, 53, 63, 67, 73–75, 79, 83, 98, 99, 135

signal – Wide and generic term referring to row of values recorded from a physical quantity evolving through time. In this manuscript, signals are considered to be digitalized, the measure is therefore discrete and not continuous. Typically a signal can be an acoustic, pressure, temperature row of values. In this manuscript, a generic signal of n sample is written as $[z_i]_{i=1}^{n}$. It can reffer to a time or frequency signal for instance.. xvi, xvii, xix, xx, 5, 7–12, 14, 15, 25, 49–51, 56, 57, 72, 73, 97, 101, 135

signature – The signal associated to an event. Note that a signature does not necessarily correspond to an observation. xvi, 8, 14, 48, 51, 53, 63, 75

1.4.2 Representing the Data

domain – See space. xx, 10, 16, 56, 97, 135

cepstral domain is a space originally used to represent speech data. The main idea is to represent the observations in a space underlying the harmonic properties of the signals. To do so, the Fourier transform is computed twice.[4]. Observations represented in this domain are noted $\mathcal{S}_q = |\mathcal{F}\{S_f\}|$, and the space unit is known as the quefrency.. 16, 30, 58, 63, 96

spectral domain refers in this manuscript as the module of the Fourier transform. observations represented in the spectral domain are noted $S_f = |\mathcal{F}\{x_i\}|$. 58, 63, 96, 135

feature – Value (or set of values) used to represent an observation. It can be a physical measurement over the observation, for instance the mean value. xvii–xx, 15, 16, 25, 27–29, 31, 36, 38, 39, 43, 52, 55, 63, 72, 80, 103, 104, 119–121, 123, 124, 136

high level – In this manuscript we refer to a high level feature as a features of dimension 1. In practice, it means that the feature vector associated to an observation can be shifted randomly without changing the information in contain. This property is of importance for machine learning algorithms. 15, 26, 30, 31, 55

low level – To the contrary, low level features have a dimension superior to 1. The feature vector associated to an observation cannot be randomly shifted without altering the information it contains. xix, 15, 16, 26, 30, 56–58, 63, 72, 96, 124

[4]Historically speaking, the cepstral transform is computed using a discrete cosine transform over the logarithm of the spectrum module. The leading idea being to underline lower frequencies and harmonic properties. We here compute the cepstral transform by applying the Fourier transform over the spectrum module. More details are given in Chapter 3.2

qualitative – Features are said to be qualitative if they are categorical. For instance `Sunny`, `Rainy` and `Windy` are quantitative features. 33

quantitative – To the contrary, numerical features are said to be quantitative. 16, 33

feature vector – Observation represented in the feature space. In this manuscrit, we write feature vector as $\mathbf{x}$ (bold font for vector notation). A set of N feature vectors is written $\mathbf{X} = \mathbf{x_i}(i = 1)^N$, with $\mathbf{x_i} = x_{i\,j=1}^{j\,d}$, for $1 \leq i \leq N$. d refers to the feature vector dimension, and x^j to the j-iest component of the feature vector, for $1 \leq j \leq d$.. xv, xvi, xix, 10, 15, 16, 25, 26, 29, 37, 49, 52, 54, 56, 59, 63, 81, 82, 96, 117, 125, 132

quefrency – Unit of the cepstral domain. 15, 58, 135

representation – See feature vector. xix, 10, 25, 26, 29, 30, 55–58, 63, 96, 117, 120, 123, 124, 135

space – We refer to space or domain as the mathematical space in which observations can be represented. It can be a feature space, which is quite generic, or a more specific space such as the time domain for instance. The time domain is the original domain of the observations while spectral, cepstral or feature domains require a transformation over the observations. Various spaces represent the same observations differently, thereby underlying different properties. The transformation can be reversible or not (with or without loss) depending on the considered space.. 15, 16, 25

feature space – Space of the features. xv, xvi, 16, 25–27, 29, 35, 38, 39, 135

input space refers to the input space of the learning algorithm. It can be a the feature space (advisable), but also the original observations space, or a low level domain (non-advisable). 26

temporal – Original domain of the observations. Compared to the recordings domain, the observations have already been preprocessed, and are noted s_k. 58, 63, 96, 135

1.4.3 Machine Learning

analysis model – See prediction model. xv–xvii, xx, 8, 17, 20, 23, 26–28, 33, 44, 48, 49, 63, 102, 107, 108

classification models are trained to predict discreet outputs (categories, classes). xviii, 25, 29, 32, 33, 35, 49, 51, 52, 117, 119, 121, 124

cost function – Function used during the training of a prediction model. The function measures the prediction errors made by the model on the learning dataset, and is minimized by the machine learning algorithm to produce the prediction model. 16, 32

learning – Similar to training. 16, 23, 32, 36, 46

loss function – See cost function. 32

machine learning is a concept englobing a wide set of methods. The concept is used for many different purposes, and it is difficult to find two similar definitions of the concept. A definition of machine learning could be a set of methods aiming at building an analysis model from a dataset. The analysis can be very varied. A general introduction on machine learning can be found in chapter 2. xv, 6, 7, 12, 15–17, 20–25, 35, 42, 55, 68, 117, 135, 138

deep learning is a subset of methods based on neural networks with a large number of hidden layers. Many different network architecture are being used for different various purposes, and this set of methods has had a great number of success during the last years. Deep learning is sometimes studied independently from machine learning, the computational aspect of neural network being relatively different (extremely large dataset, computational cost, use of low level programming language, GPU servers, etc). 17, 23, 117, 119, 120, 132, 136

supervised machine learning refers to the machine learning methods building an analysis model from a labeled dataset. xv, xix, 7, 14, 22, 27, 28, 32, 33, 36, 42, 46, 49, 63, 80, 119, 138

unsupervised machine learning refers to the machine learning methods building an analysis model from a unlabeled dataset. xv, 8, 22, 27, 119

prediction model is build during the training phase. More commonly, we refer to a prediction model as a model, or as an analysis model. 7, 14, 16, 17, 25, 32, 33, 52–54, 63

regression models are trained to predict continuous outputs. 32, 33, 35

training is the phase during which an analysis model is build by a machine learning algorithm, using a dataset. 16, 17, 23, 32, 46

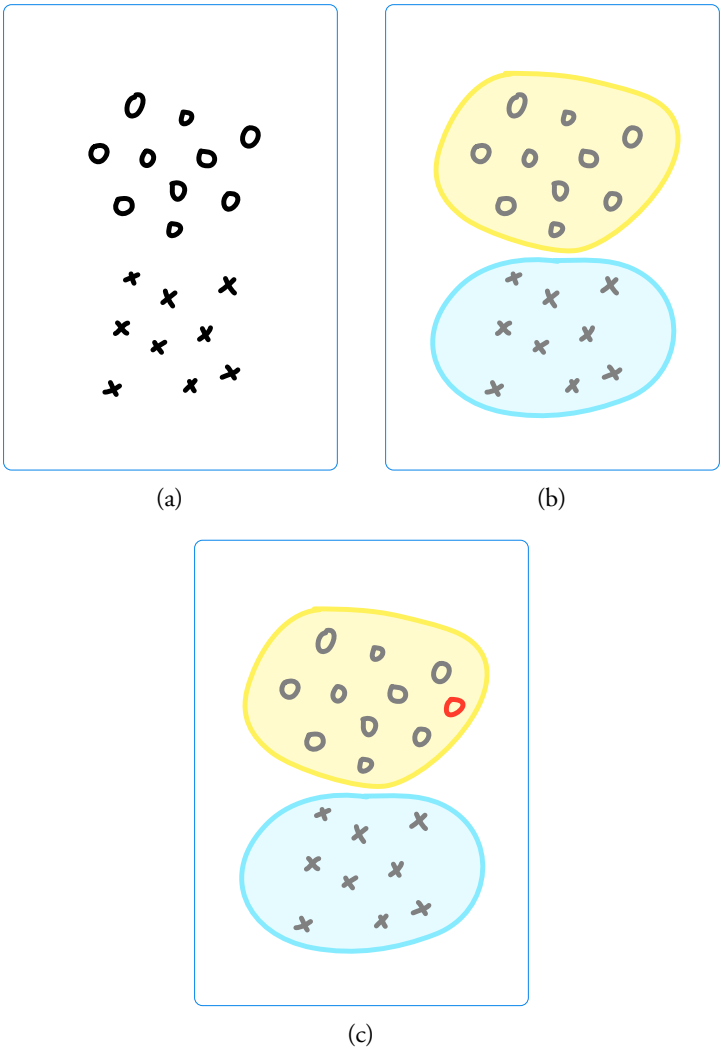
Chapter 2

Theoretical Background

Contents

Figure 4

Conceptual illustration of machine learning methods. A dataset is considered (a) and from it an analysis model separating the data into two classes is learnt (b). New data can then be automatically analyzed by the model (c).



Introduction

The term machine learning has become extremely popular during the last few years. The concept however, remains relatively generic and covers a wide set of ideas, methods and algorithms. We here propose to discuss the notion behind the buzz word *machine learning* and give a short history of its development and successes. We also give some explanations and general information regarding the methods. This short introduction on machine learning aims to be accessible to all, and thus is convenient for a first approach on the topic. More specific technical and illustrated material can be found later in the chapter body.

Learning & Intelligence: Mankind vs Machines

Artificial intelligence is a somehow fascinating field, quickly raising many expectations and controversies well outside the scientific community. Typically, many books, magazines, shows or movies have targeted artificial intelligence, presenting and discussing its numerous outcomes – real, or hypothetical. It is interesting to remark that fictional ideas well proceed the actual boom of machine learning and the first tangible results. The first automated chess player for example, was exposed to the public by Wolfgang von Kempelen in 1769. The automaton was referred to as "the Turk", and fooled many people (including Napoleon) before the trick was revealed in 1820. The Turk was actually maneuvered by a human puppeteer hidden inside it. Thanks to machine learning, what was a dream, trick and fantasy in the XVIIIth century became a reality during the XXth century. Which of today's fantasies will tomorrow exist?

One potential explanation for the enthusiasm around machine intelligence could be the questions it raises on our own intelligence and learning mechanisms. What is intelligence? How do we learn? How fast can we think? If intelligence has been defined as the ability to adapt oneself, many teachers would tell that the questions regarding human learning mechanisms are still widely open. A whole research field named pedagogy is dedicated to that problem. A proof perhaps of the enthusiasm for the potential of machines would be the numerous fictions raising those questions. The first movie displaying a robot goes back to 1927, with *Metropolis* by Fritz Lang and the False Maria robot. In 1982, *Blade Runner* raises the issue of machines and emotions and makes us wondered what makes humans human, and what would it take for a machine to be human. Those questions regarding intelligence and learning mechanisms are indeed fascinating, but they are also of great importance when thinking about teaching a machine. How should we proceed? Where should we start?

The unfortunate truth is that so far, we are not so advanced in the art of teaching a machine. Despite the common fantasy, machines will

not replace human for a long, long time.[1]Nevertheless, some methods have been developed to teach a machine specific tasks. For example to replicate a given behavior (supervised machine learning), or else to organize an extremely large number of data and automatically extract some knowledge on a given phenomenon, less biased by the human point of view (unsupervised learning). Those methods are referred to as machine learning, and are a key component in artificial intelligence. As a final remark, let us notice that *artificial intelligence* is perhaps a strong choice of words.

A Brief History of Machine Learning

The very first machine learning models are relatively old and are almost ironically, younger that *The Turk* automaton. They go back to the end of the XVIIIth century with probability theories and in particular, Bayes theorem [BP63]. The boom of machine learning methods however, was triggered by the advent of computers during the XXth century. Among the early pioneers of machine learning, Alan Turing was one of the first to seriously evoke the possibility of an intelligent machine: "*Can machine think?*". In particular, he developed the premises of the genetic algorithms [Tur50], which would later be used to estimate the evolution of a data population. He also proposed the famous Turing test (known as the imitation game) which challenges a machine to discuss with a human. The test is passed if the machine fools a human [Tur50]. So far, the test has only been passed in fictions (e.g., Ex Machina).

The term machine learning, was first used in 1959 by Arthur Samuel who defined it as a "*field of study that gives computers the ability to learn without explicitly being programmed*". A few years earlier, Samuel had also proposed the first program to automatically play checkers [Sam59]. With the increasing capacities of computers and the development of machine learning algorithms, more and more complex board games have been challenged. In 1963, Donald Michie presented a machine playing Tic-Tac-Toe. In 1992, machines could also play Backgammon [Tes95]. In 1997, Deep Blue (IBM) defeated the world chess champion Garry Kasparov [Hsu99]. In 2016 finally, the Go champion, Lee Sedol, was beaten by a machine, thereby finishing (and winning) the machine vs human boardgames race [SHM+16]. Incidentally, it is also interesting to notice that Marvin Minsky who developed the first neural network in 1952 [Min52] also published many works concerning human learning mechanisms [Min85, Min91, Min07]. He thereby stressed the influence of human learning mechanisms over machine learning. Some of Minsky writings and ideas also had an impact on a wider public. Kubrick for instance visited Minsky while writing *2001, A Space Odyssey* (released in 1968). The idea was to discuss the features of the movie robot Hal, that could be probable by 2001. Lip reading for instance was found plausible enough. Reality was not so late with a

study published in 2016 where a machine beat a human at lip reading [ASWdF16]. This relation between human thinking and artificial intelligence also inspired what is today known as deep learning and was originally based on the human brain model published in 1943 [MP43].

Modern Machine Learning Techniques

The xxist century is the time of many more machine learning achievements, but also challenges which have encouraged the development and improvement of many methods. In 2006, Netflix launched a world-wide contest for a model that would drastically improved the platform movies recommendation system. In 2009, the image database ImageNet was released, containing millions of images separated in a thousand categories. In 2010, the Kaggle platform was opened, proposing hundreds of machine learning related challenges. The availability of those many datasets lead to some of the great successes of machine learning. Among the most striking applications, we can mention computer vision with automatic objects recognition [SZ14], images automatic description [KFF15] or the automatic colorization of grayscale images [SKB13]. Concerning speech processing, recognition systems such as Apple Siri [CCC+17] or music recognition with Shazam [Wan03] are also very popular, and contributed notably to boost the popularity of machine learning to the public. More exotic applications such as generating the next *Game of Throne* chapter or transforming a photo into a Picasso style piece of art [GEB15] also generate a great interest.

First Definitions on Machine Learning

With applications covering an extremely wide spectrum, machine learning remains a generic term. It covers many different methods and algorithms and addresses a large variety of issues. A simple definition to machine learning can be a set of methods that aims at building an analysis model from a set of data (referred as learning or training dataset). Such models can have different purposes, but the general idea is to use them to position new data and compare them to the training dataset. And because of computers power, much more data can be processed than a manual analysis. In the context of big data, here lies the strength – and the need – of machine learning methods. The aim of this chapter is not to review all those methods but to give an introduction on the topic and expose the theoretical and conceptual elements used in the remaining chapters. For a more detailed presentation on those methods, the reader is welcome to consult the following reference books [DHS01, HTF09].

For now, let us illustrate the very general workflow followed by machine learning methods. Figure 4(a) illustrates a dataset that con-

tains two types of data. The first type is represented by circles while the second type is represented by crosses. As shown by Figure 4(b)), building a model consists in defining one or several boundaries between the data (i.e., separating them into different groups). The challenge here is to build boundaries that best separate data in groups. This is done learning algorithms. This chapter present some of these algorithms in detail. Once these boundaries are build, the model can be used to classify a new data in one of the two groups (Figure 4(c)). Today, those machine learning methods can be adapted to extract knowledge and valuable information from large datasets. This is the purpose of this work.

Synopsis

This chapter contains more technical material on the machine learning methods, but popularized explanations and simple illustrations are also provided to support the overall comprehension. We here aim at exposing the theoretical elements needed through this work and the proposed material is accessible without any particular background. The chapter can be read independently and outside the context of this PhD as a first introduction or tutorial on machine learning. It is organized into two different sections, both of equal importance. The first one deals with the issue of representing data: first by explaining the context and then by giving key elements answering the problematic. The second one focuses on supervised machine learning: from introducing the general mechanisms to detailing of some of the most effective algorithms.

[1] The very competition between humans and machines is not necessarily relevant given that both are ruled by different mechanisms. The comparison however, is interesting.

2.1 Signals Representations

Application fields using machine learning methods – including automatic classification algorithms – can target many different types of data, such as images, videos, sound, waves, and others. But regardless of the shape of the original data, the actual algorithms are the same and are designed to work on data with some specificities. Most of the time, the input and raw data do not match those requirements, and a first step consists in transforming the data, and considering a representation in an alternative space. This new representation of the input data matches requirements of the learning algorithms, and the considered space is often referred as the feature space. This standardization step is known as the features extraction process (Figure 5). The feature vectors are then used as input of the learning algorithms.

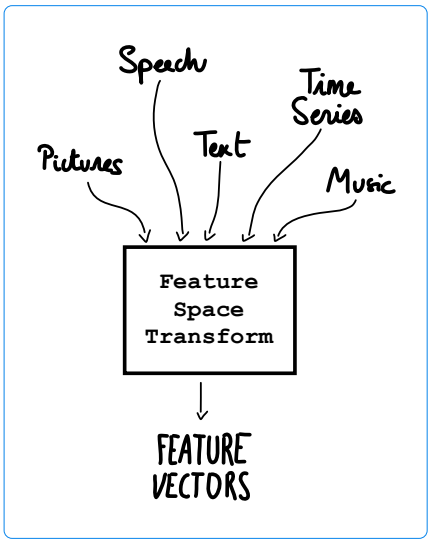


Figure 5
Uniformization of the data through the features extraction process.

2.1.1 The Importance of Signals Representation

Finding an appropriate representation for a set of signals is a very common task in signal processing. We summarize here this issue, that is the transformation from the raw recordings to the feature vectors, in the context of machine learning problems.

The main reason to use features is to place the data in a space where they are separable depending on their classes. The learning algorithm tries to build boundaries between the data in order to separate them into several groups of interest. But the learning algorithm and the model only access the data through feature vectors. So data represented in the feature space should be as separable as possible for the model to learn a meaningful boundary. Often the input data are not separable in their original space, and the transformation used from the original space to the feature space should separate them as much as possible. This of course is a complex task because it suggests that the user knows on which criteria the data should be separated.

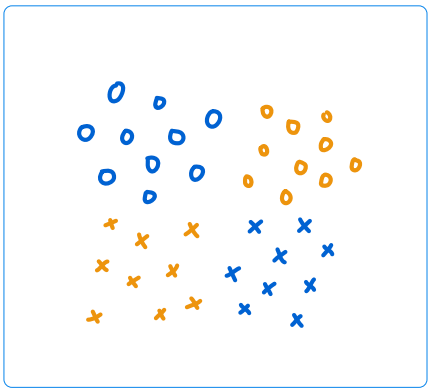
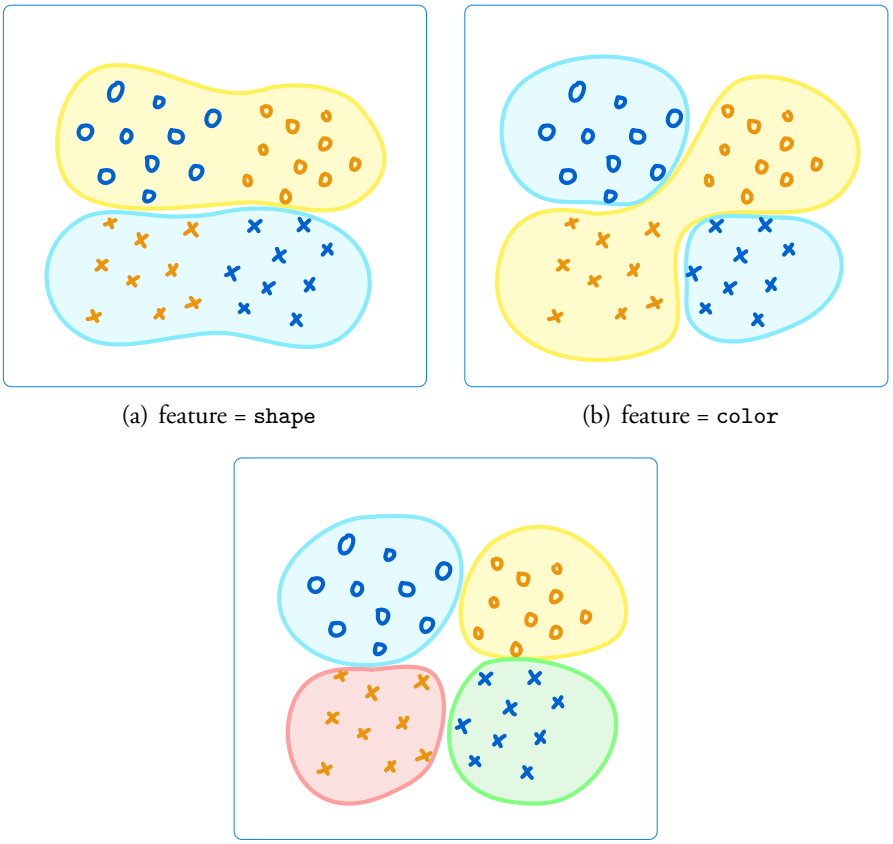


Figure 6
Simple dataset to be used to train a model.

A second argument in favor of the feature space is the curse of dimensionality, originally presented in [Bel56]. Let us consider the following notation: N represents the dataset size (number of observations), n the dimension of the data in their original space and d the dimension of the input data in the feature space (number of features). The number N of data needed to train a model increases with the dimension of the data n. Using a feature space with $d < n$ is therefore advisable. The curse of dimensionality compares the number of data N and the input space dimension (i.e., the feature space) d: considering N small compared to d leads typically to worse performances.

Finally, most learning algorithms cannot represent ordered data. They consider the various dimensions of the input data in a non ordered way. The representation used as input of the learning algorithm should therefore keep the same informative content if randomly shifted. With this condition in mind, it is advisable to use feature vectors with independent components rather than ordinate representations, such as time series, spectra or images. In this manuscript, we refer to high level features as features of dimension 1, which are computed as characteristics on the input data. The final feature vector is obtained by concatenating all the features, and the data is therefore represented by a non-ordered vector. On the contrary, we refer to low level features as ordered feature vectors. Using high level features raises the abstraction level of the representation chosen for the data and produces better results in term of classification accuracy [DHS01, HTF09].

In order to illustrate the impact and importance of the feature space, we take the example of a very simple dataset, presented in Figure 6. Given this dataset, several representations can be considered to represent the data: shape, color or position in the cartesian plan for instance. If considering the shape representation, the data will be separable as presented in Figure 7(a). If considering the color representation, data will be separable as presented in Figure 7(b). Finally, if considering the position representation, data will be separable as presented in Figure 7(c). Several remarks can be done be done on this example. The most important one would be that for a given dataset, different representations lead to different models and decision boundaries. This raises the question about which boundaries would be built if the chosen representation were



(a) feature = `shape`

(b) feature = `color`

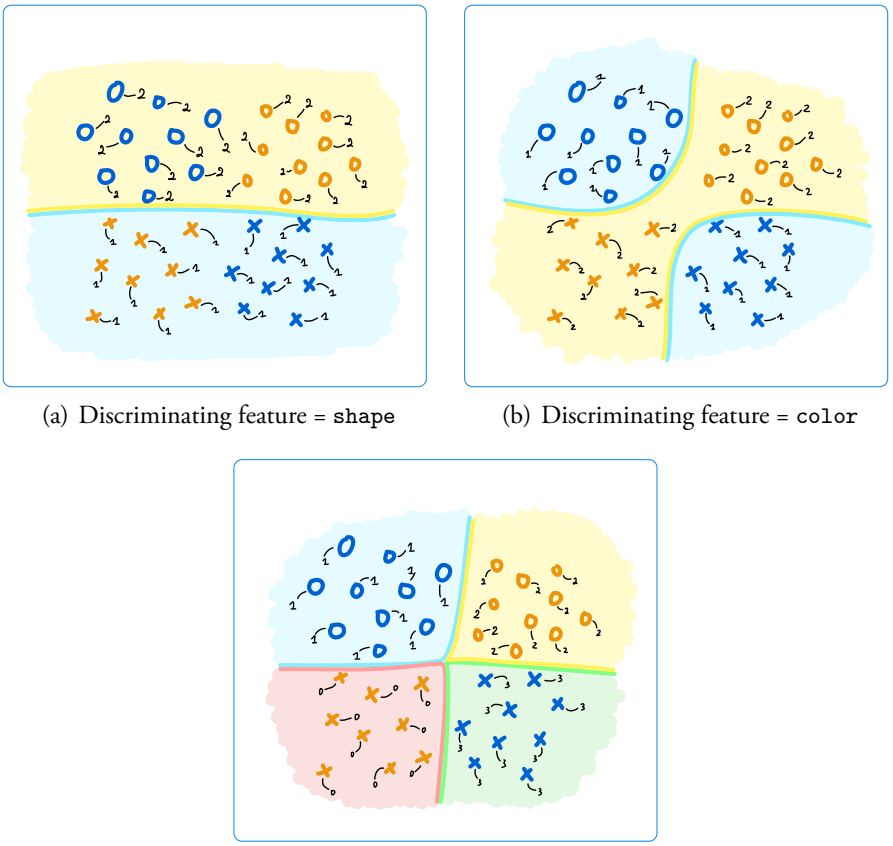
(c) feature = `position`

Figure 7
Illustration of the importance of the feature space chosen for a given dataset. Different feature spaces lead to different models. Illustration in the case of unsupervised learning.

$('shape','color','position')$. This illustration is made in an unsupervised context: a model is trained by a learning algorithm from the dataset represented in the feature space. In this context, the question of more complex representations is indeed raised. Answers to this issue however are beyond the scope of this chapter, but can be found in [DHS01, HTF09]. With this issue in mind, the benefit of supervised learning algorithms can be illustrated. A model is also trained by a learning algorithm from the dataset seen as feature vector, but additional information is also provided. The associated classes of all the training data is, referred to as labels, is also given to the learning algorithm. And from this information, a model is build. In this context (illustrated in Figure 8), the choice of which boundary to build is driven by the labels. Meaning that labels are a mean to force the boundary in one direction, but also that different labels lead to different models. We can also remark that in both supervised and unsupervised cases, the illustrated boundaries are not unique and that the choice of a specific boundary depends on the learning algorithm.

This example is visual and simplistic, but it illustrates many aspects: the importance of the features, the difference between supervised and unsupervised learning, the benefice of supervised learning, the importance of labels associated to a dataset and the impact of the learning algorithm. To conclude this part, let us notice that this reasoning is valid if the features are correctly chosen, meaning that they separate the data into their classes.

Figure 8
Illustration of supervised machine learningsupervised learning and the influence of the labels. For all three examples, features = {`shape`, `color`, `position`} and the different models come from the different labels. The discriminating feature is deduced by the learning algorithm.



(a) Discriminating feature = `shape`

(b) Discriminating feature = `color`

(c) Discriminating feature = `position`

It therefore raises the question of how to find such representation, which is the focus of the next part. This conclusion also underline the importance of the dataset. A model depends most of all on the dataset it is built upon. Meaning that the same process used on two different datasets will produce different results. Meaning that a model build on a small dataset will not be generalizable to slightly different events. And meaning that the model reproduces what is *within* the training dataset, raising questions such as the trust in predictions, or the novelty or anomaly detection. More details on the learning algorithms is given in Section 2.2.

2.1.2 How to Obtain a Representation for a Signal?

Features such as shape, color or position (introduced in the previous example) are quite simplistic for real-world data. Data recorded are often more complex, noisy, and difficult to be interpreted. The question of how to find features adapted for a given dataset is therefore challenging. Usually, two types of representation are considered: learnt features, or handcrafted (or designed) features [DHS01, HTF09].

Methods to learn representations are often referred to as *dictionary learning*, and can be done from the supervised dataset (convolutional neural networks [LBBH98]), or from the unsupervised dataset (principal component analysis [TB99], independent component analysis [Com94],

singular values decomposition [GR70], non negative matrix factorization [LS71], and others). The leading idea is to learn a dictionary of elements from the dataset that can be used to reconstruct approximate data. The data is approximated by a linear combination of the dictionary elements, and can be identified and represented by the weights associated to each element in the feature space. The main advantage of those algorithms is the final representation (i.e. the feature vectors) which is adapted for the dataset. However, such algorithms can require a large and labeled dataset to learn the representation and can be very costly (convolutional neural networks for instance). Furthermore, a learnt representation is not related to a physical value of the data, which can be an issue. Representations related to physical values of the data can indeed be used to better understand the original data (a simple illustration of this statement would be the use of Fourier transform in signal processing). During the last decade, convolutional neural networks have been particularly popular classification tools. In particular, their ability to learn a representation from a dataset is very efficient and has led to very good results in many applications, such as computer vision [GBC16, LBBH98, KSH12, SZ14]. The extremely large sets of data needed to train such feature extractors however, are one of the limitations of those tools. Chapter 6 study the possibility to use such methods on environmental data.

On the other hand, designed or handcrafted representations are based on the features extraction process, as was illustrated in the previous part. The main idea is to use descriptors to extract and quantify a physical quantity on the data, referred to as feature. All features are then concatenated into a feature vector (vector of the feature space) that will represent the data. The main issue of this method is the difficulty to find features that will effectively separate the data into their classes. This step involves knowing the data, and especially being able to explain why a given data belongs to a specific class. One advantage of those feature sets is that they can be generalized: designed for an application, they can be relevant in other fields where the data have similar properties. Moreover, this technique can be less constraining on computation times, and allows the data to be represented by physical quantities which can be of help to understand the associated physical phenomenon. Such features are used throughout this work. Another aspect related to handcraft features is the feature selection problem, which consists in finding the subset of features maximizing the model performances among a proposed feature set. Several approaches can be used, including forward selection which iteratively selects relevant features among the proposed set, and backward elimination, which iteratively eliminated irrelevant features of the proposed set [Lan94, DL97].

2.1.3 Representations Used for Transient Signals in the Literature

In this part, we present a review of the literature regarding the representations (i.e. the feature vectors) that are or have been used for the automatic classification of transient signals. Transients are defined as signals whose Fourier transform has a infinite number of components (by opposition

to stationary signals) [RG75]. Transient signals are involved in many different applicative fields, including speech, music, environmental sounds (natural or human induced), seismic signals, and others. We also focus on hand-crafted features for their help to understand and interpret the data. The many features that are used can be separated depending on their abstraction level: from low level features (i.e. raw time series) to high level features (i.e. statistical mean or standard deviation).

Originally, low level representations were used. The raw time signal was traditionally used, for example on seismic waves [FGNS96, LCMLB16, EGD+08]. The Fourier transform underlines different properties on the considered signals and can therefore be used for similar purposes, see for instance [FGNS96, URJ16]. Other law level representations are also used, including the signals envelope [FGNS96], the auto-correlation function [FGNS96, LFT03, LFPT06] or transformations based on the wavelet transform [AWO+06, LW95, WZ14]. Some features are built upon more complex transform, but are still ordinate representations and are therefore considered as low level features. The Mel Frequency Cepstral Coefficients (MFCC) feature set for instance was originally designed to model speech [ZZS01]. Several implementations can be used [ZZS01, GFK05] but the general idea is to model the signal harmonic properties. To do so, the signal energy under several frequency bands is considered. This energetic representation matches the mel scale, used to model the human hear. The discreet cosine transform is then computed. Actually, the operation consists in studying the spectrum of the signal spectrum. The final space is known as the cepstral domain, and models the periodic properties of the signal spectrum, meaning the harmonic properties of the original signal. MFCC are low level features but they are still used in many works, including to distinguish speech from music from non vocal sounds in [Foo97], for music instruments recognition in [EK00], on bio-acoustics data in [PBG+10, NTSR16, LCHH06, WTP+10, BIDL14] or [GL03], on sounds of human origin in [LBHL07, MMSFSGSP14, SDC06], or on environmental noises in [DTL11]. Feature sets based on the concept of MFCC but modified to fit the frequency range of the considered data are also used in several studies, including [BRS+07, AGC+12, IBG+09, ZTX+12] for seismic signals, but also [VFAT15, PBG+10, NTSR16] on underwater bio-acoustics data and in [CJ06] for bio-acoustics signals. On the same ideas linear predictive coding coefficients (LPC) model the signals spectral envelope and can be used as features for classification purposes, for example on seismic signals in [DPEG+03, GES+09] or [SGE+05]. LPC have been also used in classification tasks with bird calls [MC97], humpback whales [PBG+10], music signals [XMS05, BKK06], and for environmental sounds [CFGM98]. Dynamic time warping (DTW) measures the similarity between transient signals, and can also be used to represent time series in classification tasks, see for example [THMP06] and [KM98] for automatic birds recognition. Autoregressive coefficients are also used as signals representation in [Che85, KCP94] on underwater acoustic data. Features based on the energy in various frequency bandwidths can also been considered, for instance on seismic data in [BW08, LFM+09]. Generally speaking, low

level features can produce correct classification results. However all those features are ordinate representations and are therefore limited when used with most of the machine learning algorithms. In 1985 [Che85] states that "For underwater acoustic transient signal classification, well accepted features have not been developed". This statement about low level features was made for underwater acoustic transient signals but is generalizable to transient signals.

High level features are now being used in a number of studies. Originally only a few were used, seismic signals for example have been represented by their kurtosis in [SHP02, LMMB14]. On similar data, other simple features such as the maximum amplitude, or the main frequency can also be used [CMA+11, IvS08] and [CVF+09, HPM+16] propose to represent the time series by statistical features such as skewness and kurtosis. Similar features along with spectral centroid, bandwidth or threshold crossing rate are also used for other applications, including music classification tasks [SDC06, EKR04, FM00] or bioacoustics [HYYC09, Fag07, ZVH+10]. The signals lower and upper frequency are used in [ACBCB+09] for birds and amphibian calls classification. Features based information theory with entropy measurements are also considered in some studies. For instance in [EKR04] and [HMD11] for frogs sounds classification or in [ZVH+10] for the discrimination of whale and boats. Some of those features are also sometimes computed from the spectral domain, for instance in [WX10] for infrasonic data. To improve the models classification abilities, more detailed representations are now being used. In particular, feature sets with greater number of features are now being presented and show their efficiency in many applicative classification tasks. Some feature set are quite dependent on the application, such as the 30 features presented in [KOS10] and re-used [HBO12] for volcano-seismic signals. This feature set includes features resulting of trace or polarization analysis of the seismic signals for instance, which cannot be generalized to other applications. Other feature sets are more general, see for instance [TB05] who considers more than 20 features including signals shape descriptors (i.e. rate of attack and decay, temporal occurrence of the main peak), statistical moments (e.g., mean, skewness and kurtosis) and signal power (e.g., peak power, average and power standard deviation) for the automatic classification of transient anthropic signals. Similarly [MFH+17] and [PHM17] uses large number of features (40 and 71, respectively), extracted both from time and spectral domains for the automatic classification of volcano-seismic signals, and [EK00, LR04] use 20 features for music instrument recognition.

Another approach to extract features from transient signals is to consider image representations. Technically speaking, spectrograms are computed and can be pragmatically considered as images allowing one to take advantage of the large set of image processing tools available. Ridge detection and points of interest are for instance used in [DTZ+13] for bird vocalization retrieval. Similar techniques are proposed for sea mammals detection and classification [EZE14, TKB+12], for musical signals classification and in [DSN01, YS09] for discrimination of environmental noises [DTL11].

2.2 Supervised Machine Learning

2.2.1 An Overview on the Subject

Supervised machine learning algorithms are a set of methods used to build a prediction model (e.g., regression, classification, etc.) from a labeled dataset. Usually (but not necessarily), a training stage is usually needed to produce and/or optimize the model. The process is illustrated in Figure 8. Formally, the set of data used to build the model is referred to as the training or learning set and is usually noted $\mathbf{X} = \{\mathbf{x_i}\}_{i=1}^{N}$. For $1 \leq i \leq N$ and $1 \leq j \leq d$, $\mathbf{x_i}$ represents the data seen as a feature vector of dimension d: $\mathbf{x_i} = \{x_i^j\}_{j=1}^{d}$, with $x_i^j \in \mathbb{R}$. In a context of supervised learning, the associated labels describing their classes are also known. They are noted $Y = \{y_i\}_{i=1}^{N}$. The model can be symbolized by a function f, which predict the output y from the input feature vector x.

$$\begin{aligned} f : \mathbf{X} &\to Y \\ \mathbf{x} &\mapsto y. \end{aligned}$$

A supervised learning algorithm attempts to build a model represented by f from the labeled dataset $\{X, Y\}$. Depending on the algorithm, different functions f_θ (e.g., different parameters θ) are considered. The considered function f is chosen (optimized) during the learning process. In practice, a cost function $J_N(\theta)$ (or loss function) measuring the prediction errors made by the model on the training set $\{\mathbf{X}, Y\}$ is considered. The loss function can be the 0-1 function (adds 1 for each data misclassified), equation 2) or the mean square error (equation 1), depending on the nature of the prediction (classification or regression, respectively). Other loss functions can also be considered.

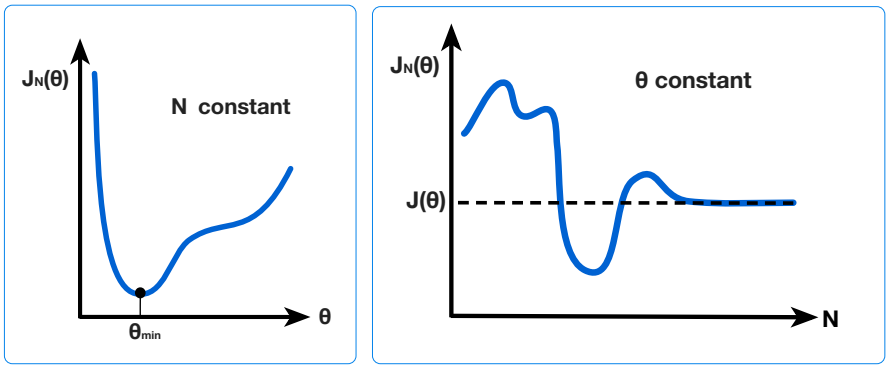
$$J_N(\theta) = \frac{1}{N} \sum_{i=1}^{N} \big(y_i - f(\mathbf{x_i})\big)^2 \tag{1}$$

$$J_N(\theta) = \sum_{i=1}^{N} \mathbb{1}\big(y_i, f(\mathbf{x_i})\big) \text{ with } \mathbb{1}(k,l) = 1 \text{ if } k = l \text{ and } 0 \text{ otherwise, } k, l \in \mathbb{R} \tag{2}$$

The cost function $J_N(\theta)$ depends on the model parameters θ and can be represented as a surface (or as a curve when $\theta \in \mathbb{R}$) as seen in Figure 9(a). During the training stage the model f associated to the minimum of the cost function $J_N(\theta)$ is chosen. This is the model minimizing the prediction errors on the training set.

$$f = f(\theta_{min}), \text{ with } \theta_{min} = \underset{\theta}{\operatorname{argmin}}\, J_N(\theta)$$

As such, the learning phase can therefore be seen as an optimization problem. A direct consequence is – in many cases – the impossibility to



(a) Optimization of $J_N(\theta)$ during the training of a model

(b) Impact of the dataset size

Figure 9

Evolution of the cost function $J_N(\theta)$. (a) N constant, evolution with θ (training of a model). (b) θ constant, evolution with N (dataset size), in this illustration the dataset is coherent, and $J_N(\theta) \to J(\theta)$

find an analytical solution and therefore the need to use iterative algorithms in order to approximate the best parameters of the model.

But the cost function $J_N(\theta)$ is also a function of the dataset size N (and a function of the dataset in general). Two remarks can here be made: First, since the cost function depends on N, two different dataset lead to two different models. This is known as the variance of the learning algorithm. Theoretically, two datasets that differ only by one data (i.e. of size N and $N+1$, with N data in common) will produce two different models. However it would be beneficial if the two models were similar since the dataset are very similar. The low variance of a learning algorithm is therefore a key point. This property is related to the model generalization capabilities and is necessary to produce efficient models. Secondly, the dependence of the cost function to N and to the dataset in general can be interpreted in term of requirements on the dataset. Typically, the bigger the dataset is, the more stable the model should be: adding one new data in training should not have a great impact on the model (low variance). But also, the training dataset should be representative of the phenomenon to be modelized. A dataset that is too small, or with very similar data that are not fully representative of the studied phenomenon will lead to a low minimum on the cost function, but to a model with poor capacities on real-world data. A example of the cost function dependence on N is illustrated in Figure 9(b).

Supervised learning algorithms can be separated into two main groups depending on the nature of the output y_i. In the case of regression models the $y_i \in \mathbb{R}$ are continuous values and the output is said to be quantitative. If the output is qualitative, Y is a discrete set and the function f is a classification model. Typically, $Y = [\![0, C-1]\!]$, with C the number of classes In the case of weather prediction for example, the output set $Y = \{'sunny','cloudy','rainy'\}$, which is in practice coded as $Y = [\![0, 2]\!]$ corresponds to a classification model[2]. However if the output refers to the outside temperature (e.g., a continuous measurement), the model will be a regression model. In all cases, the models are referred as prediction

[2] In practice, a label encoder is used on the dataset labels to transform the labels (strings, non continuous digits, etc) into the standardize set $Y = [\![0, C-1]\!]$

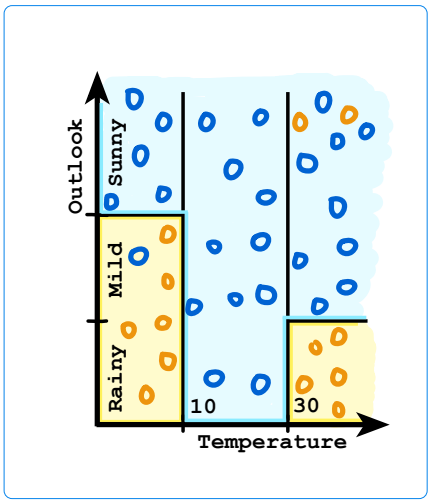
models, since the output is predicted from the input feature vectors. Supervised learning algorithms are often developed in a specific context (i.e., classification or regression) but can usually be adapted to the other.

