Fusion multi-niveaux par boosting pour le tagging automatique

Rémi Foucard

ParisTech
INSTITUT DES SCIENCES ET TECHNOLOGIES
PARIS INSTITUTE OF TECHNOLOGY

TELECOM ParisTech

2013-ENST-0093

EDITE - ED 130

Doctorat ParisTech

T H È S E

pour obtenir le grade de docteur délivré par

TELECOM ParisTech

Spécialité « Traitement du Signal et des Images »

présentée et soutenue publiquement par

Rémi FOUCARD

le 20 décembre 2013

Fusion multi-niveaux par boosting pour le tagging automatique

Directeur de thèse : **Gaël Richard**
Co-encadrement de la thèse : **Slim Essid, Mathieu Lagrange**

Jury

M. Liming CHEN, Professeur, LIRIS, Ecole Centrale de Lyon	Rapporteur
Mme Myriam DESAINTE CATHERINE, Professeur, Labri, Ecole Centrale de Lyon	Rapporteur
M. Laurent DAUDET, Professeur, Institut Langevin, Université Paris Diderot	Examinateur
M. Slim ESSID, Maître de conférences, LTCI, Télécom ParisTech	Directeur
M. Mathieu LAGRANGE, Chargé de recherche, Ircam, CNRS - UPMC Paris VI	Directeur
M. Gaël RICHARD, Professeur, LTCI, Télécom ParisTech	Directeur

TELECOM ParisTech
école de l'Institut Mines-Télécom - membre de ParisTech
46 rue Barrault 75013 Paris - (+33) 1 45 81 77 77 - www.telecom-paristech.fr

T H È S E

À Yvonne Huzar.

Remerciements

J'ai eu la chance d'être très entouré pendant ces années de doctorat. Ainsi, j'adresse un sincère et profond remerciement à tous ceux qui ont pu m'apporter leur soutien durant cette période.

Mes premiers remerciements vont à mes directeurs de thèse pour m'avoir offert l'opportunité de réaliser ce doctorat à Télécom ParisTech. Je les remercie tant pour leur compétence scientifique que pour le soutien, la motivation et la confiance dont ils ont constamment témoigné tout au long de ce travail de thèse. Leur aide a été déterminante.

Merci également aux membres du jury pour l'intérêt qu'ils ont porté à mes travaux et pour le temps qu'ils ont passé à étudier mon manuscrit. Notamment, les deux rapporteurs m'ont apporté des remarques pertinentes, qui m'ont grandement aidé à préparer la soutenance.

J'adresse un remerciement chaleureux à tous les membres du groupe AAO pour nos discussions riches d'enseignements scientifiques et non-scientifiques, mais aussi pour les très bons moments passés heureux tous ensemble. L'ambiance qu'ils ont contribué à créer constituait une des raisons de ma motivation tout au long de ce doctorat (outre l'amour de la science). J'ai également une pensée pour certains qui ne sont pas de notre lab, mais avec qui j'ai pu partager des moments précieux lors de nos voyages en conférence.

Je tiens aussi à saluer tous ceux avec qui j'ai pu partager concerts, tournées ou enregistrements au cours de ces années, notamment avec Yan Wagner et Victorine. Ces moments musicaux ont été un grand bonheur, une grande respiration et l'occasion de belles rigolades.

Je remercie particulièrement ma famille et mes proches pour leur présence bienveillante et leur soutien sans faille.

Enfin, un grand merci à Sébastien, dont l'attention, l'écoute et la patience constantes m'ont été indispensables, notamment lors de la phase de rédaction.

Résumé

Les tags constituent un outil très utile pour indexer des documents multimédias. Ce sont des labels sémantiques textuels qui décrivent n'importe quel aspect du document, aidant ainsi l'organisation et la structuration de bases de données.

Cette thèse de doctorat s'intéresse au tagging automatique, c'est à dire l'association automatique par un algorithme d'un ensemble de tags à chaque morceau. Nous utilisons des techniques de boosting pour réaliser un apprentissage prenant mieux en compte la richesse de l'information exprimée par la musique.

Une première contribution de cette thèse consiste à mieux exploiter, lors de l'apprentissage, la subtilité du lien qui existe entre un morceau et un tag. Pour ce faire, nous proposons une nouvelle fusion d'annotateurs, qui exprime mieux leurs incertitudes ou leurs désaccords. Cette nouvelle fusion est couplée à un apprentissage régressif.

Nous explorons également un ensemble de descripteurs de différents niveaux d'abstraction. Ils sont extraits du signal audio ou de données en ligne. Ces descripteurs permettent de construire une représentation riche de chaque morceau pour l'apprentissage.

Enfin, un nouvel algorithme de boosting est proposé, afin d'utiliser conjointement des descriptions de morceaux associées à des extraits de différentes durées. Cet algorithme est utilisé pour fusionner nos descriptions multi-niveaux, qui sont observables sur des durées différentes.

Abstract

Tags constitute a very useful tool for multimedia document indexing. They are textual semantic labels describing any aspect of the document, thus helping database organization and structuration.

This PhD thesis deals with automatic tagging, which consists in associating a set of tags to each song automatically, using an algorithm. We use boosting techniques to design a learning which better considers the complexity of the information expressed by music.

One first contribution from this thesis consists in better leveraging the subtlety of the link between a song and a tag, during the learning phase. To do that, we propose a new annotator fusion, which better expresses uncertainty or disagreements. This new fusion is used in conjunction with a regressive learning.

We also explore a set of descriptors belonging to different abstraction levels. They are extracted from the audio signal or online data. These features are used to build a rich representation for each song, which can be used for learning.

Finally, a new boosting algorithm is proposed, which can jointly use song descriptions associated to excerpts of different durations. This algorithm is used to fuse our multi-level descriptions, which are observable on different durations.

Table des matières

1. Introduction **1**

1.1. Indexation audio et tags 1

1.1.1. La nécessité d'une indexation de qualité 1

1.1.2. Les *tags* : des étiquettes sémantiques très répandues 2

1.2. Apprentissage automatique pour le tagging 5

1.3. Différents niveaux d'abstraction 7

1.4. Problématiques 8

1.5. Résumé des contributions 9

1.6. Structure du document 10

2. Classification pour le tagging automatique **13**

2.1. Introduction 13

2.2. Représentation des morceaux 14

2.2.1. Descriptions du signal 14

2.2.2. Données sociales et contextuelles 19

2.2.3. Le problème de la représentation des variations temporelles . . 20

2.3. L'apprentissage automatique des tags 22

2.3.1. Classification multi-labels 22

2.3.2. Algorithmes d'apprentissage 24

2.4. Fusion d'informations hétérogènes 28

2.5. Données pour le tagging automatique 30

2.5.1. Récolte des annotations 30

2.5.2. Choix d'une base de données 32

2.6. Évaluation 34

2.6.1. Cadre d'évaluation pour la classification 34

2.6.2. Validité statistique des résultats 35

2.7. Conclusion 36

3. Boosting d'arbres de décision : un cadre performant et flexible 37
3.1. Introduction . . . 37
3.2. Le boosting : une classe de méta-classifieurs . . . 38
3.2.1. Un méta-classifieur itératif . . . 38
3.2.2. Un modèle flexible . . . 40
3.3. Le cas particulier des arbres de décision . . . 40
3.3.1. Définition et construction . . . 40
3.3.2. Comportement des arbres boostés . . . 42
3.4. Adaptation à plusieurs fonctions de coût . . . 43
3.5. Le boosting pour la fusion de classifieurs . . . 44
3.6. Gestion des descripteurs manquants . . . 44
3.6.1. L'algorithme Ada-ABS . . . 44
3.6.2. Relation avec Adaboost . . . 46
3.7. Conclusion . . . 46

4. Fusion souple d'annotateurs et régression 49
4.1. L'annotation, génératrice d'incertitude . . . 49
4.2. Vers une vérité-terrain plus souple . . . 51
4.3. Fusion souple des annotateurs . . . 53
4.3.1. Méthode de fusion . . . 54
4.3.2. Validation de la méthode de fusion . . . 55
4.4. Apprentissage régressif et validation de l'approche . . . 56
4.4.1. Mode opératoire . . . 56
4.4.2. Résultats et discussion . . . 57
4.5. Conclusion . . . 58

5. Des descripteurs hétérogènes 59
5.1. Introduction . . . 59
5.2. Couvrir différents niveaux d'abstraction . . . 60
5.2.1. Timbre . . . 60
5.2.2. Harmonie . . . 64
5.2.3. Rythme . . . 64
5.2.4. Tests de performance . . . 64
5.3. Importance de l'intégration temporelle précoce . . . 67
5.3.1. Pourquoi une intégration précoce ? . . . 67
5.3.2. Étude sur la méthode d'intégration . . . 67

5.4. Influence de l'échelle de description 71
5.5. Conclusion 72

6. Décrire un morceau sur plusieurs échelles temporelles 75
6.1. Introduction 75
6.2. Travaux pré-existants sur la fusion multi-échelles 76
6.3. Algorithme de boosting pour l'analyse multi-échelles 78
6.3.1. Plage de décision 78
6.3.2. Cœur de l'algorithme 79
6.4. Deux expériences pour l'évaluation 81
6.4.1. Reconnaissance des instruments de musique 82
6.4.2. Multi-tagging 84
6.5. Conclusion 85

7. Données collaboratives et fusion multi-niveaux 87
7.1. Introduction 87
7.2. Descripteurs issus du contexte éditorial et social 89
7.2.1. Tags utilisateurs 89
7.2.2. Paroles 90
7.2.3. Image de la pochette du disque 94
7.2.4. Décennie de sortie 94
7.2.5. Tests de performance 94
7.3. Fusion multi-niveaux 96
7.3.1. Des représentations vivant à différentes échelles 96
7.3.2. Validation expérimentale 97
7.4. Conclusion 98

8. Conclusion 101

A. Métriques d'exactitude pour l'évaluation de classifieurs 107
A.1. Introduction 107
A.2. Métriques de récupération (*retrieval*) 107
A.3. Métriques de classement 110

B. Tests statistiques pour l'évaluation des prédictions 113
B.1. Introduction 113
B.2. Test de McNemar 114

B.3. Test de Student par séries appariées avec validation croisée 115

C. Liste des tags analysés pour les tests **117**

Publications de l'auteur **121**

Bibliographie **123**

Notations **139**

Index **141**

1. Introduction

La diversité culturelle est souvent synonyme de richesse, et cela est également vrai en ce qui concerne la musique. De ce point de vue, il est grisant de constater que notre société propose un accès sans précédent à une grande variété de musiques, tant dans le spectacle vivant qu'en terme d'enregistrements. Les individus possèdent des goûts très différents les uns des autres, et une même personne apprécie souvent plusieurs types de musique. L'éclectisme musical est d'ailleurs une qualité de plus en plus valorisée socialement [Cou10].

Cette diversité musicale se retrouve naturellement dans les médiathèques et les magasins de musique où, pour contenter tous les publics, les références disponibles sont souvent très nombreuses. Le client peut ainsi se retrouver un peu perdu face à un choix aussi large, c'est pourquoi des employés sont souvent là pour le conseiller. Certaines enseignes sont d'ailleurs réputées pour la qualité des conseils de leurs vendeurs, capables de proposer des disques en fonction de critères très précis.

1.1. Indexation audio et tags

1.1.1. La nécessité d'une indexation de qualité

La distribution de musique se fait de plus en plus sous format dématérialisé, au détriment du support physique [1]. Cette tendance a plusieurs conséquences :

– Premièrement, la distribution de musique dématérialisée possède un coût de distribution bien moindre que sur support physique. La diminution de cet investissement permet de proposer davantage de références peu connues, moins sus-

1. Dans certains pays, le numérique est même devenu le mode majoritaire de consommation (`http ://www.bbc.co.uk/news/entertainment-arts-18278037`).

ceptibles de se vendre. Cela augmente considérablement le nombre de références disponibles [2].

– Par ailleurs, le contact avec le vendeur disparaît. Le contact à distance étant beaucoup moins spontané, les plates-formes de distribution en ligne n'ont pas jugé utile de proposer des contacts avec des spécialistes pour recommander ou retrouver les disques. Par conséquent, l'organisation et la présentation du service en ligne deviennent primordiales.
– On note également que le contenu audio des morceaux est directement disponible pour la lecture et l'analyse. Cela facilite par exemple la pré-écoute par l'utilisateur avant d'acheter, mais permet également la diffusion de musique par flux (*streaming*), dans des services intégralement en ligne.

Avec l'augmentation de la taille des bases de données et la disparition du conseiller, les utilisateurs ont donc besoin de données très bien organisées et indexées, afin de trouver facilement ce qu'ils cherchent. En outre, les discothèques numériques personnelles sont parfois très étoffées et il est souhaitable de bénéficier d'une bonne indexation. Heureusement, la version dématérialisée permet de stocker des métadonnées permettant une indexation élaborée, et générées par des humains ou des processus automatiques. Ces derniers exploitent des données externes ou le flux lui-même, désormais exploitable directement.

1.1.2. Les *tags* : des étiquettes sémantiques très répandues

Les *tags* sont des métadonnées utilisées pour indexer le contenu multimédia. Ce sont des labels sémantiques textuels, décrivant n'importe quel aspect d'un fichier. Ils peuvent servir de mots-clés pour la recherche, ou de critères de similarité. On peut les apposer sur n'importe quel type de données. Ainsi sont-ils utilisés par des services en ligne pour indexer la musique (par exemple sur Last.fm ou MusicBrainz), mais aussi la vidéo (Youtube) et les images (Flickr). Ces étiquettes sont également de plus en plus utilisées sur les réseaux sociaux pour indexer les publications (comme sur Tumblr ou Twitter). À titre d'illustration, la Figure 1.1 montre quelques ensembles de tags que l'on peut trouver sur divers services Web, et associés directement ou indirectement au morceau « Army of me » de la chanteuse Björk.

2. À titre d'exemples, le magasin iTunes revendique actuellement 26 millions de morceaux, et Deezer, 25 millions.

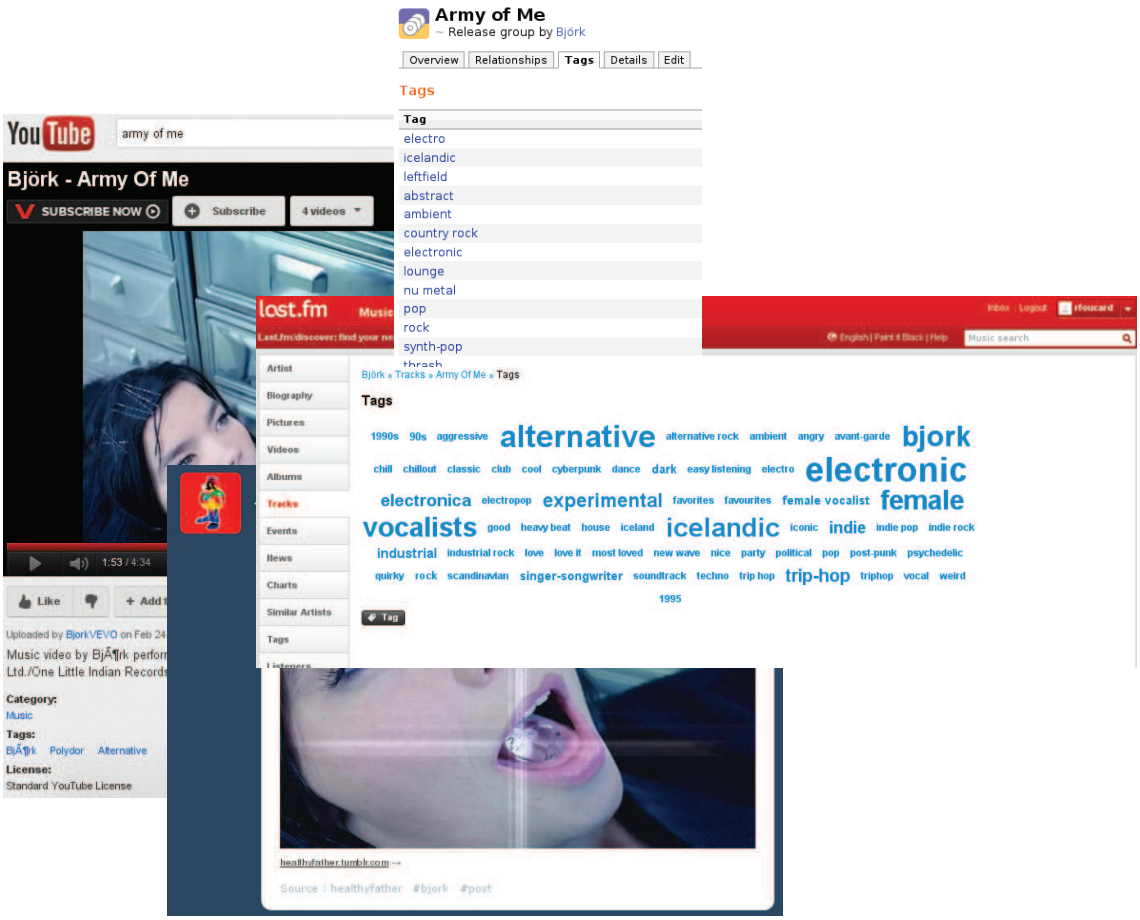


Figure 1.1.: Un exemple des tags que l'on peut trouver en cherchant « Björk Army of me » sur Last.fm, MusicBrainz, Tumblr et Youtube.