Machine learning algorithms are extremely numerous and for a full review the reader is invited to consult references such as [DHS01] or [HTF09]. We here propose a short history of supervised learning algorithms that are used today, or are the basis of more complex algorithms. The first models are based on statistics, thereby explaining why machine learning can also be referred to as statistical learning[3]. Typically, a model is build to cover the variability of the data and model their distributions. The first supervised learning algorithm is based on the Bayes theorem which was published post-mortem in [BP63] and later formalized in [Lap20]. It relates a posteriori and a priori probabilities, and is formally known as the naive Bayes classifier. The term machine learning was obviously not considered, but it is worth noticing that the first learning algorithm well preceded the birth of computers and the boom of computational capacities. Another supervised learning algorithm preceding the computer era is the linear regression with the least squares analysis, building a hyperplan minimizing the distance between the data projected onto the hyperplan and their original positions. It would have been discovered by Gauss in 1795 who did not publish it until 1809 [Gau09]. In the mean time, Legendre independently discovered it and published in 1805 [Leg05]. The method ownership is disputed and would have lead to a few debates [Sti81, Par15, Nie01]. The algorithm was originally developed for astronomical problem, but was a few years later used for biological studies by Galton, who also introduced the use of the maximizer of likelihood. The development of modern computers from the 1930s (Turing machine in 1936, ABC in 1937, the Colossus in 1942) gave a kick of interest and development of statistical and learning algorithms. In 1936, the linear discriminant analysis was proposed by Fisher [Fis36]. The model is probabilistic and leads to a hyperplan expressed as a linear combination of the training data to separate them into their classes. Models with inspiration beyond statistic are also developed. For instance, the view of the human brain as a network of small units which was proposed by McCulloth and Pitts in 1943 [MP43] lead to the perceptron. This model goes back to 1957 and is the foundation of neural networks and deep learning. It was presented by Rosenblatt [Ros57]. Logistic regression was then proposed by Cox in 1958 [Cox58] to deal with discreet categories (e.g. classification task) and model life spans. A few years later, the original ideas behind support vector machines was proposed [VL63]. At the same time, ideas behind the k-nearest neighbors algorithms were also published: in 1962 by Sebestyen with the proximity algorithm [Seb62] and in 1965 by Nilsson with the minimum distance classifier [Nil65]. The appellation nearest neighbor was first proposed with Cover and Hart in 1967 [CH67]. In 1986 Quinlan proposes yet another type of learning algorithm with the use of decision trees and the CART algorithm [Qui86]. During the last 30 years, many of those algorithms have been improved, and others have also been

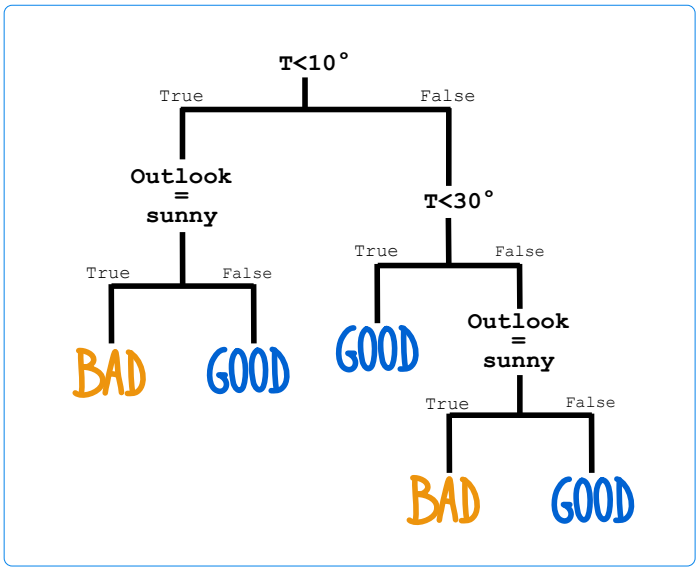
[3] Strictly speaking, machine learning and statistical learning can be distinguished, but the issue is debatable, and is well beyond the scope of this chapter.

developed. Generally speaking though, most of the original ideas behind today supervised learning algorithms have been developed during the 20st century.

Over the years, machine learning algorithms have become more and more complex. In this manuscript, two main algorithms are used, as some of today's most effective approaches. Those two algorithms are Random Forest (RF) and Support Vector Machine (SVM), and we hereafter detail their mechanisms.



(a) Partition of the feature space



(b) Associated decision tree

Figure 10

Illustration of a decision tree predicting good or bad hiking condition from two features: the weather forecast and the outside temperature. Data corresponding to good hiking conditions are represented in blue, while data representing bad hiking conditions are in orange. The partition of the feature space is illustrated in (a) and the associated decision tree is in (b).

2.2.2 Focus on Random Forest Algorithm

Random forest algorithm is based on binary decision trees, originally presented in [Qui86]. Quinlan presented the CART algorithm, for classification and regression trees, which builds a partition on the feature space. The feature space is therefore separated into various regions. Each region

is associated to its most present class. Iterative tests are made on the features representing a data that needs to be classified, thereby deciding on the region of the data and therefore of its class. See Figure 10 for an illustration.

CART algorithm explains the learning process to grow a decision tree from a labeled dataset $\{\mathbf{X}, Y\}$, with $\mathbf{X} = \{\mathbf{x_i}\}_{i=1}^{N}$ the input feature vectors of dimension d and $Y = \{y_i\}_{i=1}^{N}$ their associated labels ($y_i \in [\![1, C]\!]$). The leading idea behind the algorithm is to minimize the *impurity* of each region, that is to favor regions with data of the same class over regions with data belonging to different classes. The cost function associated to this supervised learning algorithm is therefore a measure of the regions impurity, and can either be the Gini index (Equation 3) or the cross-entropy (Equation 4) [HTF09]. The proportion of data $\mathbf{x_i}$ belonging to the class c_i for the region R_m is noted p_{R_m,c_i}. $c_i \in [\![1, C]\!]$ are the possible classes of region R_m and C is the number of classes. The cost function is to be minimized depending on the model parameters: at each split of the tree, the choice of the split feature x^j and its associated split value referred as the split point s.

$$\text{Gini index} = \sum_{k=1}^{K} p_{R_m,k_m} \cdot \left(1 - p_{R_m,k_m}\right) \tag{3}$$

$$\text{Cross-entropy} = -\sum_{k=1}^{K} p_{R_m,k_m} \cdot \log\left(p_{R_m,k_m}\right) \tag{4}$$

The process to grow a tree is iterative, and is described in the algorithm 1. While the stopping criteria is not reached, a new split is made. To find the best split, all features are considered. For each feature, all split points are considered.The selected feature is the one leading to the minimum of impurity for the new region. The stopping criteria can be the tree depth, the misclassification error gain from this split is under a given threshold, or if the regions size are under a given threshold (smaller than 5 for example in [HTF09]).

The idea beyond decision trees is quite popular for its easy interpretation. Many machine learning algorithms are seen as black boxes, and decision trees on the contrary, can help understanding the data (features ranking and easy interpretation of the model). Computation times are relatively low, which is interesting for real-time systems. The algorithm is also deterministic, meaning that the same dataset will lead to the same model. If the tree is deep enough, the model is also unbiased, meaning that it does not make systematic mistakes. Its variance however is important: two slightly different learning datasets can lead to two very different models. This last point is the main limitation of the algorithm. A model should indeed be stable if a few training data are added or removed. To partially improve this point, the tree can be pruned: some of the lower regions are merged. The misclassification error is often used as a measure

Algorithm 1: Random Forest learning algorithm

while *stopping_criteria* = *false* **do**
  for *each feature* x^j **do**
    for *each split point* s **do**
      Define a pair of half plane (two regions R_1 and R_2) in the features space:

$$R_1(j,s) = \{\mathbf{x_i} | x_i^j \leq s\}$$
$$R_2(j,s) = \{\mathbf{x_i} | x_i^j > s\}$$

      Define the associate class proportions $p_{R_1,c}(j,s)$ and $p_{R_2,c}(j,s)$:
      $\forall c \in [\![1, C]\!],\ p_{R_1,c}(j,s) = \frac{1}{N_{R_1}} . \sum_{\mathbf{x_i} \in R_1} \mathbb{1}(y_i = c)$
      $\forall c \in [\![1, C]\!],\ p_{R_2,c}(j,s) = \frac{1}{N_{R_2}} . \sum_{\mathbf{x_i} \in R_2} \mathbb{1}(y_i = c)$
    Find the best split point:
    $s = \text{argmax}_s \left(\max_c p_{R_1,c}(j,s) + \max_c p_{R_2,c}(j,s)\right)$
  Find the best feature x^j:
  $j = \text{argmax}_j \left(\max_{c1} p_{R_1,c_1}(j,s) + \max_{c2} p_{R_2,c_2}(j,s)\right)$

of a region impurity to prune a tree [HTF09]. See Equation 5 for the impurity of a region R_m attributed to the class c_m.

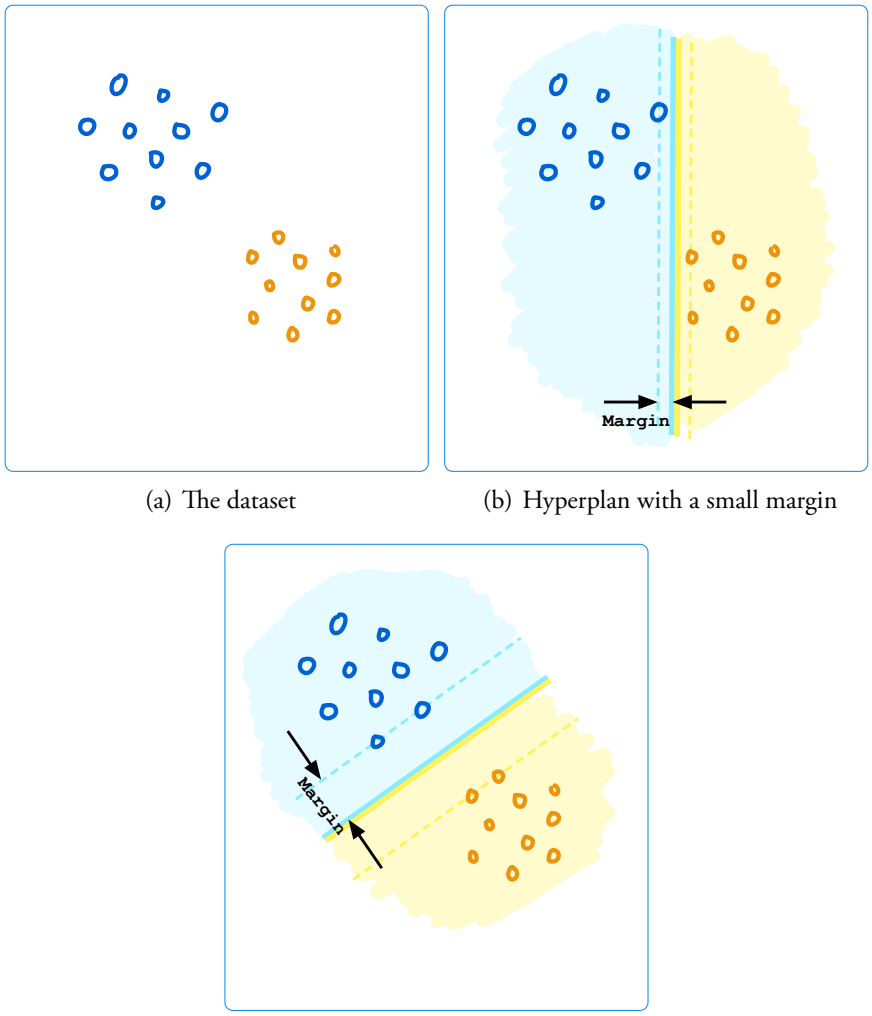
$$\text{impurity of region } R_m = \frac{1}{N_{R_m}} \sum_{i \ / \ \mathbf{x_i} \in R_m} \mathbb{1}(y_i \neq c_m) = 1 - p_{R_m,c_m} \quad (5)$$

The random forest algorithm was designed in answer to the high variance of decision trees [Ho95, Bre01]. The main idea is to build a large number B of binary decision trees and to predict the output from a majority vote of the tree forest. The idea of using several models for one task is known as bagging. Typically, a dozen to hundreds of trees can be considered. Increasing B develops the model generalization capabilities, but the phenomenon steadies itself at some point. Computation times also increase with B. In this case, each tree is build from a subset of the original learning set of data, thereby leading to different trees.[4] By doing so, the final model variance is decreased, but it remains unbiased for deep enough trees. Typically, $N' = 1/3$ or $N' = 2/3$ of the original dataset is randomly considered for each tree. Furthermore, each split is also decided on only a subset of the feature vectors. Usually, $m = \sqrt{d}$ of the features representing a data are randomly considered at each split. By doing so, the different trees are decorated and performances increased. Trees used in a random forest are not pruned: the reduce variance come from the large number of tree considered.

[4] Building different trees from the same dataset would not make sense since the CART algorithm is deterministic and would produce the same model.

Figure 11

Illustration of SVM. (a) The training dataset is used to build a hyperplan separating the two classes. Two different hyperplan are illustrated: (b) a small margin and (c) a large margin. SVM algorithm maximizes the margin, thereby selecting the margin of panel (c).



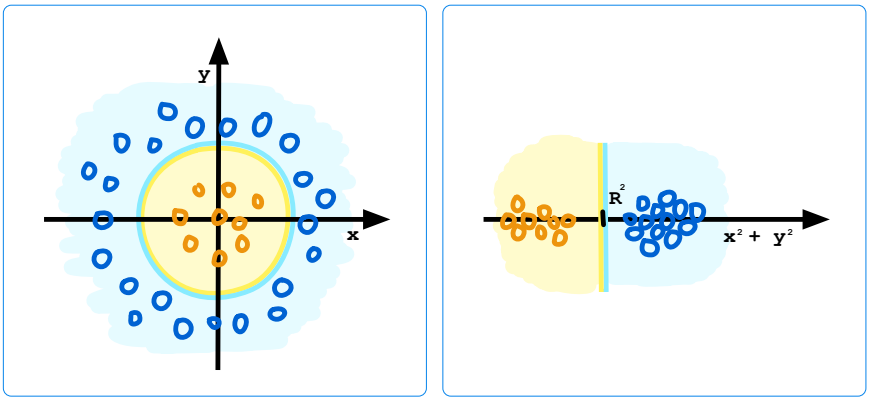
(a) The dataset (b) Hyperplan with a small margin

(c) Hyperplan with a large margin

The main advantage of random forest is its improved stability compared to decision trees. By using a forest of tree, the features are also ranked, which can used as tool for features selection. On the downside, let us remind that random forest have a random component, and are heavier on computation time that the simple decision trees.

2.2.3 Focus on Support Vector Machine Algorithm

The commonly accepted version of Support Vector Machine (SVM) algorithm is presented in [CV95]. SVM history however is quite long and goes back to the 60s [VL63]. The main idea is to find a hyperplan of the feature space separating the data into their classes. Originally, the algorithm was designed for binary classification problems (i.e., two classes only). The distance between the hyperplan and the closest data is known as the margin. The closest data to the hyperplan are referred to as the support vector (i.e., support feature vector). The distance between the margin and the data can be seen as the inverse of the cost function used during the training of a SVM model. The distance is maximized, thereby optimizing the



(a) Data are not linearly separable in the feature space

(b) Data are linearly separable in the kernel space

Figure 12

Illustration of the "kernel trick" in SVM classifier. The dataset represented in the feature space (a) is not linearly separable. The hyperplan is therefore built in the kernel space (b) where the data are linearly separable. This trick allows to separate data that are separable, but not linearly. The hyperplan represented in the feature space is therefore not linear (a).

separation between the two classes. By maximizing the margin, SVM optimizes the generalization capabilities of the final model. See Figure 11 for an illustration.

This relatively simple implementation of SVM performs well, but is limited to data that are linearly separable. For example, in Figure 12, the data are clearly separable depending on their classes but not linearly. To overcome such a limitation, the "kernel trick" was proposed [BGV92]. The idea is to use a function to transform the input data (i.e., the feature vectors $\{\mathbf{x_i}\}_{i=1}^{N}$) to a space of higher dimension. This space of higher dimension is sometimes referred to as the *feature space*, but for more clarity and to avoid confusion, we refer to it as the *kernel space* in this manuscript. Ideally, features are linearly separable in the kernel space (Figure 12(b)). The hyperplan is then chosen linearly in the kernel space, and correspond to a non-linear boundary in the feature space (Figure 12(a)). Several classical functions can be used to transform the data from the feature space to the kernel space, including polynomial or even Gaussian functions – thereby leading to a kernel space of infinite dimension. The additional computational cost is not heavy since in practice, the data do not need to be explicitly transform to the kernel space. Only the scalar product of the data with the hyperplan are computed, thereby leading to a more efficient model, with reasonable computational cost (more detail in the technical development of this section). The scalar product between two data in the kernel space is referred to as the *kernel function*. This improvement is particularly helpful on data that are not linearly separable in the feature space, which is often the case in real-case scenarios.

A second optimization of SVM is the use of a soft-margin [CV95]. By introducting a cost parameter C_{SVM}, the model tolerates some training data to be on the wrong side of the support vector hyperplans. The use of the soft margin is particularly beneficial in real-case scenario, where data are not separable. It also helps to provide a model with better generalization potential.

The litterature contains several implementions of SVM. We present here a common resolution in which SVM is a quadratic optimization problem. Its resolution does not require a greedy search. The hyperplan is defined in the feature space by the set

$$\{\mathbf{x} : g(\mathbf{h}(\mathbf{x})) = \mathbf{h}(\mathbf{x})^\intercal \boldsymbol{\beta} + \beta_0 = 0\}$$

where $\mathbf{x}$ represents a point from the feature space, and $\mathbf{h}(\mathbf{x})$ represents the same point from the kernel space. The distance between the hyperplan and the support vector is defined as the margin M:

$$M = \frac{|g(\mathbf{h}(\mathbf{x_{SV}}))|}{||\boldsymbol{\beta}||}.$$

The support vectors $\mathbf{h}(\mathbf{x_{SV}})$ are the closest data to the hyperplan in the kernel space, and the one defining the hyperplan position. They are the most difficult data to classify. Typically, $h(\mathbf{h}(\mathbf{x_{SV}})) = \pm 1$, which leads to $M = \frac{1}{||\boldsymbol{\beta}||}$. Maximizing the margin is therefore equivalent to minimizing $||\boldsymbol{\beta}||$ with respect to the correct classification of the training data. More formally:

$$\min ||\boldsymbol{\beta}||, \tag{6}$$

subject to the constraints:

$$y_i(\mathbf{h}(\mathbf{x_i})^\intercal \boldsymbol{\beta} + \beta_0) \geq 1 - \xi_i, \quad \forall i,$$

$$\xi_i \geq 0, \quad \forall i,$$

$$\sum_i \xi_\mathbf{i} \leq cst.$$

where, the slack variables $\{\xi_i\}_{i=1}^N$ describe the amount by which the training data $\{\mathbf{h}(\mathbf{x_i})\}_{i=1}^N$ can be on the wrong side of the support vector hyperplans. Equation 6 and its associated constraints can be rewritten as:

$$\min_{\boldsymbol{\beta},\beta_0} \frac{1}{2}||\boldsymbol{\beta}||^2 + C_{SVM} \sum_{i=1}^N \xi_i,$$

with respect to:

$$y_i(\mathbf{h}(\mathbf{x_i})^\intercal \boldsymbol{\beta} + \beta_0) \geq 1 - \xi_i, \quad \forall i,$$

$$\xi_i \geq 0, \quad \forall i.$$

This formulation of SVM is a quadratic optimization problem (i.e. no local minima) with constraints. But, by using Lagrange multipliers, this optimization problem can be seen as an optimization problem *without constraints*, in which the Lagrange function L_p needs to be minimized over $\boldsymbol{\beta}$, β_0 and ξ_i:

$$L_p = \frac{1}{2}||\boldsymbol{\beta}||^2 + C\sum_{i=1}^N \xi_i - \sum_{i=1}^N \alpha_i \big[y_i(\mathbf{h}(\mathbf{x_i})^\intercal \boldsymbol{\beta} + \beta_0) - (1-\xi_i)\big] - \sum_{i=1}^N \mu_i \xi_i \tag{7}$$

where $\alpha_i > 0$ and $\mu_i > 0$ are Lagrange multipliers. This formulation is known as the promal form of the optimization problem. By setting the

partial derivatives of L_p to zero, we obtain the following three conditions:

$$\boldsymbol{\beta} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{h}(\mathbf{x_i}), \tag{8}$$

$$0 = \sum_{i=1}^{N} \alpha_i y_i, \tag{9}$$

$$\text{and } \alpha_i = C_{SVM} - \mu_i, \quad \forall i. \tag{10}$$

The problem could then be solved in its primal form, but it is usually considered in its *dual* form:

$$\max_{\alpha_i} L_d,$$

subject to the constraints:

$$0 \leq \alpha_i \leq C_{SVM},$$

$$\alpha_i y_i = 0,$$

where, the dual function L_d is obtained by combining the equations 8, 9 and 10 into equation 7:

$$L_d = \sum_i \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{i'=1}^{N} \alpha_i \alpha_{i'} y_i y_{i'} \langle \mathbf{h}(\mathbf{x_i}), \mathbf{h}(\mathbf{x_{i'}}) \rangle.$$

Additional constraints are obtained using the Karush-Kuhn-Tucker conditions [KT14]:

$$\alpha_i [y_i (\mathbf{h}(\mathbf{x_i})^\intercal \boldsymbol{\beta} + \beta_0) - (1 - \xi_i)] = 0, \forall i,$$

$$\mu_i \xi_i = 0, \forall i,$$

$$y_i (\mathbf{h}(\mathbf{x_i})^\intercal \boldsymbol{\beta} + \beta_0) - (1 - \xi_i) \geq 0, \forall i.$$

The dual problem is then solved on the α_i, which then solves the SVM by identifying $\boldsymbol{\beta}$ and β_0. The parameter C_{SVM} is set by the user.

Besides, as the constraint functions are affine, the duality gap is zero and thus both primal and dual problems have the same solution. The interest of solving the problem in the dual space is computational. The solution of the dual problem needs only to compute the inner product of the data represented in the kernel space. This is the reason why the kernel trick is computationally realistic. The data do not need to be explicitly transformed into another space (which would be computationally expensive as the kernel space can be of an infinite dimension and thus significantly complicates the operations on the data). More formally, using the following notation $K(\mathbf{x_i}, \mathbf{x_j}) = \langle \mathbf{h}(\mathbf{x_i}), \mathbf{h}(\mathbf{x_j}) \rangle$ where K is the kernel function, the dual function L_d is rephrased as:

$$L_d = \sum_i \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{i'=1}^{N} \alpha_i \alpha_{i'} y_i y_{i'} K(\mathbf{x_i}, \mathbf{x_{i'}}).$$

The function h, transforming the feature vectors into the kernel space, is typically chosen so that K has a relatively low computational cost. Therefore, the problem can be solved in the dual space by only computing the scalar product of the data represented in the kernel space.

An advantage of SVM is its generalization capacities. Since the model is built on a very small subset of the data (the support vector), it is relatively resilient to outliers. The soft-margin (regularization) also helps to prevent over-fitting, which all in all, lead to robust models. The use of the "kernel trick" is another advantage of SVM, since it leads to better performance, even if the original feature space chosen by the user does not linearly separate the data. This is particularly valuable when the data are not known. However, these two advantages are related to the hyper-parameters that can notably influence the results. The choice of the hyper-parameters is therefore crucial as wrong parameters could decrease the efficency of SVM. Another disadvantage of SVM is its interpretation. Indeed, compared to other learning algorithms (e.g., Random Forest), it is not easy to interpret the outcome of the training phase.

2.3 Technical Approach: How to Proceed With Supervised Machine Learning Algorithms

We here propose a short tutorial on supervised machine learning and give some technical considerations to have in mind when working with real data and practical supervised algorithms. This section is more "hands on" than a formal description.

2.3.1 About the Dataset

It was previously stressed that any machine learning model should be built from a dataset that is representative of the phenomenon we want to model (i.e., the data should cover the variability of the phenomenon).Without such a representative dataset, the model can have a limited interest. Another important element in a dataset is the number of data. There is no generic rule to decide which number is necessary to train a model. It highly depends on the application. Supervised machine learning create a model that only replicates the behavior taught during the learning phase. When using supervised machine learning, a model can only replicate and somehow generalize the behavior taught during the learning phase. It can only recognize data similar to the one present in the learning set. The general rule is therefore to gather for each class as many data as needed to englobe and represent the variety of a class. If possible, we would therefore advise to gather a large number of data, that thoroughly describe the phenomenon under study. Preferably, the classes should be balanced in their number of data. [5]

However, when dealing with real-case scenario, the proposed dataset might be subject to constraints. For instance, the dataset can be already

[5] For machine learning consideration and to train an unbiased model. When analyzing a phenomenon, unbalanced classes already give us some information.

established and in this case, it is up to the analyst to "make things work". The rule of thumb in this context is to use as many data as possible. Some tricks can be used to increase the size of the dataset, for example, by duplicating and slightly modifying some data of the original set. This trick is known as the data augmentation process. It can be used to balance an unbalanced dataset. Moreover, if a dataset contains less than a few hundreds data, we would not necessarily recommend the use of machine learning approaches. Machine learning methods are designed to automatize or analyze large sets of data. Without a *large* set of data, the interest of machine learning approaches is disputable.

Another major issue concerning the dataset is the definition of a data for the wanted application. For example, when working on images for computer vision purposes, should the images be represented in gray scale or color? Should their size be standardized? Should their ratio be standardized? And so on. Before gathering an actual dataset, the very first question is to define the objects that will be automatically processed, in other words to define what is a data for a given application. This point will be detailed in the next chapter.

2.3.2 About the Features

The choice of the features can be done differently. Depending on the nature of the data, a set of features can already have been established, validated, and recommended. In this case, the feature set can be used as such and tests on the considered application will confirm the relevance in using this specific set of features. One of the contributions of this PhD is to propose a feature set for transient signals. If no feature set has been established and validated for the considered application, it is either possible to learn the features from the data, or to design them. This choice can be influenced by the physical meaning of the features. To design a new set of features for a given application, we would recommend to be familiar with the data. In particular, to understand why a data belong to a specific class. Indeed, designing feature is to write with math formulas the reason why a data belongs to a specific class. Depending on the chosen learning algorithm, it might be necessary to standardize the features with zero mean and unit standard deviation (typically, SVM works with distances and therefore need features with standardized values).

2.3.3 About the Learning Algorithm

To obtain an efficient model, the choice of the learning algorithm is crucial. There is no generic rule saying which learning algorithm should be chosen. With the early statistical models, assumptions on the data could help with this decision. However, with models outside the scope of statistics and complex data, the choice should be made on empirical results. According to [HTF09], the usual procedure for the testing strategy is the following. "*If we are in a data-rich situation, the best approach for both problems is to randomly divide the dataset into three parts: a training set, a validation set, and a test set. A typical split might be* 50% *for training, and* 25% *each for validation and testing. [...] [cross validation] is designed for situations where*

there is insufficient data to split it into three parts. Again it is too difficult to give a general rule on how much training data is enough; among other things, this depends on the signal-to-noise ratio of the underlying function, and the complexity of the models being fit to the data." The idea behind the use of training set, validation set and test set is to train the model with the train set and to use the validation set to find the model parameters that maximize the prediction. The test set is then used to estimate the model performances. If possible, it is crucial to use both validation and test sets when selecting the models hyper-parameters (typically the SVM kernel for instance). If a model is too simple, it will not be able to correctly learn the phenomenon represented by the data. This case is known as under-fitting. On the contrary if the model is too complex, it will give perfect results on the test set, but will not be able to generalize well to the independent data of the test set. This case is known as over-fitting. Avoiding under-fitting and over-fitting can easily be done by using the three separated sets. The prediction errors on the validation sets typically decreases with the model complexity. With the test set however, the prediction errors until the optimal model, but increases when the model is getting too close to the learning and validation set and looses its generalization capabilities. A model over-fitting can therefore be avoided by using a separated test set and monitoring the error rate. Typically, the error rate of the test set evolves with the model hyper-parameters by displaying a classical u-shaped curve (with optimum hyper-parameters at the minimum error). On the contrary, the error rate of the validation set typically decreases with the model complexity, and over-fitting is not visible on this plot.

In a situation where the dataset cannot be separated in three parts, it is advised to use cross-validation process. In this case, the global dataset is separated in two sets: the training set and the test set. Ratios of 90% - 10% or 2/3 - 1/3 are typically used. Several successive and random trials of training and test sets are done. Models are trained on the learning sets, performances evaluated on the test sets. The final results are considered as mean and standard deviations and can be used to set the models hyper-parameters and estimate its performances better than when using a single try-out.

Another point worth noticing, is that some machine learning algorithms such as SVM were originally designed on binary problems. However, it is possible to use them on a multi-class scenario by combining multiple binary classifiers with several strategies: *one-versus-all* strategy considers one class and gather the remaining classes in a second practical class while *one-versus-one* strategy considers several models of two classes, each grouping the original classes differently.

2.3.4 About the Validation Process

It exists two strategies to validate the performance of a model: (i) testing the model on the test set (assuming that there is enough data for having a training set and a validation set) or (ii) performing cross-validation. Both strategies requires metrics to measure and compare the results. The most natural metric is the *overall accuracy*, which measures the number of cor-

rect predictions, compared to the number to total predictions:

$$\text{Overall accuracy} = \frac{\#(\text{correct predictions})}{\#(\text{total predictions})}$$

The accuracy can also be evaluated class by class, for example for the class c_i, $1 \leq i \leq C$:

$$\text{Accuracy}_i = \frac{\#(\text{correct predictions in class i})}{\#(\text{data in class i})}$$

and thus, it is possible to define the *mean accuracy* to obtain a metric that is not impacted by unbalanced classes:

$$\text{Mean accuracy} = \frac{1}{C}\sum_{i=1}^{C} \text{Accuracy}_i$$

To measure the false alarm rate in a class c_i, the *precision* can be used:

$$\text{Precision}_i = \frac{\#(\text{correct predictions in class i})}{\#(\text{predicted as class i})}$$

Similarly to the accuracy, it is possible to define an overall precision and a mean precision. These metrics can be computed from the confusion matrix. For a testing process, the confusion matrix is a square matrix of size the number of classes C. It displays for each class the number of correct predictions, and the repartition of the wrong predictions on the other classes. It is a very convenient tool to analyze the results of a prediction model, in particular to see how it performs, and the main trends on the errors.

In this manuscript, we mainly use the overall accuracy and the class by class accuracy as principal metric, the idea is to optimize the good classification rate. The choice of which metric to use can be disputable, and depends on the applications. For instance, some applications require very few false detections (e.g., in medicine), while others prefer to have few false detections but not to miss any event within the classes. Moreover, these metrics are linked to each other. For instance, if the overall accuracy increases, this means that the number of correct predictions increase, and thus the overall precision increases.

As a final word on the validation process and on the metrics in general, we would like to say that objective metrics do not give enough information to validate or invalidate a process. Indeed, it is possible to obtain a very good accuray (indicating a very efficient model) with an extremely small dataset or with a very well chosen test set. Similarly, it is possible to obtain good results with cross-validation but, by not looking to the standard deviation of the metrics over the set of trials, the disparity between the different trials is not mentionned. In this manuscript, we try as much as we can to present the numerical results of the metrics, but also to discuss directly confusion matrices, and to analyze the trends behind the numerical results.

Highlights & Summary

- Machine Learning is a very wide topic, including many methods and algorithms related to the processing of datasets (from hundreds to millions of data). Learning algorithms are divided in two main groups: *unsupervised methods* and *supervised methods*. Unsupervised methods are used to perform an analysis of a dataset by grouping similar data together. The outcome of this analysis is a model. This model can then be used to classify the data contained on another dataset. The analysis is not biased by any human knowledge. Supervised methods on the other hand are used to replicate and automate a human behavior, typically for automatic classification purposes. In this case, a labeled dataset is considered: a data set containing examples of all the classes. A model is learnt from this labeled dataset, and used to analyze new data. This stage is known as training or learning. Throughout this work, we use supervised machine learning to analyze environmental data.
- Two main supervised learning algorithms are used in this work: support vector machines (SVM), and random forest (RF). If the data are represented in a space where they are as separable as possible, the learning algorithm should have a limited influence on the results.
- The first step to use machine learning algorithms is to transform data into feature vector. To do so, several methods can be used, including dictionary learning and handcrafted feature extraction. In this work we handcraft features by using their physical meaning on the data. To this end, we review in the litterature the features used to represent transient signals. Nevertheless, we explore in Chapter 6, the possibility of using dictionary learning to extract features from the data.

Chapter 3

Proposed Schemes for the Automatic Analysis of Environmental Data

Contents

Figure 13
Workflow used to build an analysis model.

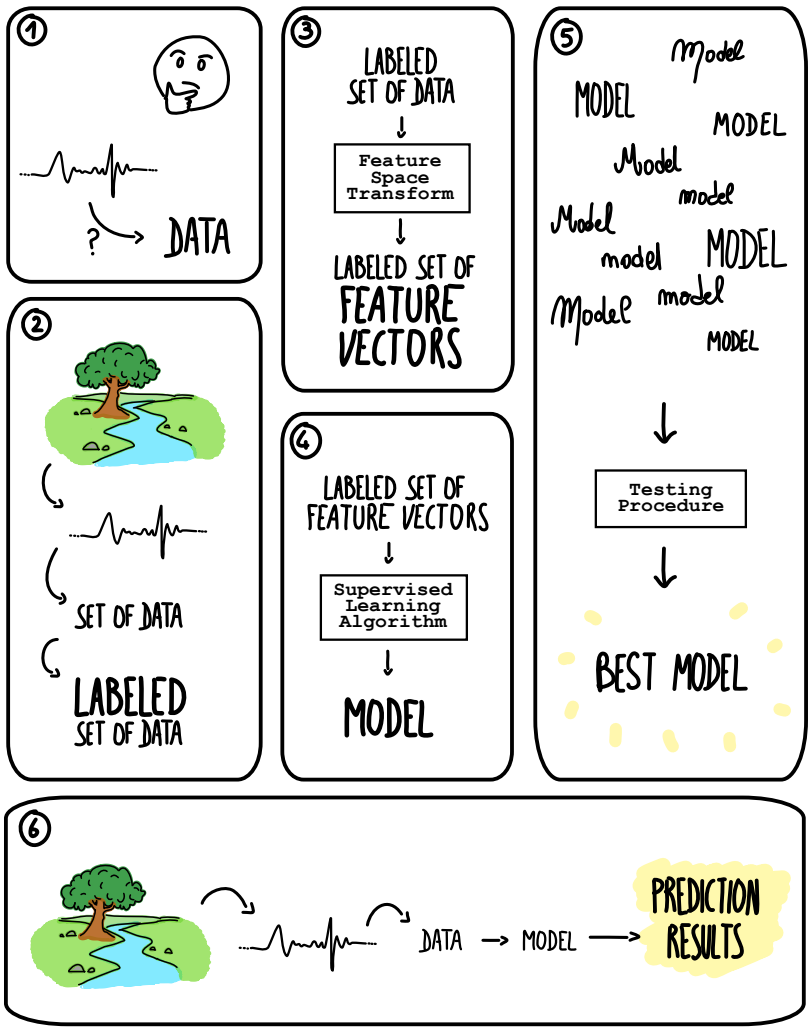
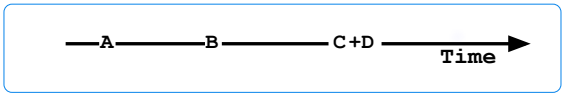
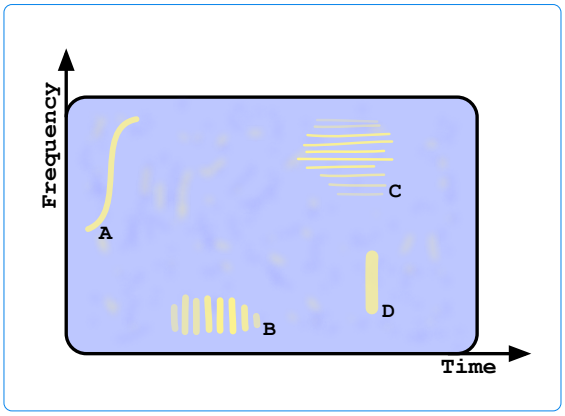


Figure 14
Schematic of four transient signatures A, B, C, and D, represented as (a) a time series or as (b) a spectrogram.



(a) Time series



(b) Spectrogram

Introduction

In this chapter, we propose and explain two automatic processing schemes of natural signals that have been developed during this PhD. These two proposed architectures answer the following problems: (i) automatic classification of environmental data (Section 3.1.1), and (ii) automatic detection and classification of environmental data (Section 3.1.2). Both architectures have been implemented in Python and are available on GitHub under the name Automatic Analysis Architecture (AAA) [Mal18]. The two automatic analysis architectures along with the associated code are the major contributions of this PhD and are illustrated in Figure 13.We present in the following each step of this workflow, from the recordings to the analysis model:

❶ The first step consists in defining what is considered as data for the learning algorithm (i.e., the objects on which the models work). For each new application, what constitutes a data must be defined. For example, in the context of this PhD, we focus on environmental monitoring. We therefore work with time series applied to acoustic (Chapter 4) or seismic (Chapter 5). The analysis of these data can be carried out in real time, or on a posteriori recordings. In context of environmental monitoring, we refer to those data as observations. A recording (i.e. from the original set of raw time series) is made of a large number of observations. The transformation from a recording to the set of observations depends on the analysis scheme. The definition of an observation is different for the two proposed architectures. Each analysis scheme is detailed in Section 3.1.1 and Section 3.1.2.

❷ A set of data used to train the analysis model should then be considered. As the proposed analysis architectures use supervised learning algorithms, the set of observations needs to be labeled. This labeling task is often carried out manually by the experts of the applicative domain, and is time consuming.

❸ Each observation is transformed into a feature vector. The reasons for using an alternative representation for the observations are exposed in Chapter 2.1. The process we propose to extract feature is detailed in Section 3.2. This process is one of the contributions of this PhD.

❹ We train an analysis model using supervised machine learning algorithms (presented in Chapter 2.2).

❺ We validate the accuracy of the model using validation methods such as cross-validation (see Chapter 2.3).

❻ The model is then operational and can be used to analyze an environment.

The context of this PhD notably influences the proposed architectures. In particular, working to develop operative tools for environmental monitoring means that the system must be validated on real-world conditions. The method should therefore be reliable, efficient and fast enough to process continuous flow of newly recorded natural signals.

Synopsis

In this chapter, we fully detail the proposed architecture but we remain relatively general with the data: the reader should keep in mind that the architecture are here proposed for environmental signals (various applications are considered in Chapters 4 and 5), but can be used on more general transient signals. For this reason, the illustrations of this chapter are made on schematic data: continuous data recorded as times series and displaying various transient events. These data can be represented as a series (Figure 14(a)) or as a spectrogram (Figure 14(b)). We first focus on describing the proposed architectures. The emphasis is then made on the feature space that is proposed for the representation of transient signals. The last section describe and explains the code imlementing the Automatic Analysis Architecture.

3.1 Proposed Architecture Schemes

3.1.1 Model for Static Analysis: Automatic Classification of Events

The first architecture scheme is dedicated to the automatic classification of environmental events. Namely, a time series recorded from an environment can be seen as a background noise on which occur some events that are represented by their associated signatures (see the schematic example of Figure 14). For instance, on acoustic signals recorded near the sea coast, one can hear the noise of a wave crushing the coast, or the call of a seagull. In this case, the wave crushing the coast, or the seagull call are events associated to their classes. Their sound is the imprint of the event on the recording (i.e., the signature), and is in this first scheme also defined as an observation. In this section, we propose an architecture that builds a model to automatically predict the class on which an observation belongs. The automatic processing of continuous signals recorded from an environment is therefore made of a detection stage to extract the observation[1], followed by the automatic classification scheme here proposed. For this reason, we refer to this model as a *static* model, to be used on a discret observations (by opposition to the second scheme exposed in Section 3.1.2 where a sliding window is used for the analysis). See Figure 15 for an illustration of the process.

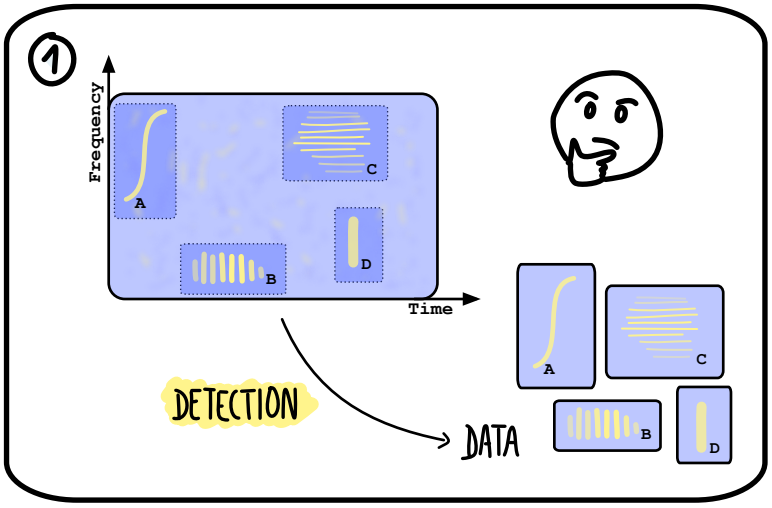


Figure 15
First step of the workflow for the first architecture. An observation is defined as a portion of a signal, filtered in its bandwidth. It contains an event and is detected manually or by an external algorithm.

In this context, a set of data is made of observations, and *each observation contains exactly one class* (positive). As shown by Figure 15, a frequency filtering might be necessary to respect such a constraint. Typically, if at a given time more than one class are present, the signals can be filtered in the corresponding bandwidths to separate them. An observation is then defined as the signal filtered in the event bandwidth. In this case, several events can occur at the same time, but in different frequency ranges. The condition "*each observation contains exactly one class*" remains valid. Visu-

[1] The automatic detection scheme usually depends on the applicative field, for example STA/LTA methods for seismic events.

ally, each observation can be represented by a time-frequency area of the recording.

A dataset of labeled observations is used to train the classification model. As explained in Chapter 2.3, the dataset (and the features) should be representative of the studied phenomenon. Typically, the number of classes present in the training set C corresponds to the number of types of events denoted #(Different Events). It also corresponds to the number of positive classes, and to the number of classes present in the test set of observations C_{test} (i.e. observations that need to be analyzed).

$$C = \#(\text{Different Events}) = \#(\text{Positive Classes}) = C_{test}$$

In that respect, the list of considered classes should be exhaustive and representative of (i) the observations that are used to train the model, but also of (ii) the observations that the model will analyze. Going back to the example of acoustic recordings of a sea coast, a set of classes associated to a training set of observations could be:

$$Classes = \{\texttt{wind}, \texttt{seagull}, \texttt{wave}, \texttt{rain}\}.$$

In this case however, an observation of thunder would be problematic to analyze since `thunder` is not in the list of classes. Again, the condition "*each observation contains exactly one class*" should be respected, and an observation can only contain a class that is present in the training set.

The model is trained from the training dataset of observations represented as feature vectors. The feature extraction process we propose is a full contribution of this work, and is detailed in Section 3.2. SVM or RF can be used as supervised learning algorithms. The classification model thus produced is then tested (with cross-validation for instance, see Chapter 2.3) and used to analyze newly recorded observations.

When analyzing newly recorded observations, it is possible to add an extra step to *detect anomalies*: observations that belong to none of the classes seen in training. The model output is made of the probabilities of the analyzed observation belonging to any of the classes. Typically, the selected and predicted class is the one associated to the highest probability:

$$c_{Predicted} = \underset{c_i \in Classes}{\text{argmax}} \; p(observation \in c_i)$$

It is however possible to apply a threshold on the output probabilities. If the selected class has a probability higher than the threshold, the prediction is confirmed. If the probability is lower than the threshold however, the prediction is rejected and an anomaly is detected. We can consider `Anomaly` as an extra virtual class that is not in the training set, but that can be detected in the testing set (and when using the model on real conditions). We refer to the *true* classes of events as the positive classes. In contrast, `Anomaly` is part of the negative classes. With this trick, the model aims is still to reproduce what has been taught during the training process, but it extends it to also reject observations that are too different from the training observations. In this scenario, each observation still contains

exactly one class, but the class can be positive (event classification), or negative (anomaly detection). In this case :

$$C = \#(\text{Different Events})$$

and,

$$C_{test} = C + 1 = \#(\text{Positive Classes}) + \#(\text{Negative Classes})$$

This architecture is used in Chapter 5 for the automatic analysis of seismic signals recorded on volcanoes.