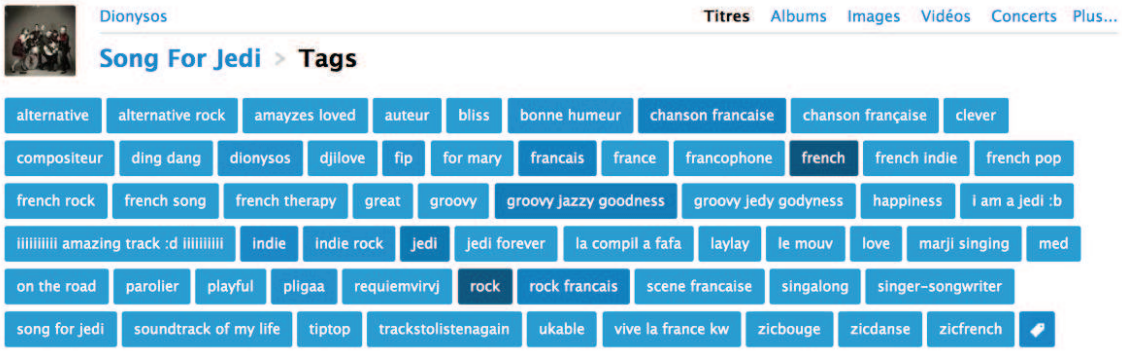


Figure 1.2.: Tags associés au morceau « Song For Jedi » de Dionysos par les utilisateurs du site Last.fm.

Les tags peuvent également être utilisés pour estimer la similarité entre les morceaux, afin de générer des listes de lecture [MCJT12] ou pour recommander des morceaux. C'est d'ailleurs une similarité par tags qui est utilisée par la radio en ligne Pandora [3].

L'annotation d'un morceau par des tags peut être effectuée par ceux qui publient le morceau (on parlera alors de tags *éditoriaux*), ou par les utilisateurs du service (tags *sociaux*). Les tags éditoriaux sont plutôt coûteux en temps et en argent car ils nécessitent un investissement humain de la part de la maison de disques ou du distributeur. Ils sont donc en général peu nombreux [4] mais fiables et bien structurés, c'est à dire que le vocabulaire des tags est cohérent, logique et aisément interprétable. Les tags sociaux, en revanche, sont construits par une communauté d'utilisateurs et leur exploitation est donc moins coûteuse que la construction de tags éditoriaux. Mais ils sont moins fiables et souvent mal structurés, puisque le vocabulaire est libre. Un exemple est donné dans la Figure 1.2, où l'on peut voir un ensemble de tags sociaux dont certains présentent un intérêt limité pour l'indexation. De plus, sur les morceaux peu populaires, ces tags sont souvent trop rares donc difficilement exploitables (c'est le fameux problème du « démarrage à froid »).

Une troisième option est l'association automatique des tags par un système informatique. Ce procédé est favorisé par la disponibilité immédiate du signal audio de tous les morceaux, qui peut être associé à d'autres sources de données. Le tagging automatique constitue un bon compromis entre les tags éditoriaux et les tags

3. `http ://www.pandora.com`
4. Une exception notable est la radio en ligne Pandora, spécialisée dans ces annotations très coûteuses, et qui réalise le tagging des morceaux de son catalogue un par un, par un collège d'experts.

sociaux. En effet, tout en restant modérément coûteux, ces tags peuvent s'avérer raisonnablement fiables, leur vocabulaire est aisément structurable, et le problème du « démarrage à froid » est inexistant.

1.2. Apprentissage automatique pour le tagging

Le tagging automatique fait en général appel à des techniques d'apprentissage pour la classification. C'est à dire que le système, en analysant un grand nombre d'exemples, apprendra lui-même à distinguer les morceaux sur lesquels un tag donné s'applique.

Un système classique de tagging automatique est présenté dans la Figure 1.3. On considère ici un tag à la fois, et le but est d'apprendre à classifier les morceaux en deux catégories : ceux sur lesquels le tag s'applique, et ceux sur lesquels il est inapproprié. La procédure comporte deux étapes :

– une étape préalable d'apprentissage, dont le but est de construire, d'après de nombreux exemples, une règle de décision sur les morceaux ;
– puis le tagging proprement dit, où cette règle peut être utilisée sur n'importe quel nouveau morceau pour décider si le tag s'applique.

Pour commencer, l'apprentissage consiste donc à analyser un grand nombre d'exemples pour apprendre à classifier de nouveaux morceaux. Il nécessite deux jeux de données connectés :

– un ensemble de morceaux de musique, représentés par leur signal audio et/ou des données provenant de services en ligne ;
– des annotations fiables, indiquant quels morceaux sont associés au tag considéré.

L'apprentissage commence par l'extraction, à partir des signaux d'entraînement, d'informations et de caractéristiques supposées pertinentes. On obtient alors un ensemble de descripteurs, généralement numériques, rassemblés dans des vecteurs $\mathbf{x}_i$. Chacun de ces vecteurs possède une valeur d'annotation associée y_i, indiquant si le tag s'applique ou non sur le morceau correspondant. Puis une technique d'apprentissage statistique va utiliser les $\mathbf{x}_i$ et les y_i pour construire une règle de décision binaire $h(\mathbf{x}) \in \{-1, 1\}$.

Lors de la phase de tagging, on doit taguer un morceau préalablement inconnu. On commence alors par extraire les mêmes descripteurs que lors de l'apprentissage, puis la fonction $h(\mathbf{x})$ est utilisée pour décider si le tag s'applique ou pas.

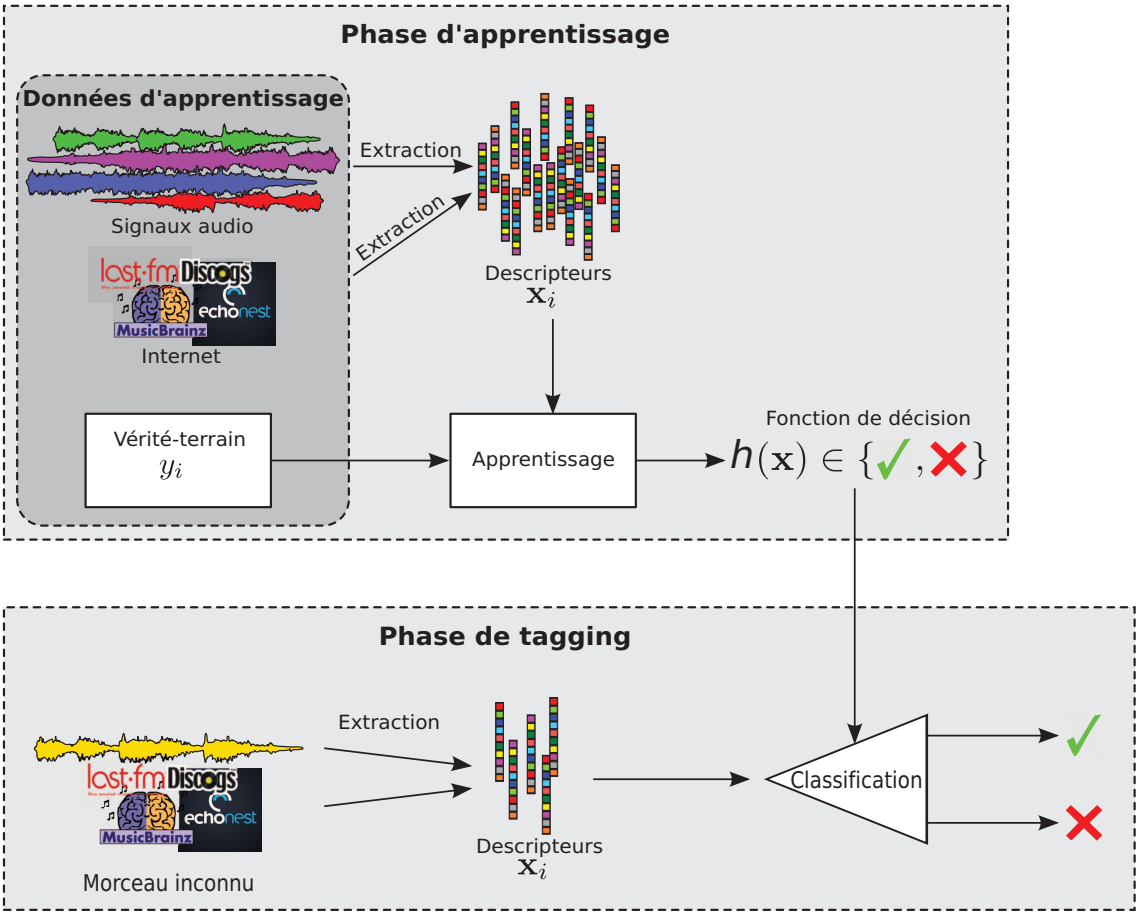


Figure 1.3.: Schéma général d'un système de tagging automatique.

Un tel système comporte trois caractéristiques principales. Tout d'abord, les données d'apprentissage doivent être représentatives de l'ensemble des données que l'on sera amené à taguer. Elles doivent également porter des annotations fiables. Ensuite, la représentation des morceaux à l'étape des descripteurs doit faire ressortir des caractéristiques pertinentes pour leur classification, et doit présenter ces caractéristiques de manière aisément exploitable. Enfin, le modèle statistique choisi pour l'apprentissage est, bien évidemment, capital (il est même souvent considéré comme le point central du système). Il doit être capable de séparer au mieux les données des deux classes (positive et négative), tout en offrant une robustesse satisfaisante au bruit et aux données aberrantes (par exemple, des morceaux mal annotés ou ambigus).

1.3. Différents niveaux d'abstraction

Dans la section précédente, nous pointons l'importance capitale d'une bonne représentation des signaux musicaux pour l'efficacité du classifieur. En effet, les caractéristiques présentées doivent être aisément exploitables, et surtout elles doivent être liées aux critères que l'on cherche à discriminer. Par exemple, la durée du morceau sera un descripteur très peu informatif pour apprendre des tags basés sur l'émotion. Par contre, si le tag à apprendre est *Morceau-Longue_durée*, alors ce descripteur sera très utile.

Puisque l'on ne connaît pas toujours *a priori* le sens de tous les tags qu'il va falloir analyser, il paraît logique d'adopter des représentations qui couvrent le plus possible d'aspects différents, tout en restant peu redondantes. Cela nécessite des descripteurs hétérogènes.

Afin d'obtenir des descripteurs différents, il est nécessaire de varier les *niveaux d'abstraction*. Pour une information, nous appelons « niveau d'abstraction » le positionnement de celle-ci entre le monde des faits, physique, et le monde des idées, des mots, des modèles et des représentations. Ainsi, pour la musique, le plus bas niveau d'abstraction dont nous disposons est le signal. On peut déjà considérer un signal numérique comme abstrait, en cela qu'il n'*est* pas le son mais constitue une représentation de celui-ci par des nombres, par ailleurs basée sur des mesures approximatives. Mais le signal est connecté de très près au monde physique, et tout son sens reste à extraire. Les tags par contre, sont des informations de beaucoup

plus haut niveau d'abstraction : elles sont bien davantage reliées à la sémantique qu'à la physique. Toutes les autres informations peuvent être placées sur cet axe : de la plus physique à la plus perceptuelle, de la plus concrète à la plus chargée de sens.

Ainsi, dans un système de tagging, il existe un immense *fossé sémantique* entre le signal et les tags que l'on cherche à estimer. En utilisant uniquement des descripteurs de bas niveau, c'est au classifieur seul que revient la tâche de franchir ce fossé. Par contre, en extrayant du signal des informations de différents niveaux d'abstraction, cet effort est partagé [ADP07]. Le problème est que ces informations ne sont pas toujours extraites sur les mêmes durées de signal et présentent des spécificités qui les rendent difficiles à exploiter conjointement.

1.4. Problématiques

Dans cette thèse, nous répondons à trois problématiques principales :

1. **Trouver de nouvelles descriptions pour mieux couvrir l'axe sémantique.**
   Les représentations classiques des morceaux résultent souvent de simples transformations du signal temporel ou du spectre, afin d'en faire ressortir certaines caractéristiques [PGS+11]. Ces caractéristiques sont en général fortement liées à l'aspect physique du signal, donc d'abstraction assez faible. Cela demande au classifieur de franchir seul le fossé sémantique pour arriver jusqu'aux tags (*cf.* section 1.3). Un système de classification pour le tagging pourrait donc bénéficier de descriptions de plus haut niveau. De plus, le fait de varier les niveaux d'abstraction des descripteurs peut constituer un bon moyen de limiter leur redondance, et ainsi améliorer leur complémentarité lorsqu'ils sont utilisés en même temps.
2. **Exploiter conjointement des descriptions issues de différents niveaux d'abstraction.**
   Les représentations de haut niveau d'abstraction sont souvent calculées sur une durée de signal plus longue que les descriptions de bas niveau. En effet, on a souvent besoin d'une certaine durée pour donner du sens à une portion de signal, tandis que certains descripteurs de bas niveau peuvent être relativement

instantanés. Cependant, si des représentations ne s'extraient pas sur les mêmes portions de signal, elles deviennent difficiles à fusionner. Une solution immédiate serait de ramener, par intégration temporelle, toutes les descriptions sur les mêmes durées. Mais ce procédé ne tient pas compte de la temporalité naturelle où chaque descripteur fait sens, ce qui peut détériorer la qualité de la description.

3. **Trouver une vérité-terrain qui reflète mieux la pertinence des tags.** Dans un schéma classique de classification, un tag peut être soit présent, soit absent sur un morceau. Or son lien avec le morceau est parfois plus subtil que cette description binaire. Un tag peut ainsi être lié à un morceau, mais parfois faiblement. Ou seulement en partie. De plus, il est courant que les auditeurs expriment des avis différents. Par exemple pour un tag *Émotion-Triste*, un morceau peut être modérément triste, ou être interprété comme triste seulement par une partie du public. Ces subtilités ne sont pas prises en compte par une association binaire or cette information peut être importante pour être a même de traiter les exemples ambigus ou mal annotés. L'apprentissage pourrait ainsi bénéficier d'une vérité-terrain plus souple, reflétant des associations plus ou moins fortes entre les tags et les morceaux.

1.5. Résumé des contributions

L'ensemble des travaux présentés dans cette thèse s'articulent autour de l'utilisation d'un outil d'apprentissage qui par sa flexibilité nous permet d'approcher les problématiques précitées. Le boosting est une technique d'apprentissage automatique particulièrement flexible et facilement adaptable (*cf.* chapitre 3). La contribution centrale de cette thèse est donc l'exploitation de schémas de boosting permettant de prendre en compte les problématiques énoncées dans la section précédente.

Les travaux présentés abordent les trois points cruciaux d'un système de classification audio (*cf.* section 1.2) : la description des morceaux, l'algorithme d'apprentissage, et les données d'entraînement. Nous explorons ainsi des descriptions nouvelles ou récentes, puis un algorithme d'apprentissage permettant de fusionner ces nouvelles données. Enfin, nous présentons également de nouveaux types d'annotations pour le tagging.

Plus précisément, nos contributions sont les suivantes :

– Nous avons conçu un nouvel algorithme d'apprentissage automatique, permettant d'utiliser conjointement des descriptions extraites à des granularités différentes (chapitre 6). Cet algorithme est basé sur le principe du boosting, et permet d'exploiter des séquences de descripteurs issus de fenêtres d'analyse différentes. Une des originalités de cette nouvelle technique est qu'elle permet de conserver l'information sur la simultanéité des descripteurs des différentes échelles.
– Puisque des descripteurs de différents niveaux d'abstraction ne sont souvent pas extraits sur les mêmes durées de signal, l'algorithme de fusion pré-cité nous permet de fusionner des descriptions présentant ce type de différences. Nous l'appliquons donc à nos nouveaux descripteurs, ce qui nous permet de garder chaque représentation à son horizon de description optimal (chapitre 7).
– Nous proposons un nouveau cadre d'apprentissage pour le tagging automatique, permettant davantage de subtilité dans la description des associations entre les tags et les morceaux (chapitre 4). Ce cadre s'appuie sur une nouvelle fusion des annotateurs, prenant en compte leurs indécisions et leurs désaccords. Cette nouvelle fusion aboutit à une vérité-terrain souple, qui sera prise en compte lors de l'apprentissage au moyen d'un algorithme de régression.
– Nous étudions des descripteurs récents ou nouveaux pour le tagging automatique, appartenant à différents niveaux d'abstraction. Certains d'entre eux sont extraits du signal (chapitre 5), tandis que d'autres proviennent du contexte social et éditorial (chapitre 7). Ces descripteurs sont évalués, ainsi que d'autres descripteurs pré-existants mais rares ou jamais utilisés pour le tagging automatique. Nous évaluons également, pour les descripteurs issus du signal, l'influence de la durée de description, et nous testons de nouvelles méthodes d'intégration (chapitre 5).

1.6. Structure du document

Le corps de ce document est organisé de la manière suivante.

Dans un premier temps, nous décrivons des techniques, concepts et données pré-existants, sur lesquels nous appuyons en partie nos contributions. Le chapitre 2 présente un état de l'art du tagging automatique par classification. Il passe en revue les trois grandes étapes d'un système de tagging automatique (représentation des

morceaux, algorithme de classification, données d'apprentissage). Il aborde également les thématiques de la fusion d'informations, qui nous intéresse particulièrement dans cette thèse (section 2.4), et de l'évaluation des systèmes de classification (section 2.6). Dans le chapitre 3, nous présentons le principe du boosting pour l'apprentissage automatique. Cette technique est utilisée tout au long des travaux présentés, car elle est particulièrement flexible et donne naissance à de nombreux algorithmes. En effet, elle permet entre autres de profiter de certaines propriétés d'autres classifieurs plus simples, de fusionner des descriptions, d'utiliser plusieurs fonctions de coût d'apprentissage, et de gérer les descripteurs manquants.

Dans le chapitre 4, nous décrivons une nouvelle configuration de tagging automatique, qui prend mieux en compte le degré du lien qui peut exister entre un tag et un morceau. Nous commençons par proposer une fusion souple d'annotateurs, qui aboutit à une vérité-terrain avec des étiquettes quantitatives continues, et non plus binaires. Puis, cette nouvelle vérité-terrain est apprise au moyen d'un algorithme de boosting régressif.

Le chapitre 5 est focalisé sur les représentations issues du signal. Nous explorons des descripteurs récents ou nouveaux pour une tâche de tagging, qui couvrent différents niveaux d'abstraction. Nous étudions également plusieurs méthodes d'intégration, et nous analysons l'influence de la durée de description sur la qualité de la classification. L'expérience montre que la durée de description optimale varie bel et bien entre les descripteurs. C'est pourquoi nous construisons un algorithme permettant de fusionner des représentations extraites sur différentes échelles. Cet algorithme, basé sur le principe du boosting, est présenté dans le chapitre 6. Dans un premier temps, nous l'évaluons sur deux tâches de classification, en utilisant les mêmes descripteurs à chaque échelle. Cela permet d'évaluer la complémentarité de descriptions faites à des échelles différentes, même si elles sont de même nature. Ensuite, dans le chapitre 7, l'algorithme de fusion multi-échelles est appliqué pour réaliser une réelle fusion de différents niveaux d'abstraction. Nous commençons par enrichir notre ensemble de descripteurs par des informations de haut niveau, et extérieures au signal. Celles-ci sont en effet extraites du contexte éditorial et social du morceau. Puis l'algorithme de fusion du chapitre 6 est utilisé pour fusionner les meilleurs descripteurs du contexte et du signal, tout en gardant chaque description à son échelle optimale (section 7.3).

Pour conclure, le chapitre 8 apportera quelques commentaires sur le travail présenté,

et suggérera quelques pistes à explorer pour son prolongement.

2. Classification pour le tagging automatique

Résumé Ce chapitre donne un aperçu de l'état actuel de la recherche en tagging automatique. Il passe en revue les principales techniques utilisées dans ce domaine pour représenter des morceaux et entraîner automatiquement un classifieur. Il aborde également les problèmes de la fusion de sources d'informations hétérogènes, de l'annotation des données, et de l'évaluation des classifieurs.

2.1. Introduction

L'estimation automatique de tags, et plus généralement la classification automatique de musique, ont donné lieu à de nombreuses publications ces dix dernières années [BMEM10]. Les travaux se placent en général dans un schéma de classification supervisée [TC02]. Dans ce type de système, on commence par extraire du signal des descripteurs appropriés, puis un algorithme de classification décide, d'après ces descripteurs, de l'attribution ou non d'un tag donné sur le signal analysé [DHS00]. La règle de décision est apprise sur un corpus d'entraînement, annoté, et permet de classifier automatiquement de nouveaux morceaux non annotés.

Souvent, les travaux se concentrent sur l'un des trois points cruciaux d'un tel système :
– les données utilisées pour l'apprentissage ;
– la représentation des morceaux ;
– le modèle adopté pour l'apprentissage et la classification.
Nous développons ces trois aspects dans la suite de ce chapitre. Pour finir, une dernière section sera dédiée à l'évaluation des techniques et des systèmes de tagging automatique.

2.2. Représentation des morceaux

La plupart des algorithmes de classification sont incapables de dégager des critères de décision pertinents en analysant directement le signal $s(n)$. Il faut donc préparer les données dans un format qui permettra au classifieur de distinguer les morceaux positifs des négatifs. Il s'agit à cette étape de capturer des caractéristiques utiles à sa classification, et de les présenter de manière aisément exploitable pour l'algorithme d'apprentissage. Nous nous penchons d'abord sur les descriptions extraites du signal, puis nous évoquons celles que l'on peut construire à partir de données extérieures.

2.2.1. Descriptions du signal

2.2.1.1. Généralités

En principe, pour extraire une représentation appropriée, le signal est d'abord découpé régulièrement en courtes portions, ou « trames », sur lesquelles le signal est considéré stationnaire.

La Figure 2.1 montre un exemple de quatre fenêtres d'analyse consécutives, servant à découper le signal.

Ces fenêtres se recouvrent souvent mutuellement. Ainsi, la grande majorité des systèmes utilisent un recouvrement de 50%, ce qui offre un bon compromis entre la résolution temporelle et la redondance [WLMM07].

Des descripteurs instantanés sont calculés sur ces courtes portions, puis les trames consécutives sont agrégées sur des durées plus longues (*intégration temporelle*).

La représentation finale d'un groupe de trames $(x_1^{(i)}, \ldots, x_D^{(i)}) = \mathbf{x}_i \in \mathcal{X}$ devra donner une description pertinente de la durée de signal qu'elle représente, tout en restant relativement compacte. En effet, les descriptions de grande dimension D sont plus longues à traiter lors de l'apprentissage. De plus, ce dernier peut être dégradé par des vecteurs trop grands, surtout si plusieurs dimensions sont peu informatives pour le critère à classifier.

Certains descripteurs extraits du signal sont très populaires. Ils peuvent être répartis en trois catégories selon l'aspect musical qu'ils caractérisent : le timbre, l'harmonie

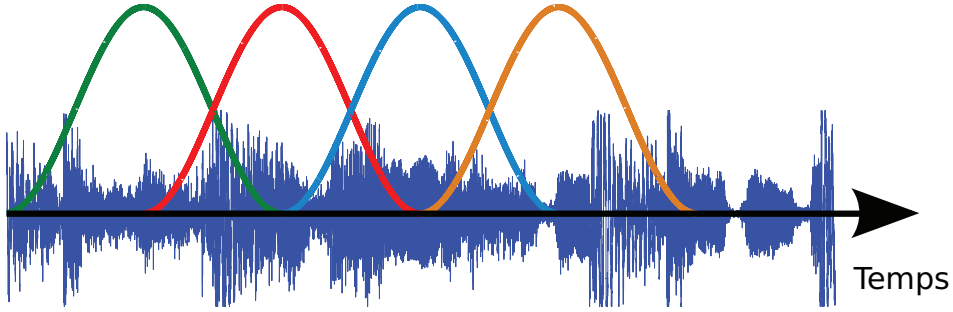


Figure 2.1.: Fenêtres d'analyse découpant une portion de signal en trames.

et le rythme [SZM06]. Le timbre est relié aux sonorités, aux instruments présents, au mixage de l'enregistrement. L'harmonie a trait aux notes entendues, leur hauteur, leurs assemblages en simultané. Le rythme prend en compte le type de mesure, le côté syncopé ou droit, les temps faibles ou forts.

D'autres taxonomies de descripteurs ont été proposées (par exemple, Weihs *et al.* utilisent quatre catégories : descripteurs à court-terme, à long-terme, sémantiques, et compositionnels [WLMM07]) mais nous utiliserons le classement par aspect musical, qui nous paraît plus intuitif et demeure le plus utilisé.

2.2.1.2. Timbre

Le timbre est un aspect souvent prépondérant lorsque l'on cherche à décrire un morceau pour la classification. De très nombreux descripteurs ont été proposés pour caractériser, entre autres, l'enveloppe temporelle ou spectrale (*cf.* Figure 2.2), la balance harmonique/bruité, ou l'évolution temporelle d'un son musical ou d'un morceau [MB03, PGS+11, FLTZ11].

Le plus utilisé de ces descripteurs est certainement constitué par les Coefficients cepstraux sur l'échelle de Mel, plus connus sous leur nom anglais *Mel-frequency Cepstral Coefficients* (MFCC) [RJ93, Log00]. Ces coefficients donnent une description de l'aspect de l'enveloppe spectrale (*cf.* Figure 2.2). On garde en général les 12 ou 13 premiers coefficients (le premier d'entre eux est parfois omis).

Les descripteurs timbraux sont souvent calculés directement à partir du signal temporel ou de transformations sur le plan temps/fréquence telles que : transformée de

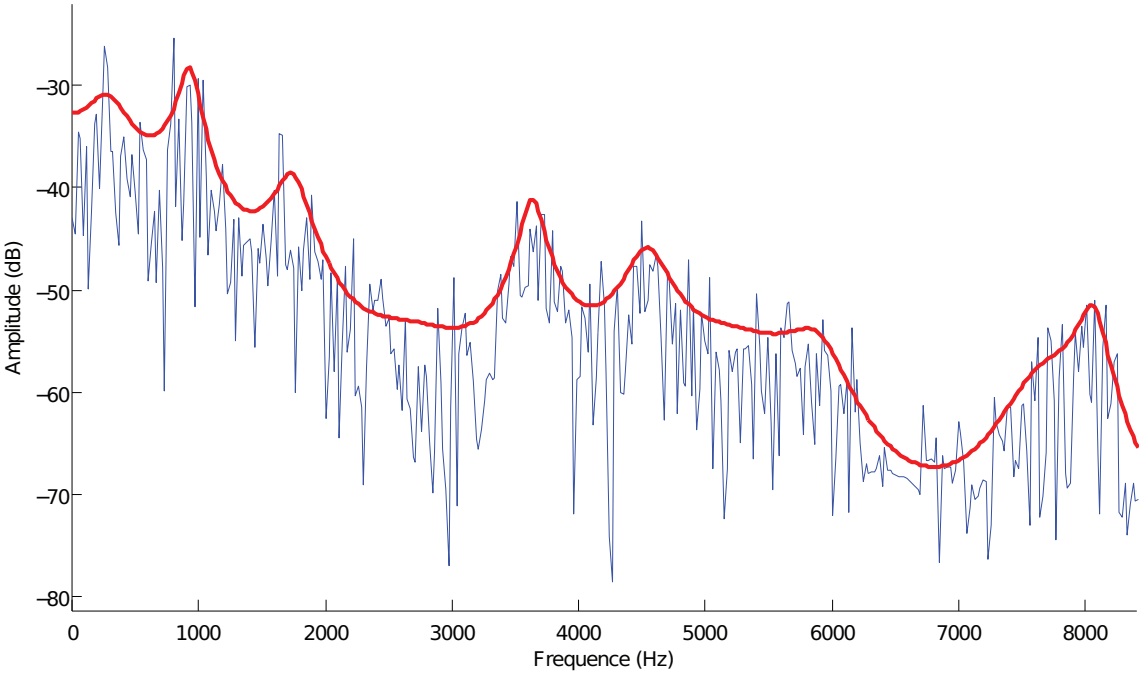


Figure 2.2.: Un spectre de signal audio et son enveloppe estimée par prédiction linéaire [Mak75].

Fourier, analyse cepstrale, modélisation auto-régressive, *etc.* Leur simplicité d'extraction et leur forte corrélation avec la physique du signal font de ces caractéristiques des descripteurs de bas niveau d'abstraction.

Même si les représentations simples fonctionnent souvent très bien, quelques études ont cherché à trouver des descripteurs plus complexes, notamment en les générant automatiquement. De nombreuses publications ont proposé des méthodes pour construire des descripteurs appris automatiquement sur les données d'apprentissage [PR07, PR09, MKRG12, KRG13]. Dans ces techniques, on choisit au départ un ensemble de quelques dizaines d'opérateurs tels que : racine carrée, maximum, autocorrélation, transformée de Fourier, filtrage passe-bande, *etc.* Puis un algorithme est utilisé pour trouver des combinaisons de ces opérateurs qui, appliquées au signal, donnent des descripteurs discriminatifs pour la tâche proposée. Bien que ce type de technique permette d'explorer un immense espace de descripteurs, il paraît probable que la simplicité des opérateurs de base ne permette de construire que des descripteurs de bas ou mi-niveau d'abstraction.

2.2.1.3. Harmonie

Les descripteurs d'harmonie décrivent le contenu tonal de la trame. Pour ce faire, des histogrammes de hauteurs ont été proposés dès les premières recherches dans le domaine [TC02]. On commence par utiliser un algorithme de détection de hauteurs multiples, puis les hauteurs estimées sont sommées dans un histogramme (en général, une case par demi-ton).

Depuis, il est devenu plus courant de décrire les hauteurs sans tenir compte de l'octave où elles apparaissent. On considère donc des classes de hauteurs qui regroupent, pour un son donné, toutes les hauteurs qui lui sont distantes d'un nombre fini d'octaves. Ces classes sont appelées *chroma*. On représente alors l'énergie de chaque chroma dans un histogramme, sans nécessairement passer par une phase d'estimation des hauteurs.

Il existe de nombreuses manières de calculer des chromagrammes. L'énergie correspondant à chaque chroma peut être calculée à partir de deux types de représentations temps-fréquence : la transformée de Fourier à court-terme (TFCT) [Fuj99, GÓ6, MD10], ou une transformée à Q constant (*Constant-Q transform,* CQT*)* [BP05, WSDR09]. La différence principale entre ces deux transformations est que l'échelle des fréquences est linéaire pour la TFCT, et logarithmique pour la CQT, ce qui est plus proche de la perception humaine. Nous utilisons pour nos travaux les chroma présentés dans [BP05], basés sur la CQT, et qui se sont montrés efficaces pour représenter le contenu tonal dans [Oud10].

Un chromagramme calculé selon cette méthode est représenté dans la Figure 2.3. Il est extrait de la chanson « S.O.S. » du groupe ABBA, à la fin du refrain, à partir du vers « *How can I even try to go on* ». On distingue bien la séquence Sib-Réb-Mib-Fa, jouée par la guitare, ainsi que les notes au chant.

Le plus intuitif est de considérer 12 chroma, correspondant aux 12 demi-tons d'une gamme chromatique, mais il n'est pas rare d'utiliser des résolutions plus fines.

Les chroma donnent une information de niveau d'abstraction moyen-bas car même si leur calcul est plutôt simple, il fait appel à des notions musicales et sont aisément interprétables.

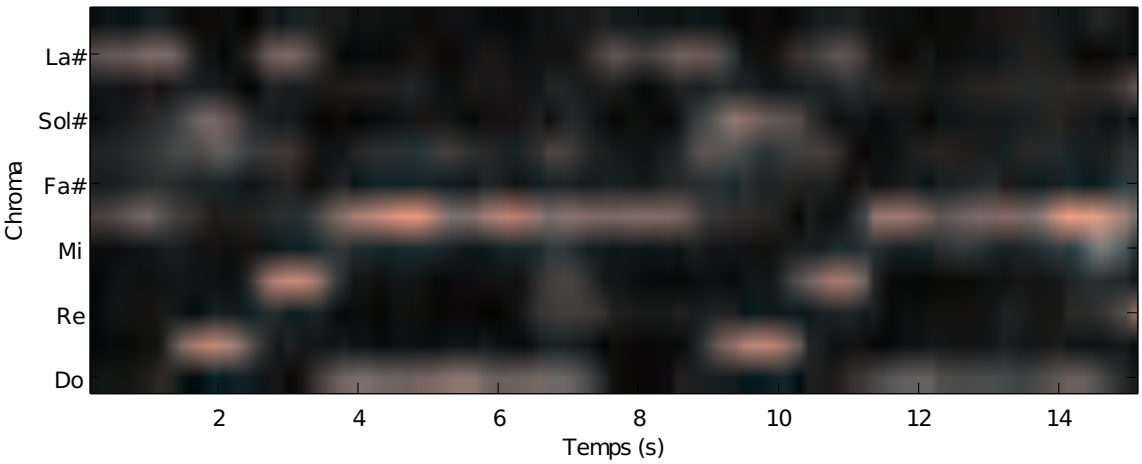


Figure 2.3.: Chromagramme représentant un extrait de « S.O.S. » du groupe ABBA.