3.1.2 Model for a Continuous Analysis: Automatic Detection and Classification of Events

In this second architecture, we propose to build a model automatically detecting and classifying environmental events. In this configuration, the time series recorded from the environment is still seen as a set of event signatures occurring over background noise. However, the definition of an observation differ from the one explained in the previous section. We here propose to continuously cover the recordings with a sliding window in time-frequency. An observation is therefore an extract of the recording that does not necessarily contain the signature of an event (i.e., no positive classes). Instead, the observation only contains background noises, a section of a signature, or even several incomplete signatures. Therefore, the prediction model automatically detects if an observation represents an event, and if it does, to which class the event belongs. As opposed to the static model, we here propose an analysis with a *continuous* model. See Figure 16 for an illustration of the process. With this architecture, the rule "*each observation contains exactly one class*" is still valid, but the class can be positive (the event signatures) or negative (background). A frequency filtering can also be justified if several classes are present at the same time. An observation is then defined by the signal within the window, and is therefore a portion of the recording filtered in a given bandwidth. With the example of the sea side recordings, the model can classify any portion of the recording (i.e., any observation) as being background noise, or containing a specific noise.

Similarly to the previous architecture, a labeled dataset of observations is used to train the prediction model. The dataset still need to represent all the variability of the studied phenomenon. This means that the dataset should include (i) observations of all the classes of interest (i.e., positive classes), and (ii) observations of the background noise. The background is here part of the negative classes. The number of classes of the training set C is therefore:

$$C = \#(\text{Different Events}) + 1$$

and the number of classes of the test set C_{test} (or the number of classes in real conditions):

$$C_{test} = \#(\text{Positive Classes}) + \#(\text{Negative Classes}).$$

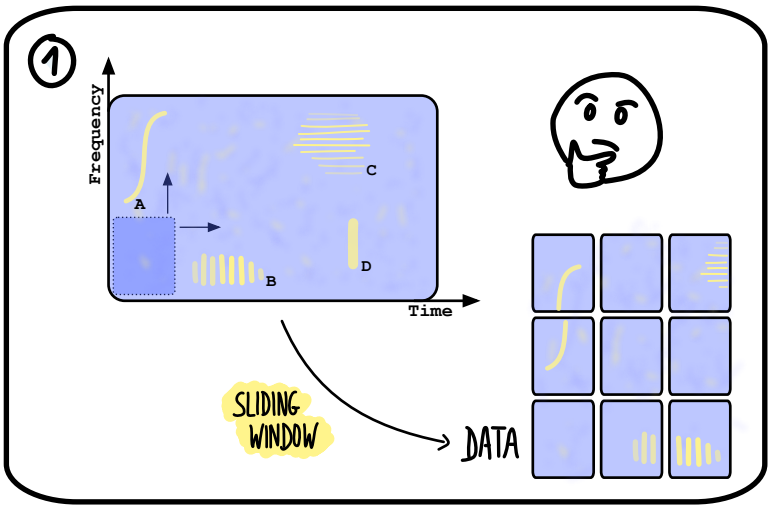


Figure 16

First step of the workflow for the second architecture. A data of observation is defined as a portion of a signal, filtered in its bandwidth. The observation can be represented by a sliding window of the time-frequency plan, and continuously covers the signal.

In this scenario, the number of classes of the training set is similar to the number of classes of the test set:

$$C = C_{test}$$

Likewise, the list of the considered classes should be representative of the observations that will be analyzed.

The prediction model is trained from the learning algorithm (SVM or RF) and the training dataset of observations represented as feature vectors. The feature extraction process is similar to the one used in the previous section. A detail description of that process is available in Section 3.2. As shown by Figure 16, the analysis of the recordings are made in two times: (i) they are scanned by the sliding window in time and in frequency, and (ii) the model classifies each observation to identify if it contains an event and if it does, the class on which it belongs to. In addition to the classification tasks, a model produced by this second architecture is also able to perform detection tasks. In that case, an extra class is considered for the background noise. Besides, the detection stage is performed by opposing the negative class (i.e., background noise) in which no event is present, to the positive classes displaying the various type of events. Moreover, it is also possible to detect anomalies by using a threshold on the output probabilities of the model. For detection tasks, we consider two negative classes: `background` and `anomaly`. Threfore, the number of classes of the training set C is equal to:

$$C = \#(\text{Different Events}) + 1 = \#(\text{Positive Classes}) + (background)$$

and the number of classes of the test set C_{test} to:

$$C_{test} = C + 1 = \#(\text{Positive Classes}) + \#(\text{Negative Classes})$$

We present in Chapter 4 an application of this code to analyze underwater acoustic recordings.

	Architecture 1	Architecture 2
Goal	Automatic *classification* of events.	Automatic *detection* and *classification* of events.
Observation	Signal within *discrete* window of the time-frequency plan: Δ_t and Δ_f are *variable*. The discrete window represents *exactly* one event (positive).	Signal within a sliding window of the time-frequency plan (*continuous*): Δ_t and Δ_f are *constant*. The sliding window represents *exactly* one event (positive or negative).
Training set	The number of classes equals the number of event types: the *positive classes*. The training set excludes mixed observations.	The number of classes equals the number of positive event + 1: the *positive classes* and the *background noise*. The training set excludes mixed observations.
	$C = \#(\text{Different Events})$	$C = \#(\text{Different Events}) + 1$
	$C_{test} = C$	$C_{test} = C$
Features	Shape descriptors extracted from various domains of the observations, see Section 3.2.	
Training	Supervised learning algorithm (e.g., SVM or RF)	
Use	1) Classification of events (observations are extracted using the same process used for the training observations). 2) Extension to anomaly detection possible (threshold on the output probabilities), and $C_{test} = C + 1$ 3) Confidence indicator on the prediction results using the output probabilities.)	

Table 1
Comparaison between the two architectures proposed for the automatic analysis of environmental signals.

3.1.3 Comparison Between the Two Architectures

In the Table 1, we compare and summarize the two previous architectures. To do so, we identify six characteristics to describe an achitecture: (i) the goal of the architecture, (ii) how an observation is defined for an architecture, (iii) the composition of a training set, (iv), the features, (v) the algorightm used during the training phase, and (vi) the possible usages of the model. In particular, we highlight for the two proposed architectures the points they have in common but we also emphasis their differences.

3.2 Proposed Feature Extraction Process

3.2.1 A Presentation of the Overall Idea

In Chapter 2.1, we explained the importance of choosing alternative representations for the observations. Such choice is important for signal processing problems but also for machine learning applications. We here propose a feature extraction scheme to represent transient signals. This feature extraction process is one of the contributions of the PhD. It is led by the following ideas:

1. Features used for machine learning application should be *high level features* (see Chapter 2.1).

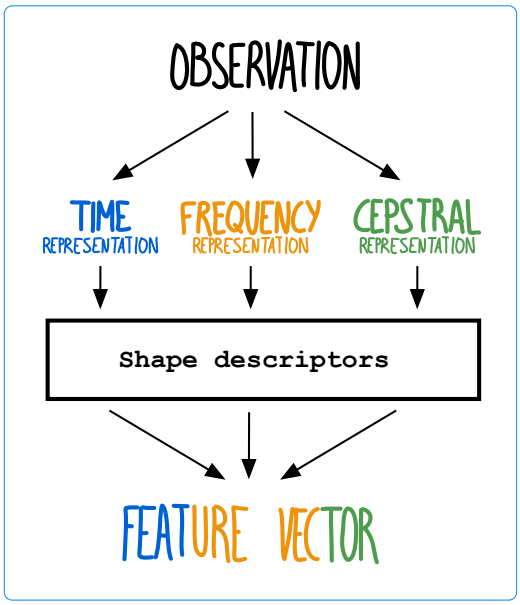


Figure 17
Feature extraction process.

2. Mathematical transforms can be used on signals to underline some given properties. For instance, we use Fourier transform to study the spectral content of a signal and its periodicity properties.

Combining those two ideas, we propose the following feature extraction scheme, illustrated in Figure 17. From one signal, several low level representations are considered. Each low level representation is associated to a classic signal domain and to its mathematical transform. A set of general shape descriptors is then extracted from each low level representation. The shape descriptors extracted from the various representations are then concatenated into the final feature vector.

3.2.2 Focus on the Shape Descriptors

We list in Table 2 a set of shape descriptors that can be used on transient signals. All of them have already been used or suggested in several works. These references can be found in the table. Besides, for each descriptor, we give its physical interpretation together with its associated formula (and computational precisions if needed). As we will see in Section 3.2.3, we can extract all these descriptors from various low level representations of an observation. All these descriptors have been implemented in the Python code we release on GitHub [Mal18].

3.2.3 Focus on the Low Level Representations

In signal processing, it is common to study alternative representations of signals to better display and understand its inherent properties. Fourier transform is the most common representation, but many others can be considered (e.g., cosine or wavelet transforms). We here base our feature extraction scheme on the idea that various low level representations underline distinct and complementary properties of a signal. One representation is not better than another, they simply highlight different aspects of a signal. For instance, the same informative content is displayed in

Table 2

General shape descriptors. The formulas are given using a generic signal $[z_i]_{i=1}^n$, which can be the observation as one of the three low level representation: s_k, S_f or S_q. i can therefore refer to time, frequency or quefrency.

Feature	Description	Formula
Mean	Mean value of the signal	$\mu = \frac{\sum_i z_i}{n}$
Standard deviation [TB05]	Measure of signal mean spread around μ.	$\sigma_z = \sqrt{\frac{1}{n-1}\sum_i (z_i - \mu_z)^2}$
Skewness [ZVH+10, Lan14]	Measure of the signal asymetry	$\frac{1}{n} \cdot \sum_i \left(\frac{z_i - \mu_z}{\sigma_z}\right)^3$
Kurtosis [ZVH+10, Lan14]	Measure of the signal peakness (compared to the Gaussian distribution)	$\frac{1}{n} \cdot \sum_i \left(\frac{z_i - \mu_z}{\sigma_z}\right)^4$
Centroid [HYYC09, Fag07]	Point (time, frequency or quefrency) around which the observation's energy is centered	$\bar{i} = \frac{1}{E}\sum_i i \cdot E_i$
RMS bandwidth [TB05]	Measurement of the bandwidth (temporal, spectral or cepstral) in which the signal's energy is spread around $\bar{i}$	$RMS_i = \sqrt{\sum_i \frac{i^2 \cdot E_i}{E} - \bar{i}^2}$
Mean skewness [TB05]	Skewness measure adjusted to the signal energy	$\sqrt{\frac{\sum_i (i-\bar{i})^3 . E_i}{E . RMS_i^3}}$
Mean kurtosis [TB05]	Kurtosis measure adjusted to the signal energy	$\sqrt{\frac{\sum_i (i-\bar{i})^4 . E_i}{E . RMS_i^4}}$
Shannon entropy[a] [EKR04, HMD11]	Measure of the informative content within the signal	$-\sum_i p(z_i) . \log_2 \left(p(z_i)\right)$
Rényi 'entropy'[b] [HMD11]	More general entropy measurement	$\frac{1}{1-\alpha} \cdot \log_2 \left(\sum_i p(z_i)^\alpha\right)$
Length [HMG+14]	Duration of the event	n
Max [Lan14]	Maximum amplitude of the signal	$\max_i \left(z_i\right)$
Min [Lan14]	Minimum amplitude of the signal	$\min_i \left(z_i\right)$
i of max [TB05]	Point (time, frequency or quefrency) of the maximum amplitude	$\text{argmax}_i \left(z_i\right)$
i of min [TB05]	Point (time, frequency or quefrency) of the minimum amplitude	$\text{argmin}_i \left(z_i\right)$
Rate of attack [TB05]	Description of the maximum increasing slope in the signal	$\max_i \left(\frac{z_i - z_{i-1}}{n}\right)$
Rate of decay [TB05]	Description of the minimum decreasing slope in the signal	$\min_i \left(\frac{z_i - z_{i+1}}{n}\right)$
Threshold crossing rate[c] [HYYC09, Fag07]	Number of time the signal crosses a given threshold. Helps describing the signal mean level and its noisyness	$\frac{\#(\text{Threshold Crossing})}{n}$
Silence ratio[c] [MZB06]	Ration between the signal length above and under a given thresholds. Helps describing the general shape of the signal.	$\frac{\#(\text{z where z<threshold})}{n}$
Max over mean [HMG+14]	Comparison between the signal peak value and its mean level	$\frac{\max_i z_i}{\mu_z}$
Min over mean [HMG+14]	Comparison between the signal lowest value and its mean level	$\frac{\min_i z_i}{\mu_z}$
Energy standard deviation[d] [Foo97]	Spead of the signal energy around the mean energy	$\sigma_E = \sqrt{\frac{1}{n-1}\sum_i (E_i - \mu_E)^2}$
Energy skewness[d] [Foo97]	Asymetry of the signal energy	$\frac{1}{n} \cdot \sum_i \left(\frac{E_i - \mu_E}{\sigma_E}\right)^3$
Energy kurtosis[d] [Foo97]	Peakness of the signal energy	$\frac{1}{n} \cdot \sum_i \left(\frac{E_i - \mu_E}{\sigma_E}\right)^3$

[a] Various bin numbers can be used for probability estimation (5, 30, 500 for instance. Experimental results showed the interest of using several estimations

[b] Similarly to Shannon entropy, various bin numbers can be used for the probability estimation. α can take several values, including $\alpha = 2$ for the collision entropy, or $\alpha = \infty$ for the Min-entropy.

[c] Signal maximum normalized to 1, and different threshold values can be used (0.2, 0.4, 0.6 and 0.8 for instance.

[d] Energy measurements are features computed from the signal energy $E_i = z_i^2$ rather than on the signal itself.

a time signal than in its complex spectrum, but the information is organized differently and a given property is often not visible in all domains. The time series is used to study the waveform, while the frequency domain helps with studying the spectral properties. We here propose to extract the general shape descriptors we proposed in the previous section. This can be done from three different low level representations of an observation $[s_k]_{k=1}^n$.

- **First representation and temporal properties** — The temporal properties of a given observation are simply best displayed on the observation itself $[s_k]_{k=1}^n$. This first low level representation describes the waveform of an observation, and is often the first one studied by the experts of the applicative field.

- **Second representation and spectral properties** — The spectral properties of an observation are displayed using the Fourier transform $\mathcal{F}\{\cdot\}$ computed on the original observation. It is often its module S_f that is considered ($S_f = |\mathcal{F}\{s_k\}|$). Therefore, thanks to such a representation, the prediction model is able to use spectral properties of the set of observations. For instance, the Fourier transform highlights periodic properties, which is of interest for pattern B of Figure 14(b) for instance.

- **Third representation and cepstral properties** — The final representation considered for the extraction of the shape descriptors can be referred to as the cepstral domain. Originally, the cepstral domain was used in speech processing through the MFCC feature set [?]. The cepstral domain models the harmonic properties of a given signal (speech recordings being highly harmonic signals), and several implementations can be used [?, GFK05]. Its variable is known as the quefrency. In this work, we define the cepstral representation by computing the Fourier transform twice on the original signal $\mathcal{S}_q = |\mathcal{F}\{S_f\}|$, and propose to use it as a general description of an observation. [2] Shape descriptors are therefore extracted from the low level representation too.

In addition to the three different low level representations, it is possible to preprocess the observations before computing the various low level representations. Indeed, depending on the application, it is important to normalize the observations in terms of mean or amplitude, to use a logarithm scale, or others. For instance, normalizing the energy of the original observation to 1 will produce a model invariant to the signal energy. Therefore, observations will only be classifyied on their shape regardless their energetic level.

[2] Among the various implementations considered for the cepstral domain, the DCT or inverse Fourier transforms can also be considered. Similarly in a context of speech processing, it is usual to apply a logarithm on the first Fourier transform in order to highlight lower frequency. In this work we compared various implementations of the Cepstral domain and decided upon using the Fourier transform twice. Typically, applying a logarithm on the spectrum did not impact the results.

Similarly, the feature vectors can be normalized in terms of mean and standard deviation for each feature. This step is not required by all learning algorithms. But, for some that we are using (e.g., SVM or neural networks), normalization helps to successfully train models.

3.3 An Overview of the Python Implementation

In this section, we briefly describe the code we wrote to implement the two proposed architectures. The README is available in Appendix B. The code is written in Python3 and uses several librairies (e.g., scikit learn for the learning algorithms). It is publicly available on GitHub [MDMM+ss] under the name *Automatique Analysis Architecture* (AAA). It is compatible with a large number of data formats (e.g., WAV, MP3, DAT). Data can be either be stored locally for a posteriori processing or accessible remotely on a server for a real time processing. All the settings for the analysis are stored in configurations files, which are organized as follow:

General – Which contains general information, such as project name (different name for different applications, different folder results, etc), path (root, to specific and feature configuration files, to the data, etc), name and settings of the learning algorithm, choice of the validation procedure, preprocessing strategy (e.g., energy normalization), and threshold values (for anomaly detection).

Specific – Specific setting depending on the type of analyze we want to do:

- USECASE1: **Continuous Analysis** – Used for the automatic and continuous processing of data. This usecase corresponds to the architecture presented in Section 3.1.2 (for automatically detecting and classifying events). This usecase can be used in an operative context.
- USECASE2: **Offline static Analysis** – Used for the automatic and discrete classification of events. It corresponds to the architecture proposed in Section 3.1.1 and Section 3.1.2, for signals that are already read and stored in `numpy.ndarray` format.
- USECASE3: **Static Analysis** – Used for the automatic and discrete classification of events, and implements the architecture proposed in Section 3.1.1.

Features – Which lists the shape descriptors functions to be extracted from the various low level representations of the observations, and lists the low level representations that are to be considered.

Each type of analysis (i.e., Continuous Analysis, Offline static Analysis, and Static Analysis) is accessible with its own script: a Python playground (to execute the code line by line, to develop and run more tests, typically for research purposes), or via a bash script (for operational use for instance). Typically, the analysis are run as follow:

- For the continuous analysis:

```
./make_usecase1.sh setting_file action verbatim
```

- For the offline static analysis:

```
./make_usecase2.sh setting_file verbatim
```

- For the static analysis:

```
./make_usecase3.sh setting_file action verbatim
```

where `setting_file` is the path to the main configuration file, `verbatim` is a digit between 0 and 3 depending on how much information we want to display and `action` is one of the four following strings: `training` to train and save a model, `analyzing` to run an analysis, `make_decision` to make decision using the output probabilities obtained in an analysis, or `display` to simply display (and save) the results of an analysis .

For research purposes or to better understand how the code works, it is possible to start an analysis in a Python playground with the following commands:

- For the continuous analysis:

```
python3 PLAYGROUND1.py
```

- For the offline static analysis:

```
python3 PLAYGROUND2.py
```

- For the static analysis:

```
python3 PLAYGROUND3.py
```

As previously exposed, the different analysis are build upon similar steps: feature extraction, training of a model, validation and use of the model. Therefore, as the aforementioned scripts used the same methods and functions, they are relatively similar. We give bellow an example for the playground_1 and the automatic classification of sparse events. The script is a high level script that uses three main classes (i.e., Python classes, not classification classes).

`Config` – which stores all the settings read from the configuration files (general and specific). The parameters are checked after loading the setting files.

`Analyzer` – which is the only object needed to run an analysis. It contains the label encoder, which encode labels (string, non consecutive digits) into digits form 0 to $C - 1$, the object used to normalize the feature (mean and standard deviation), and the model itself. The analyzer needs to be trained using the `learn` method. During the training of the model, (i) a catalogue of labeled data is read and loaded, (ii) the data are preprocessed, (iii) features are extracted and scaled and (iv) the model is trained and tested using cross-validation. The catalogue containing the labeled data has for

type `pandas.dataframe`. Such a type indicates for each labeled observation the file name where the observation is store, the observation data and time, and its class.

`Dataset` – which contains the recording that needs to be analyzed. Each file that needs to be analyzed is represented by an object of type `Recording`. This step works as follow: (i) a dataset is initialized, (ii) the dataset is analyzed, (iii) the method `makeDecision` is called to decide the final classes on which each recording belongs to (using the class probabilities obtained from the model), and (iv) the results are displayed graphically with the method `display`. The method `makeDecision` also contains the code related to the anomaly detection.

```
import json
import numpy as np
from os.path import isfile
from tools import *
from dataset import Dataset
from config import Config
from recording import Recording
from analyzer import Analyzer
import pickle

# Change if you want your screen to keep quiet
# 0 = quiet
# 1 = in between
# 2 = detailed information
verbatim = 2

# Init project with configuration file
path = '../config/general/newsettings_16.json'
config = Config(path, verbatim=verbatim)
config.readAndCheck()

# Make or load analyzer
analyzer = Analyzer(config, verbatim=verbatim)
# analyzer.load(config)  # Comment or uncomment
analyzer.learn(config) # Comment or uncomment

# Dataset that needs analyzing
# (files to analyze are specified in configuration file)
analyzedSet = Dataset(config,verbatim=verbatim)
analyzedSet.analyze(analyzer, config, save=True)
analyzedSet.makeDecision(config, save=True)
analyzedSet.display(config, onlineDisplay=False, saveDisplay=True)
```

As it was previously mentionned, it is easy to re-use this code for a new application without requiring so much programming knowledge. To do so, a user has to follow the following steps:

1. Duplicate and adjust the configuration files (i.e., the desired path, the type of analysis and the features used by the learning algorithm),
2. (Potentially) Write a reading function (if the format of the data is not supported by the current reading functions).

3. Organize the labeled catalogue as a `pandas.dataframe` object containing for each observation the information needed to get the signal, and the observation class. The observations can be read locally or remotely.

4. Run the wanted analysis, either in playgrounds scripts for research and offline analysis purposes, either using the makefiles for a more operative approach.

For more information on the code and how to use it, the reader is welcome to consult the project AAA readme in Appendix B, or to consult the GitHub deposit.

Highlights & Summary

- In this chapter, we present and describe three major contributions of the PhD.
  - We propose a scheme for the automatic classification of time series (the recordings) displaying the signatures of environmental events.
  - We propose a scheme for the automatic detection and classification of environmental events in continuous recordings of time series.
  - Finally, we propose a feature extraction scheme based on the following idea: a set of general shape descriptors are extracted from three different low level representations of an observation. The temporal is used to describe the observation waveform ($[s_k]_{k=1}^{n}$). The spectral domain is used to describe the spectral properties of an observation ($S_f = |\mathcal{F}\{s_k\}|$). And the cepstral domain is used to describe the harmonic properties of an observation ($\mathcal{S}_q = |\mathcal{F}\{S_f\}|$). The final feature vector is a concatenation of the shape descriptors extracted from each of the three low level representations. The general set of shape descriptors comes from an extensive analysis of the state of the art in various domains displaying transient signals. This proposed feature extraction scheme can be used to represent transient signals (from environmental sources or not).
- Both automatic analysis architecture are detailed, explained, and compared. The code associated to each training scheme is proposed in [Mal18]. The training of prediction models is based on supervised learning algorithms.
- In terms of terminology, we consider the positive classes representing environmental events and the negative classes which are either the background noise (empty observation), either an anomaly. We also refer to observation to describe the data used as input in the analysis model. The definition of an observation in respect to the original recordings is adjusted depending on the chosen automatic analysis scheme.

Chapter 4

Application: Monitoring of Coastal Underwater Areas Based on the Analysis of Acoustic Recordings

Contents

Introduction

In this first applicative chapter, we focus on monitoring underwater coastal areas by analyzing continuous acoustic recordings. In particular, the chapter aims at reviewing the existing methods used for Passive Acoustic Monitoring (PAM) and see how automatic analysis methods can be used for this purpose.

Seas & Coastal Areas

The global ecosystem of the Earth is ruled and influenced by many factors. With water covering around 70% of the Earth's surface, seas and oceans have a major role in the regulation of this global ecosystem. From an economical perspective, the value of the services provided by the seas is estimated at some $33 trillion per year [CddG+97], and is therefore substantial. From a biological point of view, seas and oceans are also unique areas hosting more than 240,000 different species. Given that oceans remain difficult and sometimes impossible to access however, it is particularly difficult to obtain exhaustive and reliable description about the underwater life.

In this study, we focus our attention on coastal underwater areas. Zones of shallow waters (depth < 100m) have been widely studied for the key role they held in marine environments. Among the various coastal environments, sea-grass meadows grow at maximum depths of 40m and are of particular interest. At a local scale, sea-grass meadows are the habitat of many fish species, as an environment that provides them with a source of food and where they can develop their nurseries [HHO03]. At a larger scale and because of their role in photosynthesis, sea-grass meadows are considered to be the lungs of the oceans. Coastal areas are, by definition, closer to land, and their ecosystems are particularly exposed and vulnerable to the human factor, known as anthropogenic stress. A special attention is therefore to be paid to the evolution of those areas. To do so, monitoring tools are necessary and are yet to be developed.

Monitoring of Underwater Areas & Acoustic Solutions

Several methods can be considered to monitor underwater areas, such as satellite imagery, field surveys or the analysis of acoustic signals. This chapter focuses on the use of acoustic signals. More precisely, sounds that can be heard underwater are recorded using hydrophones and then analyzed. This approach is known as Passive Acoustic Monitoring (PAM), in opposition to active methods where a known signal is emitted and its response studied for gather information on the environment. The choice of considering acoustics signals as a proxy for monitoring of sea-grass meadows instead of other approaches can be

supported by several arguments. First, water is a particularly favorable environment for sound propagation. Conversely, satellite or airborne imagery is of limited use due to the difficulty in accessing the required underwater information. This arises from the limited visibility with passive optical systems, and the lack of penetration of electromagnetic waves through the water surface with active acquisition systems. Secondly, using an acoustic-based approach allows monitoring of large areas. Indeed, large amounts of data, and even real-time recordings, are relatively easy to acquire. As the cost of data collection campaigns is relatively affordable, their use can be considered for monitoring over wide areas. Acoustic-based approaches are also less invasive and costly than in situ surveys of the sea floor. Finally, the presence of biodiversity can be seen as an indicator of the vitality of an area since fishes and sea animals generally communicate through acoustics. In this context, analyzing underwater animal sounds is a relevant choice for monitoring the biodiversity and vitality of a given coastal area. There are three different sources of underwater sounds: human (boats, divers, etc), animal (sea mammals, fishes, shells, shrimps, etc.) and environmental (storms, wind, rain, etc). Those three components are referred as anthropophony, biophony and geophony. Together they constitute the underwater acoustic landscape. By studying the acoustic landscape, information about the environment are gather, and the area can be monitored.

Originally, the use of acoustic to monitor underwater areas was related to active methods and for different purposes. Typically, the sonar is the most well known technique for underwater monitoring, and was originally developed for military purposes. Today, passive methods are also being considered, for applications that are relatively varied (military or not).

The use of PAM approaches lead to large amounts of data to be analyzed and registers in the Big Data issue. Machine learning methods are therefore naturally considered. Despite the need for automatic monitoring tools for underwater areas however, few operative approaches have been proposed. For this study, we focus on the biophony component of the acoustic landscape, targeting in particular fish sounds. Among the very few studies dealing with fish sounds classification, we can mention [VFAT15, NTSR16] and [SCSJ16] in which results are promising even if the tools are not all tested in an applicative context of underwater monitoring. The classification of bio-acoustics data is being developed in the literature, such as the sounds of sea mammals [ZVH+10], bats, frogs [HYYC09, CCL+12], birds [Fag07, ACBCB+09, THMP06] and other animals [MZB06]. However, automatic processing tools are not similarly developed in the different scientific communities. Depending on the field and data accessibility, the results have been relatively disparate. For instance, studies on speech recognition are relatively advanced, while automatic identification of musical instruments is at a more preliminary stage of development. However, to the best of our knowledge, no functional

tool on automatic fish sounds classification has been reported to this date, and thus we propose the present study as an attempt to fill this gap.

Synopsis

This chapter is built around the paper "Automatic Fish Sounds Classification" which we published in JASA is 2018 [MMDMG18]. The preliminary results also received the best paper award and was elected for a lay language paper at ASA conference 2016 [MDMMG16]. The chapter proposes to (i) review the state of the art regarding the automatic classification of acoustic transients (environmental sounds of natural, animal or human sources, speech, and music) and (ii) to use supervised machine learning approaches for the automatic detection and classification of fish sounds. The proposed tools are tested in various areas and under various experimental settings (learning and testing in different areas or at different times for instance). Several days of underwater acoustic recordings are then analyzed using the proposed tools. This chapter is best understood if the beginning of this thesis has been read. Nevertheless, this chapter can be read separately – and to this end some technical aspects are repeated. The recordings used for this study come from a partnership with the chair Chorus which studies the underwater soundscape. Thank you also to Robiah Al Wardah.

4.1 State of the Art on Automatic Methods for Passive Acoustic Monitoring

The literature regarding the automatic classification of fish sounds being extremely reduced, we here propose a survey of a wider task: the automatic classification of transient signals, including bio-acoustics, natural, human-induced, music or speech signals.

By surveying this literature, one can notice that a plethora of descriptors have been used as features for bio-acoustics signal classification. These features come from different domains such as statistics, information theory, signal and image-processing. Spectral centroid and bandwidth can be used to classify frog sounds [HYYC09]. [Fag07] also uses spectral centroid and bandwidth, and for recognition of bird species, they also include spectral roll-off frequency, spectral flux, and spectral flatness and duration. For the classification of whales and boats, [ZVH+10] uses energy centroids, standard deviation, skewness, and kurtosis, all of which are computed from the time and frequency signals. [ACBCB+09] uses the minimal and maximal signal frequency in addition to the energy in different frequency bands to classify frog and bird songs. A threshold crossing rate is used by [HYYC09] and [Fag07], and along the same lines, features based on area ratios above or below a given threshold are used by [MZB06] for the classification of bird, cat, cow, and dog calls. Regarding underwater acoustics, [NTSR16] introduces the use of Shannon entropy and call length for the recognition of fish sounds (in controlled environment though). Descriptors based on information theory are used in [HMD11], in which the authors implement Shannon and Rényi entropies to classify frog calls, and in [ZVH+10] for the discrimination of whale and boat based. Low-level coefficients and descriptors issued from various transforms of the input signals are also considered. For example, [THMP06] uses dynamic time warping and spectral ensemble average voice for bird recognition. [CCL+12] uses multistage average spectrum for frog detection. Linear predictive coefficients are also used in classification, as for calls from birds [MC97] or humpback whales [PBG+10]. [Che01] uses feature matrices from time-domain signal coding for insect and bird recognition. Finally, some studies consider features extracted from spectrograms, which can be pragmatically considered as images allowing one to take advantage of the large set of image processing tools available. For bird vocalization retrieval, for example, [DTZ+13] extracts features from spectrogram images using ridge detection and points of interest. [EZE14] also uses image processing techniques for dolphin call classification. Bowhead whale are detected and localized in [TKB+12], also using features extracted from the spectrogram. Finally, features can also be learned from the signals instead of handcrafted, for example as done in [SCSJ16] using principal component analysis for the classification of a given call of plainfin midshipman into three categories. Pertaining to the classification procedure, several machine learning techniques have been employed such as SVM used in [HYYC09, Fag07, ACBCB+09, MZB06, NTSR16, SCSJ16], neural networks (NN) are employed in [MC97, Che01, TKB+12] and [ZVH+10] while k-nearest

neighbor (kNN) are considered in [HYYC09, HMD11, EZE14, NTSR16]. Some studies also propose to use decision trees or linear discriminant analysis [ACBCB+09], distance measurements [THMP06, DTZ+13] or k-means [PBG+10]. [NTSR16] also uses Random Forest (RF).

For natural or human-induced sounds classification, very similar features are used. Statistical descriptors are used by [GL03] for multiple sounds retrieval, such as the frequency centroid, bandwidth in various frequency bands or pitch frequency and energy. To classify underwater mechanical transients, [TB05] uses a large number of perceptual features, which included energy standard deviation, skewness and kurtosis over the time or frequency axes. Image-processing techniques are also found for the description of these signals, such as by [DTL11] for classification of various sound events. To classify acoustic noise radiated by boats, [WZ14] extracts low level features based on Bark wavelet analysis and Hilbert Huang transform. Linear predictive coefficients are also used by [CFGM98] for environmental noise recognition. Regarding the learning algorithms, SVM are used in [GL03, DTL11] and [WZ14] along with distance measurements in [GL03]. kNN are also found in [TB05] and Hidden Markov Models are tested in [CFGM98].

Automatic classification for music sounds is found in several various applicative domains (e.g., content retrieval, musical instrument identification, musical genre identification), although here again, different features can be used to describe signals of interest. Statistical features for instance are used by [EK00] and [FM00] for musical instrument recognition and for timbre recognition. [EKR04] also uses statistical features along with entropy for classification of musical genre. Image processing methods have also been adopted, such as by [YS09] for spectrogram texture extraction to identify various musical instruments, and [DSN01], for musical genre classification. Learning algorithms used with musical data are similar to the ones used for the classification of biological, human or natural sounds. In particular, kNN is used in [FM00, YS09, DSN01], SVM is used in [DSN01] and linear discriminant analysis in [EKR04].

Speech classification has been mainly carried out considering Mel Frequency Cepstral Coefficients (MFCCs) as features, which are presented in Chapter 2.1.3. The success of MFCCs in speech-related studies is such that they are now considered as a reference set of features for acoustic classification purposes in general. The state of the art in automatic classification of fish sounds is very limited but both [VFAT15] and [NTSR16] use MFCCs for fish call recognition or fish individuals classification. In bio-acoustics, MFCCs are used by [BIDL14] for anuran sounds classification, by [Fag07] for bird call recognition, and by [THMP06] for bird species recognition. [PBG+10] uses MFCCs for humpback whale identification. [LCHH06] and [CJ06] modify MFCCs to fit their data, and they use them to classify frogs and crickets respectively, and for land mammal call identification. In acoustics, MFCCs are used by [GL03] and [DTL11] for multiple sounds identification, and by [LBHL07] for identification of underwater acoustic transients. They are also used by [WTP+10] for the identification of environmental sounds, and by [MMSFSGSP14] for

aircraft take-off noise classification. MFCCs are also used to describe signals to distinguish speech from music from nonvocal sounds by [Foo97], for musical instrument recognition by [EK00], and for musical genre classification by [DSN01]. Similarly to other applications, learning algorithms involved in those studies are mainly based on SVM [Fag07, GL03, DTL11, DSN01, NTSR16], neural networks [MMSFSGSP14], distance measurements [THMP06, GL03, LBHL07], k-means [PBG+10], kNN [DSN01, NTSR16], linear discriminant analysis [LCHH06], hidden Markov models [CJ06, VFAT15] or RF [NTSR16]. Fuzzy classifiers were also used in [BIDL14] and tree bases quantifier in [Foo97].

Table 3

Feature set for a generic numerical signal $[z_i]_{i=1}^n$ composed of n discrete samples and which represent an observation in one of the three considered low level domains: time s_k, frequency S_f or cepstral $\mathcal{S}_q$. E represents the signal energy. Features references are referred 'T', 'F' or 'C' depending on their computation domain respectively being time, frequency of cepstral. For instance, F28 refers to the energy kurtosis computed from the spectral domain. Features in bold font are the most valuable features (see Fig 21).

Feature	Definition	Ref.
Centroid [HYYC09, Fag07]	$\frac{1}{E}\sum_i i \cdot E_i$	T1, **F1**, C1
RMS bandwidth [TB05]	$RMS_i = \sqrt{\frac{1}{E}\cdot\sum_i i^2 \cdot E_i - \bar{i}^2}$	T2, F2, C2
Standard deviation [TB05]	$\sigma_z = \sqrt{\frac{1}{n-1}\sum_i (s[i]-\mu_s)^2}$	T3, F3, C3
Skewness [ZVH+10]	$\frac{1}{n}\cdot\sum_i \left(\frac{z_i-\mu_z}{\sigma_z}\right)^3$	T4, **F4**, **C4**
Kurtosis [ZVH+10]	$\frac{1}{n}\cdot\sum_i \left(\frac{z_i-\mu_z}{\sigma_z}\right)^4$	T5, F5, C5
Mean skewness [TB05]	$\sqrt{\frac{\sum_i (i-\bar{i})^3.E_i}{E.RMS_i^3}}$	T6, F6, C6
Mean kurtosis [TB05]	$\sqrt{\frac{\sum_i (i-\bar{i})^4.E_i}{E.RMS_i^4}}$	**T7**, F7, C7
Shannon entropy[a] [EKR04, HMD11]	$-\sum_i p(z_i).\log_2\left(p(z_i)\right)$	T, F, C 8 to 10 (**F8**)
Rényi 'entropy'[b] [HMD11]	$\frac{1}{1-\alpha}\cdot\log_2\left(\sum_i p(z_i)^\alpha\right)$	T, F, C11 to 12 (**F11, F12, C12**
Rate of attack [TB05]	$\max_i\left(\frac{z_i-z_{i-1}}{n}\right)$	T13, F13, C13
Rate of decay [TB05]	$\min_i\left(\frac{z_i-z_{i+1}}{n}\right)$	T14, F14, C14
Threshold crossing rate[c] [HYYC09, Fag07]	$\frac{\#(\text{Threshold Crossing})}{n}$	T, F, C 15 to 18 (**T15, T16**)
Silence ratio[c] [MZB06]	$\frac{\#(\text{z where z<threshold})}{n}$	T, F, C 19 to 22 (**F22, F21, F20**)
Mean	$\mu_z = \frac{\sum_i z_i}{n}$	T23, **F23**, C23
Max over mean	$\frac{\max_i z_i}{\mu_z}$	T24, F24, C24
Min over mean	$\frac{\min_i z_i}{\mu_z}$	T25, **F25**, C25
Energy measurements[d] [Foo97]	Energy standard deviation, energy skewness, energy kurtosis	T, F, C 26 to 28 **F28, T26, F17, F26**

[a] Bin numbers for probability estimation: 5, 30, 500.

[b] Bin numbers for probability estimation: 30, $\alpha = 2$ and ∞.

[c] Signal maximum normalized to 1 and different threshold values: 0.2, 0.4, 0.6 and 0.8

[d] Energy measurements are features computed from the signal energy $E(t) = x(t)^2$ rather than on the signal itself.

4.2 Proposed Approach

The architecture used for this analysis is the detailed in Chapter 3.1.2, for the automatic detection and classification of events. The anomaly detection module is also used, thresholding the output probabilities to classify each observation between: one of the positive classes (fish sounds, presented in Section 4.3.1), the background noise (i.e., detection task), and anomalies (i.e. observations not belonging to one of the classes used in training). For this application, an observation is defined as a 0.5s length piece of recording, filtered in a bandwidth. This definition is chosen in order to fulfill the one class per observation clause, detailed in Chapter 3.1. Illustration of observations are given in Section 4.3.1.

The full list of the features used to represent the observations is given in Table 3. This feature list is based on Table 2, selecting features not dependent on the time or frequency. Typically, features such as the frequency centroid are disregarded for this application.

4.3 Presentation of the Dataset

4.3.1 Seacoustic Recording Campaign

The recordings used in this study for the experimental analysis were collected in August 2014 in France during the SEACOUSTIC2014 campaign [LGI15]. The project aimed to collect data to address three issues: (i) How to determine the vitality of underwater areas; (ii) How to evaluate the anthropogenic stress on underwater areas; and (iii) To study the link between vitality of a given underwater area and the anthropogenic stress it faces. The campaign was based at the STARSEO station, near La Pointe de la Revelatta in Corsica, France (Mediterranean Sea). For more details on the SEACOUSTIC2014 project, please refer to [LGI15].

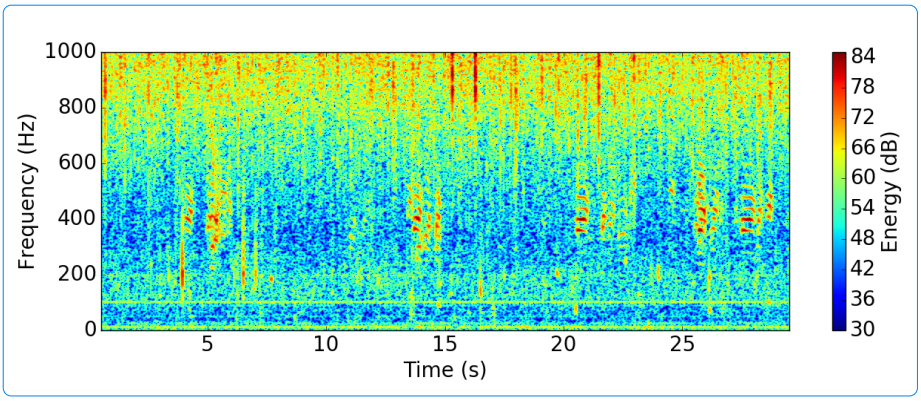
The study presented here is related to issue (i); namely, the development of tools to determine the vitality of underwater areas. The data used to test and evaluate our system are area specific, which means that self-sufficient recording devices were fastened to the sea floor and left to record continuous signals (i.e., the recordings). Specifically, between 1 day and 3.5 days of continuous recordings were collected from various marine areas. Each marine area is characterized by its depth and its sea floor (i.e., meadow, rock, sand, as given in Table 4) and the development of automatic models to classify fish sounds can help with the analysis and characterization of these areas. Hydrophones (HTI92 WB) and recorders

Table 4
Description of the area-specific recordings used in the present study.

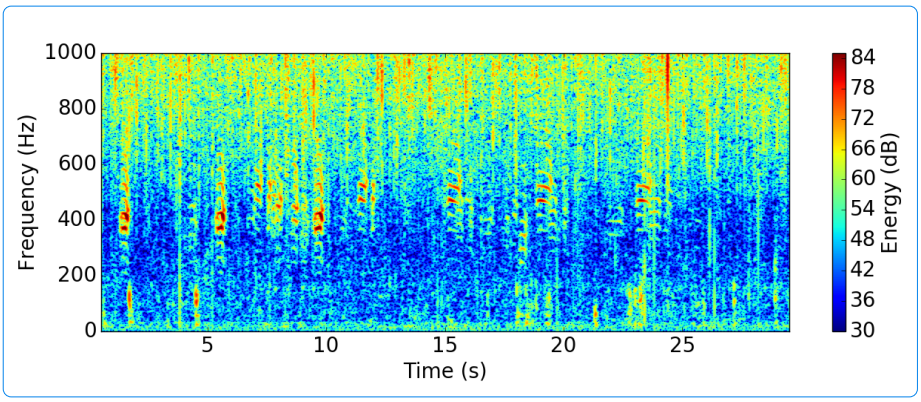
Ref.	Area	Depth	Duration
1	Healthy sea-grass meadow	-20m	3.5 days
2	Healthy sea-grass meadow	-12m	1 day
3	Lower sea-grass meadow / sand border	-38m	1 day
4	Damaged meadow + Rocks	-12m	1 day
5	Rock	-12m	1 day

Figure 18

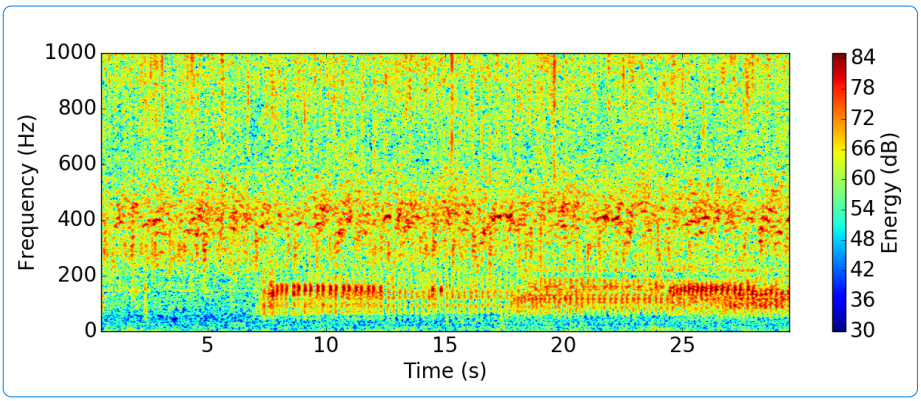
Spectrogram of recordings of the five different studies areas (presented in Table 4). Spectrogram are computed using Kaiser window of width 1024.