2.2.1.4. Rythme

Il est difficile de trouver une définition globale du rythme. On considère souvent qu'il est lié à une certaine régularité des évènements sonores. Pour décrire cette régularité, on utilise souvent un histogramme rythmique, ou tempogramme [TC02]. Ce descripteur donne la relative importance des différentes périodicités dans une fourchette de tempi perceptibles (de l'ordre de 40 à 200 BPM). De nombreuses méthodes ont été proposées pour calculer un histogramme rythmique, et la fonction de nouveauté qui sert à le construire (*cf.* par exemple [Gro12]).

On peut également extraire de l'histogramme rythmique un ensemble d'informations de plus haut niveau d'abstraction, et potentiellement très informatives pour une tâche de classification de musique [TC02]. Ce résumé (« Beat Histogram Summary », ou BHS) est calculé directement à partir de l'histogramme. Les caractéristiques utilisées sont :

– somme de l'histogramme ;
– tempo et amplitude correspondant au pic le plus important ;
– tempo et amplitude correspondant au deuxième pic le plus important ;
– rapport d'amplitude entre ces deux pics.

Cependant, un tempo absolu n'est pas toujours aisé à reconnaître. En effet, il arrive souvent que le tempo d'un morceau soit ambigu, et qu'il soit difficile de choisir entre un tempo donné et son double (deux fois plus rapide), même pour des humains. Cette

ambiguïté peut rendre superflu le fait de distinguer les octaves les unes des autres [1]. C'est pourquoi le tempogramme cyclique a été proposé, caractérisant l'importance des périodicités, à la manière d'un histogramme rythmique classique, mais à l'octave près [GMK10]. Ce descripteur est en cela comparable à un chromagramme pour l'harmonie.

2.2.2. Données sociales et contextuelles

Si les informations contenues dans le signal audio d'un morceau sont précieuses pour sa classification, des données extérieures au signal peuvent aussi s'avérer très utiles. Par ailleurs, ces dernières années, les services en ligne à dimension sociale se sont multipliés. Par conséquent, de plus en plus de données générées par les utilisateurs sont accessibles automatiquement. Ces données ne sont pas toujours exploitées pour la classification musicale, même dans des études récentes. Pourtant, certaines sont récupérables rapidement et automatiquement, avec le simple titre du morceau et le nom de l'artiste.

Bien entendu, cette récupération est parfois sujette à des erreurs ou à des cas de morceaux introuvables. Une recherche automatique peut par exemple substituer deux versions d'un même morceau (une version studio et un enregistrement de concert). Ces substitutions peuvent s'avérer plus ou moins problématiques selon le descripteur envisagé. Les descripteurs de contexte nécessitent donc un algorithme d'apprentissage capable de gérer les descripteurs manquants et plutôt robuste aux données aberrantes (« outliers »).

Les descripteurs construits à partir de données en ligne peuvent être groupés en trois catégories :

Les données éditoriales sont des données renseignées lors de la mise en ligne du morceau par l'éditeur, le distributeur ou l'utilisateur qui a partagé le morceau [BCTL07, DNBL08]. Ce sont par exemple l'année de sortie du morceau, le nom de l'artiste ou celui du label, l'image de la pochette du disque, parfois le genre musical. Certains services spécialisés dans les métadonnées, comme The Echo Nest [2], proposent des informations pouvant être considérées comme

1. On parle d'octave rythmique par analogie avec la hauteur tonale, où un changement d'octave traduit une multiplication de la fréquence par deux.

2. `http ://www.echonest.com`

éditoriales (en effet, ces données sont construites par une organisation sans le recours à des utilisateurs) [TKT10]. Dans l'ensemble, les données éditoriales sont plutôt fiables et présentées de manière aisément exploitable.

Les données sociales collectées sont issues de la communauté, et sont récupérables directement : par exemple des tags utilisateurs ou le nombre d'écoutes sur certains services [Lam08, LS09a, LSSH09, BTYL09, MCJT12]. Ces données sont souvent récupérables directement et automatiquement ; ainsi leur traitement ne diffère pas grandement des données éditoriales. Par contre, les descripteurs construits à partir de ces données sont souvent réputés moins fiables.

Les données sociales construites sont élaborées à partir de documents écrits en langage naturel. Un dictionnaire est souvent construit au préalable, regroupant les mots d'intérêt. Puis des documents sont collectés sur le Web à l'aide d'un moteur de recherche [KPS+08, TBL08, BTYL09], de forums en ligne [SSKS12], de flux RSS [CCH06], *etc.*

Même si l'Internet est encore une source de données plutôt nouvelle dans le domaine de la classification musicale, il devrait se répandre rapidement dans les prochaines années. Notamment grâce au Million Song Dataset, qui propose un million de morceaux, accompagnés de nombreuses données sociales [BMEWL11].

2.2.3. Le problème de la représentation des variations temporelles

Dans la sous-section 2.2.1, nous avons passé en revue différentes manières de décrire une portion de signal. Mais il est plutôt rare qu'une fenêtre d'observation couvre toute la longueur du signal. On obtient donc pour chaque morceau une collection de fenêtres d'observation (*cf.* Figure 2.1).

Dans le schéma « bag of frames » (*sac de trames*), généralement utilisé [BMEM10], les exemples d'apprentissage sont les fenêtres d'observation et non les morceaux ; et ils sont considérés indépendamment les uns des autres. Par conséquent, aucune information sur l'ordre des observations n'est explicitement utilisée au niveau de l'apprentissage. Dans ce cas, si l'on veut exploiter la dynamique du signal, il faut l'exprimer dans le contenu des descripteurs [ADP07].

Deux principales solutions permettent aux descripteurs de prendre en compte la temporalité. La première idée est de dériver les descripteurs. Par exemple, dans [TBTL08], Turnbull *et al.* représentent le signal par des MFCC, accompagnés de leurs dérivées première et deuxième. Le flux spectral n'est pas à proprement parler une dérivée mais il s'inscrit dans le même esprit. En effet, il représente la différence quadratique moyenne des coefficients entre deux spectres successifs. Il est donc lié à la dynamique du signal.

La deuxième technique pour exprimer la temporalité des descripteurs consiste à la faire apparaître par agrégation des observations successives. Ce procédé consiste à regrouper les vecteurs d'observation situés dans une même *fenêtre de texture*, et à les résumer par des caractéristiques significatives. Par exemple, la variance (ou la covariance) est une statistique typique pour caractériser les fluctuations dynamiques. Une méthode simpliste pour représenter l'ensemble des fluctuations temporelles est de concaténer toutes les observations intégrées, laissant ainsi au classifieur le soin d'interpréter lui-même les variations [Sla02]. Cependant, les vecteurs de description résultant de ce regroupement sont de grande dimension et relativement complexes à analyser. De plus, ce type de représentation présente un problème de phase : sur un même signal, en décalant la fenêtre de texture d'une seule fenêtre d'observation, on va obtenir des descriptions très différentes.

L'intégration par modèle auto-régressif a également été utilisée avec succès pour la classification audio [MALH07, CCL12]. Certaines agrégations sont également calculées à partir de la TFCT du descripteur sur la durée intégrée [MB03, JER09].

Le Modèle de Texture Dynamique, récemment introduit pour la musique, peut être vu comme une technique d'intégration [BCL10]. Ce modèle considère un morceau comme une réalisation d'un modèle dynamique linéaire caché. Il a été utilisé avec succès pour la segmentation, et Coviello *et al.* ont développé une technique de classification l'utilisant pour descripteur [CCL11]. Cependant, cette technique ne se place pas dans le cadre « bag of frames », puisqu'elle tire parti des corrélations entre les fenêtres d'analyse.

Figure 2.4.: Tags associés au morceau « All I Need » de Air par les utilisateurs du site Last.fm.

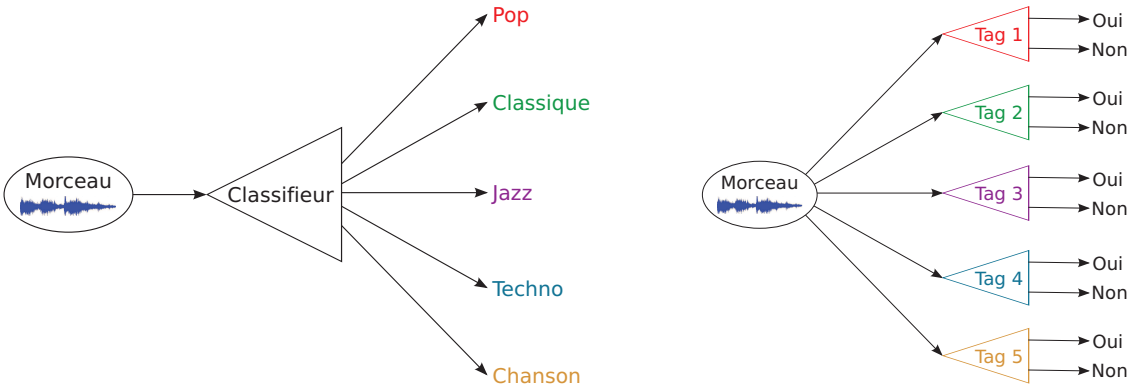
2.3. L'apprentissage automatique des tags

Une fois les morceaux représentés de manière appropriée, une technique d'apprentissage spécifique crée des règles de décision pour associer automatiquement les tags aux morceaux.

2.3.1. Classification multi-labels

Il est important, pour commencer, de bien établir la tâche à réaliser pour le tagging automatique. Le but est ici d'étiqueter chaque nouveau morceau m avec tous les labels appropriés. Ces labels sont divers et on ne dispose d'aucune information *a priori* sur leur signification. On a donc un ensemble de labels $\mathcal{L}$ et on cherche à associer à chaque morceau un sous-ensemble $L \subseteq \mathcal{L}$. On n'est donc pas dans un schéma de classification multi-classes (comme certaines classifications par genre ou par émotion), où un morceau tombe dans une seule catégorie $l \in \mathcal{L}$. On parle plutôt de *classification multi-labels*, car un morceau porte la plupart du temps plusieurs étiquettes. Beaucoup d'algorithmes proposés pour le cas mono-label sont utilisables en configuration multi-labels.

Une autre caractéristique importante du tagging automatique, est que l'on manipule les tags sans interpréter leur signification. Il paraît donc difficile d'optimiser le système d'apprentissage en fonction de la signification des tags ou des rapports sémantiques qui peuvent exister entre eux. Par exemple, les genres et les émotions sont non-exclusifs : un morceau peut obtenir plusieurs tags de genre, et plusieurs tags d'émotion. Ces tags peuvent être consensuels (un genre et ses sous-genres, des émotions



(a) Classification mono-label en genres musicaux. **(b)** Classification multi-labels.

Figure 2.5.: Schémas de classification mono-label et multi-labels.

inter-compatibles), mais rien ne l'oblige. Cette caractéristique est importante afin de pouvoir imiter le comportement d'un groupe de personnes, ayant des expériences d'écoute diverses. Ainsi sur le site Last.fm, le morceau « All I Need » du groupe Air possède plus de dix tags de genre, associés par les utilisateurs (*dream_pop*, *electronic*, *trip-hop*, ...). On peut les voir dans la Figure 2.4. Le morceau possède également des tags d'émotion plutôt contradictoires, comme *melancholic* et *warm_feelings*. On doit donc considérer comme possibles tous les sous-ensembles de tags $L \subseteq \mathcal{L}$.

Il existe plusieurs manières de reformuler un problème de classification multi-labels en un ou plusieurs problèmes mono-label [TK07]. Une configuration largement utilisée pour l'audio est de construire un classifieur binaire différent h_l pour chaque tag l. Ce classifieur a pour tâche de placer les morceaux dans les catégories *oui* ou *non*, selon que le tag s'applique ou pas : $h_l : \mathcal{X} \longrightarrow \{-1; 1\}$ [TBTL08, BTYL09]. Cette configuration est illustrée dans la Figure 2.5. Les classifieurs y sont représentés par des triangles, dont sortent les choix possibles. La sous-figure 2.5a propose d'illustrer la classification mono-label par une reconnaissance de genre musical. On voit qu'un seul classifieur est construit, avec de multiples choix. Le morceau ne portera, au final, qu'une seule étiquette. Dans le cas multi-labels par contre, on construit plusieurs classifieurs, et le nombre de tags que l'on associera au morceau peut varier (sous-figure 2.5b).

Une deuxième solution pour traiter les problèmes multi-labels est d'utiliser un unique classifieur pour gérer tous les labels en même temps, sans diviser le problème avant l'apprentissage. La sortie de ces algorithmes est donc multi-variée. Ainsi, des tech-

niques de classification comme les arbres de décision [CK01] ou le boosting [SS99] ont été adaptées pour la gestion directe de problèmes multi-labels. D'autres systèmes de ce type sont présentés dans [TK07].

Pour la classification musicale, il est plutôt rare de prendre en compte plusieurs tags à la fois. Dans [HLBE11], Hamel *et al.* ordonnent les tags selon leur probabilité d'association avec un morceau donné. Cependant, un ordre ne définissant pas directement un ensemble de tags, on a donc besoin d'une étape supplémentaire pour obtenir la décision finale [TK07].

L'empilement de classifieurs (*stacked generalization*) est une autre manière de bénéficier de l'analyse conjointe de plusieurs tags. Cette technique est basée sur une formulation classique à $|\mathcal{L}|$ classifieurs indépendants, à laquelle on ajoute une étape de généralisation. Pour commencer, un classifieur indépendant est construit pour chaque tag, à partir des vecteurs de description, comme dans la 2.5b. Puis l'ensemble des prédictions (souvent probabilisées) de ces classifieurs est utilisé comme source de description pour un second ensemble de classifieurs. Ainsi, pour prédire un tag donné, le système utilisera des estimations faites sur l'ensemble des tags, ce qui permet de tirer parti des corrélations qui peuvent exister entre les tags. L'empilement de classifieurs ou des techniques analogues ont été utilisés dans [APRB07, BMEML08, YLLC09, NTTM09].

2.3.2. Algorithmes d'apprentissage

L'algorithme d'apprentissage constitue la partie centrale d'un système de tagging automatique. Les différents algorithmes de classification audio sont utilisables que ce soit en configuration mono-label ou multi-labels. Notamment, les algorithmes utilisés pour la classification en genre musical sont également transposables à une tâche de tagging [BMEM10].

Il existe deux types de modèles de classification :

– les **modèles discriminatifs**, cherchent à séparer au mieux les exemples d'apprentissage appartenant à des classes différentes. Pour un exemple donné, on cherche directement quelle est, d'après les valeurs des descripteurs, la classe la plus probable. Cette famille d'algorithmes est donc liée à des probabilités conditionnelles de type $p(y = \mathcal{C}|\mathbf{x})$, pour une classe donnée $\mathcal{C}$.

– les **modèles génératifs**, adoptent une démarche opposée, en considérant les exemples d'apprentissage comme générés aléatoirement par un processus aléatoire. Les exemples de chaque classe $\mathcal{C}$ sont censés être générés par un processus distinct, et c'est la loi de ce processus qu'on cherche à modéliser : $p(\mathbf{x}|y = \mathcal{C})$. Pour la classification d'un nouvel exemple, la classe la plus vraisemblable sera choisie.

Les techniques d'apprentissage les plus utilisées en classification audio sont : les machines à vecteurs de support [ME05, ME08b], les modèles de mélanges gaussiens [TBTL08] et le boosting [BK06, ELBMG07]. D'autres techniques ont été suggérées, comme des réseaux de neurones [BK06], ou même une méthode proche de celle des k plus proches voisins est [SLC07]. Leur utilisation pour la classification musicale reste marginale.

Les algorithmes mentionnés ici et leur technique d'apprentissage sont décrits plus en détails dans [DHS00].

2.3.2.1. Modèle de mélange gaussien

Un modèle de mélange gaussien (*Gaussian Mixture Model*, ou GMM) réalise une modélisation générative des données d'apprentissage. Les exemples de chaque classe $\mathcal{C}$ sont donc supposés générés aléatoirement. Leur distribution de probabilité est constituée d'une somme convexe de distributions gaussiennes multivariées :

$$p(\mathbf{x}|y = \mathcal{C}) = \sum_{k=1}^{K} \pi_{k,\mathcal{C}} \cdot \mathcal{N}(\mu_{k,\mathcal{C}}, \mathbf{\Sigma}_{k,\mathcal{C}}) \tag{2.1}$$

où les $\mathcal{N}(\mu_{k,\mathcal{C}}, \mathbf{\Sigma}_{k,\mathcal{C}})$ sont des fonctions gaussiennes multivariées. En outre, puisque $p(\mathbf{x}|y = \mathcal{C})$ est une somme convexe, on a $\pi_{k,\mathcal{C}} \geq 0$ et $\sum_{k=1}^{K} \pi_{k,\mathcal{C}} = 1$.

Lors de la classification d'un nouvel exemple inconnu, la règle est simple :

$$h(\mathbf{x}) = \begin{cases} 1 & \text{si } p(\mathbf{x}|y = 1) > p(\mathbf{x}|y = -1) \\ -1 & \text{sinon} \end{cases} \tag{2.2}$$

La complexité du modèle peut être réglée au préalable en choisissant le nombre de composantes K. Pour chaque classe $\mathcal{C}$, les paramètres à estimer sont les $\pi_{k,\mathcal{C}}$, $\mu_{k,\mathcal{C}}$

et $\mathbf{\Sigma}_{k,\mathcal{C}}$. Leurs valeurs sont calculées par l'algorithme espérance-maximisation (EM) [Moo96].

L'estimation de ces paramètres par l'algorithme EM peut se révéler très coûteuse en temps (notamment, complexité cubique par rapport au nombre de descripteurs [HCS11]). C'est pourquoi Turnbull *et al.* ont utilisé une technique plus rapide, mais donnant tout de même de bons résultats pour la classification [TBTL08].

2.3.2.2. Machines à vecteurs de support

Les Machines à vecteurs de support (*Support Vector Machines*, ou SVM) [BHW10] sont l'un des classifieurs les plus utilisés à ce jour. Ces machines cherchent à définir une frontière séparant les exemples de chaque classe dans l'espace des descripteurs $\mathcal{X}$.

En supposant que les données d'apprentissage sont linéairement séparables, on cherche un hyperplan dans l'espace des descripteurs $\mathbf{w} \cdot \mathbf{x} + b = 0$, pouvant discriminer les exemples des deux classes. La décision sera alors de la forme :

$$h(\mathbf{x}) = \begin{cases} 1 & \text{si } \mathbf{w} \cdot \mathbf{x} + b > 0 \\ -1 & \text{sinon} \end{cases} \tag{2.3}$$

L'apprentissage a donc pour but de chercher un hyperplan qui vérifie $y_i h(\mathbf{x}_i) \geq 0$, c'est à dire $y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 0$ pour tout $1 \leq i \leq I$. Cependant, comme on peut le voir dans la Figure 2.6, il existe en général une infinité d'hyperplans solutions. C'est pourquoi on introduit la notion de *marge* : celle-ci désigne la distance entre l'hyperplan et les exemples les plus proches de lui. Ainsi, si l'on normalise $\mathbf{w}$ et b de façon appropriée, on peut contraindre les exemples à vérifier : $y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1$. Dans cette configuration, la marge est : $\frac{1}{||\mathbf{w}||}$. La formulation classique du problème est la suivante :

$$\begin{array}{ll} \text{minimiser} & \frac{1}{2}||\mathbf{w}||^2 \\ \text{sous les contraintes} & y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1 \quad \forall i = 1, \ldots, I \end{array} \tag{2.4}$$

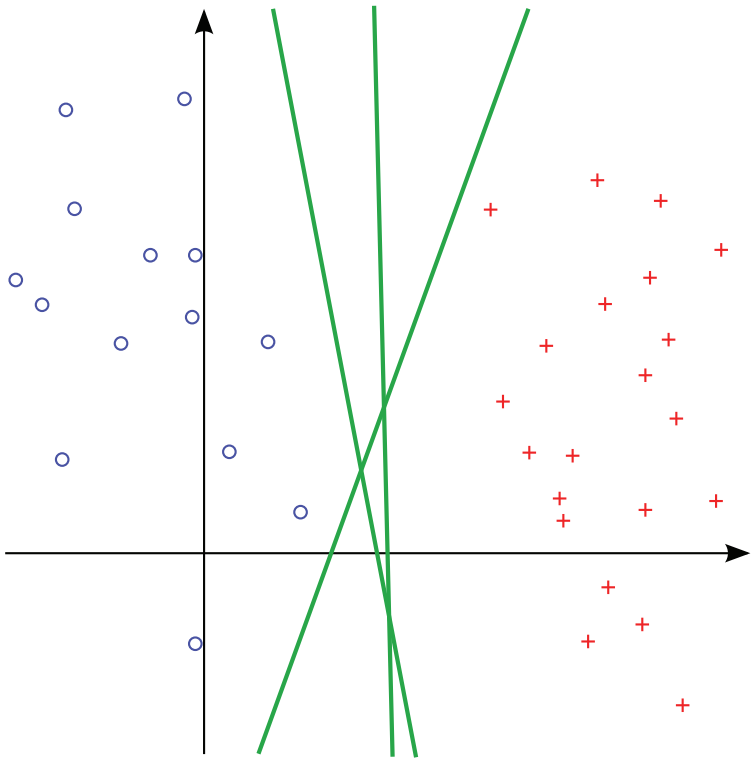


Figure 2.6.: Hyperplans séparant les données des deux classes dans un espace de descripteurs à deux dimensions.

En réalité, le problème est rarement linéairement séparable. Par conséquent, deux techniques, souvent utilisées conjointement, permettent d'adapter les SVM à des cas complexes. La première est l'introduction de *variables ressort* $\xi_i \geq 0$, permettant d'assouplir les contraintes : $y_i(\mathbf{w}\cdot\mathbf{x}_i+b) \geq 1-\xi_i$. L'apprentissage accepte donc qu'un exemple d'apprentissage jugé aberrant puisse se trouver dans la marge ou même être mal classifié. Le problème à résoudre devient :

$$\frac{1}{2}||\mathbf{w}||^2 - C\sum_{i=1}^{I}\xi_i \tag{2.5}$$

où $C > 0$ est fixé au préalable.

Une autre méthode pour traiter les cas non linéairement séparables est de projeter les données dans un espace de plus grande dimension, où elles seront séparables par un hyperplan. En réalité, comme l'apprentissage ne dépend que du produit

scalaire entre les exemples, ce produit scalaire suffit pour caractériser l'espace de redescription (cette astuce est appelée *astuce du noyau*).

Les SVM sont des outils très puissants mais nécessitent un réglage très rigoureux de leurs paramètres (paramètre C, choix et paramétrage du noyau), auquel ils se montrent particulièrement sensibles. En outre, la complexité de leur apprentissage, en temps et en mémoire, augmente rapidement avec le nombre d'exemples à traiter [LS09b]. Elle peut varier selon les implémentations et le noyau utilisés. On note que l'apprentissage est plus rapide lorsque la quantité de mémoire est grande. Pour des SVM à noyaux non-linéaires, des implémentations répandues comme SVM-*light*[3] ou Libsvm[4] présentent une complexité en temps se situant entre $O\left(N^2_{exemples} \times N_{descripteurs}\right)$ et $O\left(N^3_{exemples} \times N_{descripteurs}\right)$ [Joa98, BHW10].

2.3.2.3. Boosting

Le boosting est une technique qui consiste à entraîner un grand nombre de classifieurs basiques (dits « faibles »), de manière à les rendre complémentaires. Cette technique est décrite et analysée en détails dans le chapitre 3. Le temps d'apprentissage dépend du type de classifieur faible mais comme celui-ci est supposé assez simple, il est en général assez rapide à entraîner.

2.4. Fusion d'informations hétérogènes

En cherchant à décrire un morceau de la manière la plus complète possible, on peut parfois obtenir des descriptions très hétérogènes, provenant de différentes modalités ou extraites de manières différentes. Il existe de nombreuses manières de les fusionner. La littérature en traitement multimédia est particulièrement riche en techniques et systèmes de fusion multimodale (voir par exemple [AHEK10]).

Pour l'analyse de musique, ces techniques sont très rarement utilisées. La technique de fusion la plus utilisée est aussi la plus simple : la concaténation des vecteurs de description. Si les données sont extraites sur les mêmes trames d'analyse, c'est en

3. `http ://svmlight.joachims.org`
4. `http ://www.csie.ntu.edu.tw/∼cjlin/libsvm/`

effet une solution privilégiée. Si ce n'est pas le cas, il n'est pas rare de ramener les descriptions à des formes comparables pour pouvoir les concaténer [TC02, LS09a, ECL11, HBE12].

Les rares exemples d'autres types de fusion pour l'audio incluent [BYTL08], où les auteurs fusionnent plusieurs sources de données au moyen de l'algorithme d'Apprentissage à noyaux multiples [LCB+04, RBCG08]. Cet algorithme est basé sur l'apprentissage d'un SVM, et consiste à utiliser pour noyau K une combinaison convexe de plusieurs noyaux :

$$K = \sum_{n=1}^{N} \mu_n K_n$$

où K_n sont des noyaux et $\mu_n \geq 0$. Comme K est une somme convexe, on a également $\sum_{n=1}^{N} \mu_n = 1$. En construisant un noyau sur chaque source de données, l'algorithme permet alors d'apprendre en même temps que le SVM la combinaison des noyaux qui maximise la marge. Cet algorithme est donc utilisé dans [BYTL08] pour fusionner des MFCC, chroma, tags utilisateurs et des données issues de documents web. Le SVM avec noyaux multiples est une machine très puissante et très malléable, puisque l'on peut prendre des noyaux très différents. Cependant, cet algorithme est très complexe, nécessitant le paramétrage préalable de tous les noyaux, puis un apprentissage encore plus long qu'un SVM classique.

La fusion peut également intervenir au niveau de la décision. Dans ce cas, on mène un apprentissage séparé pour chaque source de données, puis on cherche une manière optimale de fusionner les décisions des classifieurs obtenus. Cela permet d'utiliser un classifieur différent pour les diverses sources de données. Une fusion tardive simple est la combinaison linéaire des décisions probabilisées [YLC+08]. Mais ces décisions peuvent avoir des significations ou des tendances différentes. C'est pourquoi, afin de rendre les sorties à fusionner plus homogènes, Barrington *et al.* proposent une méthode de calibration des scores [BTYL09]. Cette technique propose d'apprendre, sur les données d'apprentissage, à calibrer les prédictions des différents classifieurs, de manière à les rendre comparables [ZE02]. Elles pourront ainsi être combinées par simple moyenne (mais aussi maximum, médiane, *etc.*).

2.5. Données pour le tagging automatique

Le choix des données d'apprentissage est une question cruciale pour un système de tagging automatique. En effet, la taille des données, le choix des tags ainsi que la qualité des annotations influenceront grandement la qualité de l'apprentissage.

2.5.1. Récolte des annotations

Tout système d'apprentissage supervisé nécessite des données annotées. Ces annotations serviront de vérité-terrain à la fois pour l'apprentissage et pour l'évaluation des prédictions. Il existe plusieurs manières d'obtenir des données annotées pour le tagging automatique. Ces techniques se distinguent principalement par la taille des données qu'elles permettent de construire, et la qualité des annotations produites. D'une manière générale, les annotations rapides et peu coûteuses (donc pouvant traiter beaucoup de données) sont également de moindre qualité. Ce compromis se retrouve dans les quatre grandes méthodes d'annotation décrites pas Turnbull *et al.* [TBL08].

Questionnaire

La manière la plus simple d'obtenir des données est de demander à un certain nombre de personnes de remplir un questionnaire en écoutant les morceaux [TBTL08]. Il est très difficile ou très coûteux d'obtenir une grande quantité de données de cette manière. Par contre, cela permet d'obtenir des données relativement fiables par rapport aux autres méthodes d'annotation. Notons que le questionnaire est rédigé en amont de l'annotation. Cela permet de concentrer les auditeurs sur les tags qui nous intéressent, et qui ne sont pas forcément ceux qui auraient naturellement retenu leur attention.

Réseaux sociaux et services web

Des services en ligne comme la radio Last.fm ou le site MusicBrainz permettent aux utilisateurs d'associer des tags aux morceaux, artistes ou albums [BMEML08].

Un service spécifique donne accès aux données générées de cette manière. Cela permet d'obtenir une très grande quantité de données mais leur construction par des utilisateurs sans contraintes provoque deux problèmes.

Premièrement, l'absence d'un tag sur un morceau ne signifie pas forcément qu'il ne s'y applique pas bel et bien. Car cela peut simplement témoigner d'un désintérêt des utilisateurs pour le morceau ou pour l'information associée au tag en question. On parle parfois d'annotations « faibles » pour qualifier des données possédant cette caractéristique [LP08]. Dans le cas des tags collaboratifs, cette non-exhaustivité est très forte et on observe que les morceaux/artistes/albums les plus populaires rassemblent l'essentiel des informations, les autres disposant de très peu de données. Beaucoup de morceaux se retrouvent donc impropres à l'inclusion dans une base de données, puisque l'on sait que les données collaboratives ne deviennent informatives que lorsqu'elles sont très nombreuses. De même, certains tags sont très utilisés (notamment ceux relatifs au genre musical) tandis que d'autres, que l'on souhaiterait exploiter, ne sont pas très utilisés.

La deuxième difficulté provient de la non-restriction du vocabulaire. Cela fait apparaître d'une part des tags qui n'ont pas vraiment d'utilité pour la tâche désirée (par exemple : « *favorites* » ou « *mixtape-for-laura* »), et d'autre part, des groupes de tags ressemblants mais dont la synonymie ou l'inter-négation n'est pas évidente. Par exemple, s'il est clair qu'un morceau annoté « *opera* » peut être aussi annoté « *classical* », il n'est pas évident de regrouper « *man-singing* » et « *male-voice* » puisqu'une voix peut être parlée.

Jeu en ligne

On peut aussi sonder des auditeurs en utilisant un jeu en ligne [TLBL07, ME08a, LvA09]. Cette méthode peut fournir davantage de données qu'un questionnaire, tout en orientant, par conception du jeu, la teneur des annotations. L'idée est de créer un processus ludique incitant les joueurs à taguer les morceaux. Par exemple, dans Tag A Tune [LvA09], deux joueurs jouent l'un avec l'autre. Ils entendent un morceau de musique, et doivent le décrire sous forme de tags. En voyant les tags de l'autre, chaque joueur doit deviner s'il écoute le même morceau que son partenaire ou non.

La quantité de données disponibles est en continuelle augmentation mais reste encore bien en-dessous des tags sociaux. Chaque jeu conduit à des annotations portant

des caractéristiques différentes mais dans l'ensemble, la qualité de ces données reste moyenne, même si elle est meilleure que celle des réseaux sociaux. Par exemple, dans Tag A Tune, décrit plus haut, les annotations qui en résultent restent faibles et le vocabulaire n'est pas structuré. De plus, les auditeurs se concentrent uniquement sur les aspects les plus caractéristiques des morceaux, ce qui conduit à des annotations particulièrement parcimonieuses (très peu de tags par morceau). Par contre, les annotations créées par Tag A Tune sont plus fortement corrélées à l'audio que les tags sociaux.

Documents web

Une dernière approche pour collecter des tags est la récolte et l'analyse de documents disponibles sur le web [WE04, TBL06]. La quantité de documents consultables est pléthorique. Néanmoins, ce procédé semble peu utilisé en raison des données très bruitées qu'il produit généralement, pour la simple raison que les critiques musicaux ne décrivent pas la musique sous forme de mots-clés.

2.5.2. Choix d'une base de données

Plusieurs bases de données sont proposées à la communauté pour la recherche en tagging automatique. Les plus connues sont : Magnatagatune [LvA09] (méthode du jeu en ligne), Swat10k (ou CAL10k) [TKT10], Million Song Dataset [BMEWL11] (données en ligne) et CAL500 [TBTL08] (questionnaire). La base de données utilisée pour l'évaluation de la tâche de tagging automatique au Mirex (*Music Information Retrieval Evaluation eXchange*) est composée de deux parties : « MajorMiner Tag Dataset » (jeu en ligne), et « Mood Tag Dataset » (tags sociaux portant sur l'émotion) [5].