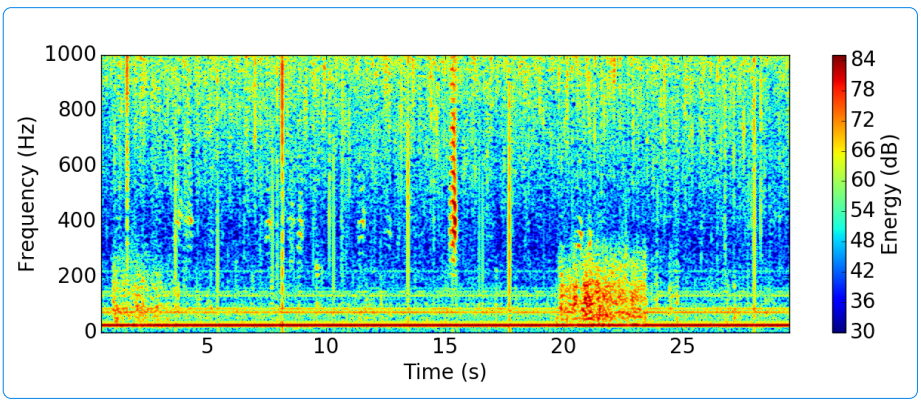
(a) Area#1



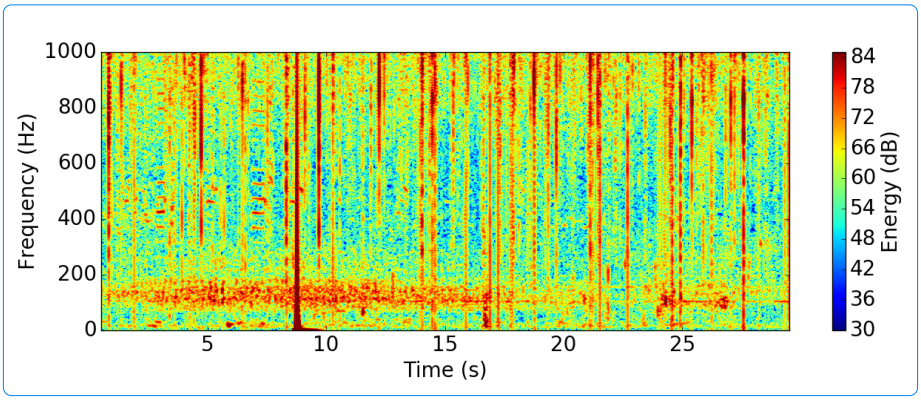
(b) Area#2



(c) Area#3



(d) Area#4



(e) Area#5

(SDA14; RTSYS) were used, which provided the signals coded on 24 bits at 256 kHz (anti-aliasing filtering was applied). Spectrograms of recordings of the five areas are presented in Figure 18. We here stress that the recordings used in the present study were not labeled, and no ground truth was available regarding their content. The labeling task was carried out specifically for the present study, and was conducted manually, by reviewing and labeling the content of part of the recordings. More details are given in Section 4.3.2.

The underwater recordings collected sounds from three different sources: animal (biophony), environmental (geophony), and human (anthropophony). Together, they form the soundscape of a given area. In this study, we focus on biophony and more specifically, on fish sounds. To this day, a direct relationship between fish sounds and individual fish or fish species has not been established. However, it is known that specific fish sounds can be associated with specific behaviors, if the fish species is known – as determined in the fish biology literature [Amo06, ASH04, DSMM+00, TF02, PVFF06, MHJ08]. An established terminology that refers to the different fish sounds is still lacking, as well as any universally accepted correspondence between fish sounds and fish behaviors [Amo06]. For this reason, we chose to name (arbitrarily) the fish sounds in the acquisitions based on their qualitative characteristics. For the various recordings, four different types of fish sounds (i.e., classes) that show distinctive acoustic signatures are recognized.Spectrograms of representative signatures along with descriptions of the four classes are shown in Figure 19, and are hereafter detailed as:

- `Impulsions` — Emissions of short duration that are separated by lags of the order of seconds.
- `Drums` — Periodic pulse trains (around at least 15 pulses) that last for at least 20 s in most of the recordings.
- `Roars` — Wideband signals in their frequency content, which are usually very energetic. These also last between about 10 s and 30 s, and occur at a frequency of around two a minute.

- `Quacks` — Short signals with a harmonic structure that are often present (up to five occurrences per second). `Quack` sounds are recognizable by their similarity to frog or duck sounds.

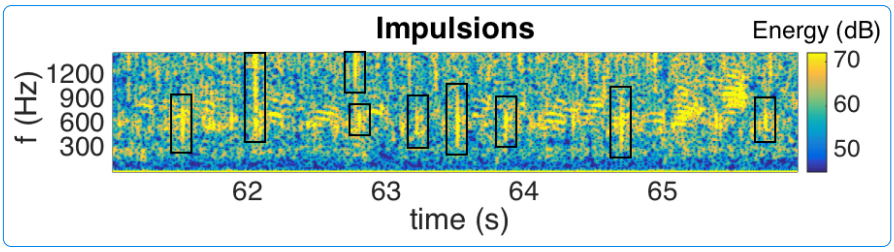
These four classes were manually identified.They are clearly distinguishable by a trained ear, and they also match the dedicated literature. Fish sounds that do not fit into this nomenclature are particularly rare in these recordings. The proposed classes are related to fish behavior, and to identify specific species, or even specific individual, the intra-class properties can be studied. Typically, it would be possible to define sub-classes: `Drums`, for example, could be sorted depending on the pulse frequency, and `Impulsions` depending on their frequency, which can vary significantly. Such sub-classes would more likely to be related to species or identification of individuals [Amo06, MHJ08].

Sounds different from those belonging to the four above mentioned classes can also be spotted in the recordings. They are mainly due to ambient or anthropic noise. Such sounds are referred to as `Unknown` when a definite structure cannot be identified by the experts, or as `Background` when background noise dominates. The four fish sound classes are referred as the *positive* classes, while `Background` and `Unknown` represent the *negative* classes. More details on how those two classes were handled will be given with the results in Section 4.4 and 4.5.

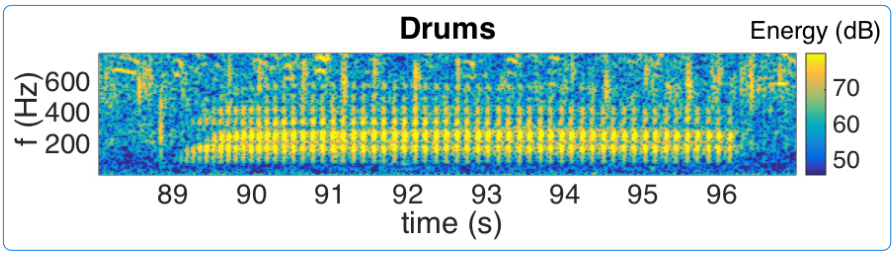
According to the literature, the sounds named here as `Impulsions` and `Drums` appear to be related to antagonistic behaviors of fish [DSMM+00, TF02, ASH04, PVFF06], while `Roars` and `Quacks` would be produced during courtship [DSMM+00, TF02]. `Impulsions` might also be linked to feeding activities according to [ASH04]. `Drums` and `Roars` could also be related to courtship behavior [MHJ08]. Additionally, it is worth noting that the fish sounds listed in the biology literature are generic: the four classes identified here are not specific to the data used in this study [Amo06]. As a consequence, the architecture we propose to automatically classify fish sounds can be used for recordings other than those used in this study.

4.3.2 From Recordings to a Labeled Dataset of Observations

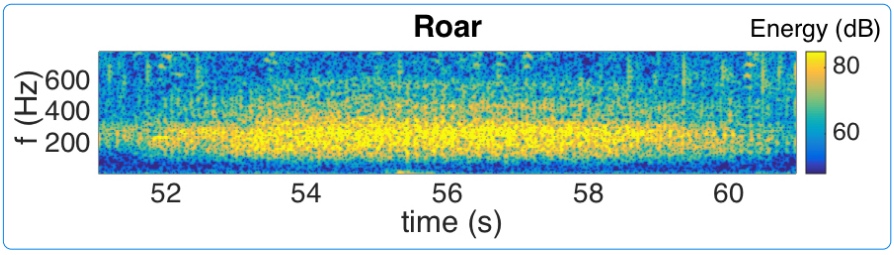
A dataset of observations is needed to train and test any classification model, with the dataset construction here. A labeled dataset is a database of signals in which each observation has been assigned to its corresponding semantic class detailed here. For our application, this meant considering a large number of fish sounds of each of the four considered classes. To distinguish fish sounds from uninteresting sounds, we also considered a fifth class of background noise. Once built, the dataset was used to train and test the classification model: the model learns to distinguish and recognize the various classes from the observations of the database. This implies that, ideally, the dataset should contain all of the variability of the phenomenon under study. As this is physically impossible, the idea was to gather as many observations as possible, to characterize a given class as completely as possible. The dataset used to build the model is directly



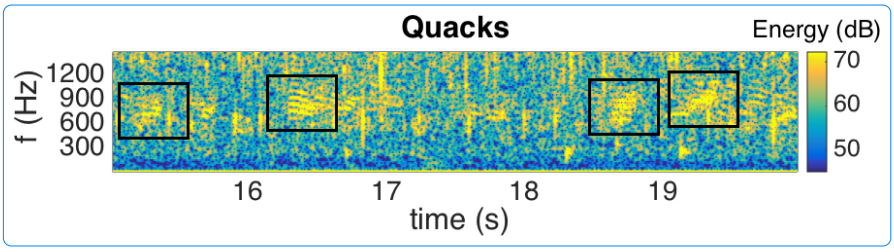
(a) Impulsions. Length: 2 ms to 20 ms. Presence: $<$ 5/s. Spectrogram with Gaussian windows on 8192 points.



(b) Drums. Length: 5 s to 20 s. Presence $<$ 5/min. Spectrogram with Gaussian windows on 16384 points.



(c) Roars. Length: 10 s to 25 s. Presence: $<$ 2/min. Spectrogram with Gaussian windows on 16384 points.



(d) Quacks. Length: 150 ms to 300 ms. Presence: $<$ 5/s. Spectrogram with Gaussian windows on 8192 points.

Figure 19

Description of the four positive classes (i.e., Impulsions, Drums, Roars, and Quacks), and their corresponding spectrograms.

linked to the model capabilities. Without data covering a wide spectrum of observations, it is very difficult for the model to analyze newly recorded data. It is therefore necessary to consider a *large* dataset that covers the range of the phenomena under study. Further explanations on supervised machine learning are given in Chapter 2.

For the present study, 913 observations were manually identified from the underwater recordings by an expert: 91 Impulsions, 114 Drums, 36 Roars, 205 Quacks, and 467 Background. We hereafter detail the process. All of these observations were extracted from continuous labeling of 10 min at the Area #3 (sand / sea-grass interface, Table 4). This particular area was selected because it appeared to host the most varied recordings. The labeled period was recorded on August 5, 2014, at 10 pm, and was selected as a particularly rich recording (i.e., gathering many fish sounds). The recordings were continuously labeled using a sliding window of fixed length $\Delta_t = 0.5s$ and two bandwidths. We chose to focus our analysis on the frequency ranges of 50 Hz to 450 Hz, and 400 Hz to 900 Hz, as most of the fish sounds in the recording were in these frequency bands. The original recordings were previously down-sampled from $f_s = 256kHz$ to $f_s = 10400Hz$. Each observation therefore had a fixed length of $\Delta_t = 0.5s$, and belonged to one of the two frequency ranges that were analyzed. The use of a sliding window of fixed size led to some calls being considered as various observations; e.g., `Drums` and `Roars` are long calls (10-30 s), and were therefore separated into several consecutive observations. The $0.5s$ window length was empirically determined as the minimum duration needed to distinguish the five classes. This was longer than a single `Impulsion` or a single `Quack`, and shorter than either a full `Drum` or a full `Roar` call. However, a minimum of $0.5s$ is needed to identify a `Drum` or a `Roar` as such. Alternatively, and depending on the data, the continuous analysis proposed in this study can be carried out on windows of different sizes and in other frequency bands. Any observation where the class was not clearly identified by the experts belonged to the `Unknown` class, and were disregarded for the learning stage (but not for the testing; see Section 4.5.1). Alternatively, observations that contained no fish sounds and no unidentified sound were labeled as `Background`. The labeling step is illustrated in Figure 20: each observation of the dataset is a signal of length 0.5 s and filtered in its bandwidth. They can be visualized as an extract of the spectrogram.

4.4 Validation of the Proposed Approach

4.4.1 Model Validation and Performance Evaluation

In this first part of the study, the proposed architecture is tested using cross-validation. A total of 913 observations belonging to five classes are considered: the four positive classes associated to fish sounds (`Impulsions`, `Drums`, `Roars` and `Quacks`), and a generic `Background` class. Cross-validation with $\alpha = 0.7$ (50 trials with 70% of the dataset used in training and 30% in testing) is used to determine the best values for the hyper-parameters of the learning algorithms, and to validate the model perfor-

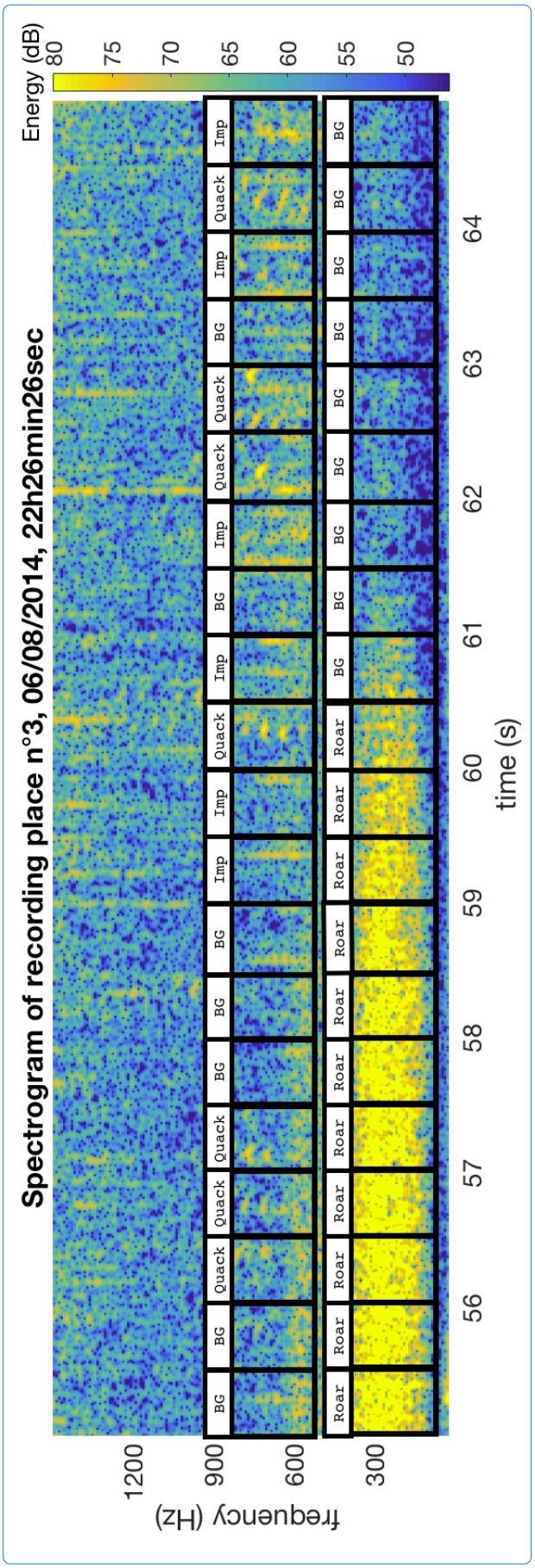


Figure 20

Illustration of the annotation process. All of the black boxes have the same width, corresponding to $0.5s$*, and the same height corresponding to the frequency range* ($f_{max} - f_{min} = 400Hz$ *in this study). An observation is a* $0.5s$ *portion of the recording, filtered between its* f_{min} *and* f_{max}*. The spectrogram was generated using a sliding Gaussian window of 16384 points.*

Table 5 *General Results for the Automatic Classification of Fish Sounds. Accuracy results are compared depending on (i) the feature set used (`time`, `frequency`, `cepstral`, `all` or MFCC) and (ii) the learning algorithm (RF or SVM). Subsets of the most important features are also considered: MVF and VF. Learning rate $\alpha = 0.7$.*

Feature set	Dimension d	Accuracy	
		RF	SVM
`All`	84	$96.9 \pm 2.0\%$	$96.5 \pm 1.6\%$
`Time`	28	$90.1 \pm 2.0\%$	$91.2 \pm 1.8\%$
`Frequency`	28	$91.1 \pm 2.7\%$	$90.7 \pm 3.1\%$
`Cepstral`	28	$91.4 \pm 3.0\%$	$90.8 \pm 2.7\%$
`MVF`[a]	3	$91.5 \pm 0.85\%$	$91.3 \pm 0.82\%$
`VF`[b]	19	$95.6 \pm 0.79\%$	$94.7 \pm 0.82\%$
`MFCC`	26	$72.5 \pm 3.3\%$	$70.0 \pm 6.0\%$

[a] with `MVF` for Most Valuable Features.

[b] with `VF` for Valuable Features

mances. When using SVM, the best accuracy of the data is obtained with a Gaussian radial basis function kernel of parameter $\gamma = 2^{-7}$, and with a cost parameter $C_{SVM} = 512$. These values are obtained after a grid search for the optimum values, and then they are kept constant. The accuracy of the results varies smoothly through the grid. For RF, the number of trees is fixed to 200, as a compromise between performance and computation time. The entropy is used as impurity measurement since it leads to better results than Gini index, $\sqrt{d}$ features are considered at each node with d the total number of features, and the trees are not pruned. The overall accuracy reaches $95.3 \pm 0.76\%$ when using all of the features and RF as classifier, and $95.0 \pm 0.88\%$ when using SVM. These results validate both the architecture and the features used. Two main conclusions are drawn from those numbers: first, the overall proposed process to automatically classify the fish sounds is validated. Second the learning algorithm has, as was expected, a limited influence on the results.

It is also relevant to discuss the limitations of the propose approach in order to better evaluate the proposed results. The current main limitation of the method is the use of a fixed window for the analysis: some observations may contain more than one class. Typically, `Quacks` and `Impulsions` can sometimes be found within the same window. The model then recognizes properties of both classes and output probabilities are split between the main classes. However, both are often lower than the threshold that was fixed, leading to their rejection in terms of classification. To overcome this limitation, the study could be carried out on temporal windows of various length: smaller windows would be less likely to contain more than one class. The use of a sliding window also prevents temporal coherence in the model; e.g., for call counting operations, a temporal regularization would help to identify complete calls from several detected observations. This is particularly relevant for the long calls, such as `Roars` and `Drums`, which are detected as a succession of windows including the same class.

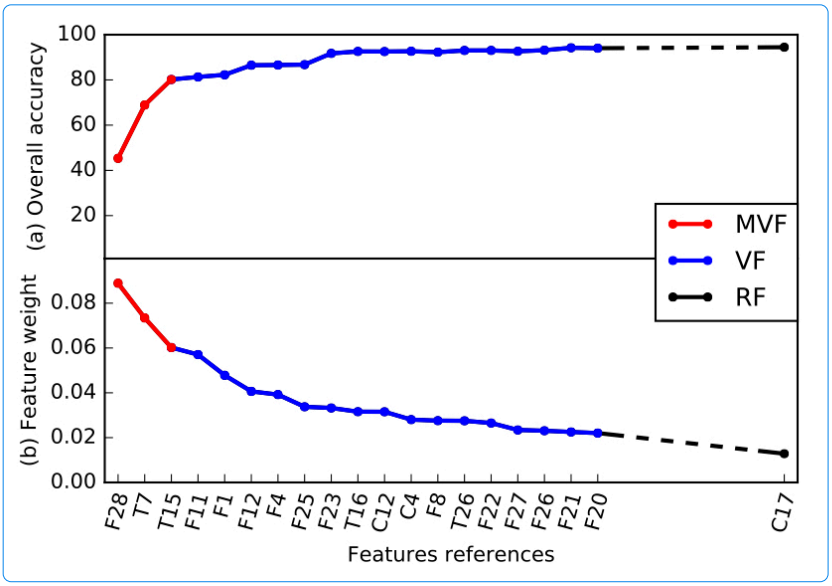


Figure 21

Evolution of (a) overall accuracy when using feature vectors of increasing dimension d. Features individual weights are shown in (b). Features are references as indicated in Table 3. In red, the Most Valuable Features (MVF), in blue the Valuable Features (VF) and in black the regular features (RF). The valuable features are highlighted in bold font in Table 3.

4.4.2 Features Selection

In this second experiment, we study the influence of the different features. In particular, we show that the features have a greater impact on the results than the choice of the learning algorithm. The accuracy when comparing the influence of the feature sets (`Time`, `Frequency`, `Cepstral` and `All`) and the learning algorithms (SVM, RF) are given in Table 5. The influence of the feature sets is of particular interest here. When comparing the accuracy of the results with `Time`: $90.1 \pm 2.0\%$; `Frequency`: $90.1 \pm 2.7\%$; and `Cepstral` features: $91.4 \pm 3.0\%$, it is interesting to note that the feature set influence is not so important in this case. Those accuracy values correspond to RF but SVM leads to the same conclusion: each domain contains enough discriminative content, but combining the three leads to better performances. This phenomenon would suggest that the discriminative information needed for the classification is spread in the various domains. Once again, numbers show that results are steady regarding the learning algorithm that is used.

As the state of the art in automatic classification of fish sounds is relatively limited, we compare the proposed method in terms of features to the use of `MFCCs` as descriptors. As explained in Chapter 2, `MFCCs` were originally designed for speech processing purposes [ZZS01, GFK05], but have since been used in many applications related to automatic classification of transient signals. Comparing `MFCCs` to the `All` features, we obtain accuracies when using `MFCCs` of $72.5 \pm 3.3\%$ for RF and $70.0 \pm 6.0\%$ for SVM, while the `All` features reach 95.3% for RF and 95.0% for SVM. The feature set we propose for this application thus leads to more accurate results, and is actually significantly better adjusted. Indeed, `MFCCs` were originally developed to represent speech data, which are particularly different from the data used in the present study, in terms of their frequency range, shape, and other details. `MFCCs` typically match the Mel scale and the human perception of sounds. We thereby stress that the features proposed here are generic, and can also be used to represent more general

Table 6

Class by class accuracy for feature vectors made of the 1st to 5th most important features, according to Random Forest feature weights. Features are designated by their references, as specified in Table 3.

	Accuracy (%)				
Background	72.1	76.0	77.2	78.6	84.2
Impulse	63.5	63.8	67.3	68.5	72.0
Drums	62.8	91.1	91.7	91.8	93.8
Quacks	59.3	57.8	58.2	61.3	65.8
Roars	67.9	92.9	94.3	93.5	95.0
New feature ref.	F28	T7	T15	F11	F1

transient signals. The same conclusion can be drawn when comparing the `MFCCs` with the `Cepstral` features, where the accuracy reaches $91.4 \pm 3.0\%$ for RF and $90.8 \pm 2.7\%$ for SVM. The comparison between these two feature sets stands as they both describe the Cepstral domain. However, the feature set we propose here performs better than the `MFCCs`, once again stressing the importance of the feature choice and validating the proposed features. One reason for this might be that the `MFCC` is an ordinate representation (low level representation), while the `Cepstral` feature set is not (high level representation). All of the data related to the use of `MFCCs` features are presented in Table 5, and all 26 `MFCC` coefficients were considered in this study.

After having estimated the influence of the features, we propose to investigate the feature selection issue, and use RF features weight to select the most important features (technical explanations in Chapter 2). More specifically, Figure 21 (b) shows the individual importance of the features and Figure 21 (a) displays the mean evolution accuracy when the dimension d of the feature vector increases: from the most important feature, to the two most important, and so on. Analysis of the features weights leads us to build two subsets of features of decreasing importance: the `most valuable features (MVF)` (Figure 21, red dots) and the `valuable features (VF)` (Figure 21, blue dots). The `MVF` and `VF` are selected as the minimum features subsets leading to stable results: using more features does not significantly increase the performance of the classification system. The accuracy when using `MVF` and `VF` are also reported in Table 5. The `MVF` contains only three features: the energy kurtosis computed from the spectrum (`F28`), the mean kurtosis computed from the waveform (`T7`), and the threshold crossing rate also computed from the waveform (`T15`). Considering the mean accuracy over the five classes when using the `MVF`, it reaches 91.5% and 91.3% for RF and SVM, respectively. This result is essential for real-time applications and embedded systems with limited computational costs and storage capabilities. If we consider the 19 first features with `VF` (including features from `MVF`), the global accuracy reaches 95.6% and 94.7% for RF and SVM, respectively. Similar numbers are obtained when using the feature set `All`. This result shows that all of the features are actually not needed to obtain good classification results, and it also reflects on correlations between some of the features. Furthermore, it is worth noting that features of `VF` contain descriptors computed on the signals represented in the time, frequency, and cepstral domains, encour-

aging to consider several representations of each observation.

The impact of the most important features on the class by class accuracy is also particularly relevant and is reported in Table 6. In particular, it reveals that valuable features are not equally important depending on the considered class. The second most important feature for example (T7, mean kurtosis computed from the time domain) has a great impact on Background, Drums and Roars, has no effect on Impulse, but is detrimental to Quacks since their accuracy drops from 59.3% to 57.8% when using a second feature to represent the observations.

4.5 A Posteriori Analysis of the Dataset

4.5.1 Model Validation on Continuous Data from Different Areas

This section reports on the use of our proposed method to automatically analyze continuous underwater recordings. The overall goal of this study is not a real time analysis for continuous monitoring, but an a posteriori study, used to gather knowledge on a considered environment after a data gathering campaign.

We train a model on the dataset presented in Section 4.3.2 that contains 91 Impulsions, 114 Drums, 36 Roars, 205 Quacks and 467 Background observations. These observations are extracted from a continuous labeling of 10 min of recording on August 5, 2014, at 10 pm. The model is trained with SVM and the All features, on five classes (four positive fish sound classes, and the background) . The model is then used in an applicative context to analyze continuous recordings in two different configurations. The first one tests the model performance on continuous recordings: the test signals were recorded on August 5, 2014, between 10:27 pm and 10:37 pm, that is half an hour after the acquisitions used for training the model. The two data sets (i.e., the one used for training and the one for test) were recorded on the same area. The second test configuration considers a set of recordings that was randomly selected among the other recording areas. These were registered in Area #2 given in Table 4

		True Class (ground truth)						Precision
		B	D	I	Q	R	U	
Predicted Class	B	**969**	2	2	-	2	23	97.1%
	D	-	**133**	1	-	-	2	97.8%
	I	-	-	**208**	-	-	-	100%
	Q	5	-	-	**245**	-	2	97.2%
	R	-	-	-	-	**58**	1	98.3%
	U	39	16	11	40	13	**624**	84.0%
Accuracy		95.7%	88.1%	93.7%	86.0%	79.5%	95.7%	

Table 7
The five considered classes are Background (B), Roars (R), Drums (D), Quacks (Q) and Impulses (I). A sixth class for rejected observations is considered and referred as Unknown (U). The average accuracy reaches: 93.4%. *Testing observations recorded at a different time than learning observations.*

(sea grass at 12 m in depth, compared to sea grass/sand interface at a depth of 38 m, for the learning observations) on August 8, 2014, at 11 pm. In both configurations, the recordings are continuously processed in the two bandwidths on which this study is focused (i.e., 50-450 Hz and 500-900 Hz), with a sliding window of duration 0.5 s. To reduce the computational burden in both the training and validation processes, no overlap is considered between consecutive windows. The threshold value is empirically fixed to $0.8 \in [0, 1]$. Classification results are presented in Tables 7 and 8. In this configuration, class by class accuracy and precision results are presented, in order to better explicit the false alarm rates.

The first configuration was decided to test the model on continuous signals that were recorded at a different time than the learning observations, in order to avoid similarities between the signals. The confusion matrix of this test is presented in Table 7 and several conclusions can be drawn. First, very few errors are noticed. Regarding `Drums`, 133 observations are correctly detected, and two mislabeled in the `Background` class. Similarly, 208 `Impulsions` are correctly detected, two observations are confused for `Background` and one for `Drums`. Comparable results are obtained for `Quacks` and `Roars`, with 245 and 58 correct detections for no and two errors, respectively. The associated accuracy is therefore extremely high going up to 95.7%. The study of the `Unknown` observations (anomaly detection) is also particularly relevant since we observe 743 observations for which the model does not reach any decisions. Among those, 624 are also not identified by experts, either because the observation is too noisy to be identified, or because it contains more than one class, or because the sound does not match any of the four positive classes. Between 11 (`Impulsions`) and 39 (`Background`) observations per class should have been attributed to a class, suggesting that the probability threshold could be decreased, and adjusted to each class. Precision results also lead to particularly valuable information, since they are systematically above 97% for the four fish sound classes: the false alarm rate is extremely low, which clearly validates the use of the model in real conditions. Results from the experiment are particularly conclusive since the overall high performance and the precision prove the interest of such tools for analyzing continuous data.

The second set of observations are considered to analyze the model generalization capacities since the model is tested on recordings from a different underwater area. Results of this experiment are presented in the confusion matrix in Table 8 and very similar conclusions can be drawn from the detected observations. Almost all detected observations are correctly assigned to their classes. Depending on the class, a maximum of 25 errors out of 885 correct detections are found for the `Background` class, and regarding positive classes, 6 errors for 46 detections and 4 errors for 25 detections are made respectively for `Impulsions` and `Drums`. It is also interesting that no `Roars` are detected, which is confirmed by the analysis done by experts. Very good detection results are therefore achieved, even when the model is tested in a different area compared to the learning observations. It is relevant to notice that in this case, the number of rejected

observations (Unknown class) is greater: 1024 compared to 743 in the previous configuration. Out of those observations, 651 are also rejected by the experts, but the others are missed by the model. Generally speaking, accuracy and precision rates are lower than when the training and testing recording places are similar, but still relatively high. A conclusion on those results is the need to lower the threshold in this configuration: when the learning and testing areas are different, test observations are less likely to be similar to the ones used in the training, and the probabilistic outputs of the model are therefore lower than in the previous configuration. Those results recommend the use of such a method for the continuous and real-time analysis of underwater recordings. In particular, large datasets can be automatically analyzed and conclusions can be drawn regarding the fish populations, their evolution in time, and their movement from one area to another. As for the computation times, each set of recordings (i.e., duration of ten minutes) was analyzed in about 4 min on a laptop computer, thereby validating the use of this method for real-time applications. As a reminder, `all` features were used for this analysis, and thus the computation times could be decreased if only selected features are used.

We now discuss the need to perform anomaly detection when analyzing continuous recordings, and the difficulty to set the thresholds for each class. Without thresholding the model output probabilities (i.e., no anomaly detection), all of the windows are classified between the four positive classes and `Background`. However, according to the interpretation done by experts, some of the windows are `Unknown': if the fish are far away and the effects of the propagation are non-negligible, if different classes occur in the same window (sometimes up to three), or if the sound does not fit in any of the classes (geophony, anthropophony), it is not possible for the experts to classify these observations. It is therefore necessary to threshold the output prediction probabilities to reject such observations. The thresholding operation, however, raises the issue of the threshold choice: if it is too high, only well-defined observations will be detected, and many will be missed; if it is too low, many observations that are `Unknown` for the experts will be forced into a class. Ideally, a different threshold should be decided upon for each class. A promising

		True Class (ground truth)						Precision
		B	D	I	Q	R	U	
Predicted Class	B	**885**	6	4	-	-	21	96.6%
	D	3	**40**	-	-	-	8	78.4%
	I	16	-	**21**	-	-	-	56.8%
	Q	6	-	-	**256**	-	-	97.7%
	R	-	-	-	-	-	-	-
	U	207	24	15	127	-	**651**	63.6%
Accuracy		79.2%	57.1%	52.5%	66.8%	-	95.7%	

Table 8

The five considered classes are Background (B), Roars (R), Drums (D), Quacks (Q) and Impulses (I). A sixth class for rejected observations is considered and referred as Unknown (U). The average accuracy reaches: 80.9%. *Testing observations recorded at a different time and place than learning observations.*

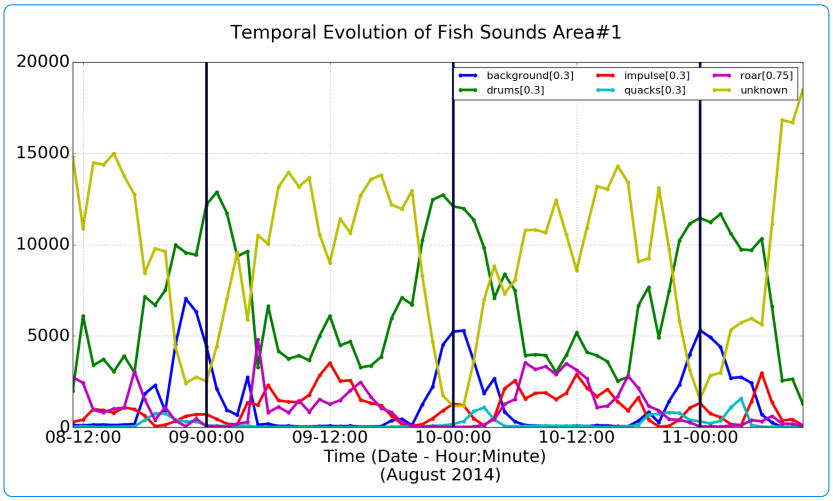


Figure 22

Hourly evolution of the number of observations for each class. Recording area#1. For each class, the threshold are indicated in the legend and are fixed to 0.3 for all the classes except for roars, where it is fixed to 0.75. Day to day patterns are revealed.

development of the existing model would be to perform an analysis of the `Unknown' observations in order to detect new classes of sounds. For example, classes related to anthropophony or geophony can be considered, and we can in particular think of boat, rain, or thunder sounds.

4.5.2 Analysis of Various Recording Areas

The previous experiments have validated, analyzed and tested the model in various conditions, including testing in a different recording area. The associated results where quantified, using a ground truth that was necessary to validate the approach. In this section, we consider the previous model, and use it to analyze the entire set of recording described in Table 4.

We first focus on the Area#1 (Table 4), since it presents the longest monitoring period, with 3.5 days of continuous recording. Figure 22 is obtained by counting hourly the number of observations for each class. The evolution over the recording period is displayed class by class. Several conclusions can be drawn from the analysis. A day-to-day pattern is clearly visible, for all present classes. In particular, `Drums` and `Quacks` peak during the night and remain to lower level during day time. `Impulse` display a similar increase during the night time, but with higher level during the day. A day to day pattern is also visible on the two negative classes: `background` and `unknown`. `background` appears more during the night, while `unknown` observations are predominant during the day. This would suggest that night time are either composed of fish noises, or of silence, while day time would display sounds other than the five classes used in learning. Typically, boats sounds are frequent during the day and have similar signature to `Roars` in the considered bandwidths. The question of the threshold to use for this analysis is crucial, and is currently a major limitation of the tool. For this experiment, all classes had a threshold of 0.3 except Roars. Increasing the threshold would simply increase the number of `Unknown` observation, but the curves of the `Background`, `Impulse`, `Drums` and `Quacks` would remain very similar with a lower mean value. For `Roar`, the threshold was set high, in order to discriminate them from boats. The relatively low number of `quacks` is interesting, and was ex-

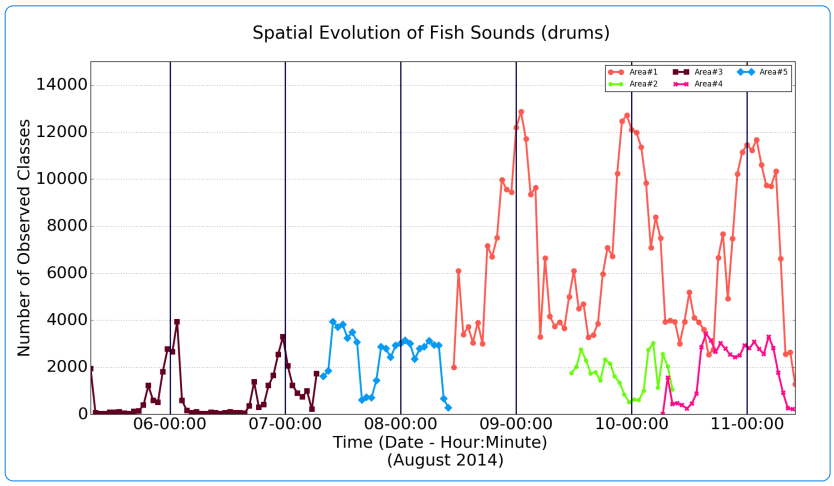


Figure 23

Hourly evolution of the number of observations for each class. Recording area#1. Thresholds are fixed to 0.3 for all the classes except for roars, which it fixed to 0.95. Day to day patterns are revealed.

pected higher. One possible explanation is that as seen in the previous experiment, `Quacks` are relatively sensitive to the change of area. Typically, this area is analyzed with a model trained on a different area, and `Quacks` are probably missing. A study with a model trained on `Quacks` of both area could be interesting and relevant. The graph associated to the remaining areas are displayed in Appendix C.

The proposed tool can be used to study the temporal evolution within one area, but also to study the spatial evolution of the acoustic landscape. Figure 23 displays the evolution of the number of `Drums` observations across the five recording areas. The curves mean levels are relatively different, which can result either of the threshold settings, either of the difference between the habitat. Concerning the repartition of `drum` observations, areas #1 and #3 have very similar conclusions. Both being close to sea-grass meadow, the analysis is consistent. It is interesting to see that conclusions regarding Drum observation are consistent from one area to the other. Similarly, curves associated to the areas #4 and #5 have similar repartitions. Both are areas with rocks. The graph associated to the other classes are displayed in Appendix C.

Highlights & Summary

- The study of the acoustic landscape allows to gather information about an environment by studying the associated sounds (passive acoustic monitoring). To monitor underwater areas, we consider the use of machine learning to automatically analyze recordings made underwater, in coastal areas. We here focus on a specific component of the acoustic landscape: biophony, and in particular fish sounds.
- Four positive classes corresponding to four different behavior of fish are considered, along with two negative classes: the background noise, and anomalies. Around a thousand of observations are considered are were manually identified. This dataset is unique.
- Automatic detection and classification is applied using the first architecture presented in Chapter 3. Anomalies are also detected. The analysis is carried out using 0.5 time windows, and in two different bandwidths. Four different fish sounds are considered, among with the background noise.
- Results validate the use of such method to analyze large dataset. Cross-validation results reach $96.9 \pm 2.0\%$, and results in real condition reach 93.4% and 80.9% for a classification tested on the same area than learning but at a different time, and on different areas, respectively.
- The influence of the learning algorithm is limited, but the features have a greater influence on the accuracy levels. The success of the approach lies in the proposed feature set, where observations are described by shape descriptors in three different low level representations (time, spectral and cepstral).
- Analysis carried out on five different recording areas reveals day to day pattern in the different fish sounds classes.
- One of the main limitations of this method is the need to set the detection threshold. This limitation also underline the importance to work with the field experts, to compare manual and automatic results, to adjust the tools. The proposed tools offer an analysis, but the experts knowledge is needed to interpret such results.

Chapter 5

Application: Monitoring of Volcanoes Based on the Analysis of Seismic Recordings

Contents

Figure 24

Mount Merapi, Indonesia. Photos by Marielle Malfante.

(a) Mount Merapi at sunset, view from Pos Pengamatan Gunung Merapi Jrakah

(b) Merapi crater

(c) Recording and transmission station below the dome of Mount Merapi

Introduction

This second applicative chapter focuses on monitoring volcanoes by analyzing continuous seismic recordings. In particular, the chapter aims at reviewing the current methods used for the monitoring of volcanoes and see how automatic analysis methods can fit in existing monitoring frameworks.

The Volcanic Threat

Generally speaking, volcanoes are a well known and intriguing phenomenon, despite the continuous threat they represent. Several cultures and life habits illustrate this dual facet: captivation, attraction and even deference or deification of volcanoes are opposed to the fear and fatalism toward eruptive episodes. Volcanic grounds are particularly fertile for farming which historically, lead to the establishment of villages and towns in the vicinity of volcanoes, but also to major disasters for the populations. We can quote the most commonly known, for instance the Vesuve eruption in 79 (Italy), or the Santonin in -1646 (Greece) which incidentally lead to the Atlantis myth. The deadliest eruption would be the Tambora in 1815 with more that 71,000 victims (Indonesia) [Opp03]. Worldwide, 29 millions of people life in the vicinity ($<$ 10km) of one of the 1508 active volcanoes, therefore being directly exposed to the thread of an eruption [BJS+17]. The need for an effective monitoring is therefore not to be demonstrated, and the development of efficient methodologies along with innovative and operational tools to mitigate risks related to volcanic unrest (prevention, crisis management and recovery) are of key importance [UI15]. This challenging task has been a concern for the scientific community for many years.

Volcanoes Monitoring

Historically, volcanic monitoring was mainly visual, with observations and evolution study of gas clouds exhausting the crater. During the latest decades, the monitoring of volcanoes has developed itself, and now includes the study of many parameters including visual indicators (camera, thermal imaging), the evolution of chemical composition (gas, magma) or geophysical measurements (ground deformation, inclinometry, or seismic waves for instance) [McN96]. In particular, seismic waves are used to investigate traces of tectonic events and events precursors to an eruption. Beneath the volcano, movements of magma or gas regularly occur and eventually lead to magma rising up the magmatic conduit. Such movements are the source of seismic waves (i.e. earthquake) that can be recorded on the earth surface, on the volcanoes slopes or vicinity. By recording and analyzing the seismic recordings, experts are able to propose a physical interpretation for the volcano activity, and to better understand eruptive

events. More specifically, seismic recordings contain recurrent patterns, which are related to a physical behavior of the volcano. The monitoring task of the seismic recordings is therefore a detection and classification task of the various patterns. The overall goal is double: predicting and prevention eruptions, but also understanding the phenomenon involved in active volcanic episodes. This problem is therefore a direct application of the automatic analysis methods proposed in Chapter 3.

The Interest in Using Machine Learning Methods

A mean of 540 persons per year are killed by volcanoes [BJS+17], and most of the 1508 worldwide active volcanoes are not monitored. The monitoring of a volcano is indeed a heavy task, requesting sensors, communication with the observatories, maintenance and of course, data analysis. Typically, the monitoring of seismic recordings is mostly a manual task, and is an obvious limitation. Automatic tools to analyzing continuous data, and help the experts in the decision making process concerning safety recommendation could be used. The easier the monitoring task gets, the more volcanoes can be thoroughly monitored. It is however very important to underline that all the automatic processing tools proposed for this purpose aim at helping the experts, and in no case to replace their knowledge and expertise. The human brain has an incredible faculty to learn and recognize patterns and anomalies with a great reliability. Artificial Intelligence tools cannot and will not replace the human expertise, but they can ease the monitoring task. In particular, the use of supervised learning tools is here considered to replicate the classification task, taught to the machine via a set of examples (labeled set of seismo-volcanic observations).