Le Million Song Dataset possède une quantité gigantesque de données, mais les tags sont peu fiables et faiblement annotés, le vocabulaire n'est pas structuré et les annotations sont extrêmement creuses. Magnatagatune et Swat10k se situent à un niveau intermédiaire tant pour la taille des données que pour la fiabilité des annotations. Notamment, les tags CAL10k sont tirés du site Web de la radio Pandora [6], dont les

5. Les détails sur cette base de données peuvent être consultés à l'adresse : `http ://www.music-ir.org/mirex/wiki/2013 :Audio_Tag_Classification`

6. `http ://www.pandora.com`

annotations sont réputées élaborées, fiables et très bien structurées. Cependant, son mode de collecte conduit à des annotations faibles, comme pour Magnatagatune. En outre, les annotations de ces bases sont très parcimonieuses (cette parcimonie est accentuée par la faiblesse des annotations et la mauvaise structuration du vocabulaire). La base CAL500, malgré sa taille modeste, possède les données les plus fiables. Nous décidons de privilégier la fiabilité des annotations en choisissant d'utiliser cette base.

La base de données CAL500 a été élaborée par l'équipe du Computer Audition Lab (University of California, San Diego). Elle contient 500 morceaux pop, annotés par la méthode du questionnaire. Chaque morceau est annoté en moyenne par trois ou quatre étudiants. Aucun présupposé n'est émis quant à l'expertise de ces annotateurs. Par contre, le questionnaire est très guidé. Par conséquent, les annotations qui résultent de cette enquête sont très propres, structurées, porteuses de sens et relativement fiables. De plus, CAL500 est la seule base qui possède des annotations fortes, c'est à dire qu'une valeur d'annotation négative signifie effectivement que le tag considéré ne s'applique pas au morceau.

Les caractéristiques de CAL500 nous conduisent à choisir cette base pour nos expérimentations, pour les raisons suivantes. Comme évoqué dans le chapitre d'introduction, les tags produits par notre système pourront être utilisés à la fois pour la recherche par mots-clés mais également pour une tâche de recommandation de morceaux similaires. Comme le vocabulaire des tags de CAL500 est bien structuré, un système entraîné correctement sur cette base pourra produire des annotations utiles et exploitables pour les tâches sus-mentionnées. Des tags pourront ainsi être produits automatiquement à la fois pour contrer le problème du démarrage à froid, mais aussi pour compléter un ensemble de tags sociaux lorsque ces derniers sont présents. C'est pour cette dernière raison que nous trouvons important d'exploiter des annotations fortes. En effet, des annotations de même qualité que les tags sociaux auront moins de chances d'être utiles en complément de ceux-ci.

Les 174 tags de CAL500 sont regroupés en six catégories, décrites dans le Tableau 2.1. Les tags des catégories *Genre* et *Émotion*, et certains de la catégorie *Général*, sont doublés de leur négation (par exemple, on trouve les tags *Émotion-Joyeux* et *Émotion-Pas_Joyeux*). Afin de renforcer l'expressivité des données, nous gardons uniquement les tags associés à au moins 50 morceaux (comme dans [BYTL08]). Ces tags sont listés dans l'Appendice C.

Catégorie	Description	Exemples
Genre	Genre musical	*Rock_classique*, *Électronique*
Émotion	Émotion ou état d'esprit	*Tendre*, *Triste*
Instrument	Instrument présent ou instrument solo	*Guitare_acoustique*, *Batterie*
Général	Considérations générales sur le morceau	*Texture_acoustique*, *Tempo_rapide*
Utilisation	Contexte d'écoute suggéré	*Lors_d'une_fête*
Voix	Description de la voix lead	*Émouvante, Puissante*

Tableau 2.1.: Catégories de tags de la base CAL500.

2.6. Évaluation

L'évaluation d'un système de classification musicale demande une certaine rigueur afin de bien comprendre la signification des métriques choisies.

2.6.1. Cadre d'évaluation pour la classification

L'évaluation des prédictions a pour but, à partir des prédictions d'un classifieur sur un ensemble de morceaux, de fournir des chiffres ou des courbes qui décrivent précisément les propriétés dont on souhaite juger. Ces propriétés dépendent de l'application envisagée pour le système de classification. Dans notre cas, on peut par exemple envisager une simple recherche de morceaux par tag, comme suggérée dans le chapitre d'introduction. On peut également imaginer un système de recommandation de morceaux par similarité sémantique, en rapprochant des morceaux portant des tags similaires (c'est d'ailleurs la méthode utilisée par la radio en ligne Pandora).

Les métriques mathématiques d'exactitude, bien que parfois remises en question pour leur signification uniquement quantitative [Law08, LWM+09], sont largement utilisées. Ce type de métriques mesure mathématiquement le degré de correspondance entre les prédictions et la vérité-terrain [HKTR04]. Nous les utilisons aussi car elles sont cohérentes avec applications potentielles de nos classifieurs, même si elles sont légitimement critiquées.

Dans cette thèse, nous utilisons deux métriques de *ranking*, c'est à dire des métriques qui évaluent la liste de morceaux, ordonnée par probabilité de présence d'un tag donné. Cette liste est comparée à la vérité-terrain binaire. Notre première mesure

est la *Mean Average Precision* (MAP). Elle peut être interprétée comme la précision moyenne, pour chaque valeur de rappel. Nous utilisons aussi l'aire sous la courbe *Receiver Operating Characteristic* (AROC). Cette courbe représente le rappel par rapport au taux de fausse alarme, calculé à chaque élément de la liste ordonnée.

L'Appendice A détaille le calcul et l'interprétation des principales métriques d'exactitude, dont la MAP et l'AROC.

2.6.2. Validité statistique des résultats

Afin d'obtenir des résultats fiables, toutes les évaluations effectuées dans le cadre de cette thèse ont été réalisées en validation croisée de 10 *folds*. Cela signifie qu'un ensemble de 450 morceaux est d'abord construit pour l'apprentissage, et les 50 morceaux restants sont utilisés pour les prédictions. Puis on répète l'opération neuf autres fois, tout en veillant à ce que les ensembles de prédiction soient disjoints. Les performances que nous présentons sont toutes obtenues par moyenne entre les dix folds. Ainsi, en répétant l'entraînement dix fois, on lisse certains comportements d'apprentissage liés aux particularités des données.

Pourtant, même avec une telle validation croisée, certaines différences observées entre deux mesures de performance, trop ténues, peuvent être dues au hasard. C'est pourquoi il est parfois indispensable de recourir à un test statistique, afin de vérifier la significativité des différences.

Par exemple, lorsque les performances sont mesurées en termes de taux de bonne reconnaissance, sur un seul jeu de données, le test de McNemar [Eve77] apparaît tout à fait indiqué.

Dans notre cas, le test de Student par séries appariées avec validation croisée [Die98] est particulièrement approprié pour tester la significativités d'évaluations telles que nous les menons sur CAL500.

Ces tests sont décrits plus en détails dans l'Appendice B.

2.7. Conclusion

Nous avons passé en revue les différents aspects du tagging automatique et plus généralement de la classification musicale. Ce chapitre évoque les principales techniques pour élaborer un système de classification automatique, mais également la question des données d'apprentissage et de l'évaluation, indispensables pour mener et interpréter des expériences.

Globalement, on constate que le cadre établi par Tzanetakis & Cook il y a plus de dix ans [TC02] a été beaucoup repris et exploré pendant des années, mais on observe qu'aujourd'hui, beaucoup de travaux tentent de le dépasser. Les descriptions se diversifient, prennent parfois en compte la dynamique temporelle, sont intégrées de diverses manières et sur diverses durées, et le signal n'est plus la seule source d'information. En outre, ces descriptions diverses appellent des algorithmes permettant de fusionner plusieurs types de données.

3. Boosting d'arbres de décision : un cadre performant et flexible

Résumé Ce chapitre présente l'algorithme de boosting pour l'apprentissage automatique, son procédé et ses fondements mathématiques, notamment dans le cas du boosting d'arbres. La plasticité de cet algorithme lui permet d'être adapté à plusieurs fonctions de coût, d'être utilisé pour la fusion d'autres classifieurs, ou de gérer les descripteurs manquants.

3.1. Introduction

Parmi les nombreuses techniques d'apprentissage automatique évoquées au chapitre 2 (*cf.* page 24), il en est une qui possède une configuration particulièrement flexible et aisément modulable : le boosting.

En effet, plutôt qu'un algorithme, le boosting est un principe donnant naissance à de nombreux algorithmes, à choisir, à combiner ou à construire selon les besoins. De plus, cette classe d'algorithmes s'appuie toujours sur l'exploitation itérative d'un autre algorithme d'apprentissage, dit *faible*, que l'on peut choisir. Bien entendu, ce choix influence grandement le comportement général du système.

Ce principe a été imaginé pour des problèmes de classification [FS96] mais sa plasticité le rend, comme nous allons le voir, tout à fait approprié pour des problèmes de régression.

3.2. Le boosting : une classe de méta-classifieurs

3.2.1. Un méta-classifieur itératif

Comme nous l'avons précisé dans l'introduction, le boosting est un principe général, dont la déclinaison donne naissance à de nombreux algorithmes.

Le principe de base est de combiner les sorties de nombreux classifieurs « faibles », supposés peu précis, mais dont la réunion donnera de bonnes performances. L'apprentissage consiste à entraîner itérativement plusieurs versions d'un modèle de classifieur faible, en influençant cet apprentissage de manière à rendre complémentaires les différents classifieurs qui en résulteront.

On retrouve bien ce principe dans l'algorithme Adaboost [FS96, FS99], peut-être le plus connu des algorithmes de boosting. L'algorithme Adaboost utilise une pondération des exemples d'apprentissage, pour représenter leur importance relative. Plus le poids d'un exemple est élevé, plus on accorde d'importance à la bonne classification de cet exemple. Ce sont ces poids qui vont permettre de construire des classifieurs faibles complémentaires les uns des autres : en concentrant chaque classifieur faible sur les exemples qui font défaut aux autres.

Adaboost est détaillé dans l'Algorithme 3.1. Au départ, les poids $w_{1,i}$ sont tous identiques. Puis, à chaque itération r, un classifieur faible est entraîné ; il doit tenir compte des valeurs des poids $w_{r,i}$. À la fin de l'itération, les poids des exemples bien classifiés sont diminués. Ainsi, au fil des itérations, les exemples difficiles à classifier verront leur poids augmenter, ce qui concentrera l'effort sur eux.

Dans la décision finale, la contribution de chaque classifieur faible recevra un coefficient, calculé en fonction de son taux d'erreur pondérée. La décision finale sera une somme seuillée de ces contributions. D'un point de vue théorique, le seuil t de cette décision est égal à zéro. En augmentant t, l'algorithme classifiera moins de morceaux comme positifs, ce qui pourra donner une plus grande confiance dans la bonne classification de ces morceaux positifs (donc une meilleure précision, *cf.* Appendice A).

L'entraînement d'Adaboost, comme les SVM, ne se contente pas d'apprendre à bien classifier les exemples d'apprentissage : il maximise une marge, ce qui lui donne une bonne capacité à généraliser.

Algorithme 3.1 L'algorithme d'apprentissage Adaboost.

Paramètre: Exemples annotés $(\mathbf{x}_i, y_i)$, $1 \leq i \leq I$
Paramètre: Modèle de classifieur faible $\mathcal{H}$

$w_{1,i} \longleftarrow \frac{1}{I}$
pour $r = 1, \ldots, R$ **faire**
  $w_{r,i} \longleftarrow \frac{w_{r,i}}{\sum_{j=1}^{I} w_{r,j}}$ *// Normaliser les poids*
  Entraîner le classifieur h_r avec le modèle $\mathcal{H}$ et les poids $w_{r,i}$, sur les cibles y_i
  // Calculer le taux d'erreur pondérée
  $\epsilon_r \longleftarrow \sum_i w_{r,i} \, \mathbb{1}_{h_r(\mathbf{x}_i) \neq y_i}$
  // Coefficient associé à h_r
  $\alpha_r \longleftarrow \log \frac{1}{\beta_r}$, où $\beta_r = \frac{\epsilon_r}{1-\epsilon_r}$
  // Mettre à jour le poids des exemples
  pour chaque exemple i **faire**
    si $\mathbf{x}_i$ bien classifié par h_r **alors**
      $w_{r+1,i} \longleftarrow w_{r,i} \, \beta_r$
    sinon
      $w_{r+1,i} \longleftarrow w_{r,i}$
    fin si
  fin pour
fin pour

Sortie: $H(\mathbf{x}) = \begin{cases} 1 & \text{si} \quad \sum_r \alpha_r h_r(\mathbf{x}) > t \\ 0 & \text{sinon} \end{cases}$

3.2.2. Un modèle flexible

On trouve de nombreux algorithmes basés sur le principe du boosting. L'algorithme Adaboost lui-même a été décliné en plusieurs versions lui permettant, par exemple, de traiter des problèmes multi-classes. Mais des algorithmes différents ont également été créés, notamment pour faire de la régression logistique, pour utiliser des classifieurs faibles ayant des sorties réelles [FHT00], ou pour des problèmes de ranking [FISS03]. Nous allons, dans la suite de ce chapitre, voir plus en détails quelques exemples d'algorithmes proposant des adaptations du principe de boosting.

Mais la configuration des algorithmes de boosting varie non seulement par la procédure de boosting en elle-même, mais aussi par le choix du modèle de classifieur faible. En effet, il n'est pas important que le classifieur faible possède de bonnes performances pour le type de données analysées (d'où sa qualification de « faible »). Il a d'ailleurs été montré qu'à chaque itération r, tant que le classifieur faible h_r prédit mieux que le hasard ($\epsilon_r < 0.5$), il fait baisser l'erreur du classifieur fort [FS97].

Le fait de prédire mieux que le hasard (en termes d'erreur pondérée) est en fait la seule supposition que l'on fait sur le modèle de classifieur faible, même si ses prédictions sont à peine meilleures. Cela permet d'orienter le choix de ce modèle en fonction d'autres critères, comme l'interprétabilité de l'apprentissage, la robustesse face à certaines transformations de l'espace de description, la rapidité d'apprentissage, *etc.*

3.3. Le cas particulier des arbres de décision

Le boosting et les arbres de décision entretiennent des connexions étroites. Notamment, les arbres boostés présentent de bonnes performances et des propriétés avantageuses. Nous évoquerons ces propriétés dans un second temps, après avoir présenté le principe et la construction des arbres.

3.3.1. Définition et construction

Les arbres de décision sont des classifieurs récursifs très simples conceptuellement, mais souvent utilisés pour leur bon comportement en combinaison [LP08]. Dans

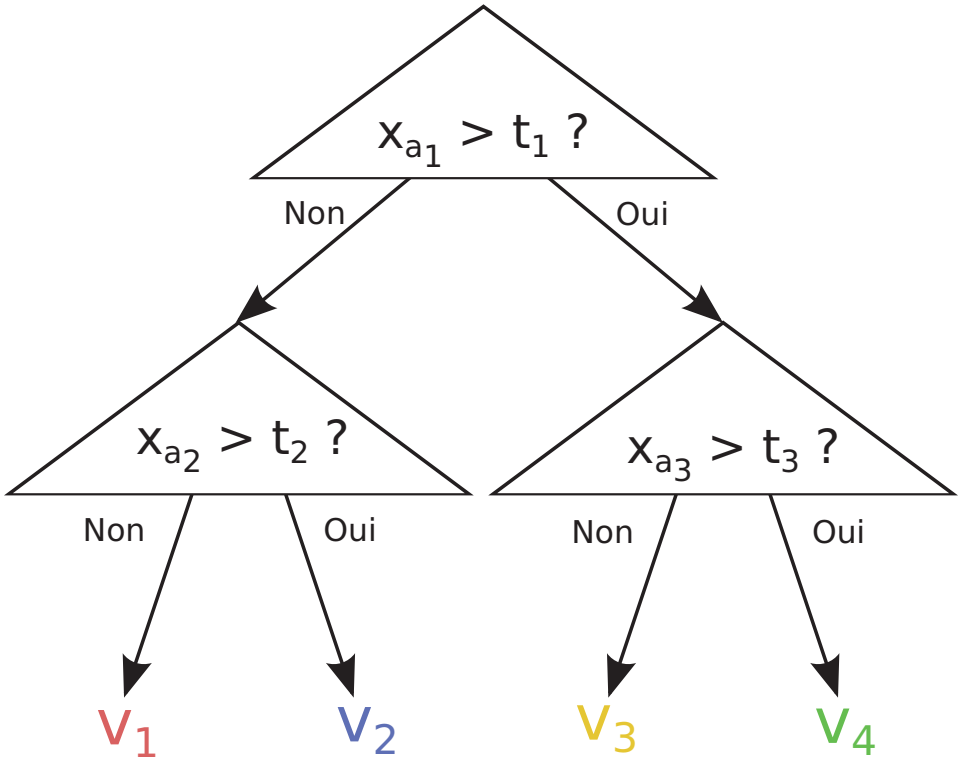


Figure 3.1.: Un arbre de décision binaire de profondeur 2.

cette thèse, nous considérerons uniquement des arbres binaires, car il est prouvé que les autres arbres pouvaient toujours se ramener au cas binaire [HTF09].

Dans un arbre de décision, la classification d'un exemple suit un chemin depuis la racine vers une feuille. Un exemple d'arbre est représenté dans la Figure 3.1. À chaque nœud n rencontré, on compare un coefficient x_{d_n} du vecteur $\mathbf{x}$, à un seuil donné t_n [1]. Selon que la valeur de x_{d_n} est inférieure ou supérieure à ce seuil, le chemin de classification continue dans le fils gauche ou droit du nœud n. Lorsque l'on atteint une feuille, le classifieur retourne la valeur inscrite dans celle-ci.

Il existe plusieurs manières de construire un arbre de décision. Les deux techniques principales sont CART (*Classification And Regression Trees*) [YH99] et C4.5 [Qui93]. Ce sont des techniques récursives, où chaque nœud contient un ensemble d'exemples d'apprentissage (on commence avec la racine, qui contient toutes les données). On va alors chercher une variable à tester et un seuil, de manière à séparer cet ensemble en deux sous-ensembles les plus « purs » possibles. Cela signifie que chaque sous-

1. Il existe également des arbres dont les nœuds utilisent plusieurs variables. L'opération de test prend alors la forme $x_{d_n} + x_{d'_n} > t_n$.

ensemble contiendra des exemples appartenant autant que possible à une seule classe. Il existe plusieurs manières de mesurer la pureté d'un nœud. Nous garderons le critère de Gini pour ses bonnes propriétés lors de la construction d'arbres [DHS00].

Les arbres de décision peuvent être construits avec une complexité raisonnable par rapport à d'autres méthodes (*cf.* sous-section 2.3.2 p. 24). La librairie Scikit-learn [2] [PVG+11] propose par exemple une version optimisée de CART, construisant un arbre en $O\left(N_{exemples} \times \log\left(N_{exemples}\right) \times N_{descripteurs}\right)$ opérations.

3.3.2. Comportement des arbres boostés

Dès son introduction dans la littérature spécialisée en apprentissage automatique, la technique du boosting a suscité un vif intérêt dans la communauté des chercheurs en apprentissage, mais aussi en statistiques [FHT00].

On remarqua alors rapidement que le choix des arbres de décision en tant que classifieur faible donnait des propriétés tout à fait intéressantes et appréciables [Fri01]. Pour commencer, les performances sont très bonnes. À l'époque, Breiman nomme même cette configuration « le meilleur classifieur *presse-bouton* » (« *best off-the-shelf classifier* ») [Bre98], dans le sens où elle ne nécessite aucun pré-requis ou réglage en fonction des données à traiter. Cette constatation est d'autant plus intéressante que les arbres de décision sans boosting ne sont pas toujours très performants.

Il faut également noter que d'autres propriétés des arbres de décision se transfèrent au boosting d'arbres : l'apprentissage réalisé est ainsi aisément interprétable, rapide, et effectue une sélection embarquée d'attributs. De plus, la nature des arbres, et leurs propriétés statistiques permettent de donner naissance à certains algorithmes de boosting qu'on ne peut utiliser qu'avec ce classifieur faible (*cf.* section 3.4).

Le choix des arbres de décision comme classifieur faible apparaît donc tout à fait approprié, et c'est celui que nous choisissons pour les expériences présentées dans cette thèse.

Pour le boosting, il n'est pas rare d'utiliser de simples arbres à seulement deux branches (« souches de décision »), qui consistent donc en une classification utilisant un simple seuil sur un attribut. Cette extrême simplicité est comblée par la

2. `http ://scikit-learn.org/stable/modules/tree.html#complexity`

grande quantité d'arbres que l'on va booster. D'ailleurs, il est difficile de distinguer un clair avantage pour le boosting, entre des « souches » et des arbres plus complexes [FHT00]. Nous avons également pu constater cet équilibre dans une expérience préliminaire [FELR11], c'est pourquoi lors des travaux présentés dans cette thèse, nous utilisons uniquement des arbres à deux branches.

En outre, les souches de décision comportent l'avantage d'être très rapides à construire (et leur temps de classification est constant). Le boosting de ces modèles est donc en principe une configuration rapide à entraîner.

3.4. Adaptation à plusieurs fonctions de coût

Dans leurs travaux, Friedman, Hastie & Tibshirani [FHT00, Fri01, HTF09] ont remarqué que l'on pouvait voir le boosting comme une descente de gradient, utilisant la perte exponentielle comme fonction de coût : $L_{\exp}(y,H(\mathbf{x})) = \exp(-yH(\mathbf{x}))$. On peut alors généraliser la procédure à d'autres fonctions de coût [Fri01], notamment des coûts pour la régression, ou des coûts différents pour la classification [LS10].

Malheureusement, l'utilisation de ces autres fonctions ne génère pas toujours des procédures aussi simples que l'élégant système de pondération d'Adaboost. Cependant, pour certaines d'entre elles, l'algorithme peut être simplifié si l'on utilise des arbres de décision comme classifieurs faibles.

Par chance, une des fonctions les plus utilisées pour mesurer l'erreur de régression, donne un algorithme de boosting très simple et intuitif. Il s'agit de l'erreur quadratique : $L_{\text{quad}}(y,H(\mathbf{x})) = (y - H(\mathbf{x}))^2$. Les calculs sont décrits dans l'Algorithme 3.2. Il s'agit tout simplement, à chaque itération, de tenter de prédire les résiduels de prédiction des itérations précédentes.

Le boosting est donc utilisable pour résoudre des problèmes de classification et de régression, avec différentes fonctions de coût.

3.5. Le boosting pour la fusion de classifieurs

La fusion de classifieurs peut être utile pour réunir plusieurs sources de description, ou pour appliquer un traitement spécifique à différentes parties des données.

Le boosting semble tout à fait pertinent pour cette tâche. Les classifieurs à fusionner seront alors utilisés dans le boosting comme classifieurs faibles. Ils seront tous entraînés à chaque itération, puis le plus performant sera sélectionné pour le classifieur fort final. Cette technique de fusion a été utilisée de nombreuses fois, notamment pour l'analyse d'images [VJ01], et plus récemment pour le tagging audio [BTYL09]. Cette fusion de plusieurs classifieurs faibles permet, en outre, de réaliser facilement un apprentissage à noyaux multiples [JL10].

On démontre facilement que cette configuration est bel et bien équivalente à un algorithme de boosting, tel que l'Algorithme 3.1, avec un unique classifieur faible. Dans ce cas, le classifieur faible réalise lui-même la fusion. Son modèle est donné dans l'Algorithme 3.3.

Cette adaptation pour la fusion est tout à fait compatible avec les modifications de fonction de coût décrites dans la section 3.4.

3.6. Gestion des descripteurs manquants

Il arrive parfois que des descripteurs ne soient pas disponibles sur certains exemples d'apprentissage. C'est par exemple le cas lorsqu'on cherche à récupérer un descripteur sur Internet et que le morceau n'est pas trouvé sur le site voulu (*cf.* p. 19).

3.6.1. L'algorithme Ada-ABS

Il est possible de pallier l'absence d'un descripteur au niveau du classifieur faible : celui-ci donne alors une réponse sur chaque exemple quoi qu'il arrive, et le boosting se passe normalement. Il existe d'ailleurs des techniques pour permettre à des arbres de décision de donner une réponse, même en l'absence d'un descripteur [DHS00].

Algorithme 3.2 L'algorithme d'apprentissage régressif LS_Boost.

Paramètre: Exemples annotés $(\mathbf{x}_i, y_i)$, $1 \leq i \leq I$
Paramètre: Modèle de régresseur faible $\mathcal{H}$

initialiser les cibles $c_{1,i} = y_i$

pour $r = 1, \ldots, R$ **faire**
Entraîner le régresseur h_r avec le modèle $\mathcal{H}$ sur les cibles $c_{r,i}$

// Calculer les résiduels de prédiction
$c_{r+1,i} = c_{r,i} - h_r(\mathbf{x}_i)$
fin pour

Sortie: $H(\mathbf{x}) = \sum_r h_r(\mathbf{x})$

Algorithme 3.3 Modèle de classifieur faible pour la fusion de classifieurs par boosting.

Paramètre: Exemples annotés $(\mathbf{x}_i, y_i)$, $1 \leq i \leq I$
Paramètre: Poids des exemples w_i
Paramètre: Modèles de classifieurs à fusionner $\mathcal{H}_1, \ldots, \mathcal{H}_M$

Entraîner les classifieurs h_m avec les modèles $\mathcal{H}_m$ et les poids w_i, sur les cibles y_i

// Calculer l'erreur pondérée
$\epsilon_m \longleftarrow \sum_i w_i \, \mathbb{1}_{h_m(\mathbf{x}_i) = y_i}$

// Meilleur classifieur
$m_{\text{best}} = \text{argmin}_m \, \epsilon_m$

Sortie: Retourner $H_{m_{\text{best}}}$

Cependant, Smeraldi *et al.*, dans [SDPS10], proposent un moyen de gérer l'absence de descripteurs au niveau du boosting. Ils montrent aussi que cette méthode obtient de meilleurs résultats que la gestion des descripteurs manquants par le classifieur faible.

Dans cette méthode, si le classifieur faible h_r a besoin de lire un descripteur qui manque sur l'exemple $\mathbf{x}_i$, il s'abstient de répondre : $h(\mathbf{x}_i) = 0$. La mise à jour des poids pour l'itération suivante $r+1$ tient compte des exemples sur lesquels h_r n'a pas répondu. On a donc désormais $h(\mathbf{x}_i) \in \{-1,0,+1\}$ mais les labels sont toujours binaires : $y_i\{-1,+1\}$.

3.6.2. Relation avec Adaboost

Dans le cas où le classifieur faible ne s'abstient jamais, on peut montrer facilement que l'algorithme présenté dans [SDPS10] est équivalent à Adaboost.

Premièrement, dans Ada-ABS, dans le cas où $W_a = 0$ (donc pas d'abstention), la règle de mise à jour à la fin de l'itération devient :

$$w_{r+1,i} \longleftarrow \begin{cases} \frac{1}{2W_b} w_{r,i} & \text{si} \quad \mathbf{x}_i \text{ est bien classifié par } h_r \\ \frac{1}{2W_m} w_{r,i} & \text{si} \quad \mathbf{x}_i \text{ est mal classifié par } h_r \end{cases} \tag{3.1}$$

Puisque les poids ne sont que des quantités relatives, cette mise à jour revient à multiplier les exemples bien classifiés par $\frac{W_m}{W_b}$. Ce rapport est égal au β_r de l'Algorithme 3.1.

D'autre part, en utilisant ce β_r, le calcul du coefficient α_r devient :

$$\alpha_r \longleftarrow \log \frac{W_b}{W_m} = \log \frac{1}{\beta_r} \tag{3.2}$$

3.7. Conclusion

Nous avons vu au cours de ce chapitre que le boosting était un principe d'apprentissage performant, et surtout très plastique. En effet, son aspect modulaire (l'abs-

Algorithme 3.4 Ada-ABS : Adaboost pour des classifieurs faibles qui s'abstiennent.

Paramètre: Exemples annotés $(\mathbf{x}_i, y_i)$, $1 \leq i \leq I$
Paramètre: Modèle de classifieur faible $\mathcal{H}$

$w_{1,i} \longleftarrow \frac{1}{I}$

pour $r = 1, \ldots, R$ **faire**
Entraîner le classifieur h_r avec le modèle $\mathcal{H}$ et les poids $w_{r,i}$, sur les cibles y_i

// Poids des exemples non classifiés, bien classifiés et mal classifiés
$W_a \longleftarrow \sum_i w_{r,i} \mathbb{1}_{h_r(\mathbf{x}_i)=0}$
$W_b \longleftarrow \sum_i w_{r,i} \mathbb{1}_{h_r(\mathbf{x}_i)=y_i}$
$W_m \longleftarrow 1 - W_a - W_b$

// Coefficient associé à h_r
$\alpha_r \longleftarrow \log \frac{W_b}{W_m}$

// Mettre à jour le poids des exemples
pour chaque exemple i **faire**

$$w_{r+1,i} \longleftarrow \begin{cases} w_{r,i} / (W_a + 2\sqrt{W_b W_m}) & \text{si} \quad \mathbf{x}_i \text{ n'est pas classifié par } h_r \\ w_{r,i} / (W_a \sqrt{W_b / W_m} + 2W_b) & \text{si} \quad \mathbf{x}_i \text{ est bien classifié par } h_r \\ w_{r,i} / (W_a \sqrt{W_m / W_b} + 2W_m) & \text{si} \quad \mathbf{x}_i \text{ est mal classifié par } h_r \end{cases}$$

fin pour
fin pour

Sortie: $H(\mathbf{x}) = \begin{cases} 1 & \text{si} \quad \sum_r \alpha_r h_r(\mathbf{x}) > t \\ 0 & \text{sinon} \end{cases}$

traction faite sur le classifieur faible, la décomposition de la solution en une somme de fonctions) rend très facile son adaptation à certains problèmes spécifiques. Il peut ainsi être utilisé avec différents classifieurs faibles de propriétés différentes, être adapté pour minimiser plusieurs fonctions de coût et gérer la fusion de différents classifieurs ou des descripteurs manquants.

Nous avons vu que l'utilisation du boosting avec des arbres constituait une configuration aussi efficace qu'adaptative. Elle nécessite en effet très peu de réglages, même pour fusionner des sources de données. Cette propriété est très avantageuse par rapport à d'autres techniques comme des SVM à noyaux multiples (*cf.* section 2.4).

Ces propriétés en font une technique de choix pour l'apprentissage automatique, et nous en tirons profit au cours des travaux présentés dans cette thèse.

4. Fusion souple d'annotateurs et régression

Résumé Dans ce chapitre, nous nous intéressons au calcul des valeurs-cibles pour l'apprentissage. Cette vérité-terrain, construite en fusionnant les réponses de plusieurs annotateurs, est généralement binaire. Pourtant, l'application d'un tag donné à un morceau n'est pas toujours évidente. Nous proposons donc de construire une vérité-terrain continue, tenant compte de l'incertitude qui peut émerger des annotations. Les nouveaux scores seront appris par boosting régressif.

4.1. L'annotation, génératrice d'incertitude

Dans la plupart des cas, le tagging automatique est considéré comme une tâche de classification bi-classes : pour un morceau donné, un tag est soit présent, soit absent. Cela peut correspondre aux valeurs-cibles 1 et -1 pour l'apprentissage. Cependant, la collecte des annotations se fait la plupart du temps d'après le jugement et l'expérience d'un certain nombre d'auditeurs. Or la musique étant une source complexe de données, les réactions des auditeurs à un morceau de musique sont rarement aussi claires et unanimes qu'un simple *oui* ou *non*. Une incertitude sur l'association du tag peut ainsi émerger des annotations.

Cette incertitude peut émerger à deux niveaux : incertitude individuelle et désaccord inter-annotateurs.

Pour commencer, un annotateur peut exprimer un doute lors de sa réponse. Par exemple, pour CAL500, le questionnaire dispose de cinq niveaux de pertinence pour les tags d'émotion, et de quatre niveaux pour les tags de présence d'instruments

(Absent, *Peut-être*, *Présent* et *Au-premier-plan*). Même pour certains tags qui pourraient *a priori* être bien représentés par une valeur binaire (comme la tonalité, majeure ou mineure), un annotateur peut explicitement répondre qu'il ne sait pas. Cela qui introduit une troisième valeur de réponse possible. Cette diversité dans les choix de réponses est tout à fait justifiée par le désir de décrire au mieux la réaction des auditeurs. Cependant, elle peut apparaître mal exprimée par des scores binaires.

Imaginons par exemple que tous les annotateurs répondent « *Je ne sais pas* » pour le tag *Morceau-Enregistrement_en_studio*. C'est en effet un tag parfois difficile à trancher car la différence entre le son de studio et de concert ne s'entend pas toujours distinctement. Il paraît donc tout à fait approprié de laisser les annotateurs exprimer leur indécision. Mais dans ce cas, une fusion binaire forcera le morceau à porter une des deux étiquettes (*Studio* ou *Concert*), ce qui peut sembler arbitraire.

La deuxième source d'incertitude est constituée par les désaccords entre les différents annotateurs. En effet, il est de plus en plus rare qu'une base de données soit annotée par une seule personne : disposer de plusieurs annotateurs permet d'obtenir des résultats plus fiables, reflétant mieux une réaction générale des auditeurs au morceau considéré. Mais dans ce cas, il n'est pas rare que ces annotateurs donnent des réponses différentes, voire contradictoires.