One particular challenge when using supervised learning algorithms on volcanic data is the time scale at which we work. A volcano evolves slowly. A thousands of years can be necessary for a volcano to grow, and centuries can pass between two eruptions. But as specified in Chapters 2 and 3, the labeled dataset used to train a prediction model on a given phenomenon must fully describe the considered phenomenon. This constraint is not fulfilled when using seismo-volcanic signals. On a similar note, the notion of ground truth is extremely relative when studying volcanoes. Physical interpretations relating the classes of observations to a physical behavior of a volcano are hypothesis, and the absolute ground truth is not known. All the signals are also relatively similar, and are difficult to classify, including by the observatories experts. Furthermore, some classes of observations occur relatively often (every day), while other are much less frequent (not every year), which eventually lead to very unbalanced problems and complicates the problem event more. Finally, the physical shape and structure of a volcano evolves with time. An eruption usually changes the dome shape, meaning that the propagation of the

seismic wave through the volcano is altered, and that the seismic signal recorded on the various stations evolves, despite having similar causes. All those specificities make the classification of volcano-seismic observations a complex and laborious tasks, and need to be considered to provide accurate, efficient and useful monitoring tools. The classification task for volcano-seismic observations is particularly complex: observations are very similars, and classes are difficult to distinguish. See for example Figure 33 at the end of this chapter, which illustrates the difficulty to classify the observations of Mount Merapi.

Synopsis

This chapter contains (i) an analysis of the state of the art in automatic methods used for the monitoring of volcano-seismic data (automatic detection of relevant events, and their automatic classification), (ii) the architecture used for the automatic analysis and its specificities, and study results on two volcanoes. The first one is Ubinas, in Peru (Figure 25), which proposes a 6 years of labeled seismic data and is used to validate the method and to perform an a posteriori study on the volcano. The second one is Merapi, in Indonesia (Figure 24), which is monitored by the Balai Penyelidikan dan Pengembangan Teknologi Kebencanaan Geologi (BPPTKG) for whom we installed the automatic classification architecture in an operative context, for a real time monitoring. Those two volcanoes are closely monitored: they are close to cities (Arequipa and Yogyakarta, respectively), and eruptions can have dramatic effects. More details about both volcanoes are given in the body of the chapter.

This chapter is based on two papers that we published in IEEE Signal Processing Magazine [MDMM+18] and Journal of Geophysical Research [MDMM+ss], for an analysis of the state of the art and presentation of the architecture, and for a detailed analysis of Ubinas volcano respectively. An article regarding the franco-indonesian cooperation with BPPTKG and Merapi volcano is also to be included in a book published by the franco-indonesian embassies. This chapter is best understood if Chapters 1, 2 and 3 have been read. However, reminders of the technical points are made, and most of the chapter can be read independently from the rest of this PhD thesis.

Figure 25

Ubinas volcano and Ubinas village, Peru. Photos by Melquiades Àlvarez.

(a) Ubinas volcano

(b) Crater of Ubinas volcano

(c) Ubinas volcano during an active phase

(d) Ubinas volcano, from Ubinas village

5.1 State of the Art about the Automatic Monitoring of Volcanoes

The main task concerning the monitoring of volcanoes using seismic recordings consists in detecting relevant events, and to classify them. Traditionally, those two steps are done separately and we here review the main methods used to automatize the tasks. The detection stage is often done manually, see for example [AWO+06, LFT03, GES+09, FGNS96, SGE+05, KOS10]. Some automatic detection processes have been proved to be efficient though, and are relatively commonly used in operative contexts. The STA/LTA method (short term average, long term average) is by far the most popular detection method and was originally presented in [All78, All82]. It consists in measuring the energy of the considered observation (short term average), and to compare it to the neighbor energy level (long term average). It is widely used in operational context and in published works, including [IvS08, BAMOA13]. Other detection systems are also considered, among which measurements of the signal kurtosis [LMMB14] or optimal filtering [LCMLB16]. Results are satisfactory or promising but present the same limitation. In particular, they have been proved to be efficient for well-separated events, or for some specific volcano-seismic classes. Typically, events which are notable longer than the average or emergent events are not detected. Those signals are difficult to detect because their starting and end points are not always clearly detectable, especially when the analysis is carried out on relatively short observations. Emergent signatures are also likely to be overlapped with other events signatures, which also complicates the issue. Another issue of volcano-seismic events detection is the high variability in events durations: they can last less than 10 seconds for some to several days. The manual tuning of the methods parameters (setting thresholds, window lengths, etc) is also a limitation, and many of those approaches are tested on relatively small datasets (few hundreds or less than a hundred samples in some cases), or on datasets including only a given class of signals. To our knowledge, there is no established procedure to detect volcano-seismic events in continuous recordings when (i) numerous signals occur in a short period of time (hundreds per hour) which is the case during an eruption: in this scenario signals associated to different events (not necessarily of the same type) can occur overlapped in time and show very different amplitudes; and (ii) for emergent signals, i.e. signals whose amplitude increases and decreases very slowly.

Once extracted (manually or using an automatic detection algorithm) volcano-seismic events need to be classified into one of several classes related to a physical behavior of the volcano. This information is then used to analyze the volcano and predict eruptions. In many observatories, this classification task is still done manually but the literature offers some studies on the subject. Hidden Markov Models (HMM) have been used [AWO+06, GIC+09, CAG+09]. Neural Network are also popular, but with very various results [LFT03, DPEG+03, LFPT06, IvS08, CVF+09, GES+09, FGNS96, SGE+05, LFM+09]. Bayesian classifiers were also tested in [OGDC06], but with limited results. Support Vec-

tor Machines algorithm also have been used, for instance in [GES+09, LFM+09, BAMOA13] with excellent results and very similar method than studies using Random Forests as learning algorithm, see for example [MFH+17, HPM+17]. Some studies also tried using unsupervised models, such as [DPEG+03] with Principal Component Analysis (PCA), [LFM+09, KOS10, EGD+08] with Self Organizing Maps (SOM) or [LFM+09] Cluster Analysis (CA). Results however, are highly variable. The main gap in the state of the art regarding the automatic classification of volcano-seismic events remains the lack of effective and operational tools. Operational tools set up in volcanic observatories are only mentioned in [MFH+17, BRS+07, KOS10]. Many studies provide interesting results, but cannot process continuous recordings or segmented events, or are validated on rather small datasets. The features used to represent the observations are reviewed in Chapter 2.1.3. The latest works show a improvement in using a larger number of high level features, but a large number of study still use feature set that could be improved. Too simplistic features struggle to embrace and represent the signals properties.

As previously said, detection and classification are traditionally done as two separate tasks in volcano-seismic analysis, but a few studies have developed architecture to process a continuous analysis. Such architecture are based on HMM, see for instance [IBG+09, BRS+07, Ohr01, HBO12]. Once again however, results are very fluctuating.

5.2 Proposed Approach

The approach proposed for this study targets the automatic classification task, and aims at being robust and operational in monitoring routines. We therefore use the architecture for a static analysis, presented in Chapter 3.1.1.

In this scenario, an observation is made of a seismic recording displaying a volcano-seismic event. The signal is high-pass filtered above 1Hz, in order to remove the tide component, but also to match short band sensors. The signal is also normalized in term of its energy, which is computed and normalized to 1. This is done to build a model that is available for all observations, regardless of their amplitude level. It is worth noting that most state-of-the-art methods are based on the energy levels [LFPT06], and therefore cannot be used to detect non energetic observations. Those observations constituting the dataset are extracted by automatic methods (STA/LTA for instance), or manually. They are manually labelled into classes of meaningful importance regarding the volcano behavior.

The observations are then transformed into feature vectors. The features used to represent the observations are listed in Table 9, along with their associated reference number (used in Section 5.3.3). Those features are the one presented in Chapter 3.2. More precisely, the shape descriptors from the table are computed on three low level representations of each observation $[s_k]_{k=1}^{n}$: in time s_k, in frequency S_f, and in quefrency $\mathcal{S}_q$. A classification can then be trained from the labeled dataset of observations.

Feature	Definition	Ref.
Length	$n = \text{length}(z)$	1
Mean	$\mu_z = \frac{1}{n}\sum_i z_i$	2
Standard deviation	$\sigma_z = \sqrt{\frac{1}{(n-1)}\sum_i (z_i - \mu_z)^2}$	3
Skewness	$\frac{1}{n}\sum_i \left(\frac{z_i-\mu_z}{\sigma_z}\right)^3$	4
Kurtosis	$\frac{1}{n}\sum_i \left(\frac{z_i-\mu_z}{\sigma_z}\right)^4$	5
i of Central Energy	$\bar{i} = \frac{1}{E}\cdot\sum_i E_i \cdot i$	6
RMS bandwidth	$B_i = \sqrt{\frac{1}{E}\sum_i i^2 \cdot E_i - \bar{i}^2}$	7
Mean skewness	$\sqrt{\frac{\sum_i (i-\bar{i})^3 E_i}{E\cdot B_i^3}}$	8
Mean kurtosis	$\sqrt{\frac{\sum_i (i-\bar{i})^4 E_i}{E\cdot B_i^4}}$	9
Shannon entropy[a]	$-\sum_i p(z_i)\log_2\left(p(z_i)\right)$	10 to 12
Rényi "entropy"[b]	$\frac{1}{1-\alpha}\cdot\log_2\left(\sum_i p(z_i)^\alpha\right)$	13 to 18
Rate of attack	$\max_i\left(\frac{z_i-z_{i-1}}{n}\right)$	19
Rate of decay	$\min_i\left(\frac{z_i-z_{i+1}}{n}\right)$	20
Ratios	min/mean, max/mean	21 to 22
Energy descriptors	Signal energy, maximum, average, standard deviation, skewness, kurtosis	23 to 28
Specific values	min, max, i of min, i of max, threshold crossing rate, silence ratio	29 to 34

[a] Bin numbers for probability estimation: 5, 30, 500.

[b] Bin numbers for probability estimation: 5, 30, 500, $\alpha = 2$ and inf.

Table 9

List of features used to represent volcano-seismic observations. Features computed for a signal $[z_i]_{i=1}^n$ (which can correspond to an observation in temporal, frequency, or cepstral domains). $E = \sum_{i=1}^n z_i^2$ and $E_i = z_i^2$ describe the signal energy and the energy at sample i, respectively. Features are referenced by a letter-number system, for instance, feature F3 is the standard deviation computed from the observation spectral representation.

In the following, we show the relevant of this approach on two volcanoes. First, a very large database of volcano-seismic signatures of Ubinas volcano is studied to show the relevance of the proposed approach. Typically, the approach is tested, validated and studied. In particular, the impact of the features is detailed. Six years of volcano-seismic recordings of Ubinas are then analyzed. We then use the dataset to simulate a continuous and operative monitoring of Ubinas. We typically show that the approach was validated, and then implemented for the monitoring on Merapi volcano.

5.3 Ubinas Volcano: Validation of the Approach and a Posteriori Analysis

5.3.1 Ubinas Volcano

The first application aims at validating the proposed approach for the monitoring of volcanoes, but also to perform an a posteriori analysis over 6

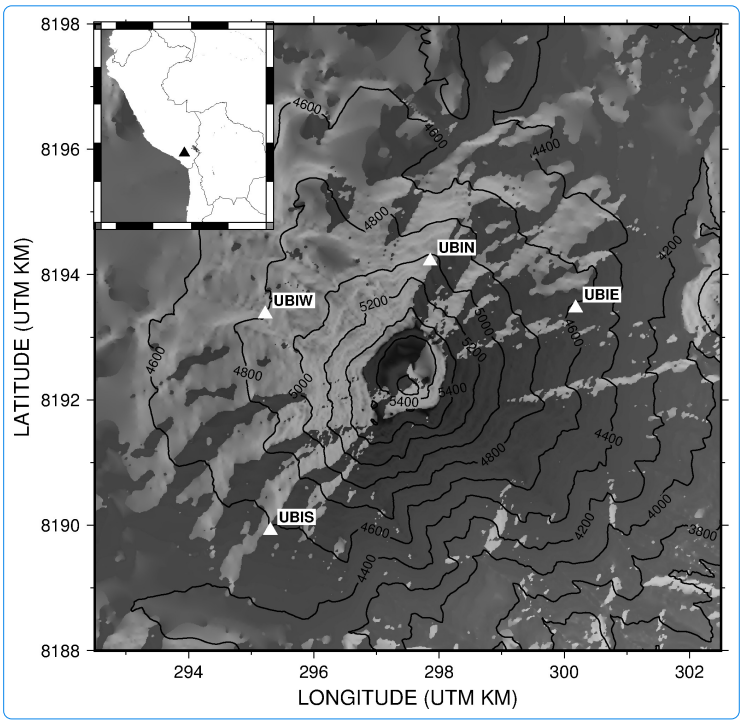


Figure 26

Map of Ubinas Volcano with the locations of the permanent IGP seismic network indicated (white triangles). The data used in this study are recorded at UBIW station. Insert, top left: location of Ubinas Volcano (black triangle) within Peru.

years of volcano-seismic recordings. The data comes for the observatory monitoring Ubinas volcano, in Arequipa, Peru (Figure 25). This study involves the participation of Jean-Philippe Métaxian from IPGP (Paris, France), Orlando Macedo from the University Nacional San Agustin (Arequipa, Peru) and Adolfo Inza, from the Instituto Geofisico del Peru (Arequipa, Peru).

Ubinas Volcano is an andesitic stratovolcano in southern Peru, (16° 22' S, 70° 54' W; altitude, 5672 m). It is considered to be the most active volcano in Peru, and is closely monitored by the *Instituto Geofísico del Perú* (IGP). After nearly 40 years of quiescence, Ubinas Volcano erupted in 2006. Three eruptions have occurred since 2006, from 2006 to 2011, from 2013 to 2014, and in 2016. Ubinas Volcano is monitored seismically by the IGP since 2006 [MMT+09], with the cooperation of the VOLUME project (funded by the European Commission 6th Framework Program) and the Institut de Recherche pour le Developpement (France). The first permanent telemetric station (i.e., `UBIW`) was equipped with a short-period vertical 1-Hz sensor that was installed in May 2006 on the northwest flank of Ubinas Volcano [MMT+09]. Three additional stations were added in 2007 (i.e., `UBIN`, `UBIE`, `UBIS`). `UBIN` was equipped with a broadband vertical sensor, and the other stations had short-period sensors. In addition, `UBIN` and `UBIS` were equipped with bi-axial tiltmeters with 0.1 micro-radian resolution [IMM+14]. These four stations have been working permanently since 2007 (Figure 26). The data are recorded continuously with a sampling rate of 100 Hz, and they are then transmitted in real time to Cayma Volcanological Observatory in Arequipa (Peru). The analysis is here based on seismic data from the vertical component of `UBIW` station (see Figure 27 for an example of recording).

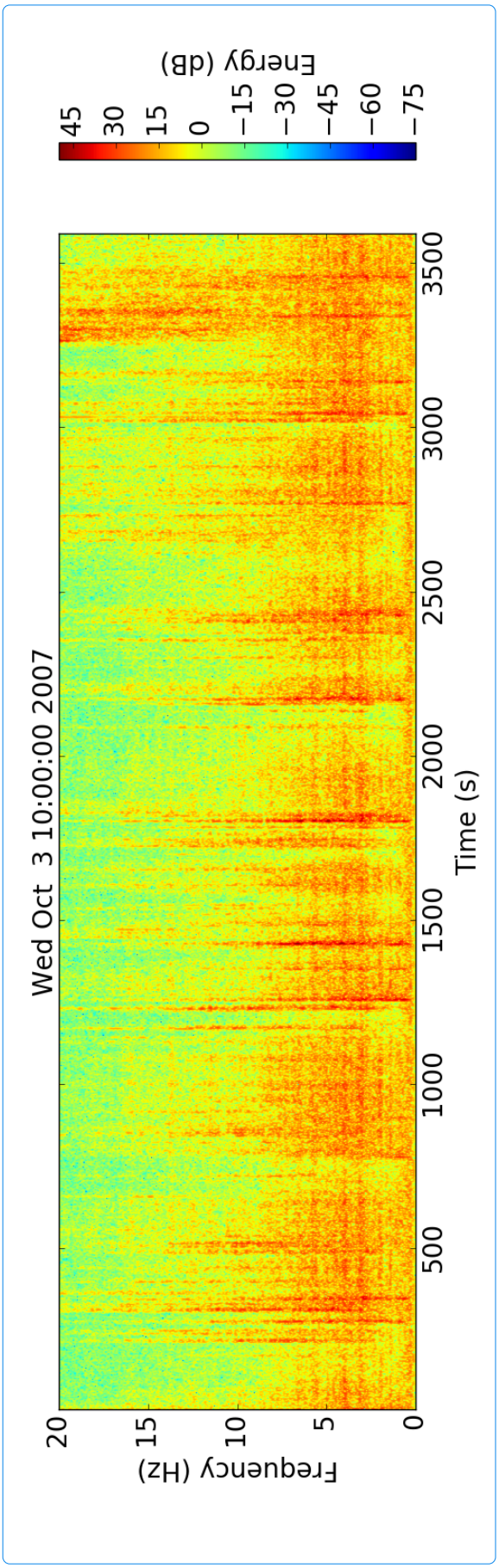


Figure 27

Seismic recording of Ubinas, station UBI.UB, *3/10/2007 at 10am.*

The data used for the present study came from a catalog of $N = 109,609$ seismic observations of volcanic events that were recorded between May 2006 and October 2011. Each observation was manually identified and extracted by the IGP team in Arequipa Observatory. Six very heterogeneous classes of signals were defined by the IGP after 10 years of observations of Ubinas Volcano. Each class is associated to a physical state or activity of the volcano. These types of signals are observed for other volcanoes, and have been described for many years in the literature [LCS+94, Cho96, NLBO00]. All the classes of events that can be recorded on Ubinas are illustrated in Figure 28 and are listed as follow:

1. `Long-period` (`LP`; 95243 observations over 6 years) - This signals come from fluid processes. They are interpreted as a time-localized pressure excitation mechanisms, followed by the response of a fluid-filled resonator [CM13]. Different models have been developed to explain the resonance observed for `LP` events, including in particular the fluid-filled crack model [Cho86] and the fluid-filled conduit model [NLBO00]. A wide variety of volcanic processes can produce the excitation mechanism that triggers crack or conduit resonance, in particularly for lava dome growth for andesitic volcanoes [NLBO00, MMQ+08, CM13].

2. `Tremors` (`TR`; 12309 observations over 6 years) - They are defined by a sustained amplitude that can last from seconds to days, and occur over a frequency range from 1Hz to 9Hz [McN92]. The literature reports that many characteristics of `LP` events, and in some cases also of explosion quakes (see below), are commonly associated with `TR`. Virtually, all eruptions are accompanied by `TR` [McN92]. Visual observations at Ubinas volcano suggests that `TR` are associated with magma extrusion and sporadic or continuous gas and ash emissions [MMT+09].

3. `Explosions` (`EXP`; 159 observations over 6 years) - Explosions are associated to sudden magma extrusion, and ash and gas emission. Physically, they are related to fragmentation processes in the conduit, as are observed on many andesitic volcanoes [DYB+02, Ohm06, IYT+08]. For Ubinas volcano, `EXP` are related to destruction of the magmatic plug [IMM+11]. Especially, there is an overall acceleration of the `LP` rate over the 2 to 3 hours before Vulcanian `EXPs` for Ubinas volcano [TLM+11]. This occurrence of a large number of `LP` events before `EXPs` is consistent with the dome growth process.

4. `Volcano-tectonic` (`VT`; 1315 observations over 6 years) - This class is related to brittle-failure events. They are associated to stress changes that are induced by magma movement [CM13]. There are relatively few `VT` events at Ubinas Volcano, compared to `LP` events, but their number increased from 2006 to 2011. `VT` events are more numerous at the end of an eruption period.

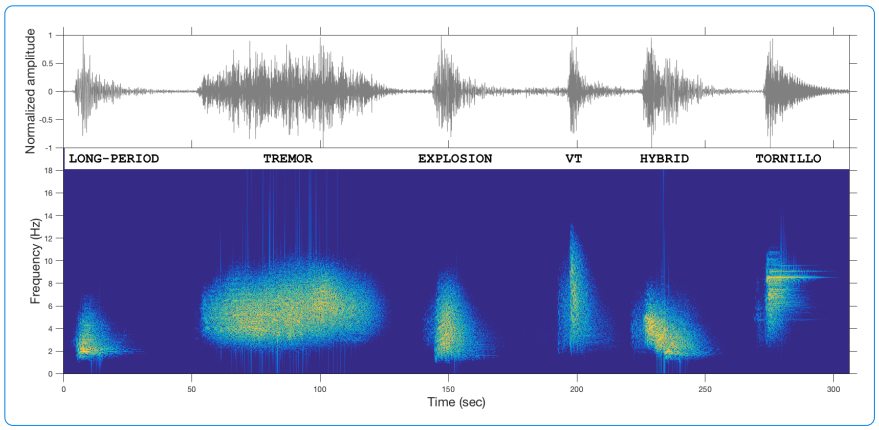


Figure 28
Waveform and spectrogram (Gaussian window of 512 samples width) of volcano seismic signals recorded at Ubinas Volcano. Six observations are displayed. The amplitude is linear and has been normalized.

5. `Hybrid` (`HYB`; 474 observations over 6 years) - Those observation have characteristics of both `VT` and `LP` events, with high-frequency of onset followed by low frequencies. `HYB` events are introduced by [LCS+94] to describe events observed at Redoubt Volcano. They have also been observed for Soufrière Hills Volcano, Montserrat Volcano [WMLP98], and Mount St. Helens Volcano [HB07], where they are related to dome growth.

6. `Tornillos` (`TOR`; also known as screw events; 109 observations over 6 years) - They are related to resonating fluid-filled conduits or cavities, and have a very limited distribution of frequencies and a very slowly decaying coda. `TOR` can be considered as a specific type of `LP` events with a long duration coda that is composed of harmonic oscillations. They were observed at Galeras Volcano before several eruptions in 1992 and 1993 [NTG+97]. `TOR` are rare for Ubinas Volcano, but they appear to be more common at the beginning and end of eruptive periods.

5.3.2 First Result: Performance Evaluation

In order to validate the methodology and to estimate results that can be expected by the proposed method, cross-validation is performed on the dataset. To consider relatively balanced classes, 1 year of observations are gathered for classes that are highly frequent (*i.e.*, `LP`, `TR`, `VT` events), and all of the observations available for the less frequent classes (*i.e.*, `HYB`, `EXP`, `TOR` events). In total, $70,856$ volcano-seismic observations are considered for this first experiment. For each class, a fraction α of the observations is used to train the model. Because all the learning data need to be loaded in the computer memory at the same time, the number of training observation per class is also limited to $N_{max} = 800$. Therefore for each class c_i with $1 \leq \ i \leq \mathbf{C}$, $N_{i,training} = \min(\alpha \cdot N_i, N_{max})$ with 70% ($\alpha = 0.7$ the learning rate) of the signals are randomly chosen to train the model. The remaining $N_{i,testing} = \max\left(N_i \cdot (1-\alpha), N_i - N_{max}\right)$ labeled observations are used to test the model. The process is repeated 50 times to obtain statistically stable results. In the following, the results are expressed as means and standard deviation over the 50 trials. Using this process, a comparison

Table 10

Confusion Matrix for a analysis modelmodel trained with RF, 50-fold cross-validation with $\alpha = 70\%$. Overall accuracy: $92.5 \pm 0.45\%$.

		True Class (ground truth)						Precision
		LP	TR	VT	EXP	HYB	TOR	
Predicted Class	LP	**57504**	457	4	1	8	-	99.2%
	TR	3911	**4764**	-	1	3	1	54.9%
	VT	372	10	**487**	5	12	3	54.8%
	EXP	112	8	6	**41**	-	-	24.6%
	HYB	128	6	14	1	**119**	-	44.4%
	TOR	2	0	3	0	0	**28**	82.8%
Accuracy		92.7%	90.8%	94.6%	84.8%	83.6%	87.6%	

is made between 2 learning algorithms: RF and SVM. The observations are represented using the features of Table 9, extracted from three representations of the observations (time, spectral, cepstral), see Chapter 3.2 for more details. Using RF, the overall accuracy reaches $92.5 \pm 0.45\%$ compared to $92.1 \pm 0.54\%$ with SVM (RBF kernel, $C_{RBF} = 10$, $\gamma = 0.01$). Those results validate the effectiveness of the proposed architecture along with the feature choice, and illustrate the limited influence of the learning algorithm.

The best model is in this scenario the one trained using SVM. The associated confusion matrix is proposed in Table 10 provides more details about the class-by-class results. For instance, the best classified classes are VT with 487 out of 514 (94.6%) and LP, with 57504 correctly classified observations compared to 62029 considered (92.7%).The confusion matrix can also be used to analyze the model limitations. Most of the prediction errors are divided up between: (i) LP events mixed with TR; (ii) HYB events mixed with VT and LP events.; and (iii) a bias for EXP and VT toward LP.Those mistakes are seen through the precision rates which are related the the false detection rates, and are low for some classes. Physical interpretation of these results is valuable, as all of the errors made by the model translate into physical similarities between the signals. For example, to parallel the main three prediction errors above: (i) LP events and TRs are in the same frequency range and can overlap, and typically a LP event can be found within a TR. This will confusing the model, which predicts a single class at a given time. [MMT+09] also showed that on some occasions, LP events occur in a repeated way, and can be separated at the beginning by some tens of seconds, before merging into a TR (*e.g.*, before an EXP). (ii) HYB events have characteristics of both LP and VT events, and they can belong to one class or the other. The analysis of this error is thus particularly valuable, as these results can help volcanologists to better analyze the seismicity, the relations between classes of events and their evolution with time. The ability of these methods to process very large datasets of recordings is essential for volcanic observatories.

5.3.3 Second Result: Features Influence and Selection

The second experiment illustrates the influence of features chosen to represent the observations. The same cross-validation process is used to com-

pare the influence that features can have on the classification results. In particular, the use of several computation domains for the features is studied. The overall accuracies are reported in Table 11 vary from $78.4\% \pm 1.0\%$ when using `cepstral` features and SVM, to $93.5\% \pm 0.50\%$ when using `frequency` features and SVM. It is particularly interesting to note that for the application, the frequency features appear to be particularly discriminative. The manual classification is often based on the frequency content of the observations which could explain the results.

As explained in Chapter 2.2, RF model estimates the features weights and can be used as a feature selection tool. In particular, we investigate the possibility to reduce the feature vectors dimension given that some features are strongly correlated. As a reminder, this dimension should be kept as low as possible to avoid accuracy losses due to the curse of dimensionality (see Chapter 2.1). We intentionally did not use compression algorithms (*e.g.*, principal component analysis), so as to maintain the physical meanings of the features. Figure 29 shows the features ranked by importance using the RF feature weights. It also displays the cross-validation accuracies when representing the observations by feature vectors of dimension d with $1 \leq d \leq D$, composed of the d most important features. From this analysis, two subsets of the most important features are extracted: the three `most valuable features` (Figure 29, `MVF`, filled squares) and the 13 `valuable features` (Figure 29, `VF`, filled and empty squares, which include the `most valuable features`). These features are selected as subsets leading to a notable gain in accuracy with a reasonably small dimension. The accuracy reaches $84.4\% \pm 0.50\%$ when considering the `most valuable features`. This result is interesting, as it shows that good classification rate can be obtained with a very restrictive number of features. In particular, this is of importance for real-time or embedded systems and applications. Indeed, the `most valuable features` contain the three most important features (3% of the total features), which are (i) `F4`: the spectrum skewness, which measures the spectrum asymmetry; (ii) `F5`: the spectrum kurtosis, which measures the spectrum peak-ness; and (iii) `T7`: the RMS bandwidth in time, which measures the mean length of the signal around which the energy is accumulated. The physical analysis of these features is, as such, a good indicator of the signals for the experts. In particular, while the feature ranks based on RF can be used to select such relevant features, these features are themselves interesting to analyze, as they gather and embody the information-discriminating observation

Features	Dimension	SVM	RF
`All`	102	$92.1 \pm 0.54\%$	$92.5 \pm 0.45\%$
`Most Valuable Features`	3	$84.4 \pm 0.50\%$	$84.2 \pm 0.75\%$
`Valuable Features`	13	$86.9 \pm 0.60\%$	$90.3 \pm 0.52\%$
`Time`	34	$83.9 \pm 0.89\%$	$87.2 \pm 0.57\%$
`Frequency`	34	$93.5 \pm 0.50\%$	$91.3 \pm 0.45\%$
`Cepstral`	34	$78.4 \pm 1.0\%$	$79.3 \pm 0.76\%$

Table 11

Influence of the feature set used to represent the observations. Comparison of the accuracies when using random forest versus support vector machine, for the different feature sets.

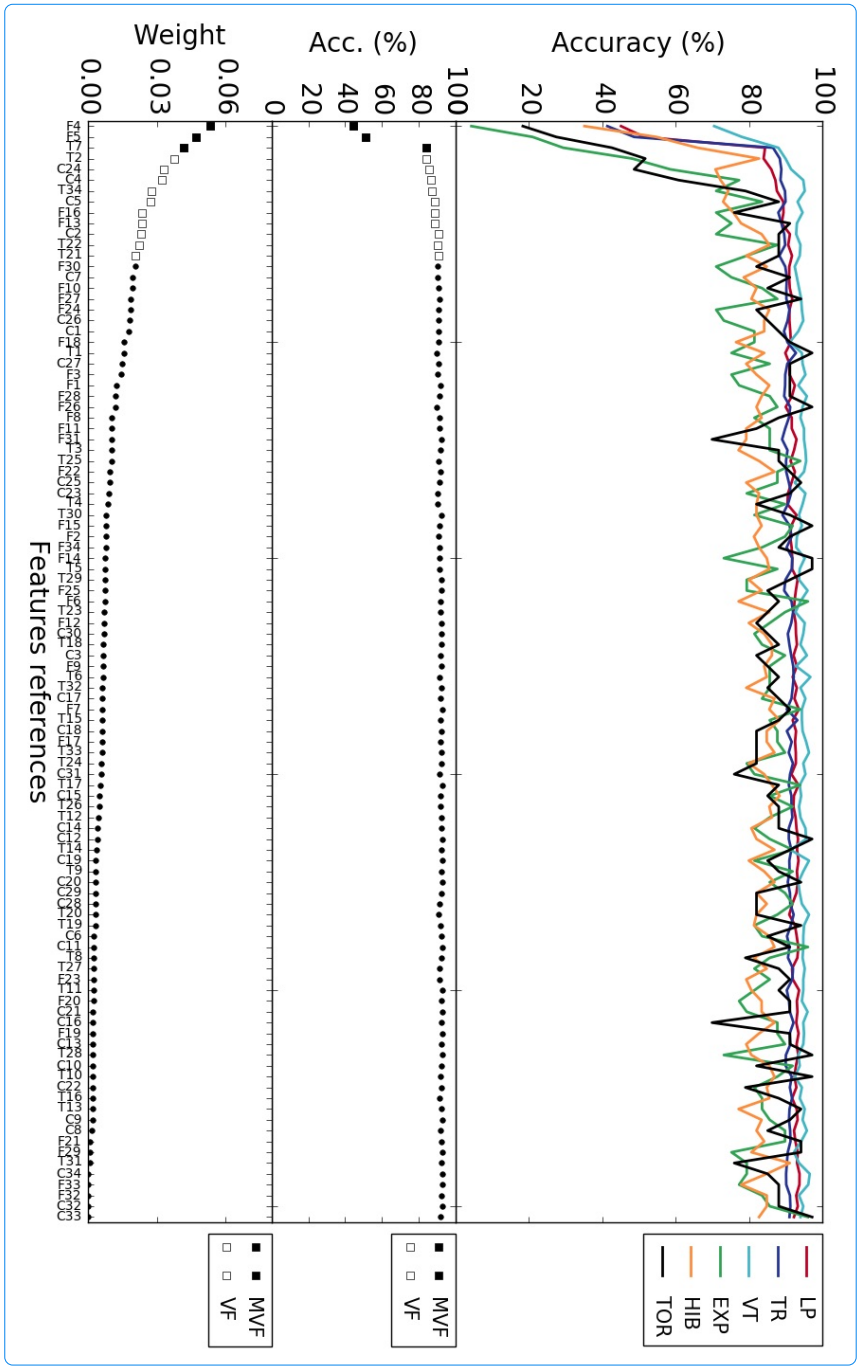


Figure 29

Importance and selection of features. Bottom subplot: Feature weights (mean weights on 1-year cross-validation models, trained with random forest on `All` *features; 100 trees;* $\alpha = 0.7$*). Middle subplot: Mean accuracies (models trained with random forest; 100 trees; on the* d*-st most important features with* $1 \leq d \leq D$*,* $D = 102$*). Top subplot: Mean accuracies for each class* c_i*,* $1 \leq i \leq C$*. The three* `most valuable features` *(MVF; filled squares) and the 13* `valuable features` *(VF; filled and empty squares) are indicated. Features are referenced by a letter-number system as given in Table 9.*

across the classes. Similarly, the 13 most important features were extracted (13% of the features) for the feature set of the `valuable features` (which includes the `most valuable features`). This feature set is particularly interesting as it provides a performance that is close to that of the whole feature set, at $90.3 \pm 0.52\%$ when using RF. It is worth noticing that when using the `valuable features`, RF performs notably better than SVM, which is consistent since RF was used to select the feature set. It is also interesting to note that the `valuable features` are features from the three computation domains, as time, frequency, and cepstral, thereby confirming the interest in considering these three different domains for signal representation.

5.3.4 Third Result: Analysis Over Six Years of Volcano-seismic Recordings

Depending on the volcanic activity and the volcano structure which could significantly evolve with time, observations within a same class can have a significant variability and change over time. To estimate this evolution, we consider a RF model trained on the $N_{max} = 800$ first observations of each class (less if 800 are not available), and to test it on the six years recordings. The overall accuracy over all the classes is of 59.7%, which is relatively low compare to the previous results. We therefore propose to study the evolution of the dominant class: Long Period LP, with 95,094 events. The monthly evolution of LP accuracy from 2006 to 2011 is displayed in Figure 30. We here remind that the $N_{max} = 800$ first LP observations are recorded in few days only (June 2006). LP overall accuracy is of 61.0%, which once again is relatively low compared to cross-validation results, but its evolution is curious and worth analyzing.

In particular, the accuracy collapses in May 2007, which reveals a significant change in the signal shapes compared to the training observations (May 2006). The accuracy drops from more than 95% in average to less than 10%. This period also corresponds to a sharp decrease in the number of LP events. This result was discussed with the team of Ubinas Volcano Observatory, in Arequipa. A manual revision of the seismic signals for the period of May-July 2007 was performed to determine whether the observatory analysis might have confused LP events with other signals. It turned out that the classification criteria had been improved since the beginning of the creation of the catalog in 2006. This evolution in the manual classification criteria results from the experience acquired during the other eruptions of Ubinas Volcano, as well as the signal classification of other volcanoes (Ticsani and Sabancaya). The new criteria used for the manual classification are now similar to those used for all of Peruvian volcanoes. This revised analysis thus showed a difference between the two manual classifications, starting from May 25, 2007. Some of the LP events were indeed mislabeled, essentially as VT events or TRs, or to a lesser extent, as HYB events. Part of the accuracy collapse observed from May 25, 2007, can thus be explained by this confusion between the classes. At the same time, the new classification showed a similar decrease in the number of LP events, which would instead suggest a change in activity, and therefore

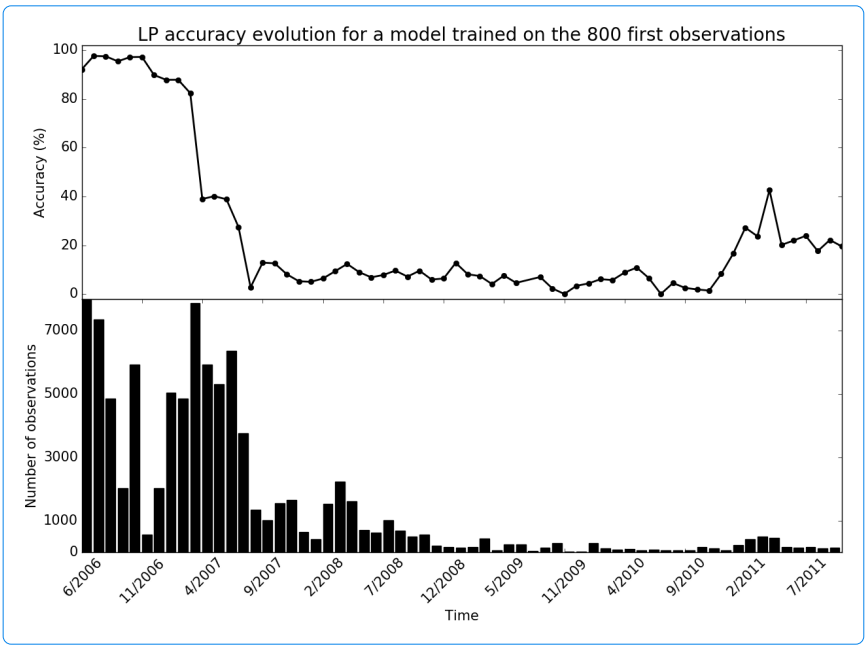


Figure 30

Monthly accuracy evolution for the LP observations, with the model trained on the first 800 observations of each class (all recorded in May 2006 for the LP events).

a change in the characteristics of these signals. [MMT+09] observed a strong temporal variations of the degassing and seismic activity in a period starting in November 2006. A backward migration of magma in the crater has been observed on images taken in December 2006, compared to previous images taken in April and October 2006 (Figure 5 in [MMT+09]). The receding of magma in the conduit has been observed over several months. Other images of the crater taken by IGP Arequipa Observatory respectively on April 17, 2007, June 8, 2007 and August 26, 2007 show a clear drop of the magma level between April and June 2007. The backward migration of magma in the conduit implies modifications of seismic source positions and possibly mechanisms due to the complexity of the geometry of the conduit and time modifications of the coupling between magma and the conduit walls. Magma migration can also affect local stress conditions or conduit cavity properties. This can obviously explain modifications of the characteristics of the LP events. The decrease of LP accuracy is starting in January 2007 and its strengthening between April and May 2007, which is coincident with observations of volcanic activity. The fundamental point is here that Figure 30 is a proof of a change in the volcano-seismic signals from May 2007, that is physically interpreted as a volcanic phase leading to a change in Ubinas structure and/or a internal activity. These modifications were not detected by manual classification, while the automatic classification perfectly identified them, This result is therefore particularly important since the use of automatic methods of classification revealed inconsistencies in the original manual classification. The use of such methods therefore have an interest beyond the automatic classification, and can be used as a physical analysis tool to study the evolution of a volcano.

5.3.5 Fourth Result: Simulation of an Operative Monitoring

In this last experiment, we focus on simulating an operative monitoring context, where Ubinas would be continuously monitored throughout the 6 years of recordings. To do so, a new classification model is trained each month using the past observations (i.e. observations happening before the considered month). This experiment raises the issue of the number of observations to use in training, since only a relatively small number of observations are available for the training stage of the first models (especially for rare classes). From the previous experiment, we also know that the labeled dataset present a large number of label mistakes, and that observations evolve within a given class. This new experiment therefore raises the ability of the model to follow those changes, and to adapt itself to new observation shapes. The models are trained on a maximum of $N_{max} = 800$ observations per class, which take very duration to collect depending on the considered class. Typically for very frequent classes such as LP, only a few days are necessary to gather N_{max} observations: new observations are therefore considered for each newly trained model. For less frequent classes such as Tr, few months are needed to gather the N_{max} observations: the first models will be then trained on less than 800 examples, but once this limit is reached, only the 800 most recent observations will be used for training. Finally, for very sporadic classes such as Hybrids HYB, the whole dataset does not contain N_{max} observations, therefore each monthly model will be trained on the available observations. For the first months, the number of observations can be very low (≤ 10).

Overall accuracy over all the classes and the 6 years reaches 80.3% in this classification, and as in the previous experiment we study the evolution of the class by class accuracy. Figure 31 displays the evolution for LP which is the most represented class and can be compared to the previous experiment results. It is interesting to interpret the performance of the

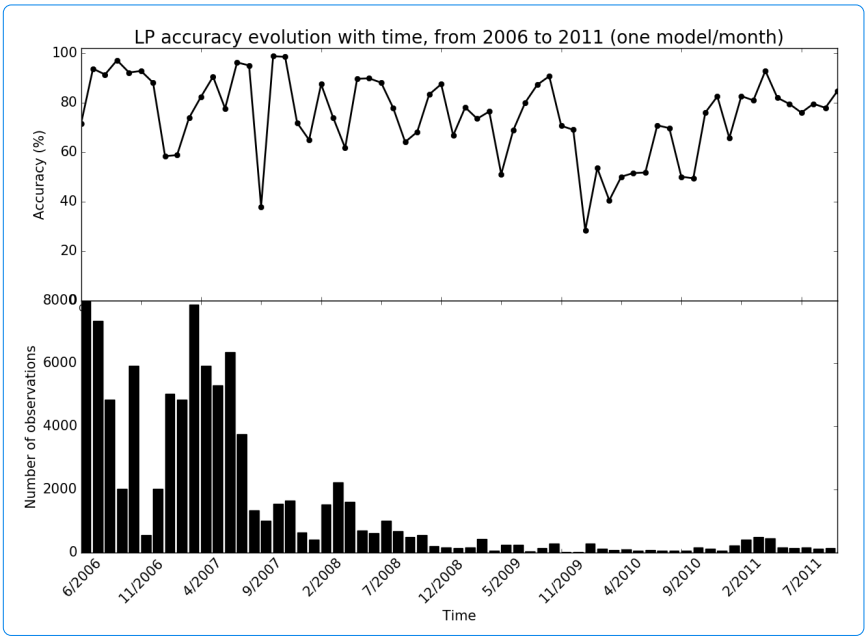


Figure 31

Accuracy evolution of LP, from 2006 to 2011. A new model is trained every month, on a maximum of $N_{max} = 800$ observations per class.

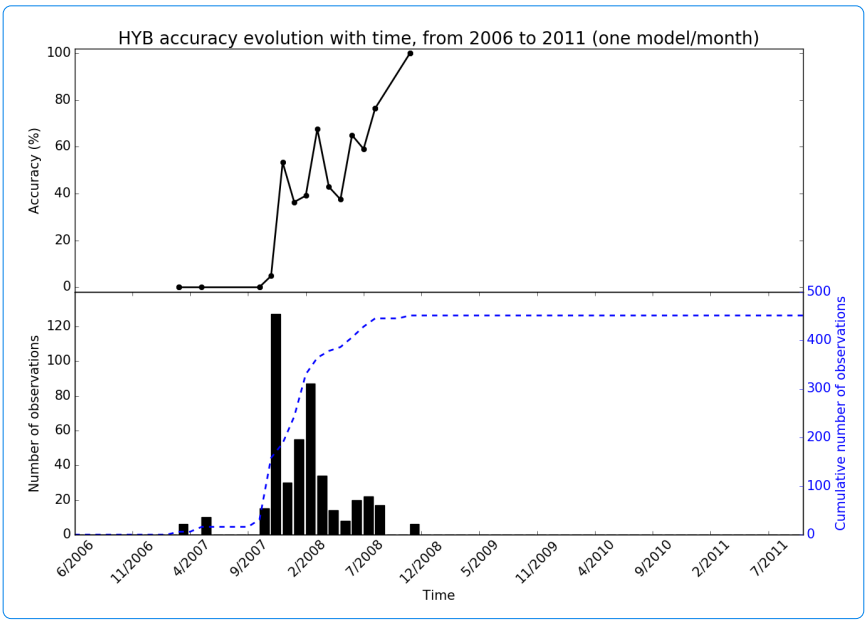


Figure 32
Accuracy evolution of Hybrid classification, from 2006 to 2011. A new model is trained every month, on the past data (the maximum limit of 800 observations is never reached, the class is particularly sporadic).

evolving model compared with cross validation results and with the static model of Section 5.3.4. Indeed, the previous section revealed that observations within a same class do evolve with time. By using a newly trained model every month, we manage to follow and learn the classes evolutions. Global performances are indeed much higher in this configuration than in the previous one. However, the observations evolution is still visible with sudden accuracy drops in the plot: for example in September 2007, the accuracy drops from 95% to 38% and increases again to 98% in October. With the sudden change in signals shapes, observations from September are predicted with a model trained on data that are no longer representative of the class. As soon as the model has been updated however (in October), accuracy increases again. This phenomenon is observed on the six years and is notably visible between stable phases in the volcanic activity.

Studying the accuracy evolution of sporadic classes is also relevant to see the influence of the number of observations in training. However it presents a difficulty since very few observations are available. Typically for `HYB`, only six observations are available for the first model including this class: accuracy levels are then notably low (see Figure 32). With an increasing number of observations used in training however (blue plot displaying cumulative number of observations), the accuracy level steadily increases to very good performances (100% in November 2011). For this class, around 500 observations are needed for the model to learn the variability of the `Hybrids` signals.

5.4 Merapi Volcano: an Operative Approach

Merapi is a strato-volcano situated in Yogyakarta province in Indonesia (Figure 24), and is the most active volcano of Indonesia. It is currently continuously monitored by the observatory Balai Penyelidikan dan Pengembangan Teknologi Kebencanaan Geologi (BPPTKG) with a minimum

team of 15 persons. Mount Merapi erupts relatively often (around every 5 years), with typical eruptions consisting in a growth and partial collapse of the dome, leading to pyroclastic flows. The last erruption of this type started on August 17th, 2018. The areas surrounding the volcano are highly populated, and the city of Yogyakarta (400.000 people) is situated at 25km of the volcano. The monitoring task is therefore if crucial importance in order to avoid casualties, and regular safety recommendation made by the BPPTKG to the local authorities. Merapi is currently in an active phase with several phreatic eruptions since May 11th, 2018.

Mount Merapi has been seismically monitored by the observatory BPPTKG since 1924 and is therefore a well known volcano. The various classes of events are well known and well defined, with effective procedures for the manual classification. Those classes are hereafter detailed, and spectrograms are presented in Figure 33.

1. `Tect` - Tectonic seisms, of non volcanic origin. They are related to the subduction, and are produced at hundreds of kilometers from the volcano. They can be recorded from stations outside of the volcano monitoring sensors.

2. `Tectloc` - Tectonic seisms, of non volcanic origin, but locally produced.

3. `Tele` - Tele-seisms, which are tectonic seisms of very distant source (thousands of kilometers).

4. `Volcano Tectonic` of type `A` or `B`, which reveal a tectonic fracture. The two types correspond to a deep (A) or shallow (B) source. Those events are usually of high energy and are relatively frequent, and last between 5 to 20 seconds.

5. `Low Frequency` (`LF`), which reveal a fluid movement, such as gas or magma. This class of signal is relatively rare, but should not be missed for its physical interpretation is important. Usually, `LF` have a relatively low amplitude.

6. `Multi Phase` (`MP`), which reveal a magma friction in the conduit. The events are made of several unidentified phases (4 or 5) followed by a coda and usually last around 5 seconds. `VT` of small amplitude could be `MP`.

7. `Rockfall`, also known as `Guguran`, which correspond to rock falling. The observations are emergent and usually energetic signals. They last from seconds to minutes and can be very varied depending on the path taken by the falling rocks.

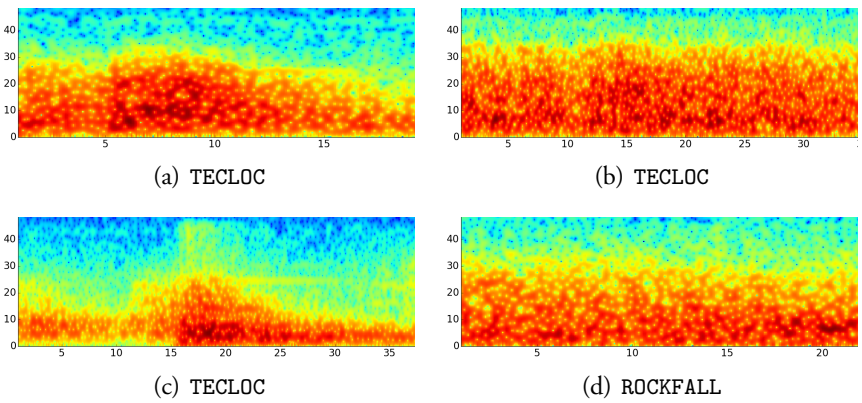
8. `Pyroclastic flow`, which usually last around 20 minutes and have varied shaped depending on the cause. Lava exiting the dome lead to rather continuous signals, while a collapsed lava dome lead to more discrete events.

9. `Tremors` which are related to the eruption are display continuous flows of gas and/or magma exiting the dome. They are different from Explosions, which are extremely rare at Merapi, and not even considered as a class.

A collaboration was started with Agus Budi Santoso from BPPTKG and François Beauducel from IPGP to deploy the AAA [Mal18] architecture for the automatic classification of volcano-seismic observations of Mount Merapi. Before July 2017, the seismic monitoring was done manually from the paper seismograms, and a record was kept counting how many observations of each class did occur on each day. Since July 2017, the monitoring has become mainly numerical with WebObs platform, which was developed and deployed by François Beauducel and the IPGP team at BPPTKG. This change of support for the monitoring has a great impact, since the geochemical and geophysical data are available on a server for the BPPTKG team. The fact that various physical measurements can be obtained on the same post and under the same architecture is a great evolution, and opens the door to mome complex analysis. Since the deployment of WebObs at BPPTKG, a labeled dataset of 1265 volcano-seismic observations is constituted. The events are detected using STA / LTA methods and are then classified by the operators. Manual detections can also be added to the dataset.

The AAA module which is presented in Chapter 3.3 and in Appendix B was added in March 2018. Nine months of labeled observations to build a RF classification model. The classification task can therefore be suggested using the AAA module. For a new detected observation, the latest classification model is interrogated, and proposes a list of probabilities of the observation belonging to the classes. In this operative context, it is important to leave the final decision to the experts, and the AAA module is used as an extra tools for the decision making process.

The system is currently used at the BPPTKG observatory, Indonesia.

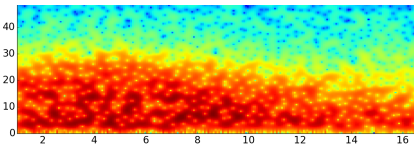


(a) `TECLOC` (b) `TECLOC`

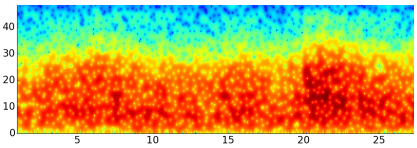
(c) `TECLOC` (d) `ROCKFALL`

Figure 33

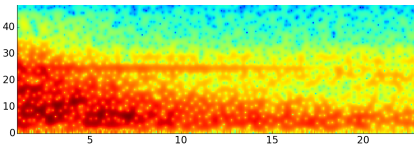
Spectrograms of observations of Mount Merapi. The spectrograms are purposefully small in order to compare the different observations. This figure underlines the difficulty to correctly classify seismo-volcanic observations.



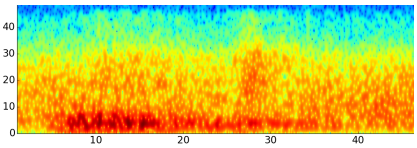
(e) ROCKFALL



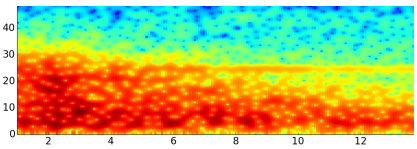
(f) ROCKFALL



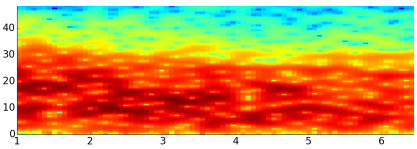
(g) VTA



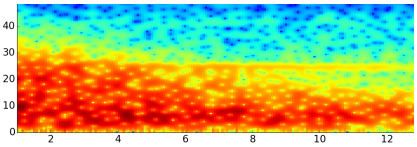
(h) VTA



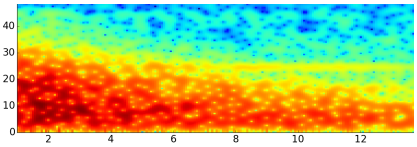
(i) VTA



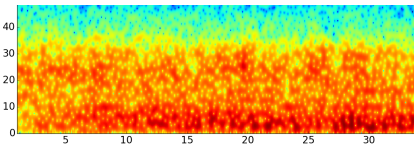
(j) VTB



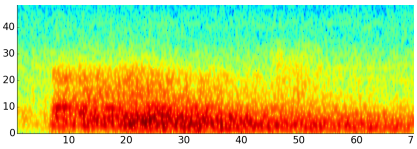
(k) VTB



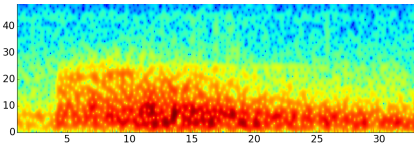
(l) VTB



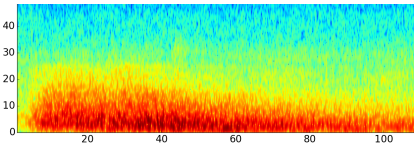
(m) TECT



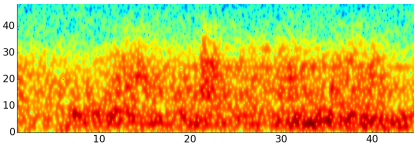
(n) TECT



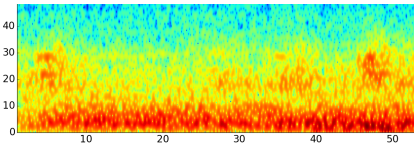
(o) TECT



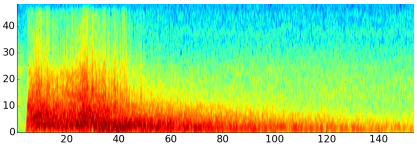
(p) TELE



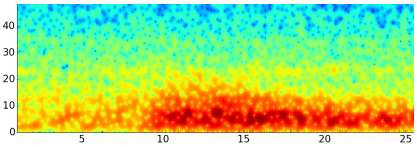
(q) TELE



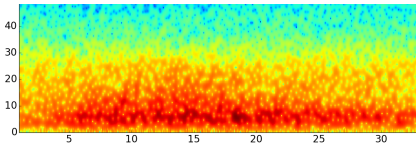
(r) TELE



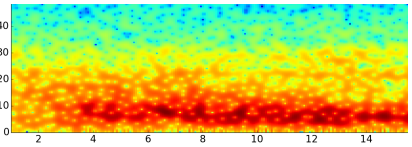
(s) TELE



(t) TPHASE



(u) TPHASE



(v) TPHASE

Highlights & Summary

- Volcanoes are responsible for the death of 540 persons per year. The deadliest eruption would be the Tambora in 1815 with more that 71,000 victims (Indonesia) [Opp03]. Eruptions can also last several years. Reliable tools for the monitoring of volcanoes are extremely important, with huge societal, environmental and economical consequences.
- Volcanic monitoring is done by studying the evolution of various parameters, including the seismicity around and on the volcano. Seismic recordings bear the signature of various classes of volcanic events, which are related to the volcano behavior by the experts. By detecting and classifying the volcano-seismic events, experts can study and better understand the volcano, and predict eruptions. The detection of such event is manual, semi-automatic or automatic with the use of methods such as STA / LTA (which has limitation, but is robust enough to be used in applicative contexts) The classification remains manual in the observatories, even if some studies have been conducted (but very often on limited number of events, classes or for a short duration not necessarily representative of the volcano)
- The AAA code is used here for the automatic classification of volcanic event.
- The method is validated on 70, 856 volcano-seismic observations of Ubinas (Peru). The accuracy reaches $92.5 \pm 0.45\%$ with a model trained using RF algorithm and `All` features. The influence of the learning algorithm is limited, but the impact of the feature choice is illustrated. In particular, feature selection can be considered: and with 13% of the features (called the `Valuable Features`), the accuracy drop from $90.3 \pm 0.52\%$.
- On Ubinas volcano, 6 years of data are analyzed (109, 609 volcano-seismic observations). Results show that the automatic analysis exceed the manual one, and reveled inconsistencies in the original labeled dataset. Those results have been discussed and communicated to the observatory of Arequipa who manually re-analyzed the dataset. They confirmed the change of behavior of the volcano, that was missed by the experts and detected by the automatic analysis.
- Another study is conducted to simulate an operative monitoring: a new prediction model is trained each month with the available labeled data. For evolving classes, this allows the model to learn the observation changes and keep up with the classification task. For relatively rare classes, the prediction models get better and better at classifying, with the increasing number of observation available for training. The study was done on Ubinas volcano, and lead to a collaboration with the observatory BPPTKG monitoring Mount

Merapi in Indonesia, where the AAA module was added to WebObs, the monitoring software.

- Given the clear impact that machine learning tools can have on the day to day monitoring of volcanoes, the many perspectives should be explored, and lead to effective and operational tools. To this effect, the AAA code is available on GitHub [Mal18], and is open both for observatories and for contributions.

Chapter 6

Transfer Learning & Deep Learning Tools

An investigation on their relevance for the automatic classification of transient signals

Contents

Introduction

This manuscript exposes and gives answer to the issue of environmental monitoring and the need to develop dedicated and operative tools. So far, the first chapters were linearly organized. Typically, Chapter 1 presents the issue. Chapter 2 gives the necessary backgrounds in machine learning and signal processing. Chapter 3 presents the proposed tools and the contribution of this PhD thesis: the automatic detection, classification, and anomaly detection schemes. Finally, Chapters 4 and 5 illustrate the proposed analysis schemes onto two different applications: underwater acoustic and volcano-seismic. In this final chapter, we propose to challenge one of our work hypothesis. In particular, we examine the possibility of using learnt features.

Looking Back on Feature Vectors

The automatic analysis schemes proposed in this manuscript are build upon the representation used to transform the considered observations (the data) into feature vectors. The importance of considering reliable feature vectors is exposed in Chapter 2, and as explained, feature vectors can either be learnt or hand-crafted (i.e. designed). In this PhD, we chose to work with hand-crafted features, and the proposed features extraction scheme is one of the main contribution of this manuscript (fully detailed in Chapter 3). This choice was lead by the applicative and operational point of view leading this PhD: hand-crafted features have a physical interpretation which is desirable for the various experts of the applicative fields. However, it would have been possible to use learnt features. This open question is the focus of this chapter: can we learn the representation to be used on the observations? What would the results be compared to the proposed feature extraction scheme (i.e., hand-crafted features)? In particular, the study addresses the use of convolutional neural networks as a feature extraction process in classification.

Synopsis

In the following, we recall the theoretical backgrounds needed for introducing deep learning and convolutional neural networks (Section 6.1) If the reader is already familiar with those notion, it is possible to go directly to 6.2. The transfer learning paradigm and the proposed approach are detailed in Section 6.2. Finally, the underwater acoustic dataset presented in Chapter 4 is used to compare the various feature extraction schemes. Results are reported in Section 6.3.

Strictly speaking, this study is not necessary for the overall comprehension of this manuscript. The main issue and the proposed solutions can be understood without reviewing this last chapter. This chapter is an investigation of the relevance of using alternative tools

(namely transfer learning and deep learning tools): a work hypothesis that was set in Chapter 3 is challenged and we here propose an alternative path for the automatic processing schemes. The chapter is relatively independent from the rest of this manuscript. We would nevertheless advice the reader to have a good overall idea of this manuscript, and to have read the theoretical chapters (Chapters 2 and 3) to better understand the study that we here propose. For this study, a number of theoretical elements that were not necessary for the rest of the manuscript need to be introduced. By nature, this chapter is therefore both theoretical and applicative. Similarly to Chapter 2, we popularize and illustrate the theoretical concepts on simple examples. We hope that this approach will help with the overall comprehension of this study. This chapter can be read as a first introduction on deep learning tools. Thank you to Omar Mohammed for his collaboration on this study.

Notations used in this chapters are specific to deep learning and convolutional neural network tools, and can overlap some previous conventions of notation used throughout this manuscript. Concerned notations are redefined, and are in accordance with the classic notations used for deep learning tools.

6.1 Background on Convolutional Neural Network

A convolutional neural network is a neural network architecture that is used both (i) to learn (and extract) features from a dataset and then (ii) to classify the data. It is based on the deep learning paradigm [GBC16]. An incredible amount of resources can be found on the subject, either as academic material or as online classes or tutorial. This section is a first illustrated introduction on the subject, and for a deeper approach the reader can refer to [GBC16] for instance.

Neural Networks (NN) are a set of learning algorithms. Each NN is based on small units called neurons or perceptrons, that are organized in layers. At each layer, the output y of a neuron is computed as a linear combination of the outputs of the neurons from the previous layer (a bias can be added). A non linear function f is also applied:

$$y = f\left(\sum weight.neuron + bias\right)$$

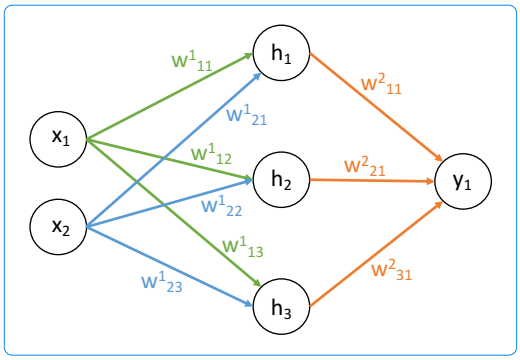
This layer is known as a *dense* layer and several stacked dense layers form a network *architecture* known as a Multi-Layer Perceptron (MLP) and is illustrated in Figure 34. In this figure, the input is of dimension 2 x_1 and x_2 (on the left). Each unit of the input is used with various weights (i.e. coefficients) and in the non linear function to compute the units h_1, h_2 and h_3 of the hidden layer. Theoretically, $h_i = f(x_1 w^1_{i,1} + x_2 w^1_{i,2} + b^1_i)$ for $i \in \{1, 2, 3\}$. Similarly, each unit of the hidden layer is used with various weights and in a non linear function to compute the output y_1, here represented of dimension 1: $y_i = f(h_1 w^2_{1,i} + h_2 w^2_{2,i} + h_3 w^2_{3,i} + b^2_i)$ for $i = 1$. A condensed representation can be used with vector notations: the input $\mathbf{x}$ is linked to the hidden layer $\mathbf{h}$ and to the output $\mathbf{y}$: $\mathbf{y} = f(\mathbf{W}^\intercal \mathbf{h})$. The weights of a network are defined and optimized[1] during the training stage. The number of layers, numbers of neurons at each layer and the choice of the non linear functions define the network *structure* and are considered as hyper parameters of the algorithm. NN is a very generic term, and different NN architectures are used to answer very different issues. Typically, NN can be used for supervised or unsupervised learning.

The deep learning paradigm is therefore built upon MLP and emerged with (i) the improvement of training algorithms with stochastic gradient descent, (ii) the use of GPU as training hardware and (iii) the emergence of huge datasets, and in particular ImageNet [RDS+15]. Deep Neural Networks (DNN) in particular, refer to neural networks with a large number of layers, either dense or of other architectures (typically, the convolutionnal layers that are later explained). The number of the so called hidden layers (i.e. layers between the input layer and the output layer) can typically range from a dozen to hundred in some cases [SZ14, HZRS16, HZC+17, SVI+16]. The training phase of such networks is therefore much more complex and computationally costly.

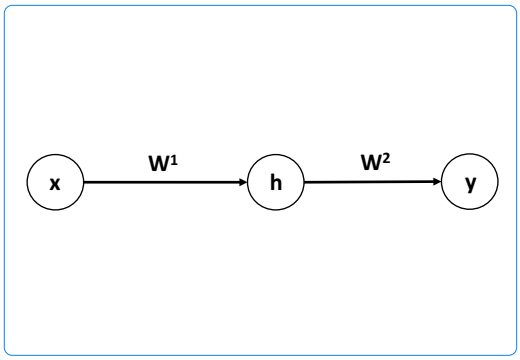
[1] In a nutshell, a cost function is defined on the network and the weights are iteratively tuned in order to minimize the cost function. For more detail on the training process of neural networks, please refer to [GBC16].

Figure 34

Schematic illustration of a MLP with the input $\boldsymbol{x}$*, one hidden layer* $\boldsymbol{h}$ *and the output* $\boldsymbol{y}$*. Each unit or neuron of a layer is linked to the neurons of the previous layer through a set of weights. The condensed view can be used with matrix notations . The nolinearity function is not represented given the increasing complexity of detailed view.*



(a) Detailed view



(b) Condensed view

Convolutional Neural Networks (CNN) refer to a neural networks architecture which is often within the scope of deep learning. We focus of this particular architecture by explaining and illustrating the mechanisms of a trained CNN. We then focus on the mechanisms of convolutional layers and finally we briefly talk about the training procedure.

A CNN is a network architecture based on convolutional layers. CNNs were originally designed to work on images as input signals [LBBH98] and part of their success is due to the emergence of ImageNet database and its 14,000,000 images [RDS+15, KSH12, SZ14]. Very large dataset are indeed requested to train a CNN. CNNs are also particularly efficient in terms of data representations (please refer to Chapter 2.1 for more information on the importance of the representation chosen for the data). They are as much features learning algorithms than classifiers [GBC16, LBBH98, KSH12, SZ14]. The architecture of a CNN is illustrated in Figure 35. An often large number of convolutional layers C_i are considered, followed by a small number of denses layers D_i (two denses layers are represented in the illustration). The first layers are known as bottom layers, the last ones as top layers. Convolutional and dense layers do not have the same purpose: the convolutional layers transform the input images into features (process hereafter detailed) which are then classified by the dense layers. Once again, the number of convolutional layers, number of dense layers, dimension of each layer and non linearity functions define the network structure and are hyper parameters of the CNN architecture.

A dense layer can be studied and explained from the scale of a unit or neuron. A convolutional layer is a dense layer with some constraints, and is more easily explained from the scale of the layer. Technically speaking, the convolutional layer implements weights sharing, meaning that the blue weights and green weights from Figure 34 are identical: $w^1_{11} = w^1_{21}$, $w^1_{12} = w^1_{22}$ and $w^1_{13} = w^1_{23}$. Many of those weights are also set to 0. The computational cost of training a convolutional layer is therefore drastically reduced compared to a dense layer (less parameters have to be estimated). Conceptually speaking, the output of a computational layer is its input filtered by a set of filters. The filtering is computed by the convolution between the input signal and the weights which represent the filter, and explains the name of *convolutional* layer[2]. This view of convolutional layers as filtering the data by given filters is a direct consequence of the weights sharing. To better understand the process, Figure 36 illustrates and details the effect of two convolutional layers on a 1D signal[3] (a time series for example).

The input is composed of background noise and given patterns, such as `A`, `B`, `AB` or `AAA`. The signal is therefore to be reconstructed from the basic elements `A` and `B`. Let us assume that those basic elements are the filters of the first layer. The output of the first layer is then made of two signals: the original signal filtered by `A` and `B`. `A` and `B` are the features at this layer. At the second layer, the process is repeated, but the input is now made of two signals. The filters (and therefore features and weights) then have a second dimension and are expressed as a combination of the previous layer features (and therefore filters and weights). Here, two filters are considered, and since the features from the first layers are very basic, the second layer features can easily be understood. The first one represents the `AB` pattern. The second one represents the `AAA` pattern. By adding a layer to the network, we consider features that are combinations of the previous layers features. The strength and representation abilities of convolutional networks lies in this point. A direct consequence however, is that finding out what is represented by a filter after a few convolutional layers is very

[2]Theoretically, the filtering is implemented using a correlation and not a convolution. But since the weights are learnt during the training of a CNN, the "mirror" operation between convolution and correlation does not change the reasoning. The deep learning community is closer to informatics that to signal processing, which perhaps can explain the misuse of language for CNN.

[3]CNNs were originally developed for images but for the sake of clarity, we here present an example on 1D signals. As explained, the various dimensions involved in a CNN increase rapidly, the concepts are more easily understood with simple schematic time series.

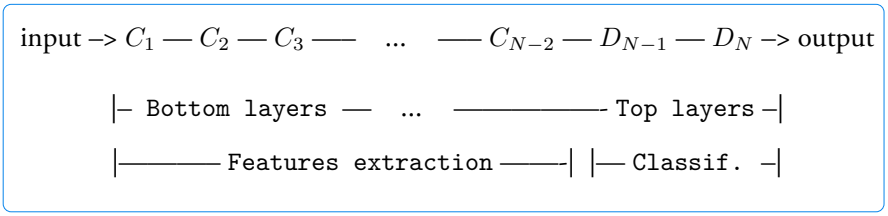


Figure 35
Schematic view of a CNN, with N_L layers including the convolutional layers C_i, with $1 \leq i \leq N_L - 2$ (features extraction) and the dense layers D_{N_L-1} and D_{N_L} (classification).

complex. This example also shows the importance of the filters - i.e. of the weights - which need to be adapted to the signal content. As previously mentioned, weights are set and optimized during the training procedure, which we now briefly expose.

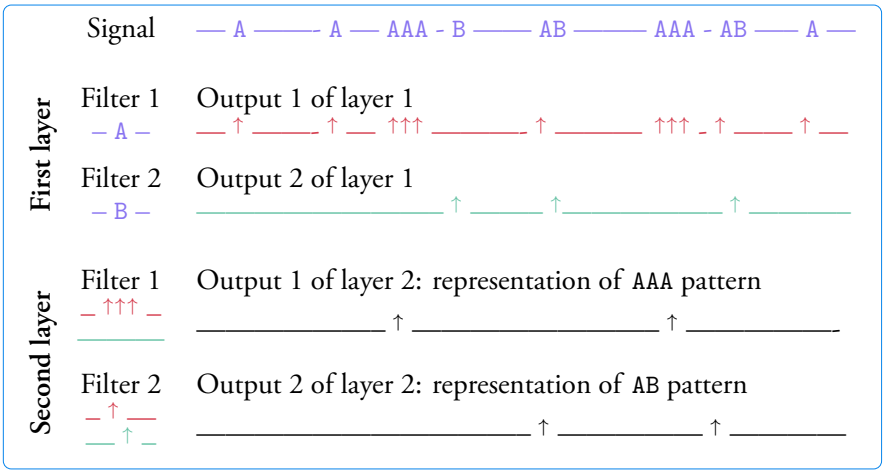


Figure 36
Schematic illustration of the effect of a CNN on a 1D signal.

The training procedure is conceptually relatively simple, but required many years and trials before CNNs reached their current success, originally with AlexNet architecture [KSH12]. GPUs and their computational power were one of the key points allowing the development and successful training of CNNs, along with the release of massive datasets and in particular of ImageNet [RDS+15]. At its simplest level, the training process of a neural network goes as follows. A cost function is defined for the network, estimating the classification errors on the training dataset. This function has the network weights as parameters. The aim of the training process is to find weights that minimize the cost function. Because of the non linear functions used in neural networks, the cost function is not convex[4]. However, convex optimization methods are used and have led to very good results (without theoretical support however). In particular, a stochastic gradient algorithm is used to iteratively update the network weights. The gradient is often estimated using the back-propagation algorithm [RHW86]. For more details on the subject, please refer to [GBC16].

CNNs that are used today have structures that are built upon the previously mentioned architecture, but with some improvements and developments. We can in particular mention pooling layers [KSH12], dropout, regularization, more complex architectures such as Inception which created "loops" in some parts of the network thereby confronting features from different layers [SVI+16], or the efforts that are made to compress networks and reduce computational costs, for example with MobileNets [HZC+17].

[4]Just in case: The question of the function convexity when trying to find a minima in high dimension is of importance. Analytics solutions are usually not possible to compute, so iterative algorithms are used to find the minimum. With convex functions (think "u-shaped" functions), iterative algorithms can be used to find the global minimum (think "bowl rowling along the side of a bowl"). With a non convex function however, the result depends on the starting point and may be a local minima.

One major limitation of CNNs relies the number of weights that need to be trained (from tens of thousands to millions), and the size of the involved dataset. During the training process of a neural networks, the weights of each unit have to be learnt from the input data. In other words, the filters used to represent the data, along with more general weights of the classification layers are learnt. When dealing with deep learning models, the dataset size is the main issue for applications where large labeled dataset are difficult to obtain, or where the dataset size is small with respect to the number of free parameters in the network. In practice, tricks such as data augmentation can be used to develop the size of a given dataset [KSH12]. A second limitation is the computational cost. Even if GPUs made the training of deep neural networks possible, the involved computational cost is much greater than when dealing with conventional machine learning approaches [GBC16].

As a conclusion on CNNs, we underline their great ability to learn representations from images. They are mainly used for image classification, but a key point in their success is their feature extraction process. Features extracted with CNN are therefore dependent on (i) the dataset (i.e. different applications lead to different features), (ii) on the network structure and (iii) on the learning process that is used. However, the constraints to train a CNN are such that in this configuration, their use is limited to huge labeled datasets. This constraint is often not realistic for environmental applications, and those tools are therefore out of reach.

In the following, we propose to use transfer learning techniques to benefit from CNNs feature extraction process without needing to train them.

6.2 Transfer Learning & Proposed Approach

CNNs require extremely large labeled datasets to be trained, it is difficult for applications beyond image processing to use them. To benefit from their feature extraction abilities, it is possible to (i) use a network trained and validated on a given application to another dataset, or (ii) to resume training of an already trained network, with the new dataset. In the first case, the top dense layer is removed (since the classification layer is specific to the training dataset) and the output features can be directly fed into a classic machine learning algorithm. The second procedure is known as fine-tuning and allows to adapt a network to a different application without the constraints on the size of the labeled dataset. In particular, the top dense layer is removed, and replaced with a new one, adjusted to the new dataset. This is done in order to define a new cost function, thereby allowing to resume the network training. Both procedures are within the scope of transfer learning: how can we use knowledge gained from training for one application to another application? Both procedures have also been proved their interest in domains were CNNs could not be trained from scratch and sometimes have led to interesting discoveries (DeepArt for instance [GEB15]). In [NPdS17], training from scratch is compared to fine tuning, and using an already fully trained network to

extract features from hyperspectral images and remote sensing purposes.

In the following study, we propose to test the first approach to extract features from environmental data and to consider several already trained networks that have shown their effectiveness on image vision. To do so, our process presented in Chapter 3 is adapted as follows:

1. The observations are represented as images using the spectrogram computation (low level representation).
2. Spectrogram images are then fed to different neural networks (removing the top layer), leading to various possible feature sets. Such features can be referred as deep features.
3. A learning algorithm is then used to train a classification model, as done previously.
4. Cross-validation on the labeled dataset is used to select the configuration leading to the best results.

This study is conducted on the underwater acoustic dataset and the conducted experiments along with their results are presented in the next section.

6.3 Classification of Underwater Acoustic Observations Using Deep Features

Three issues are raised and addressed in the following.

1. Comparison between the proposed features and deep features obtained with various deep convolutional networks,
2. Evaluation of the influence of the deep feature vectors dimension on classification results,
3. Layer selection for the deep feature vectors extraction (on the network leading to the best classification results).

All the following experiments consider the underwater acoustic datasets described in Chapter 4.Spectrograms are computed using Kaiser windows of size $n = 1024$ with on overlap of 90% between two successive windows. The fast Fourier transform is computed on $N = 1.5 * n$ (with n the length of an observation) points. The spectrogram are represented using a decibel scaling.

6.3.1 Deep Features *versus* Classic Features

In this first experiment, we compare classification results when using deep features with respect to those obtained by using the previously introduced features (see Chapter 2, 3 and 4). In particular, four different CNNs are considered to extract the deep features: `VGG16` [SZ14],

`ResNet50` [HZRS16], `MobileNet` [HZC+17] and `InceptionV3` [SVI+16]. Each one was trained on the ImageNet dataset, and the top layer was removed. Those networks are among the most effective ones among pre-trained publicly available networks. Keras library is used to extract features from CNNs and all tests are run using on a M2000 NIVIDIA Quadro GPU card with 768 cores and 4 GB of memory.

Cross-validation results for SVM (linear kernel) and RF are summarized in Table 12. Several remarks can be made on those results. First, among the four different CNNs, `ResNet50` systematically performs better than the others, with $89.5 \pm 1.15\%$ for RF and $93.9 \pm 0.97\%$ for SVM. The lowest results are obtained with `VGG16`, with $75.0 \pm 1.99\%$ for RF and $86.8 \pm 1.40\%$ for SVM. Secondly, accuracy is repeatedly lower with deep features than with the proposed features ($96.9 \pm 2.0\%$ for RF and $96.5 \pm 1.6\%$ for SVM). However, it is worth noting that the dimension of deep features much greater than the 84 proposed features. At this point, it is not possible to say if the lower results obtained with deep features are related to a less informative content or to their high dimension (due to the curse of dimensionality which could influence the results, see Chapter 2). Therefore, the issue of the feature vectors dimension and their impact on the accuracy is raised. In addition, results are systematically greater with SVM (with linear kernel) than RF. The accuracy gain when using SVM can vary from 4.4% with `ResNet50` to 11.9% with `VGG16`. It is worth noticing that the network leading to the smaller accuracy difference (`ResNet50`) is also the one with the smaller output dimension. This latter point also rises the question of the feature vectors dimension.

Before discussing the impact of the feature vectors dimension in the next part, let us notice that computation times for the deep feature extraction and for the models training in high dimension is much higher than in the previous tests. In particular, the training stage is noticeably longer for SVM (especially when the dimension is high). As an example, it took almost 50 minutes to train the SVM model on the `InceptionV3` features (dimension 131072) against 2min37 with RF. This study is on GPU, and for theoretical studies those computation times are not a problem. It might become an issue for real time processing however. Furthermore, we remind that all the previous experiments were run on CPU only, and involved much lower computation times. This remark is not essential for the results interpretation but has a clear impact when considering real time applications and deployment of the classification tools.

6.3.2 Influence of the Feature Vectors Dimension

In this second experiment, we study the impact of size of the feature vectors (i.e., number of features) have on the classification results. Two main questions are raised:

1. Is `ResNet50` leading to the best results compared to the other three networks because of its smaller dimension?
2. Is SVM performing better than RF because of the high dimension, or is SVM more adapted for this dataset?

Table 12

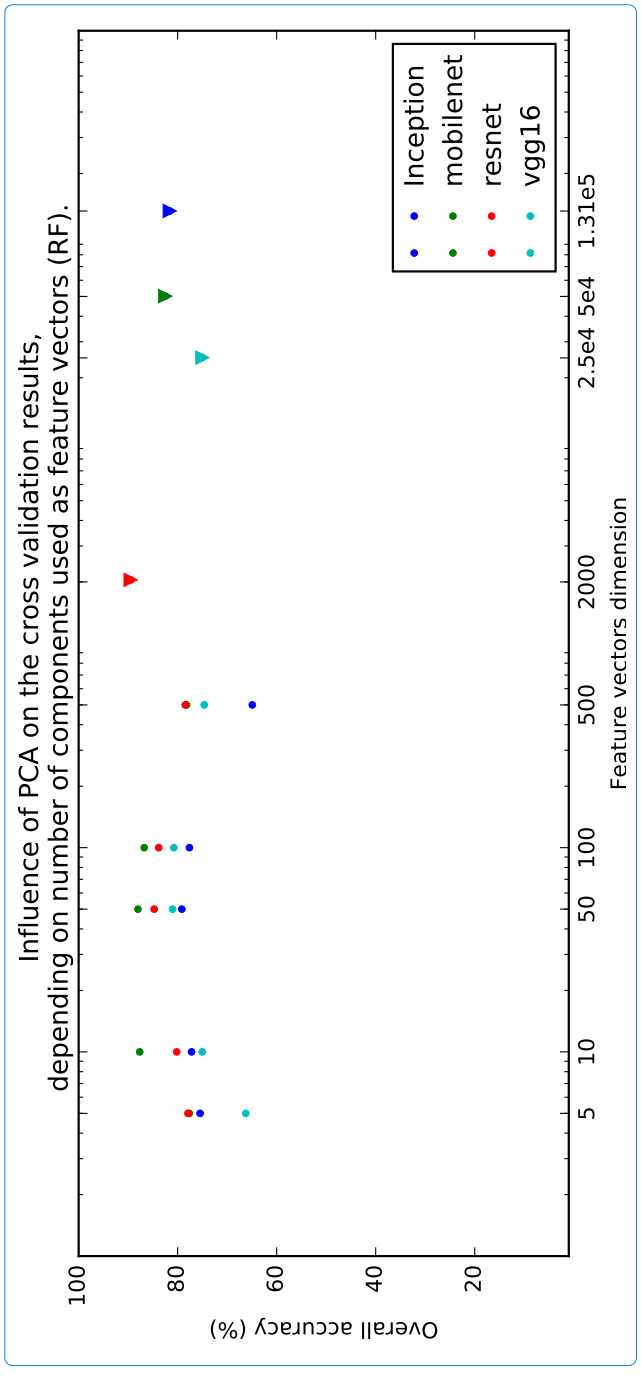
Comparison between the use of `All` features and features computed from various CNNs. Results are obtained using SVM and RF algorithms and cross-validation ($\alpha = 0.7$ on the learning set, 50 trials). The FTC (features computation time) columns indicate the time needed to extract the features from the considered networks. Learning computation times are also indicated between parenthesis.

Feature set	Dim d	FCT	X-validation	
			RF	SVM
`InceptionV3`	131072	30s	$80.6 \pm 0.93\%$ (157s)	$88.4 \pm 1.72\%$ (2876s)
`MobileNet`	50176	12s	$81.9 \pm 1.04\%$ (36s)	$91.5 \pm 1.37\%$ (1370s)
`ResNet50`	2048	23s	$89.5 \pm 1.15\%$ (23s)	$93.9 \pm 0.97\%$ (24s)
`VGG16`	25088	32s	$75.0 \pm 1.99\%$ (20s)	$86.8 \pm 1.40\%$ (755s)
`All`	84	-	$96.9 \pm 2.0\%$	$96.5 \pm 1.6\%$

To answer both questions, we use Principal Component Analysis (PCA) to compress the feature vectors output by the various CNNs. Five to 500 components are kept as the new feature vectors, and the comparison between the four CNNs is run again. Results are presented in Figure 37. The top graph presents results using RF as a classifier, and the lower one when using SVM. Accuracy values depending on the feature vectors dimensions (CNNs and PCA) are reported (color dots), and accuracy values from the previous part (CNNs without PCA) are displayed for reference (colored triangles).

`ResNet50` still performs very well with PCA (red dots), but `MobileNet` tends to have better results (green dots). The difference between those two networks is generally small with 1.85% mean increase in the accuracy from `ResNet50` to `MobileNet`, but is more pronounced when using RF compared to SVM. When using PCA, `ResNet50`, `Inception` and `MobileNet` perform better with SVM while results with `VGG16` are greater with RF.

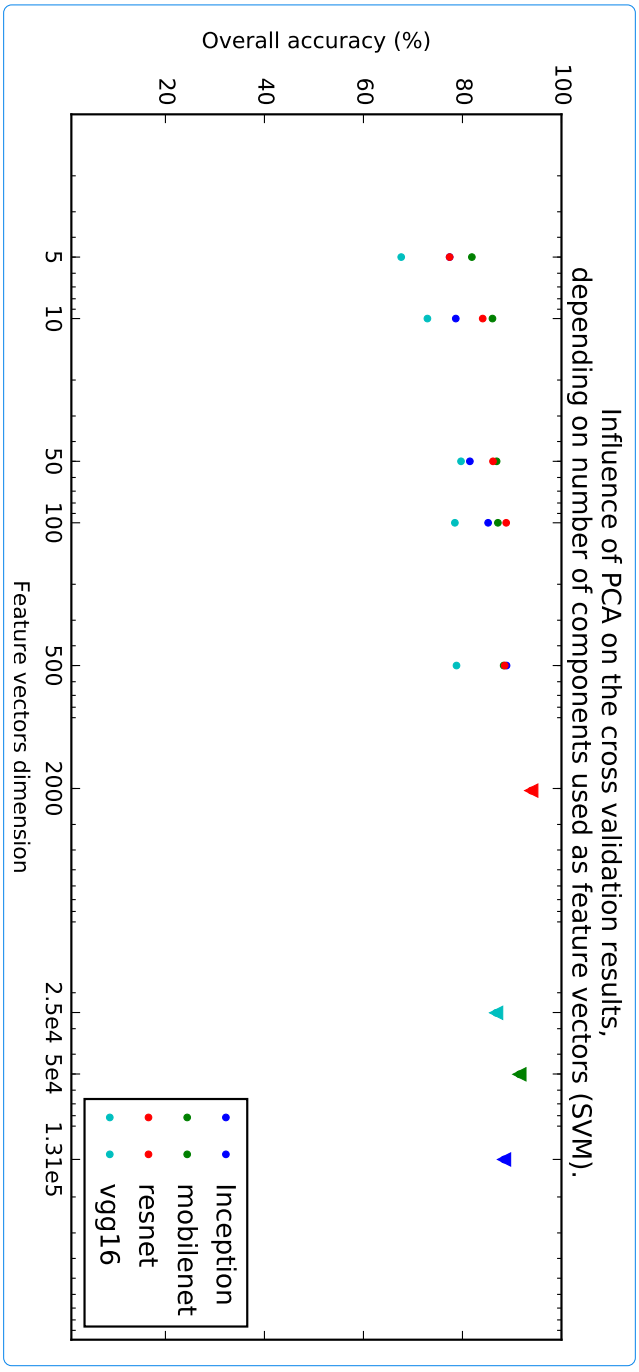
Results are in general speaking quite different from RF to SVM. The general trend for RF accuracy evolution would be to increase with the PCA dimension. A maximum is reach for feature vectors of 50 features, and accuracy then to decreases. This observation could have been related to the curse of dimensionality, but the comparison with results from the previous part rules out this hypothesis (feature vectors of larger dimension leading to greater accuracy levels). Results without PCA are equivalent (`VGG16`) or superiors (`ResNet50`, `MobileNet` and `InceptionV3`) than with PCA and 500 components. There is no clear tendency between accuracy without PCA and PCA with 50 dimensions (best results are obtained when considering RF and PCA): `ResNet50` and `Inception` perform better without PCA, `VGG16` has similar results in both configuration, while `MobileNet` performs notably better with the use of PCA. There is no clear interpretation on RF and the input data dimension, but the use of PCA with RF would not necessarily be recommended. If used, the number of principal components to keep should be considered as a hyper-parameter of the



(a)

Figure 37

Use of PCA on the deep feature vectors, with different components kept as input of the learning algorithm. RF is consider on the top graph, SVM with linear kernel on the lower graph. Triangles display results when using the full deep feature vectors (no PCA).


Influence of PCA on the cross validation results,
depending on number of components used as feature vectors (SVM).
Overall accuracy (%)
20
40
60
80
100
Feature vectors dimension
5
10
50
100
500
2000
2.5e4
5e4
1.31e5
Inception
mobilenet
resnet
vgg16

(a)

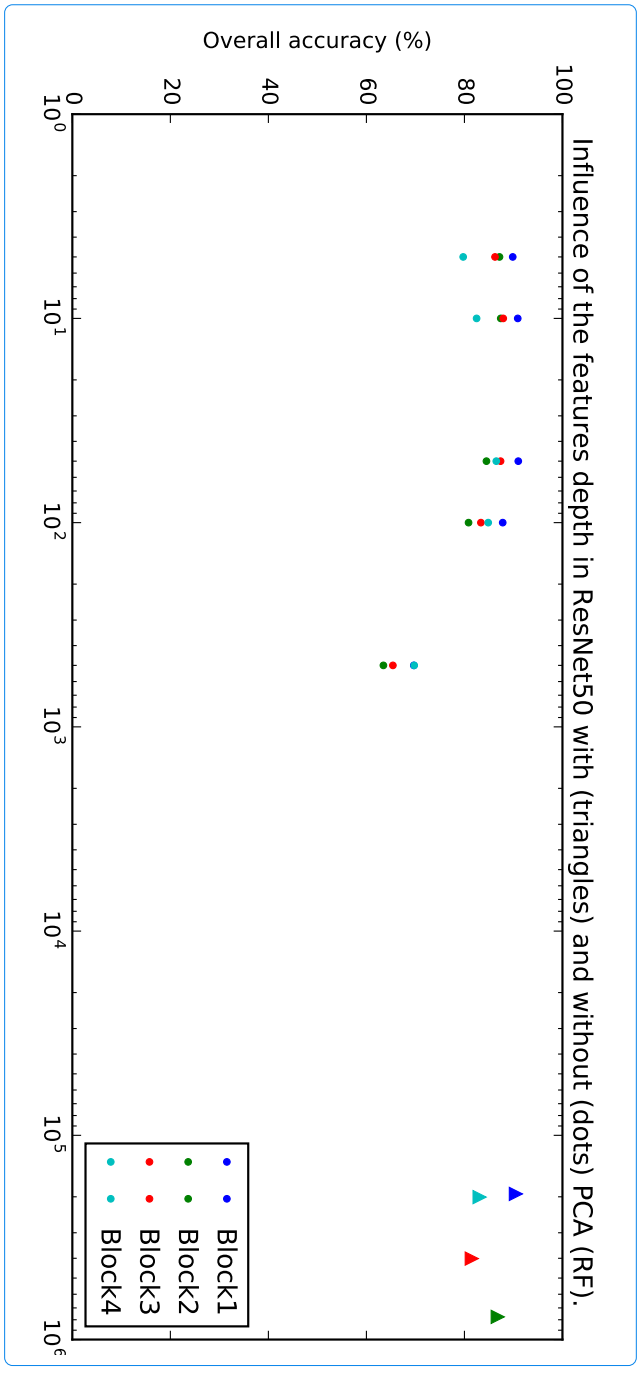
problem.

This interpretation with SVM as learning algorithm is quite different, since the general trend is toward better results with higher dimensions, whether PCA is used or not. This observation could be explained by the use of a linear kernel: the more separable the input data are, the better for classification results. With this configuration, the use of PCA which compacts the data informative content would not be recommended.

6.3.3 Layer Selection for Deep Features Extraction

As explained in the theory of CNNs, features learnt at each layer are built upon the previous layer features. Meaning that the deeper the layer is, the more complex the feature are. Some studies have also reported that the first layers of CNNs often learn the same basic features: first segments, then edges, then more complex shapes [YCBL14, DJV+14]. The shapes of interest are mainly built from vertical or horizontal segments when working with the underwater acoustic dataset. In those conditions, it might be interesting to use features from the bottom layers (simple shape features) rather than from the top layers (complex shapes and features).

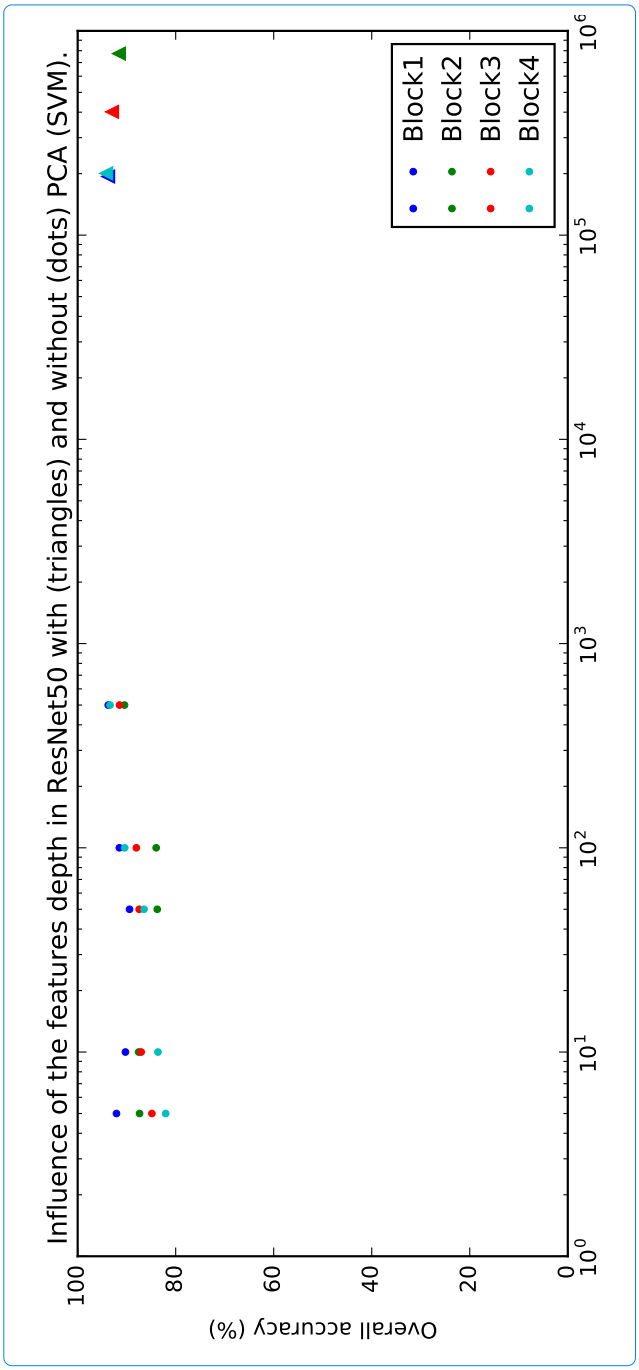
In this experiment, we compare the features extracted at different layers from `ResNet50`. In particular, `ResNet50` architecture is made of four blocks of layers, and features extracted from each of those four blocks are considered and will be referred as `block1` to `block4` feature sets. Feature set `block1` being the simplest and `block4` the most complex. Results comparing those four feature vector sets are reported in Figure 38. The figure displays accuracy levels for each of the four feature sets and put them in relation with the feature vectors dimensions. `block1` and `block4` features vectors have similar dimensions, 193.600 and 200.704 respectively. They lead to similar accuracy levels when used with SVM ($93.8 \pm 1.17\%$ and $94.2 \pm 1.14\%$, respectively) but `block1` leads to better results with RF ($90.5 \pm 1.18\%$ and $83.1 \pm 2.07\%$, respectively). The second and third feature sets perform worse in terms of classification accuracy, but also have larger dimensions, 774,400 and 401,408 respectively. To compare the effect of the different feature vectors regardless of their dimensions, the same experiment is conducted after dimension reduction with PCA. `block1` features perform better than the other feature sets regardless of the learning algorithm and the number of components extracted from the PCA. Features from the bottom layers of `ResNet50` therefore seem to be interesting for this application, even if in this configuration, the best results are obtained with `block4` features and with SVM. In this configuration, the overall accuracy is almost equal to when using the proposed and handcrafted features.



(a)

Figure 38

Layer selection performed on ResNet50. Dimension reduction of the deep features vectors is also considered with the use of PCA (50 components). Results are proposed using SVM and RF algorithms and using cross-validation process.


Influence of the features depth in ResNet50 with (triangles) and without (dots) PCA (SVM).
Overall accuracy (%)
0
20
40
60
80
100
10^0
10^1
10^2
10^3
10^4
10^5
10^6
Block1
Block2
Block3
Block4

(b)

Highlights & Summary

- In this chapter, many notions regarding deep learning are introduced. In particular, Convolutional Neural Networks (CNN) are presented: CNN are a specific architecture of Neural Network (NN), particularly efficient to (i) learn representations and (ii) classify images. The main limitation of CNN however is their training, which require very large datasets (more than is available for applications within the scope of this PhD). For this reason, CNN are not necessarily adapted for to our work conditions (i.e. relatively small dataset).
- To benefit from CNNs ability to learn representations, it is possible to use CNNs that are already trained (and are publicly available). The network is then used only to extract features, which are then fed to a classic learning algorithm to build a classification model. This strategy is used for applications where large enough labeled datasets are not available. It is part of the transfer learning theory.
- Different CNNs are therefore consider to extract features that have been learnt. The idea is to compare this approach to the use of handcrafted features (which were used in the previous chapters): does learnt features perform better or worse in term of classification of transient signals?
- The study in conducted on the fish sounds database (presented in Chapter 4). The observations are represented as spectrogram and fed to an already trained CNN to extract the feature vectors. Among `VGG16`, `ResNet50`, `MobileNet` and `InceptionV3`, `ResNet50` performs better in terms of classification accuracy. The size of the feature vectors when using one of those four CNN varies significantly. A study is therefore conducted when reducing the dimension of feature vectors to vectors of similar sizes between the four networks. The conclusions regarding the benefit of using dimensionality reductions tools are unclear (different behavior of RF and SVM classifiers), and `MobileNet` tends to better accuracy levels than `ResNet50`.
- Systematically, the handcrafted features lead to greater accuracy levels than the deep features. The difference can be important in some configurations, but not necessarily.
- Computation times are manageable when using GPUs, but clearly above the few seconds needed in the proposed configuration of Chapter 3.

Chapter 7

Conclusion

In this final chapter, we propose to summarize and step back from the manuscript: what has been done, but also what is yet to do, and what would future developments include.

Contents

7.1 Three Years in a Nutshell

This PhD work deals with the use of machine learning tools to automatically analyze natural signals, for environmental monitoring purposes. The leading idea is to take continuous recordings within the considered environment. The analysis of the physical value (temperature, acoustic, seismic, etc) that is recorded allows to better understand the environment, and therefore monitor it. Given the usually large amount of data that need to be processed, the use of automatic analysis tools are considered. To this end, this thesis proposes tools for the automatic processing of natural signals.

Very few operational tools are presented in the literature: the use of machine learning tools for environmental monitoring is being considered, but most studies target an a posteriori analysis of an environments. Datasets tend to be small, and the methods only valid under some specific constraints that do not usually match real-world conditions. The use of machine learning for a continuous environmental monitoring is not yet widespread. A specificity of the work proposed in this thesis is therefore the exploratory aspect it presents: finding the right theoretical tools, finding how to adapt them for this subject, test them, and finally use them in operative context.

From the theoretical point of view, this manuscript provides the necessary background in signal processing and machine learning (Chapter 2). The proposed tools are presented in Chapter 3: an architecture for the automatic classification of environmental observations and an architecture for the automatic detection and classification of environmental events. Both architectures rely on the feature space used to represent the observations: a set of general shape descriptors gathered from the literature in various applicative fields, and are extracted from three representations of the observations, namely in time, frequency, and quefrency (frequency of frequency). This feature extraction scheme lead to a precise description of the observations in various domains, thereby underlying complementary properties. Both architectures can also be extended to detect anomalies among the observations. They can be used in a fully automatic mode, or to suggest the classification results but leaving the final decision to an operator (use of output probabilities). This last feature is required in some applicative contexts where safety recommendations depend of the analysis results. The Python implementation of the proposed tools is known as Automatic Analysis Architecture (AAA). It is presented, and is available on GitHub.

The automatic detection and classification scheme is tested and validated for the monitoring of underwater coastal areas in Chapter 4. The soundscape is studied, and reveals numerous information on the area. We here target the fish sounds frequency band ($< 1k$Hz), where four different fish sounds have been manually identified (four positive classes), along with background noise and anomalies (two negative classes). Around 1.000 observations of 0.5 seconds have been The analysis is done in various frequency bands. The model is tested using cross-validation and leads to accuracy level above 96%. Further analysis is conducted regarding the in-

fluence of the features, the learning algorithm, and the model limitations. The model is then validated on two different areas, the one used to record the learning observations (but at a different time), and a different one. Accuracy levels exceed 93% on the same area, and 80% on the different area. An analysis of the whole set of recordings is then performed (five recordings areas, several days of measurements), and revealed day-to-day pattern.

The automatic classification scheme is tested for the monitoring of volcanoes using seismic recordings in Chapter 5. The models are validated on the volcano Ubinas (Perù): first performing cross-validation on one year of recordings (70, 856 events displayed), then by analyzing six years of recordings. The study of accuracy levels is particularly relevant: it confirmed the use and effectiveness of such methods, but also revealed a large number of inconsistencies in the original dataset. Many observations had been wrongly classified, due to a change in the volcano structure. The information was originally missed by the human analysis, and the automatic classification scheme lead to better classification results in a context of volcano-seismic crisis. After demonstrating the use of such methods, an interest for operative tools was strong, and a partnership was developed with BPPTKG observatory in Indonesia, for the monitoring of Mount Merapi. Ubinas data were used to simulate and validate the operative scheme, where a new model is regularly trained on the latest observations available. Once validates, the scheme was deployed in the observatory, and is currently under testing.

Finally, a study investigating the possibility and relevance of using features extracted using deep learning tools has been investigated in Chapter 6. In particular, convolutional neural networks are explained and used in a context of transfer learning. Convolutional neural network already trained and validated on images dataset are used of the observations represented as spectrogram images. The comparison between the proposed feature extraction scheme and the use of the so called deep feature is made. In particular, several networks are compared, and the influence of the final feature vector size is investigated (curse of dimensionality). Results are in favor of the handcrafted features, even if deep feature lead to honorable results. This final study if conducted on the fish sounds set of observations.

This manuscript proposes operational tools and exposes some use-cases of automatic environmental monitoring. The field being relatively new and very little explored however this work is a first step, which also opens many more prospects.

7.2 For Future Developments

7.2.1 About Volcano-seismic Analysis

Regarding the automatic analysis of volcano-seismic observations: this work illustrates the automatic classification task, where the observations of volcanic events have been detected, either manually, either using STA/LTA method. It would be interesting to test the automatic detection and classi-

fication scheme on the volcano-seismic recordings, and to compare results (similarly to what was done with the underwater acoustic recordings). The approach we propose normalizes the observations energy, while manual or STA/LTA detection are mainly based on the recordings energy. Detection results using the automatic scheme would likely lead to more detected events, especially of small energy. This analysis would therefore (i) analyze more observations, perhaps leading to new classes of signals with new physical interpretations, and (ii) would be independent of any other methods, the input being the continuous recordings. At the time, one difficulty for this study would be the validation phase, which would include the manual review of the detected results. Typically, some observations are not in the labeled dataset but do correspond to events. This task of course underline the importance of the expert knowledge in those analysis, but is obviously limited by the time needed to perform such results review.

7.2.2 About the Time Window and the Various Lengths of Events

Second, the analysis is today performed using a sliding window, which is a limitation when dealing with events of very various length. Typically on the fish dataset, `Roars` and `Impulse` have very different lengths. Long events such as `Roars` being considered stationary, the use of the current methods remains effective, but does not answer the counting issue. The current model cannot make the difference between a continuous call lasting on several observation windows and independent calls. A solution could be to use relatively short analysis window, and to use a time regularization on top of the classification results. Typically, Hidden Markov Models could be used. This prospect remains true when considering the volcano-seismic data.

This second prospects introduces the idea of hierarchical models, where a top model would work on the input of one or several lower models. This could be used as was explained to train simpler model on what we could call the elementary units observations, whose prediction results would in turn be used to train models of a higher abstraction level. This idea incidentally, is related to the deep learning paradigm. Hierarchical models could also be used to cover geographical variety: typically for the volcanoes monitoring using seismic recordings, a model could be trained on each recording station, and a top model could take the final classification decision, based on all the lower models outputs. In case of an eruptions, it is common that stations are damaged one after the other. A single model needed the input of all the station would therefore be useless when the first station got damaged and, therefore not be advisable. With hierarchical models, even the top models would be inefficient with the first damaged station, but the lower models would still be accessible. This configuration would also lead to more reliable results since the final classification decision would be made based on all the stations. Studies going in this direction would probably lead to very interesting results.

7.2.3 About the Model Output

Another development would be to modify the learning algorithm, in order to have non probabilistic outputs. At the moment, the output probabilities of the classes always sum to 1, leading to the limitation that two classes cannot be detected at the same time. Depending on the application, this constraint can be limiting. Typically for the fish sounds, the observation length was set to 0.5s. Consequently, a relatively large number of observations displayed a mix a several classes (often with incomplete patterns). The possibility to detect more that one class at a time would be beneficial and is worth investigating. The need for a frequency filtering and an analysis in various bandwidth in the case of underwater acoustic analysis could even be challenged. This development is particularly true for passive acoustic monitoring, but would also be relevant for volcano-seismic signals, where some observations of long duration can be overlaped with other events signatures (typically, `tremors`).

7.2.4 About the Labeling Constraint

The studies presented in this manuscript are mainly based on supervised learning. But machine learning includes many more techniques with promising developments. One of the most natural prospects is to consider the use of semi-supervised learning methods. The current biggest limitation to supervised learning is the need to have labeled dataset to train models, which is always complicated when dealing with environmental data. Natural data which are here the main interest are usually not formally processed, and few labeled datasets of natural signals are available. In the context of environmental data, the context is relatively different from machine learning approaches for computer vision or speech processing. The goal is not the development of new algorithms, but the development of effective, operational and reliable tools. Lowering the labeling constraint is therefore a major issue. To this end, semi-supervised learning methods can be considered: a large set of observations is considered, but they do not need to all be labeled [ZGL03].

Similarly, the use of transfer learning methods would be worth investigating. Chapter 4 investigated the possibility to record the training dataset on one location and to use it on observations recorded on a different area for the automatic classification of fish sounds . Results are satisfactory, but the accuracy level still dropped of 13% from the known location (learning area) to the unknown one. Some transfer learning methods where a model trained on a dataset can be adjusted to another one would be relevant to consider. This development would also be fitting for volcano-seismic applications. A majority of volcanoes are not monitored, and require years of labeled data before considering the automatic monitoring is a very heavy constraint. Being able to re-use labeled observations from another volcano could be considered. By doing so, a comparison between the various volcanoes would also be made, which would be relevant for the overall comprehension of volcanoes in general.

7.2.5 About Mapping and Unsupervised Analysis

Finally, unsupervised models could also be considered. From the various studies presented in this manuscript, we would suggest to use supervised and unsupervised models in parallel: using the supervised model to reproduce and automate the human classification, and the unsupervised model to run an unbiased[1] analysis. The studies on volcano-seismic observations revealed that the human analysis was not always relevant, which considering the studied phenomenon and the time scale at which they evolve is relatively understandable. Using both types of analysis would probably lead to different results, but would be interesting regarding the comprehension of volcanoes in general. The use of supervised and unsupervised models for parallel analysis would similarly be relevant for underwater applications where some sounds are clearly identified, but where the global environment has many unknown.

Among the unsupervised analysis, mapping techniques would be particularly interesting for monitoring applications, or to compare several applications. Typically, several maps related to different time scales (minute, hour, day for instance) could be build for the monitoring of a volcano, and each new observation would be displayed in real time on the maps. This tool could then be used to compare several unknown volcanoes, to study a unknown volcano in comparison with a known one, etc. Self-Organizing Maps could be considered as a first approach [Koh90], along with t-distributed Stochastic Neighbor Embedding (t-SNE) [MH08]. This idea could be translated to the underwater monitoring, and in general, to many applications that require a continuous analysis of the data flow, when the studied phenomenon is rather unknown.

7.2.6 Finally, About the Data[2]

We finish this manuscript on a longer term perspective by saying that generally speaking, the collection of even more data is another highly relevant development. This remark however requires the data to be well thought about. This point is perhaps a bit imprecise, but some projects have collected data without enough thinking on the basic requirements of the signals (e.g., sampling frequency), leading to a set of data that is of little use. Similarly, the big data tendency have shown that incredible amounts of data can be gathered. Once again this point is beneficial only if the data can be analyzed and related to a physical meaning. Among the future and hypothetical projects, the comparison of all volcanoes signals and behavior would be a nice challenge. It could be use for a better understanding of the volcano mechanisms, but only in real time, typically to study the relations between physically close volcanoes. On a similar idea, the analysis of the sea floor at a larger scale would be highly relevant. The presented and suggested tools could be used as exploration tools, typically to explore the very deep sea (with all the material constraints of course). Considering machine learning tools as comparison tools would lead to many other prospects where an unknown environment would be compared and positioned to a known one. Such applications would be of use in all the

[1] less biased by human knowledge at least

projects related to the exploration of an environment: either a volcano, the very deep seas, or hypothetically, environments well outside the Earth.

[2]We finish this manuscript by discussing the *data*: the need to extract information from a large amount of data is the foundation of this PhD thesis. It is lead by the data, it is about "making the data speak", and it is about great projects and great scientific questions that can be answered thanks to data. This manuscript opens with a quotation by G. Box: "*All models are wrong, but some are useful*". With the idea in mind, we go from data to models, and from models to knowledge. Knowledge comes from the data.

Appendix A

List of publications

This PhD has lead to several publications and communications in general, which are here detailed.

A.1 Scientific papers

- Marielle Malfante, Jérôme I. Mars, Mauro Dalla Mura, and Cédric Gervaise. Automatic fish sounds classification. *The Journal of the Acoustical Society of America*, 143(5):2834–2846, 2018
- Marielle Malfante, Mauro Dalla Mura, Jean-Philippe Métaxian, Jerome I. Mars, Orlando Macedo, and Adolfo Inza. Machine learning for volcano-seismic signals: Challenges and perspectives. *IEEE Signal Processing Magazine*, 35(2):20–30, 2018
- Marielle Malfante, Mauro Dalla Mura, Jerome I. Mars, Jean-Philippe Métaxian, and Orlando Macedo. Automatic Classification of Volcano Seismic Signals. *Journal of Geophysical Research*, 2018 in press

A.2 International acted conferences

- Marielle Malfante, Omar Mohammed, Cedric Gervaise, Mauro Dalla Mura, and Jerome I. Mars. Use of deep features for the automatic classification of fish sounds. In *OCEANS'18 MTS/IEEE*, Kobe, Japan, May 2018
- Marielle Malfante, Mauro Dalla Mura, Baptiste Boullay, Jean-Philippe Métaxian, and Jerome I. Mars. Apprentissage statistique: classification automatique de signaux volcano-sismiques. In *XXVIème colloque GRETSI (GRETSI 2017)*, Juan-Les-Pins, France, September 2017

- Marielle Malfante, Mauro Dalla Mura, Jerome I. Mars, and Jean-Philippe Métaxian. Machine Learning for Automatic Classification of Volcano-Seismic Signatures. In *25th European Signal Processing Conference (EUSIPCO 2017)*, European Signal Processing Conference Proceeding, pages 2457–2461, August 2017

- Marielle Malfante, Mauro Dalla Mura, Jerome Mars, Orlando Macedo, Adolfo Inza, and Jean-Philippe Métaxian. Automatic classification of seismo-volcanic signatures. In *EGU General Assembly Conference Abstracts*, volume 19, page 8842, 2017

- Marielle Malfante, Mauro Dalla Mura, Jerome I Mars, and Cedric Gervaise. Automatic fish sounds classification. volume 139, pages 2115–2116. ASA, 2016, which lead to the best paper award. The presentation was also selected for a lay language paper available at http://acoustics.org/3pab4-automatic-classification-of-fish-sounds-for-environmental-purposes-marielle-malfante/.

A.3 Non acted conferences and informal communications

A.3.1 Talks

- Marielle Malfante. Machine Learning for the Automatic Classification of Natural Signals: Application to Underwater Acoustics & Volcano-seismics. Bandung and BPPTKG Yogyakarta (Indonesia), 2017

- Marielle Malfante. Machine Learning for the Automatic Classification of Natural Signals: Application to Underwater Acoustics & Volcano-seismics. Institut de Physique du Globe de Paris (IPGP), 2017

- Marielle Malfante, Mauro Dalla Mura, Jerome I Mars, and Cedric Gervaise. Machine Learning & Bioacoustics: automatic detection & classification of fish sounds', Serenade, France (Brest), 2016

A.3.2 Tutorials

- Marielle Malfante. Machine Learning: Introduction on general concepts & focus on supervised algorithms. Training day for the participants of a course organized by BEST (Board of European Students of Technology), Grenoble, 2017

- Marielle Malfante. Supervised Machine Learning & Applications to the Automatic Classification of Fish Sounds, In Workshop Chorus entitled *Signal Processing for Passive Acoustic Monitoring*, France (Grenoble), 2016

A.3.3 Poster presentation

- Marielle Malfante. Automatic Classification & Detection of Fish Sounds. In Deep Learning Summer School, Canada (Montreal), 2016
- Marielle Malfante. Automatic Classification of Underwater Acoustics Data: Application to Fish Sounds. In Machine Learning Summer School, Spain (Cadiz), 2016

Appendix B

Readme of the AAA module

The following pages contain the Readme associated to the Automatic Analysis Architecture (AAA) project. The code is available under CeCILL license on GitHub. More detail on the architecture that are implemented can be found in Chapter 3. This code was used to perform the studies on underwater acoustic recordings of Chapter 4 and on volcano-seismic recording of Chapter 5.

Automatic Analysis Architecture

Welcome to this automatic classification scheme! Please carefully read the following before asking questions :)

This Automatic analysis scheme should be credited as follow:
Automatic Analysis Architecture
Marielle MALFANTE, Jérôme MARS, Mauro DALLA MURA
Univ. Grenoble Alpes, CNRS, Grenoble INP, GIPSA-lab, 38000 Grenoble, France

What is this code for?

This code answers three different purposes:

1. The automatic classification of continuous signals, stored in recording files (.wav, .sac, etc)
2. The automatic classification of discrete (sparse) events, stored as numpy.array objects.
3. The automatic classification of discrete (sparse) events, with real time conditions and data requests.

The roadmap to help you chose a usecase is simple:

- If your data are continuous recordings in which you want to detect and classify certain classes of event, go for Usecase 1.
- If your data are already detected events that you want to classify, you have one more question to answer.
    - Are you data stored in recordings ? If so, read and shape your data in numpy.array format and go for Usecase 2.
    - If you are dealing with real time data request, then go for Usecase 3.

Set up and requierements needed to run the code

This code was developed under Python 3, and needs the following libraries. Those libraries need to be previously installed.

- `numpy==1.13.3`
- `scipy==0.18.1`
- `pandas==0.19.2`
- `matplotlib==1.5.3`
- `numpy`
- `obspy==1.0.2`

- `python_speech_features==0.4`
- `sympy==1.0`
- `soundfile==0.8.1`
- `scikit-learn==0.18.1`

To install the correct version of python, along with the library, you can use miniconda environment manager.

1- Download and install miniconda: `https://conda.io/miniconda.html` NB: By installing miniconda, your `.bachrc` will be modified with the following line :

> # added by Anaconda3 4.3.0 installer
> export PATH="/home/user/anaconda3/bin:$PATH"

We suggest you replace them by

> "$PATH:/home/user/anaconda3/bin"

It will leave your computer configuration unchanged (in particular, your previous versions of python will still be used)

2- Create and activate your working environment (in a terminal session):
`conda create -n AAA python=3.6`
`source activate myEnvName`

3- Install the libraries:
`pip install --upgrade pip`
`pip install -r AAA_requierements.txt`.

4- Run the code (see next section)

5- Quit the working environment:
`source deactivate`

How to run the code?

Each usecase can be run from two different ways, depending on your preferences and what you intend to do with this code.

Option 1

Either using a bash script and path to settings files as input arguments. Bash scripts will run the appropriate Python scripts and properly save and display results. This method is 'the official one'. In your favorite terminal window, start by moving to the `automatic_processing` folder using `cd` command. Then run one of the usecase makefile as follow:

```
    bash make_usecase1.sh setting_file action verbatim
or

    bash make_usecase2.sh setting_file verbatim
or

    bash make_usecase3.sh setting_file action verbatim
```

setting_file are stored in the configuration folder and contain all settings needed to run an analysis. Depending on the usecase you have choosen, the information requested in setting_files can change. Please refer yourself to the setting_file sections for more details.

Option 2

Either by using Python playgrounds scripts, if you need a playground where to experiment. Those script are easily found, they all are called something like `PLAYGROUND_USECASE1.py` or similarly. Path to settings and input arguments are 'hard coded' at the beginning of each playground file. To run on of those scripts, simply go to the `automatic_processing` folder and run one on the following commands:

```
    python3 PLAYGROUND_something_something.py
```

Both configurations are basically going torward the same analysis but you might prefer one or the other depending on what you want to do.

More detail on the input arguments

- `setting_file` : path to the setting file. Traditionnaly, setting file are stored in the `config` folder. The next section gives details on the formatting of configuration files.
- `verbatim` :
    - 0 - quiet
    - 1 - some information regarding general steps
    - 2 - more detailed information
    - 3 - all details
- `action` :
    - training to train and save a model
    - analyzing to run the analysis
    - make_decision to make decision from the output probabilities output from the analysis
    - display if it is of interest to you.

Obviously, the various actions should be run in the order ... analysis cannot be run withour a trained model.

When running the code for a new application, a specific folder is created for all results.

Configuration files

All the settings related to a new project or a new run are indicated in a setting main setting.

It contains information regarding the project paths, the considered application, the signals preprocessing, the features used (linked to a dedicated feature configuration file), and the learning algorithms.

Extra information regarding the wanted analysis, the data to analyze and display parameters are indicated in a separate configuration file, which format depends on the usecase.

So, for each configuration, 3 configuration files are considered:

- the general setting file, contained in `config/general` folder
- the feature setting file, contained in `config/specific/features` folder
- the configuration file specific to the wanted analysis, contained in `config/specific/usecaseXX/`

Commented examples for each of the setting files are available (but keep in mind that json files do not support comments, so those files are simply there as examples.)

More info

If you still have questions, try running and exploring the code. The playground files are relatively easy to play with.

If you still have question, fell free to ask !

Contact: marielle.malfante@gmail.com

Appendix C

Automatic classification of fish sounds: a few more graphs

This annex presents all the graph of the analysis on the fish sounds dataset, presented in Chapter 4.

C.1 Temporal Evolution

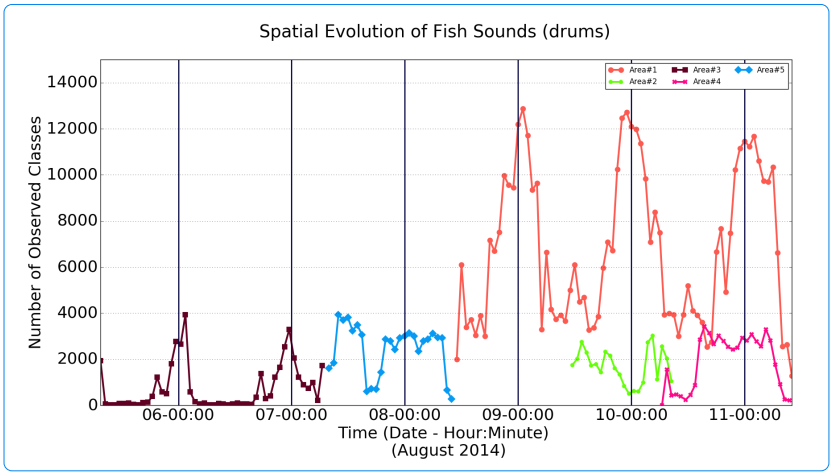
Each figures represents for a considered area the temporal evolution of the number of classified observations across the six classes.

C.2 Geographical Evolution

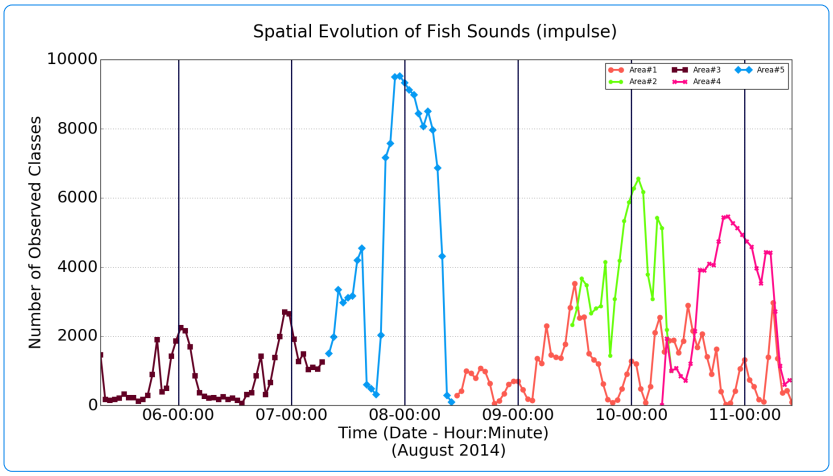
Each figure represents for a considered class the geographical evolution of the number of detected observations across the five areas.

Figure 39

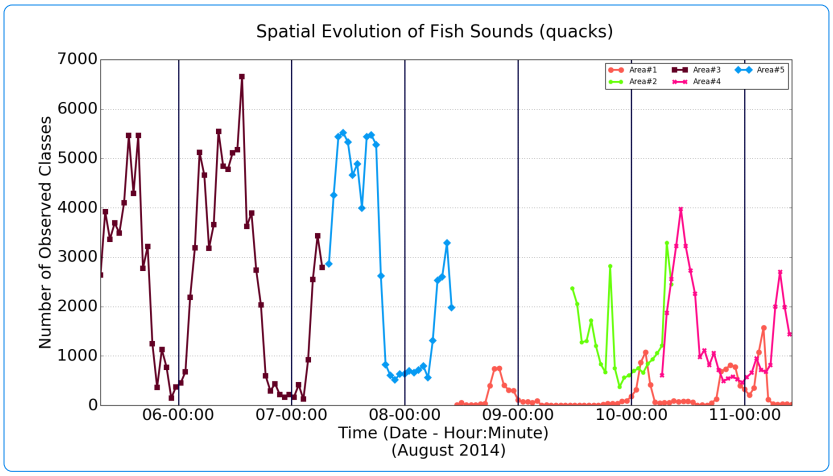
For a considered area, the temporal evolution of the number of classified observations across the six classes



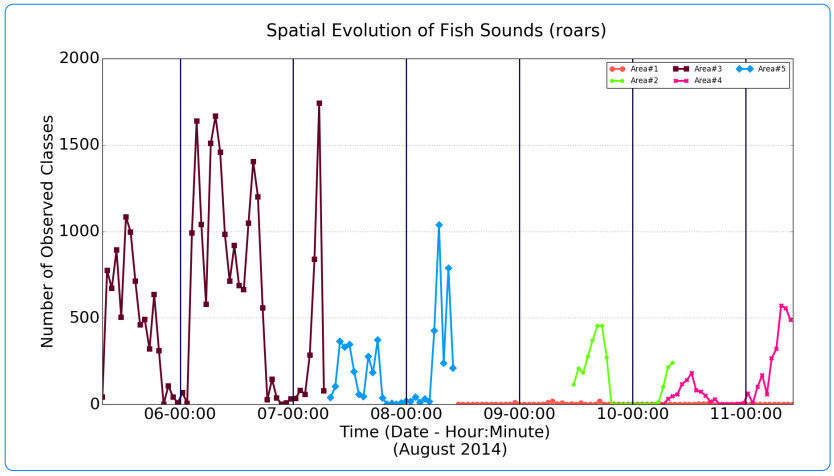
(a) Drums



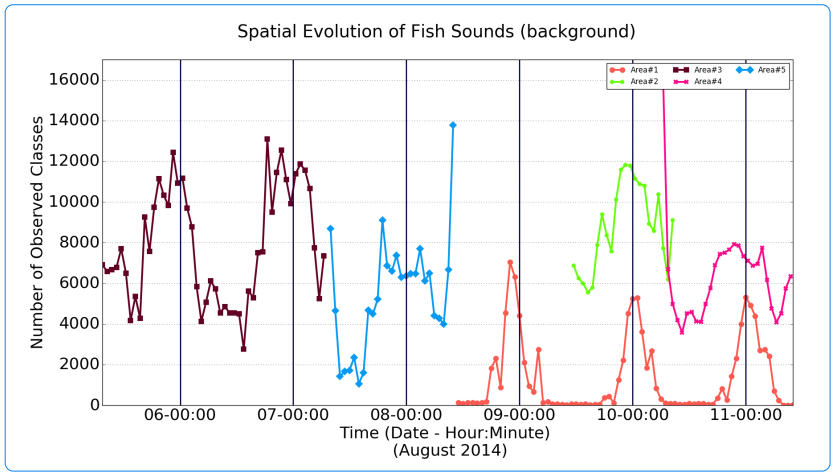
(b) Impulse



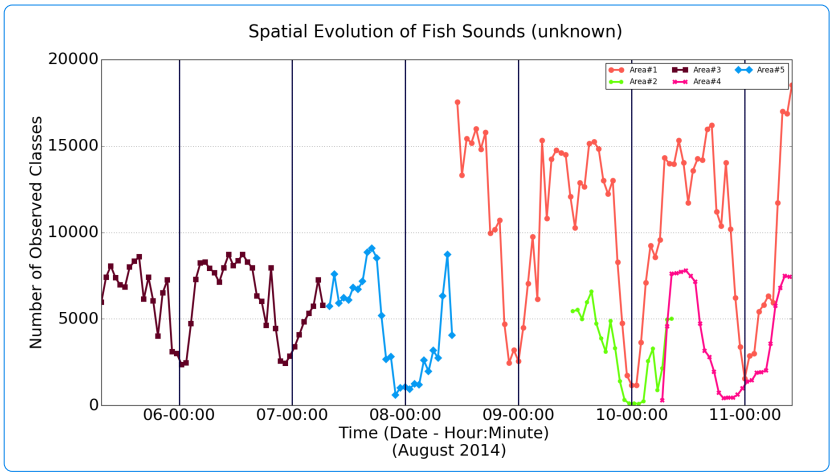
(c) Quacks



(d) Roars



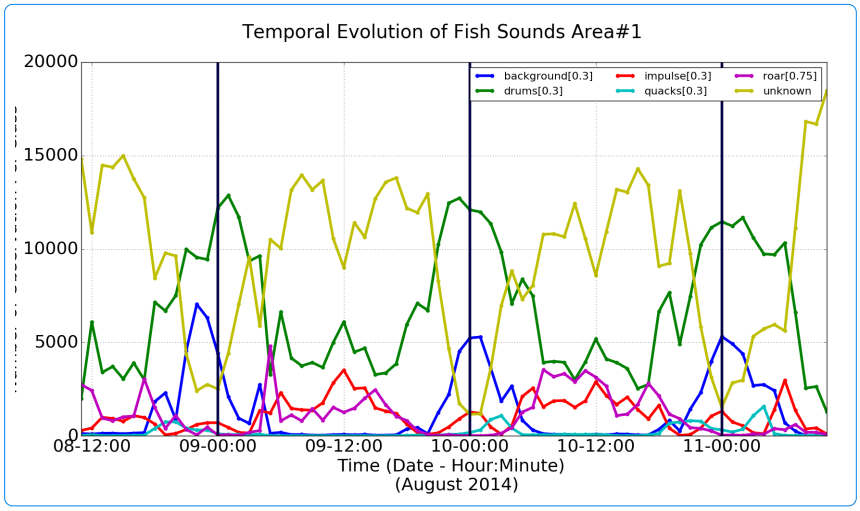
(e) Background



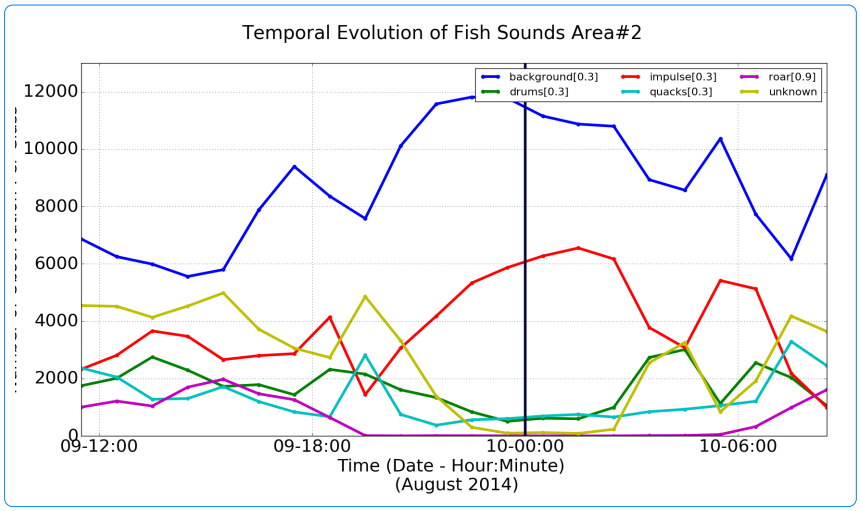
(f) Unknown

Figure 40

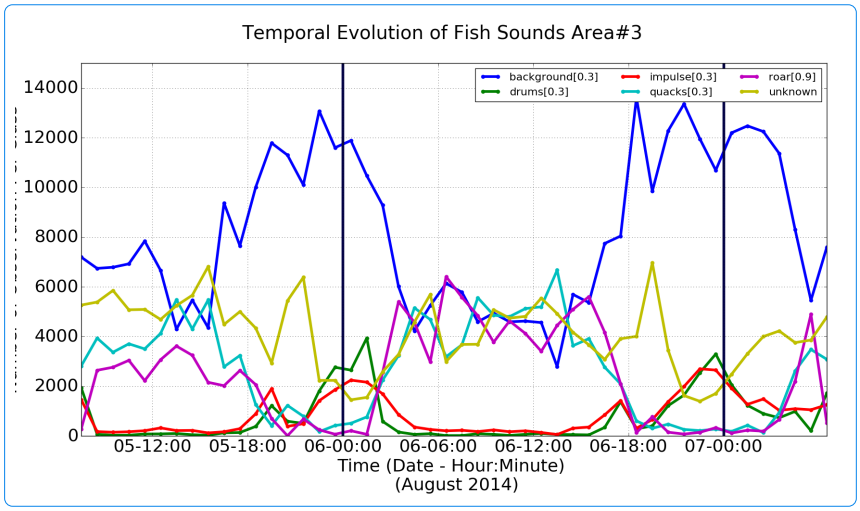
For a considered class, the geographical evolution of the number of detected observations across the five areas



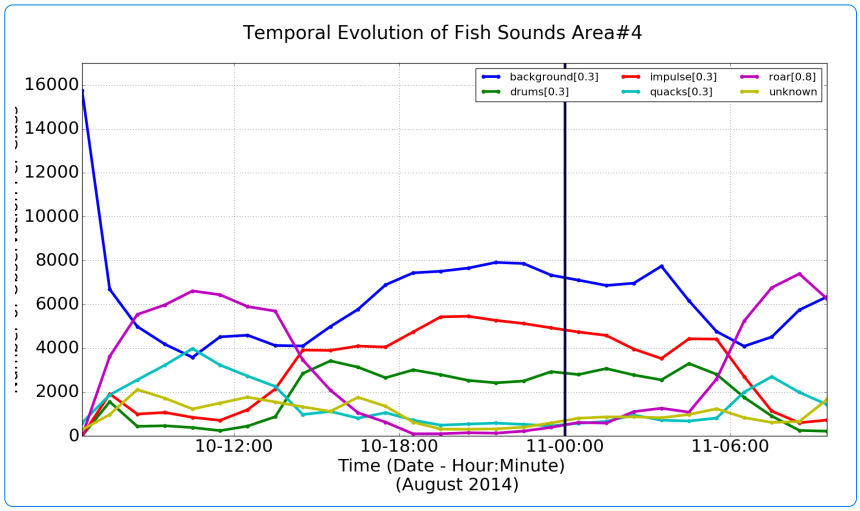
(a) Area#1



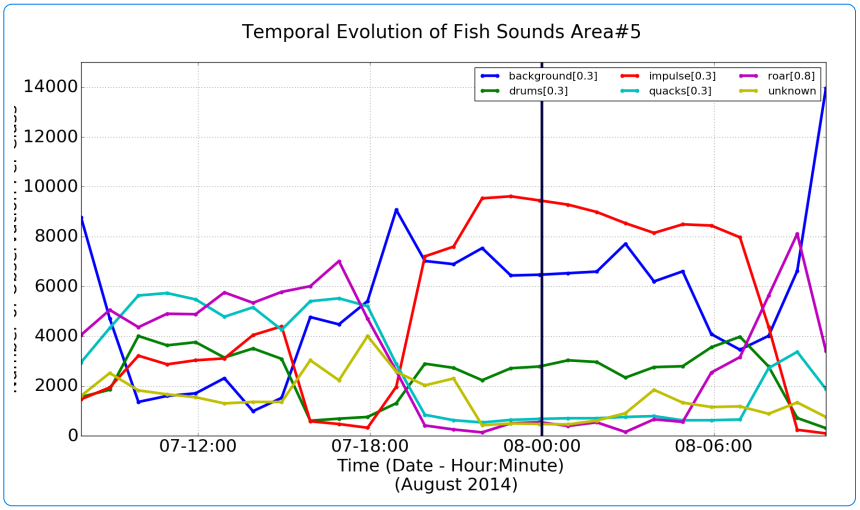
(b) Area#2



(c) Area#3



(d) Area#4



(e) Area#5

Bibliography

[ACBCB+09] Miguel A Acevedo, Carlos J Corrada-Bravo, Héctor Corrada-Bravo, Luis J Villanueva-Rivera, and T Mitchell Aide. Automated classification of bird and amphibian calls using machine learning: A comparison of methods. *Ecological Informatics*, 4(4):206–214, 2009.

[AGC+12] Isaac Alvarez, Luz Garcia, Guillermo Cortes, Carmen Benitez, and Ángel De la Torre. Discriminative feature selection for automatic classification of volcano-seismic signals. *IEEE Geoscience and Remote Sensing Letters*, 9(2):151–155, 2012.

[All78] Rex V. Allen. Automatic earthquake recognition and timing from single traces. *Bulletin of the Seismological Society of America*, 68(5):1521–1532, 1978.

[All82] Rex Allen. Automatic phase pickers: Their present use and future prospects. *Bulletin of the Seismological Society of America*, 72(6):S225–242, 1982.

[Amo06] M. Clara P. Amorim. Diversity of sound production in fish. 1:71–105, 2006.

[ASH04] Maria Clara Pessoa Amorim, Yorgos Stratoudakis, and Anthony D Hawkins. Sound production during competitive feeding in the grey gurnard. *Journal of Fish Biology*, 65(1):182–194, 2004.

[ASWdF16] Yannis M Assael, Brendan Shillingford, Shimon Whiteson, and Nando de Freitas. Lipnet: Sentence-level lipreading. *arXiv preprint arXiv:1611.01599*, 2016.

[AWO+06] Paolo Alasonati, Joachim Wassermann, Matthias Ohrnberger, H Mader, C Connor, and S Coles. Signal classification by wavelet-based hidden markov models: application to seismic signals of volcanic origin. *Statistics in Volcanology*, (1):161–174, 2006.

[BAMOA13] Manuele Bicego, Carolina Acosta-Munoz, and Mauricio Orozco-Alzate. Classification of seismic volcanic signals using hidden-markov-model-based generative embeddings. *IEEE Transactions on Geoscience and Remote Sensing*, 51(6):3400–3409, 2013.

[Bel56] Richard Bellman. Dynamic Programming and Lagrange Multipliers. *Proceedings of the National Academy of Sciences of the United States of America*, 42(10):767–769, 1956.

[BGV92] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. A Training Algorithm for Optimal Margin Classifiers. In *Proceedings of the fifth Annual Workshop on Computational Learning*, pages 144–152, 1992.

[BIDL14] Carol Bedoya, Claudia Isaza, Juan M Daza, and José D López. Automatic recognition of anuran species based on syllable identification. *Ecological Informatics*, 24:200–209, 2014.

[BJS+17] Sarah K Brown, Susanna F Jenkins, R Stephen J Sparks, Henry Odbert, and Melanie R Auker. Volcanic fatalities database: analysis of volcanic threat with distance and victim classification. *Journal of Applied Volcanology*, 6(1):15, 2017.

[BKK06] Emmanouil Benetos, Margarita Kotti, and Constantine Kotropoulos. Musical instrument classification using non-negative matrix factorization algorithms and subset feature selection. In *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, volume 5, pages V–V. IEEE, 2006.

[BP63] Thomas Bayes and Richard Price. An essay towards solving a problem in the doctrine of chances. *Philosophical Transactions (1683-1775)*, 53:370—-418, 1763.

[Bre01] Leo Breiman. Random forests. *Machine learning*, 45:5–32, 2001.

[BRS+07] M Carmen Benítez, Javier Ramírez, Jos C Segura, Jess M Ibanez, Javier Almendros, Araceli García-Yeguas, and Guillermo Cortes. Continuous hmm-based seismic-event classification at deception island, antarctica. *IEEE Transactions on Geoscience and remote sensing*, 45(1):138–146, 2007.

[BW08] Moritz Beyreuther and Joachim Wassermann. Continuous earthquake detection and classification using discrete hidden markov models. *Geophysical Journal International*, 175(3):1055–1066, 2008.

[CAG+09] Guillermo Cortés, Raúl Arámbula, LigdamisA Gutiérrez, Carmen Benítez, Jesús Ibánez, Philippe Lesage, Isaac Alvarez, and Luz Garcia. Evaluating robustness of a hmm-based classification system of volcano-seismic

events at colima and popocatepetl volcanoes. In *Geoscience and Remote Sensing Symposium, 2009 IEEE International, IGARSS 2009*, volume 2, pages II–1012. IEEE, 2009.

[CCC+17] Tim Capes, Paul Coles, Alistair Conkie, Ladan Golipour, Abie Hadjitarkhani, Qiong Hu, Nancy Huddleston, Melvyn Hunt, Jiangchuan Li, Matthias Neeracher, Kishore Prahallad, Tuomo Raitio, Ramya Rasipuram, Greg Townsend, Becci Williamson, David Winarsky, Zhizheng Wu, and Hepeng Zhang. Siri On-Device Deep Learning-Guided Unit Selection Text-to-Speech System. *Proc. Interspeech 2017*, pages 4011–4015, 2017.

[CCL+12] Wen-Ping Chen, Song-Shyong Chen, Chun-Cheng Lin, Ya-Zhung Chen, and Wen-Chih Lin. Automatic recognition of frog calls using a multi-stage average spectrum. *Computers & Mathematics with Applications*, 64(5):1270–1281, 2012.

[CddG+97] Robert Costanza, Ralph d'Arge, Rudolf de Groot, Stephen Farber, Monica Grasso, Bruce Hannon, Karin Limburg, Shahid Naeem, Robert V. O'Neill, Jose Paruelo, Robert G. Raskin, Paul Sutton, and Marjan van den Belt. The value of the world's ecosystem services and natural capital. *nature*, 387(6630):253–260, 1997.

[CFGM98] Christophe Couvreur, Vincent Fontaine, Paul Gaunard, and Corine Ginette Mubikangiey. Automatic classification of environmental noise events by hidden markov models. *Applied Acoustics*, 54(3):187–206, 1998.

[CH67] Thomas Cover and Peter Hart. Nearest neighbor pattern classification. *IEEE transactions on information theory*, 13(1):21–27, 1967.

[Che85] C. H. Chen. Automatic recognition of underwater transient signals - a review. In *ICASSP '85. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 10, pages 1270–1272. IEEE, 1985.

[Che01] E.D. Chesmore. Application of time domain signal coding and artificial neural networks to passive acoustical identification of animals. *Applied Acoustics*, 62(12):1359–1374, 2001.

[Cho86] Bernard Chouet. Dynamics of a fluid-driven crack in three dimensions by the finite difference method. *Journal of Geophysical Research: Solid Earth*, 91(B14):13967–13992, 1986.

[Cho96] Bernard A. Chouet. Long-period volcano seismicity: its source and use in eruption forecasting. *Nature*, 380(6572):309–316, Mar 1996.

[CJ06] Patrick J. Clemins and Michael T. Johnson. Generalized perceptual linear prediction features for animal vocalization analysis. *The Journal of the Acoustical Society of America*, 120(1):527–534, 2006.

[CM13] Bernard A. Chouet and Robin S. Matoza. A multi-decadal view of seismic methods for detecting precursors of magma movement and eruption. *Journal of Volcanology and Geothermal Research*, 252(Supplement C):108 – 175, 2013.

[CMA+11] Andrea Cannata, Placido Montalto, Marco Aliotta, Carmelo Cassisi, Alfredo Pulvirenti, Eugenio Privitera, and Domenico Patanè. Clustering and classification of infrasonic events at Mount Etna using pattern recognition techniques. 185:253–264, 2011.

[Com94] Pierre Comon. Independent component analysis, a new concept? *Signal processing*, 36(3):287–314, 1994.

[Cox58] David R. Cox. The regression analysis of binary sequences. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 215–242, 1958.

[CV95] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[CVF+09] Gloria Curilem, Jorge Vergara, Gustavo Fuentealba, Gonzalo Acuña, and Max Chacón. Classification of seismic signals at villarrica volcano (chile) using neural networks and genetic algorithms. *Journal of volcanology and geothermal research*, 180(1):1–8, 2009.

[DHS01] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification*. John Wiley. Sons Eds, 2001.

[DJV+14] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *International conference on machine learning*, pages 647–655, 2014.

[DL97] Manoranjan Dash and Huan Liu. Feature selection for classification. *Intelligent data analysis*, 1(3):131–156, 1997.

[DPEG+03] Edoardo Del Pezzo, Anna Esposito, Flora Giudicepietro, Maria Marinaro, Marcello Martini, and Silvia Scarpetta. Discrimination of earthquakes and underwater explosions using neural networks. *Bulletin of the Seismological Society of America*, 93(1):215–223, 2003.

[DSMM+00] Manuel E. Dos Santos, Teresa Modesto, Ricardo J. Matos, Matthew S. Grober, Rui F. Oliveira, and Adelino Canario. Sound production by the lusitanian toad fish, halobatrachus didactylus. *Bioacoustics*, 10(4):309–321, 2000.

[DSN01] Hrishikesh Deshpande, Rohit Singh, and Unjung Nam. Classification of music signals in the visual domain. In *Proceedings of the COST-G6 Conference on Digital Audio Effects*, pages 1–4, 2001.

[DTL11] Jonathan Dennis, Huy Dat Tran, and Haizhou Li. Spectrogram image feature for sound event classification in mismatched conditions. *IEEE signal processing letters*, 18(2):130–133, 2011.

[DTZ+13] Xueyan Dong, Michael Towsey, Jinglan Zhang, Jasmine Banks, and Paul Roe. A novel representation of bioacoustic events for content-based search in field audio data. In *Digital Image Computing: Techniques and Applications (DICTA), 2013 International Conference on*, pages 1–6. IEEE, 2013.

[DYB+02] Timothy H. Druitt, S. R. Young, Brian Baptie, Costanza Bonadonna, Eliza S. Calder, A. B. Clarke, Paul D. Cole, Chloe L. Harford, Richard A. Herd, Richard Luckett, Graham Ryan, and Barry Voight. Episodes of cyclic vulcanian explosive activity with fountain collapse at soufrière hills volcano, montserrat. *Geological Society, London, Memoirs*, 21(1):281–306, 2002.

[EGD+08] Antonietta M. Esposito, Flora Giudicepietro, Luca D'Auria, Silvia Scarpetta, Marcello G. Martini, Mauro Coltelli, and Maria Marinaro. Unsupervised neural analysis of very-long-period events at stromboli volcano using the self-organizing maps. *Bulletin of the Seismological Society of America*, 98(5):2449–2459, 2008.

[EK00] Antti Eronen and Anssi Klapuri. Musical instrument recognition using cepstral coefficients and temporal features. In *Acoustics, Speech, and Signal Processing, 2000. ICASSP'00. Proceedings. 2000 IEEE International Conference on*, volume 2, pages II753–II756. IEEE, 2000.

[EKR04] Shahrzad Esmaili, Sridhar Krishnan, and Kaamran Raahemifar. Content based audio classification and retrieval using joint time-frequency analysis. In *ICASSP-2004, IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 5, pages 9–12, 2004.

[EZE14] Mahdi Esfahanian, Hanqi Zhuang, and Nurgun Erdol. Sparse representation for classification of dolphin whistles by type. *The Journal of the Acoustical Society of America*, 136(1):EL1–EL7, 2014.

[Fag07] Seppo Fagerlund. Bird Species Recognition Using Support Vector Machines. *EURASIP Journal on Advances in Signal Processing*, 2007(1), 2007.

[FGNS96] Susanna Falsaperla, Salvatore Graziani, Giusepp Nunnari, and Salvatore Spampinato. Automatic classification of volcanic earthquakes by using multi-layered neural networks. *Natural Hazards*, 13(3):205–228, 1996.

[Fis36] Ronald A Fisher. The use of multiple measurements in taxonomic problems. *Annals of human genetics*, 7(2):179–188, 1936.

[FM00] Ichiro Fujinaga and Karl MacMillan. Realtime recognition of orchestral instruments. In *Proceedings of the International Computer Music Conference (ICMC2000)*, volume 141, pages 141–143, 2000.

[Foo97] Jonathan Foote. A similarity measure for automatic audio classification. In *Proc. AAAI 1997 Spring Symposium on Intelligent Integration and Use of Text, Image, Video, and Audio Corpora*, 1997.

[Gau09] Carl Friedrich Gauss. *Theoria motus corporum coelestium in sectionibus conicis solem ambientium auctore Carolo Friderico Gauss*. sumtibus Frid. Perthes et IH Besser, 1809.

[GBC16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[GEB15] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*, 2015.

[GES+09] Ferdinando Giacco, Antonietta Maria Esposito, Silvia Scarpetta, Flora Giudicepietro, and Maria Marinaro. Support vector machines and mlp for automatic classification of seismic signals at stromboli volcano. In *Proceedings of the 2009 Conference on Neural Nets WIRN09: Proceedings of the 19th Italian Workshop on Neural Nets, Vietri Sul Mare, Salerno, Italy, May 28–30 2009*, pages 116–123, 2009.

[GFK05] Todor Ganchev, Nikos Fakotakis, and George Kokkinakis. Comparative evaluation of various mfcc implementations on the speaker verification task. In *Proceedings of the SPECOM*, volume 1, pages 191–194, 2005.

[GIC+09] Ligdamis Gutiérrez, Jesús Ibañez, Guillermo Cortés, Javier Ramírez, Carmen Benítez, Virginia Tenorio, and Álvarez Isaac. Volcano-seismic signal detection and classification processing using hidden markov models. application to san cristóbal volcano, nicaragua. In *Geoscience and Remote Sensing Symposium, 2009 IEEE International, IGARSS 2009*, volume 4, pages IV–522. IEEE, 2009.

[GL03] Guodong Guo and Stan Z Li. Content-based audio classification and retrieval by support vector machines. *IEEE transactions on Neural Networks*, 14(1):209–215, 2003.

[GR70] Gene H Golub and Christian Reinsch. Singular value decomposition and least squares solutions. *Numerische mathematik*, 14(5):403–420, 1970.

[HB07] Rebecca M. Harrington and Emily E. Brodsky. Volcanic hybrid earthquakes that are brittle-failure events. *Geophysical Research Letters*, 34(6), 2007.

[HBO12] Conny Hammer, Moritz Beyreuther, and Matthias Ohrnberger. A seismic-event spotting system for volcano fast-response systems. *Bulletin of the Seismological Society of America*, 102(3):948–960, 2012.

[HHO03] Ken Jr Heck, Cynthia Hays, and Robert Orth. Critical evaluation of the nursery role hypothesis for seagrass meadows. *Marine Ecology Progress Series*, 253:123–136, 2003.

[HMD11] Ng Chee Han, Sithi V. Muniandy, and Jedol Dayou. Acoustic classification of australian anurans based on hybrid spectral-entropy approach. *Applied Acoustics*, 72(9):639–645, 2011.

[HMG+14] Clément Hibert, Anne Mangeney, Gilles Grandjean, Christian Baillard, Diane Rivet, Nikolai M. Shapiro, Claudio Satriano, Alessia Maggi, Patrice Boissier, Valérie Ferrazzini, and Wayne Crawford. Automated identification, location, and volume estimation of rockfalls at piton de la fournaise volcano. *Journal of Geophysical Research: Earth Surface*, 119(5):1082–1105, 2014.

[Ho95] Tin Kam Ho. Random decision forests. In *Document analysis and recognition, 1995., proceedings of the third international conference on*, volume 1, pages 278–282. IEEE, 1995.

[HPM+16] Clément Hibert, Floriane Provost, Jean-Philippe Malet, André Stumpf, Alessia Maggi, and Valérie Ferrazzini. Automated classification of seismic sources in large

database using random forest algorithm: First results at piton de la fournaise volcano (la réunion). In *EGU General Assembly Conference Abstracts*, volume 18, page 12895, 2016.

[HPM+17] Clément Hibert, Floriane Provost, Jean-Philippe Malet, Alessia Maggi, André Stumpf, and Valérie Ferrazzini. Automatic identification of rockfalls and volcano-tectonic earthquakes at the piton de la fournaise volcano using a random forest algorithm. *Journal of Volcanology and Geothermal Research*, 340:130–142, 2017.

[Hsu99] Feng-hsiung Hsu. Ibm's deep blue chess grandmaster chips. *IEEE Micro*, 19(2):70–81, 1999.

[HTF09] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. The Elements of Statistical Learning. *Elements*, 1:337–387, 2009.

[HYYC09] Chenn-Jung Huang, Yi-Ju Yang, Dian-Xiu Yang, and You-Jia Chen. Frog classification using machine learning techniques. *Expert Systems with Applications*, 36(2):3737–3743, 2009.

[HZC+17] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv preprint arXiv:1704.04861*, 2017.

[HZRS16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[IBG+09] Jesús M Ibáñez, Carmen Benítez, Ligdamis A Gutiérrez, Guillermo Cortés, Araceli García-Yeguas, and Gerardo Alguacil. The classification of seismo-volcanic signals using hidden markov models as applied to the stromboli and etna volcanoes. *Journal of Volcanology and Geothermal Research*, 187(3-4):218–226, 2009.

[IMM+11] Lamberto A. Inza, Jérome I. Mars, Jean-Philippe Métaxian, Gareth S. O'Brien, and Orlando Macedo. Seismo-volcano source localization with triaxial broadband seismic array. *Geophysical Journal International*, 187(1):371–384, 2011.

[IMM+14] Lamberto A. Inza, Jean-Philippe Métaxian, Jérome I. Mars, Christopher J. Bean, Gareth S. O'Brien, Orlando Macedo, and Daria Zandomeneghi. Analysis of dynamics of vulcanian activity of ubinas volcano, using multicomponent seismic antennas. *Journal of Volcanology and Geothermal Research*, 270:35–52, 2014.

[IvS08] Malte Ibs-von Seht. Detection and identification of seismic signals recorded at krakatau volcano (indonesia) using artificial neural networks. *Journal of Volcanology and Geothermal Research*, 176(4):448–456, 2008.

[IYT+08] Masato Iguchi, Hiroshi Yakiwara, Takeshi Tameguri, Muhamad Hendrasto, and Jun ichi Hirabayashi. Mechanism of explosive eruption revealed by geophysical observations at the sakurajima, suwanosejima and semeru volcanoes. *Journal of Volcanology and Geothermal Research*, 178(1):1 – 9, 2008. Dynamics of Volcanic Explosions: Field Observations, Experimental Constraints and Integrated Modelling of Volcanic Explosions: Field Observations, Experimental Constraints and Integrated Modelling.

[KCP94] Amlan Kundu, George C Chen, and Charles E Persons. Transient sonar signal classification using hidden markov models and neural nets. *IEEE Journal of Oceanic Engineering*, 19(1):87–99, 1994.

[KFF15] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3128–3137, 2015.

[KM98] Joseph A Kogan and Daniel Margoliash. Automated recognition of bird song elements from continuous recordings using dynamic time warping and hidden markov models: A comparative study. *The Journal of the Acoustical Society of America*, 103(4):2185–2196, 1998.

[Koh90] Teuvo Kohonen. The self-organizing map. *Proceedings of the IEEE*, 78(9):1464–1480, 1990.

[KOS10] Andreas Köhler, Matthias Ohrnberger, and Frank Scherbaum. Unsupervised pattern recognition in continuous seismic wavefield records using self-organizing maps. *Geophysical Journal International*, 182(3):1619–1630, 2010.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[KT14] Harold W Kuhn and Albert W Tucker. Nonlinear programming. In *Traces and emergence of nonlinear programming*, pages 247–258. Springer, 2014.

[Lan94] Pat Langley. Selection of relevant features in machine learning. In *Proceedings of the AAAI Fall symposium on relevance*, volume 184, pages 245–271, 1994.

[Lan14] Nadège Langet. *Détection et caractérisation massives de phénomènes sismologiques pour la surveillance d ' événements traditionnels et la recherche systématique de phénomènes rares*. PhD thesis, University of Strasbourg, 2014.

[Lap20] Pierre Simon Laplace. *Théorie analytique des probabilités*. Courcier, 1820.

[LBBH98] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[LBHL07] Taegyun Lim, Keunsung Bae, Chansik Hwang, and Hyeonguk Lee. Classification of underwater transient signals using mfcc feature vector. In *Signal Processing and Its Applications, 2007. ISSPA 2007. 9th International Symposium on*, pages 1–4. IEEE, 2007.

[LCHH06] Chang-Hsing Lee, Chih-Hsun Chou, Chin-Chuan Han, and Ren-Zhuang Huang. Automatic recognition of animal vocalizations using averaged mfcc and linear discriminant analysis. *Pattern Recognition Letters*, 27(2):93–101, 2006.

[LCMLB16] Román A Lara-Cueva, Andrés Sebastián Moreno, Julio C Larco, and Diego S Benítez. Real-time seismic event detection using voice activity detection techniques. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 9(12):5533–5542, 2016.

[LCS+94] John C. Lahr, Bernard A Chouet, Christopher D. Stephens, John A. Power, and Robert A. Page. Earthquake classification, location, and error analysis in a volcanic environment: Implications for the magmatic system of the 1989–1990 eruptions at redoubt volcano, alaska. *Journal of Volcanology and Geothermal Research*, 62(1-4):137–151, 1994.

[Leg05] Adrien Marie Legendre. *Nouvelles méthodes pour la détermination des orbites des comètes*. F. Didot, 1805.

[LFM+09] Horst Langer, Susanna Falsaperla, Matteo Masotti, Renato Campanini, Salvatore Spampinato, and Alfio Messina. Synopsis of supervised and unsupervised pattern classification techniques applied to volcanic tremor data at mt etna, italy. *Geophysical Journal International*, 178(2):1132–1144, 2009.

[LFPT06] Horst Langer, Susanna Falsaperla, Tanya Powell, and Glenn Thompson. Automatic classification and a-posteriori analysis of seismic event identification at Soufrière Hills volcano, Montserrat. *Journal of Volcanology and Geothermal Research*, 153(1-2 SPEC. ISS.):1–10, 2006.

[LFT03] Horst Langer, Susanna Falsaperla, and Glenn Thompson. Application of artificial neural networks for the classification of the seismic transients at soufriere hills volcano, montserrat. *Geophysical research letters*, 30(21), 2003.

[LGI15] J Lossent, C Gervaise, and L Di Iorio. Cartographie de la biophonie des écosystèmes côtiers. pages 1–5, 2015.

[LMMB14] Nadège Langet, Alessia Maggi, Alberto Michelini, and Florent Brenguier. Continuous kurtosis-based migration for seismic event detection and location, with application to Piton de la Fournaise volcano, La Réunion. *Bulletin of the Seismological Society of America*, 104(1):229–246, 2014.

[LR04] Arie Livshin and Xavier Rodet. Musical instrument identification in continuous recordings. In *Digital Audio Effects 2004*, pages 1–1, 2004.

[LS71] William H Lawton and Edward A Sylvestre. Self modeling curve resolution. *Technometrics*, 13(3):617–633, 1971.

[LW95] Rachel E Learned and Alan S Willsky. A wavelet packet approach to transient signal classification. *Applied and computational Harmonic analysis*, 2(3):265–278, 1995.

[Mal18] Marielle Malfante. Automatic Analysis Architecture (AAA), 2018. URL: `https://github.com/malfante/AAA`.

[MC97] Alex L McIlraith and Howard C Card. Birdsong recognition using backpropagation and multivariate statistics. *IEEE Transactions on Signal Processing*, 45(11):2740–2748, 1997.

[McN92] S.R. McNutt. Volcanic tremor. *Encyclopedia of Earth System Science*, 4:417–425, 1992.

[McN96] S. R. McNutt. *Seismic Monitoring and Eruption Forecasting of Volcanoes: A Review of the State-of-the-Art and Case Histories*, pages 99–146. Springer Berlin Heidelberg, Berlin, Heidelberg, 1996.

[MDMB+17] Marielle Malfante, Mauro Dalla Mura, Baptiste Boullay, Jean-Philippe Métaxian, and Jerome I. Mars. Apprentissage statistique: classification automatique

de signaux volcano-sismiques. In *XXVIème colloque GRETSI (GRETSI 2017)*, Juan-Les-Pins, France, September 2017.

[MDMM+17] Marielle Malfante, Mauro Dalla Mura, Jerome Mars, Orlando Macedo, Adolfo Inza, and Jean-Philippe Métaxian. Automatic classification of seismo-volcanic signatures. In *EGU General Assembly Conference Abstracts*, volume 19, page 8842, 2017.

[MDMM+18] Marielle Malfante, Mauro Dalla Mura, Jean-Philippe Métaxian, Jerome I. Mars, Orlando Macedo, and Adolfo Inza. Machine learning for volcano-seismic signals: Challenges and perspectives. *IEEE Signal Processing Magazine*, 35(2):20–30, 2018.

[MDMMG16] Marielle Malfante, Mauro Dalla Mura, Jerome I Mars, and Cedric Gervaise. Automatic fish sounds classification. volume 139, pages 2115–2116. ASA, 2016.

[MDMMM17] Marielle Malfante, Mauro Dalla Mura, Jerome I. Mars, and Jean-Philippe Métaxian. Machine Learning for Automatic Classification of Volcano-Seismic Signatures. In *25th European Signal Processing Conference (EUSIPCO 2017)*, European Signal Processing Conference Proceeding, pages 2457–2461, August 2017.

[MDMM+ss] Marielle Malfante, Mauro Dalla Mura, Jerome I. Mars, Jean-Philippe Métaxian, and Orlando Macedo. Automatic Classification of Volcano Seismic Signals. *Journal of Geophysical Research*, 2018 in press.

[MFH+17] Alessia Maggi, Valérie Ferrazzini, Clément Hibert, François Beauducel, Patrice Boissier, and Amandine Amemoutou. Implementation of a multistation approach for automated event classification at piton de la fournaise volcano. *Seismological Research Letters*, 88(3):878–891, 2017.

[MH08] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[MHJ08] David A. Mann, Anthony D. Hawkins, and J. Michael Jech. *Active and passive acoustics to locate and study fish*, volume 32. Springer, 2008.

[Min52] Marvin Minsky. A neural-analogue calculator based upon a probability model of reinforcement. *Harvard University Psychological Laboratories, Cambridge, Massachusetts*, 1952.

[Min85] Marvin Minski. The society of mind. *Columbia: Simon & Schuster*, 1985.

[Min91] Marvin Minsky. Society of mind: a response to four reviews. *Artificial Intelligence*, 48(3):371–396, 1991.

[Min07] Marvin Minsky. *The emotion machine: Commonsense thinking, artificial intelligence, and the future of the human mind*. Simon and Schuster, 2007.

[MMDMG18] Marielle Malfante, Jérôme I. Mars, Mauro Dalla Mura, and Cédric Gervaise. Automatic fish sounds classification. *The Journal of the Acoustical Society of America*, 143(5):2834–2846, 2018.

[MMG+18] Marielle Malfante, Omar Mohammed, Cedric Gervaise, Mauro Dalla Mura, and Jerome I. Mars. Use of deep features for the automatic classification of fish sounds. In *OCEANS'18 MTS/IEEE*, Kobe, Japan, May 2018.

[MMQ+08] Seth C. Morgan, Stephen D. Malone, Anthony I. Qamar, Weston A. Thelen, Amy K. Wright, and Jacqueline Caplan-Auerbach. Seismicity associated with renewed dome building at mount st. helens, 2004-2005: Chapter 2 in a volcano rekindled: the renewed eruption of mount st. helens, 2004-2006. Technical report, 2008. Report.

[MMSFSGSP14] Miguel Márquez-Molina, Luis Pastor Sánchez-Fernández, Sergio Suárez-Guerra, and Luis Alejandro Sánchez-Pérez. Aircraft take-off noises classification based on human auditory's matched features extraction. *Applied Acoustics*, 84:83–90, 2014.

[MMT+09] Orlando Macedo, Jean-Philippe Métaxian, Edu Taipe, Domingo Ramos, and Adolfo Inza. Seismicity associated with the 2006-2008 eruption, Ubinas volcano. pages 262–270, 2009.

[MP43] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[MZB06] Dalibor Mitrovic, Matthias Zeppelzauer, and Christian Breiteneder. Discrimination and retrieval of animal sounds. In *Multi-Media Modelling Conference Proceedings, 2006 12th International*, pages 5–pp. IEEE, 2006.

[Nie01] Yves Nievergelt. A tutorial history of least squares with applications to astronomy and geodesy. In *Numerical Analysis: Historical Developments in the 20th Century*, pages 77–112. Elsevier, 2001.

[Nil65] Nils J Nilsson. *Learning machines: Foundations of trainable pattern-classifying systems*. McGraw-Hill, 1965.

[NLBO00] J Neuberg, R Luckett, B Baptie, and K Olsen. Models of tremor and low-frequency earthquake swarms on montserrat. *Journal of Volcanology and Geothermal Research*, 101(1):83 – 104, 2000.

[NPdS17] Keiller Nogueira, Otávio AB Penatti, and Jefersson A dos Santos. Towards better exploiting convolutional neural networks for remote sensing scene classification. *Pattern Recognition*, 61:539–556, 2017.

[NTG+97] Lourdes Narváez M, Roberto a. Torres C, Diego M. Gómez M, Gloria Patricia Cortés J, Héctor Cepeda V, and John Stix. 'Tornillo'-type seismic signals at Galeras volcano, Colombia, 1992–1993. *Journal of Volcanology and Geothermal Research*, 77(1-4):159–171, 1997.

[NTSR16] Juan J Noda, Carlos M Travieso, and David Sánchez-Rodríguez. Automatic taxonomic classification of fish based on their acoustic signals. *Applied Sciences*, 6(12):443, 2016.

[OGDC06] Mauricio Orozco, Marcelo E García, Robert PW Duin, and César G Castellanos. Dissimilarity Based Classification of Seismic Signals at Nevado del Ruiz Volcano. *Earth Sciences Research Journal*, 10(2):57–65, 2006.

[Ohm06] Takao Ohminato. Characteristics and source modeling of broadband seismic signals associated with the hydrothermal system at satsumaalwojima volcano, japan. *Journal of Volcanology and Geothermal Research*, 158(3):467 – 490, 2006.

[Ohr01] Matthias Ohrnberger. Continuous automatic classification of seismic signals of volcanic origin at Mt. Merapi, Java, Indonesia. 2001.

[Opp03] Clive Oppenheimer. Climatic, environmental and human consequences of the largest known historic eruption: Tambora volcano (indonesia) 1815. *Progress in physical geography*, 27(2):230–259, 2003.

[Par15] Quirino Paris. The dual of the least-squares method. *Open Journal of Statistics*, 5:658–664, 2015.

[PBG+10] Federica Pace, Frederic Benard, Herve Glotin, Olivier Adam, and Paul White. Subunit definition and analysis for humpback whale call classification. *Applied Acoustics*, 71(11):1107–1112, 2010.

[PHM17] Floriane Provost, Clément Hibert, and Jean-Philippe Malet. Automatic classification of endogenous landslide seismicity using the random forest supervised classifier. *Geophysical Research Letters*, 44(1):113–120, 2017.

[PVFF06] Eric Parmentier, Pierre Vandewalle, Bruno Frederich, and ML Fine. Sound production in two species of damselfishes (Pomacentridae): Plectroglyphidodon lacrymatus and Dascyllus aruanus. *Journal of Fish Biology*, 69(2):491–503, 2006.

[Qui86] J. Ross Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986.

[RDS+15] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[RG75] Lawrence R. Rabiner and Bernard Gold. Theory and application of digital signal processing. *Englewood Cliffs, NJ, Prentice-Hall, Inc., 1975. 777 p.*, 1975.

[RHW86] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533, 1986.

[Ros57] Frank Rosenblatt. *The perceptron, a perceiving and recognizing automaton*. Cornell Aeronautical Laboratory, 1957.

[Sam59] Arthur L Samuel. Some studies in machine learning using the game of checkers. *IBM Journal of research and development*, 3(3):210–229, 1959.

[SCSJ16] Farook Sattar, Sarika Cullis-Suzuki, and Feng Jin. Acoustic analysis of big ocean data to monitor fish sounds. *Ecological Informatics*, 34:102–107, 2016.

[SDC06] Christian Simmermacher, Da Deng, and Stephen Cranefield. Feature analysis and classification of classical musical instruments: An empirical study. In *Industrial Conference on Data Mining*, pages 444–458. Springer, 2006.

[Seb62] George S. Sebestyen. Decision-making processes in pattern recognition. 1962.

[SGE+05] Silvia Scarpetta, Flora Giudicepietro, Eugène C. Ezin, Simona Petrosino, Edoardo Del Pezzo, Marcello Martini, and Maria Marinaro. Automatic classification of seismic signals at mt. vesuvius volcano, italy, using neural networks. *Bulletin of the Seismological Society of America*, 95(1):185–196, 2005.

[SHM+16] David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

[SHP02] Christos D Saragiotis, Leontios J Hadjileontiadis, and Stavros M Panas. Pai-s/k: A robust automatic seismic p phase arrival identification scheme. *IEEE Transactions on Geoscience and Remote Sensing*, 40(6):1395–1404, 2002.

[SKB13] Austin Sousa, Rasoul Kabirzadeh, and Patrick Blaes. Automatic Colorization of Grayscale Images, 2013.

[Sti81] Stephen M Stigler. Gauss and the invention of least squares. *The Annals of Statistics*, pages 465–474, 1981.

[SVI+16] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the Inception Architecture for Computer Vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, 2016.

[SZ14] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[TB99] Michael E Tipping and Christopher M Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(3):611–622, 1999.

[TB05] Simon Tucker and Guy J. Brown. Classification of transient sonar sounds using perceptually motivated features. *IEEE Journal of Oceanic Engineering*, 30(3):588–600, 2005.

[Tes95] Gerald Tesauro. Temporal difference learning and td-gammon. *Communications of the ACM*, 38(3):58–68, 1995.

[TF02] Robert F. Thorson and Michael L. Fine. Crepuscular changes in emission rate and parameters of the boatwhistle advertisement call of the gulf toadfish, Opsanus beta. *Environmental Biology of Fishes*, 63(3):321–331, 2002.

[THMP06] Hemant Tyagi, Rajesh M Hegde, Hema A Murthy, and Anil Prabhakar. Automatic identification of bird calls using spectral ensemble average voice prints. In *Signal processing conference, 2006 14th European*, pages 1–5. IEEE, 2006.

[TKB+12] Aaron M Thode, Katherine H Kim, Susanna B Blackwell, Charles R Greene Jr, Christopher S Nations, Trent L McDonald, and A Michael Macrander. Automated detection and localization of bowhead whale sounds in the presence of seismic airgun surveys. *The Journal of the Acoustical Society of America*, 131(5):3726–3747, 2012.

[TLM+11] Paola Traversa, Olivier Lengliné, Orlando Macedo, Jean-Philippe Métaxian, Jean-Robert Grasso, Adolfo Inza, and Edú Taipe. Short term forecasting of explosions at ubinas volcano, perú. *Journal of Geophysical Research: Solid Earth*, 116(B11), 2011.

[Tur50] Alan Mathison Turing. Computing machinery and intelligence. *Journal of the Mind Association*, LIX(236):433–460, 1950.

[UI15] UN-ISDR. International Strategy for Disaster Reduction. *70th UN General Assembly*, (20c):22 December 2015, 2015.

[URJ16] Kathi Unglert, Valentina Radić, and A. Mark Jellinek. Principal component analysis vs. self-organizing maps combined with hierarchical clustering for pattern recognition in volcano seismic spectra. *Journal of Volcanology and Geothermal Research*, 320:58–74, 2016.

[VFAT15] Manuel Vieira, Paulo J Fonseca, M Clara P Amorim, and Carlos JC Teixeira. Call recognition and individual identification of fish vocalizations based on automatic speech recognition: An example with the lusitanian toadfish. *The Journal of the Acoustical Society of America*, 138(6):3941–3950, 2015.

[VL63] Vladimir Vapnik and Alexey Lener. Pattern recognition using generalized portrait method. *Automation and remote control*, 24:774–780, 1963.

[Wan03] Avery Li-Chun Wang. An industrial strength audio search algorithm. In *Ismir*, volume 2003, pages 7–13. Washington, DC, 2003.

[WMLP98] Randall A. White, Angus D. Miller, Lloyd Lynch, and John Power. Observations of hybrid seismic events at soufriere hills volcano, montserrat: July 1995 to september 1996. *Geophysical Research Letters*, 25(19):3657–3660, 1998.

[WTP+10] Jason Wimmer, Michael Towsey, Birgit Planitz, Paul Roe, and Ian Williamson. Scaling acoustic data analysis through collaboration and automation. In *e-Science (e-Science), 2010 IEEE Sixth International Conference on*, pages 308–315. IEEE, 2010.

[WX10] Wang Wei and Shan Xinjian. Dynamic fuzzy clustering for infrasound as a precursor of earthquakes. In *Image and Signal Processing (CISP), 2010 3rd International Congress on*, volume 8, pages 3582–3586. IEEE, 2010.

[WZ14] Shuguang Wang and Xiangyang Zeng. Robust underwater noise targets classification using auditory inspired time–frequency analysis. *Applied Acoustics*, 78:68–76, 2014.

[XMS05] Changsheng Xu, Namunu Chinthaka Maddage, and Xi Shao. Automatic music classification and summarization. *IEEE transactions on speech and audio processing*, 13(3):441–450, 2005.

[YCBL14] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014.

[YS09] Guoshen Yu and Jean-Jacques Slotine. Audio classification from time-frequency texture. In *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, pages 1677–1680. IEEE, 2009.

[ZGL03] Xiaojin Zhu, Zoubin Ghahramani, and John D Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the 20th International conference on Machine learning (ICML-03)*, pages 912–919, 2003.

[ZTX+12] Qianwei Zhou, Guanjun Tong, Dongfeng Xie, Baoqing Li, and Xiaobing Yuan. A seismic-based feature extraction algorithm for robust ground target classification. *IEEE Signal Processing Letters*, 19(10):639–642, 2012.

[ZVH+10] Serge Zaugg, Mike Van Der Schaar, Ludwig Houégnigan, Cédric Gervaise, and Michel André. Real-time acoustic classification of sperm whale clicks and shipping impulses from deep-sea observatories. *Applied Acoustics*, 71(11):1011–1019, 2010.

[ZZS01] Fang Zheng, Guoliang Zhang, and Zhanjiang Song. Comparison of different implementations of mfcc. *Journal of Computer Science and Technology*, 16(6):582–589, 2001.

Abstract

This manuscript summarizes a three years work addressing the use of machine learning for the automatic analysis of natural signals. The main goal of this PhD is to produce efficient and operative frameworks for the analysis of environmental signals, in order to gather knowledge and better understand the considered environment. Particularly, we focus on the automatic tasks of detection and classification of natural events. This thesis proposes two tools based on supervised machine learning (Support Vector Machine, Random Forest) for (i) the automatic classification of events and (ii) the automatic detection and classification of events. The success of the proposed approaches lies in the feature space used to represent the signals. This relies on a detailed description of the raw acquisitions in various domains: temporal, spectral and cepstral. A comparison with features extracted using convolutional neural networks (deep learning) is also made, and favours the physical features to the use of deep learning methods to represent transient signals. The proposed tools are tested and validated on real world acquisitions from different environments: (i) underwater and (ii) volcanic areas. The first application considered in this thesis is devoted to the monitoring of coastal underwater areas using acoustic signals: continuous recordings are analyzed to automatically detect and classify fish sounds. A day to day pattern in the fish behavior is revealed. The second application targets volcanoes monitoring: the proposed system classifies seismic events into categories, which can be associated to different phases of the internal activity of volcanoes. The study is conducted on six years of volcano-seismic data recorded on Ubinas volcano (Peru). In particular, the outcomes of the proposed automatic classification system helped in the discovery of misclassifications in the manual annotation of the recordings. In addition, the proposed automatic classification framework of volcano-seismic signals has been deployed and tested in Indonesia for the monitoring of Mount Merapi. The software implementation of the framework developed in this thesis has been collected in the Automatic Analysis Architecture (AAA) package and is freely available.

Keywords — Automatic classification, Underwater acoustics, Volcano-seismic, Dimensionality reduction, Supervised machine learning, Deep learning

Résumé

Ce manuscrit de thèse résume trois ans de travaux sur l'utilisation des méthodes d'apprentissage statistique pour l'analyse automatique de signaux naturels. L'objectif principal est de présenter des outils efficaces et opérationnels pour l'analyse de signaux environnementaux, en vue de mieux connaitre et comprendre l'environnement considéré. On se concentre en particulier sur les tâches de détection et de classification automatique d'événements naturels. Dans cette thèse, deux outils basés sur l'apprentissage supervisé (Support Vector Machine et Random Forest) sont présentés pour (i) la classification automatique d'événements, et (ii) pour la détection et classification automatique d'événements. La robustesse des approches proposées résulte de l'espace des descripteurs dans lequel sont représentés les signaux. Les enregistrements y sont en effet décrits dans plusieurs espaces: temporel, fréquentiel et quéfrentiel. Une comparaison avec des descripteurs issus de réseaux de neurones convolutionnels (Deep Learning) est également proposée, et favorise les descripteurs issus de la physique au détriment des approches basées sur l'apprentissage profond. Les outils proposés au cours de cette thèse sont testés et validés sur des enregistrements in situ de deux environnements différents : (i) milieux marins et (ii) zones volcaniques. La première application s'intéresse aux signaux acoustiques pour la surveillance des zones sous-marines côtières : les enregistrements continus sont automatiquement analysés pour détecter et classifier les différents sons de poissons. Une périodicité quotidienne est mise en évidence. La seconde application vise la surveillance volcanique : l'architecture proposée classifie automatiquement les événements sismiques en plusieurs catégories, associées à diverses activités du volcan. L'étude est menée sur 6 ans de données volcano-sismiques enregistrées sur le volcan Ubinas (Pérou). L'analyse automatique a en particulier permis d'identifier des erreurs de classification faites dans l'analyse manuelle originale. L'architecture pour la classification automatique d'événements volcano-sismiques a également été déployée et testée en observatoire en Indonésie pour la surveillance du volcan Mérapi. Les outils développés au cours de cette thèse sont rassemblés dans le module Architecture d'Analyse Automatique (AAA), disponible en libre accès.

Mots-clé — Classification automatique, Acoustique sous-marine, Volcano-seismique, Réduction de dimension, Apprentissage statistique supervisé, Apprentissage profond