Par exemple, dans la Figure 4.1, on voit que sur le morceau « Army of Me », deux des quatre annotateurs ont répondu « *Présent* » pour le tag *Instrument-Batterie*, tandis que les deux autres ont considéré qu'il n'y avait pas de batterie (ces derniers auraient plutôt identifié une boîte à rythmes). Puisqu'un jeu d'annotations n'est jamais parfaitement objectif, il importe finalement moins de savoir qui a raison [1], que de constater que les auditeurs ont réactions différentes à l'écoute du morceau de Björk. Cette dernière constatation exprimera mieux l'expérience des auditeurs face au contenu audio du morceau, ce que l'on cherche en général à modéliser. Pourtant, ce type de réponses contradictoires est le plus souvent fusionné vers un score binaire, ce qui exprime mal le désaccord.

1. Il s'agit en fait d'une boîte à rythmes, d'après le livret du disque.

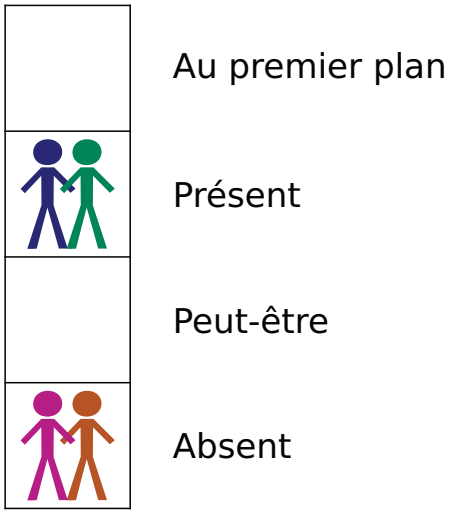


Figure 4.1.: Annotations du tag *Instrument-Batterie* pour le morceau « Army of Me » de Björk dans CAL500.

4.2. Vers une vérité-terrain plus souple

Dans la littérature, certains types d'annotations demeurent largement traités comme des informations objectives, bien que les données soient souvent annotées par des humains. L'annotation a alors plutôt pour but de s'approcher le plus possible de cette vérité objective. C'est le cas par exemple pour l'instrumentation, où il existe effectivement une vérité factuelle. Mais c'est aussi le cas dans beaucoup d'études sur la reconnaissance du genre musical. En effet, bien que le genre soit souvent discutable [AP08, SCBG08], la vérité-terrain est principalement représentée par des catégories rigides, définitives et souvent imperméables. Scaringela *et al.* opposent même le genre à des « catégories perceptuelles » [SZM06]. En conséquence, les morceaux sont alors rangés dans des catégories précises et rigides, et on cherche à construire un classifieur pour ranger les morceaux dans la bonne classe. Il est par contre moins courant d'affirmer que l'émotion véhiculée par un morceau constitue une donnée parfaitement objective [Bow09]. C'est pourquoi plusieurs travaux sur la reconnaissance d'émotions déjà ont tenté de rendre compte de la précision et de la variété des réponses des annotateurs.

Par exemple, les différentes émotions peuvent être situées dans un espace continu en deux ou trois dimensions (*valence*, *intensité*, éventuellement *dominance*). La procédure d'annotation consiste alors à situer les morceaux dans l'espace des émotions, ou à les ordonner suivant une dimension. Les scores-cibles sont obtenus en prenant

la moyenne des réponses individuelles. On peut alors formuler la tâche d'analyse d'émotions comme un problème de régression [YLSC08] ou de *ranking* [YC10]. Dans le cas de la régression, on cherche à prédire une valeur continue, tandis que le *ranking* consiste à ordonner les documents. Ces deux tâches permettent d'avoir des cibles à apprendre plus souples que de simples catégories bien définies. Cette formulation convient très bien à la tâche de reconnaissance d'émotions, mais elle ne construit pas de catégories. Elle nécessiterait donc des étapes supplémentaires de traitement pour être adaptée au tagging automatique.

L'apprentissage multi-instances peut également être utilisé pour assouplir l'une des deux catégories d'exemples (positifs ou négatifs). Son principe est de considérer un processus d'annotation effectué à une granularité plus grossière que les exemples d'apprentissage. Par exemple, si un morceau est annoté « Instrument-Piano », cela ne signifie pas que le piano est audible à tous les instants. Ainsi, avec des durées de description de quelques secondes, certains exemples annotés avec ce tag peuvent décrire des moments où le piano est absent. Le problème est le même si l'on cherche à appliquer sur des morceaux des tags associés auparavant à l'album entier ou à l'artiste. Chaque groupe d'exemples annotés positifs en même temps contient donc un nombre indéterminé d'exemples négatifs. Des algorithmes ont été proposés pour estimer, pendant l'apprentissage, le nombre de ces exemples erronés. Mandel & Ellis ont appliqué des SVM de ce type au tagging automatique audio [2] [ME08b].

Comme nous l'avons vu au chapitre 2 (sous-section 2.6.1), il est courant d'évaluer un algorithme de tagging sur une tâche de *ranking* des morceaux [BYTL08, TBTL08, BTYL09]. On demande alors au système de prédiction, non pas d'estimer la catégorie binaire des exemples de test, mais de les ordonner selon leur probabilité d'association avec le tag en question. Le système n'a alors pas besoin de décider quels exemples seront positifs et lesquels seront négatifs. Ce procédé permet d'observer le comportement de l'algorithme de manière plus détaillée qu'avec des prédictions binaires. Cependant, les données d'apprentissage et d'évaluation possèdent là encore une vérité-terrain binaire, qui comporte les limitations citées plus tôt. Même ces prédictions souples sont donc comparées à une vérité-terrain binaire.

Dans [YLLC09], les auteurs tirent parti des corrélations entre les tags pour construire

2. Little & Pardo, dans [LP08], constatent également qu'un instrument est rarement présent sur toute la durée d'un morceau. Cependant, ils mènent un apprentissage classique en considérant simplement les exemples faussement positifs comme des données mal annotées.

plusieurs catégories intermédiaires ordonnées, qui représentent différents niveaux de confiance. Par exemple, des annotations *Morceau-Très_dansant* et *Morceau-Calme* rencontrées sur un même morceau apparaîtront relativement peu fiables. En exploitant les corrélations des tags, on peut ainsi enrichir les annotations. Pour chaque tag, au lieu des catégories *Oui/Non*, on en obtient alors quatre : *Certain*, *Probable*, *Peu-probable* et *Très-improbable*. Cependant, ces catégories sont fabriquées et déduites à partir des scores binaires. L'incertitude des annotateurs est donc seulement supposée, au lieu d'être directement exploitée.

On peut également enrichir les sorties des classifieurs en ordonnant les tags selon leur probabilité d'association avec un morceau donné [HLBE11]. Mais là encore, l'ordre est obtenu à partir de scores binaires.

Globalement, pour l'assouplissement de la vérité-terrain, on distingue deux idées :
- pour la reconnaissance d'émotion, on utilise des représentations plus riches pour décrire les états d'esprit en évitant de dessiner des catégories. Les annotations sont alors précises et souples, et les différents annotateurs sont souvent fusionnés par moyenne.
- pour le tagging automatique, on tente de créer des catégories plus riches ou plus perméables, ou encore d'exploiter les corrélations entre les labels, même si ceux-ci restent fondamentalement binaires.

4.3. Fusion souple des annotateurs

Tout en restant dans une tâche de tagging, nous cherchons à exploiter directement l'information sur l'incertitude des annotations, afin de mieux exprimer les doutes et de permettre un apprentissage plus flexible. C'est pourquoi nous proposons une fusion d'annotateurs qui produit une vérité-terrain continue, et non plus seulement cantonnée aux valeurs *oui* ou *non*. Ces scores sont compris entre -1 (présence du tag très improbable) et 1 (le tag est très certainement associé au morceau).

Des scores souples sont déjà proposés par le Computer Audition Lab avec la base de données CAL500, mais ceux-ci ne sont pas pleinement documentés et leur méthode de construction à partir des annotations individuelles demeure, dans certains cas, difficiles à comprendre. Par exemple, pour le tag *Morceau-Rythmique_puissante*, sur

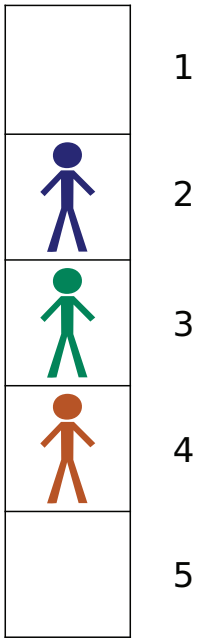


Figure 4.2.: Annotations du tag *Morceau-Rythmique_puissante* pour le morceau « Boogie Woogie Bugle Boy » des Andrews Sisters dans CAL500.

le morceau « Boogie Woogie Bugle Boy » des Andrews Sisters, les trois annotateurs ont répondu comme illustré dans la Figure 4.2. Le score proposé par le Computer Audition Lab est -1, ce qui ne rend pas compte des réponses mitigées des annotateurs. De plus, le score du tag *Morceau-Pas-Rythmique_puissante* (négation du tag précédent) est également de -1.

Pour ces raisons, nous proposons de nouveaux scores souples, à partir des réponses individuelles des annotateurs.

4.3.1. Méthode de fusion

Pour commencer, chaque réponse possible est convertie en une valeur $v \in [-1,1]$. Les valeurs consécutives sont régulièrement espacées. De plus, $v = -1$ et $v = 1$ doivent toujours correspondre à des réponses possibles. Par exemple, pour les tags indiquant la présence d'un instrument de musique, il existe quatre réponses possibles (*cf.* section 4.1). Ces choix sont donc convertis en -1, $-\frac{1}{3}$, $\frac{1}{3}$ et 1.

Ensuite, pour un tag et un morceau donnés, il existe plusieurs manières de fusionner les réponses des différents annotateurs. Pour les scores binaires de CAL500, un

tag est considéré comme « positif » si 80% des sujets jugent que le tag s'applique [BYTL08]. Parmi d'autres méthodes de fusion, on trouve : le vote majoritaire (le score correspond alors à la catégorie la plus souvent choisie), ou bien prendre la moyenne (éventuellement seuillée) des valeurs d'annotations individuelles. Le vote majoritaire permet d'exprimer l'indécision individuelle des annotateurs, mais ne tient pas compte de la variabilité des réponses. La moyenne seuillée reflète également mal l'incertitude puisqu'elle produit des scores binaires ; c'est pourquoi nous choisissons de prendre la moyenne simple des scores individuels. La moyenne est aussi utilisée par Yang *et al.* dans [YLSC08, YC10], mais pour un type différent de vérité-terrain (*cf.* section 4.2).

Soit V_c le score continu correspondant au morceau considéré. On a donc :

$$V_c = \frac{1}{K} \sum v_k \tag{4.1}$$

où v_k est la valeur correspondant au choix de l'annotateur k, et K est le nombre d'annotateurs.

Pour les tags « négatifs » (par exemple *Émotion-Pas-Joyeux*), la valeur est simplement $V = -P$, où P est la valeur associée au tag « positif » correspondant.

Sur l'exemple de la Figure 4.2, cette nouvelle méthode de fusion donne un score de 0 pour le tag *Morceau-Rythmique_puissante*, ce qui résume plutôt bien les réponses des annotateurs.

4.3.2. Validation de la méthode de fusion

Pour valider les scores souples obtenus avec cette méthode, nous mesurons leur accord avec les scores binaires fournis par le Computer Audition Lab avec CAL500. Le coefficient Kappa de Cohen [LK77] est conçu pour ce type d'évaluation. Il prend en entrée deux ensembles d'annotations binaires. Il est calculé selon l'expression :

$$\kappa = \frac{a - e}{1 - e} \tag{4.2}$$

où a désigne le taux d'accord entre les deux ensembles d'annotations, et e est la probabilité empirique d'obtenir un accord par hasard entre deux annotateurs

d'ensembles différents, sur un élément donné. Soient $V = \{v_1, \ldots, v_I\}$ et $W = \{w_1, \ldots, v_I\}$ les deux ensembles d'annotation. On a donc :

$$a = p(v_i = w_i) \tag{4.3}$$

La probabilité e est calculée à partir de la distribution statistique de chaque ensemble d'annotations sur les différentes classes :

$$e = p(v_i = -1) \cdot p(w_i = -1) + p(v_i = 1) \cdot p(w_i = 1) \tag{4.4}$$

De manière à obtenir des valeurs comparables, nous construisons donc de nouveaux scores binaires V_b, qui correspondent aux scores souples reconstruits V_c. Les nouveaux scores binaires sont obtenus par seuillage de nos valeurs souples :

$$V_b = \begin{cases} 1 & \text{si} \quad V_s > t \\ -1 & \text{sinon} \end{cases} \tag{4.5}$$

Le seuil qui produit l'accord le plus élevé ($t = 0{,}64$) donne un Kappa de Cohen moyen de $\kappa = 0{,}80$ entre les deux ensembles d'annotations. D'après [LK77], cette valeur traduit un accord excellent. C'est donc ce seuil que l'on utilisera pour construire V_b.

4.4. Apprentissage régressif et validation de l'approche

Nous menons maintenant une expérience afin de déterminer si l'information sur l'incertitude des annotations, perdue lors d'une fusion binaire, est utile à l'apprentissage des tags.

4.4.1. Mode opératoire

Pour cette expérience, on va entraîner deux systèmes de prédiction. Afin de limiter le biais dû aux différences structurelles entre les algorithmes d'apprentissage, les deux

systèmes seront basés sur un algorithme de boosting, avec le même nombre d'itérations (500). Le premier sera un algorithme de classification Adaboost classique, utilisant les scores binaires recréés V_b. Nous choisissons ces derniers plutôt que les scores du Computer Audition Lab car ils sont plus proches des scores souples V_s. Pour apprendre les V_s en revanche, un algorithme de régression semble plus adapté. Le boosting a déjà été adapté avec des coûts de régression, et nous utiliserons un coût quadratique, très courant, et qui donne naissance à l'algorithme LS_Boost, décrit page 45 [Fri01].

Le but de cette expérience est de démontrer l'utilité de la fusion souple des annotateurs, même pour une tâche finale de tagging, où la décision finale est binaire (on choisit d'indexer ou non un morceau avec un tag). C'est pourquoi nous évaluerons la capacité des deux systèmes à prédire la vérité-terrain binaire V_b. En outre, si l'on cherchait à estimer les scores souples V_s, le système de classification serait clairement désavantagé puisque les scores binaires dont il dispose pour l'apprentissage ne permettent pas de déduire les V_s.

Les descripteurs utilisés pour cette expérience sont tous issus du signal : descripteurs psycho-acoustiques, MFCC, chroma (*cf.* chapitre 5), taux de passage par zéro, dispersion, asymétrie et kurtosis spectraux [Pee04]. L'échelle choisie donne des fenêtres d'analyse de $2\,s$.

4.4.2. Résultats et discussion

Les résultats sont présentés dans le Tableau 4.1. On constate clairement l'avantage du système régressif, utilisant la fusion souple. La différence des performances a été vérifiée au moyen d'un test de Student par séries appariées avec validation croisée (*cf.* Appendice B). Selon ce test, la différence est significative, avec 99% de confiance. Cela prouve donc que l'information sur l'incertitude des annotations est effectivement utile pour l'apprentissage automatique et la prédiction des tags.

Il est important de remarquer que le système régressif ne nécessite pas davantage de données que l'autre système. Le gain de performance est obtenu simplement par un traitement différent de ces données d'annotation. Cette expérience montre ainsi que les données perdues lors d'une fusion binaire d'annotations se révèlent utiles pour l'apprentissage.

Méthode de fusion	MAP (en %)	AROC (en %)
Binaire	46	67
Souple	50	71

Tableau 4.1.: Performances des prédictions avec les fusions d'annotateurs binaire et souple.

4.5. Conclusion

Dans ce chapitre, nous avons décrit une manière de fusionner les annotateurs, qui préserve l'information sur l'incertitude de l'association d'un tag à un morceau, grâce à des scores continus au lieu des habituels oui/non. Nous avons également proposé d'utiliser le boosting régressif pour apprendre les scores obtenus par cette fusion. Les tests montrent que cette configuration mène à un meilleur apprentissage des tags que la classification binaire.

Les chapitres suivants décrivent des expériences indépendantes de celle présentée dans le présent chapitre. Les contributions que nous présentons dans la suite sont tout à fait compatibles avec une fusion d'annotateurs souple et un cadre de régression. Cependant, sans minimiser le résultat que nous venons d'exposer, nous menons les expériences suivantes dans un cadre de classification binaire, afin de mieux les insérer dans une configuration classique et d'étudier isolément les gains apportés par les différents outils proposées.

5. Des descripteurs hétérogènes

Résumé Ce chapitre est focalisé sur la description des morceaux de musique. On y décrit des descripteurs communs, ainsi que de nouvelles représentations, afin de mieux couvrir l'axe sémantique des trois principaux aspects musicaux du signal (timbre, harmonie, rythme). Ensuite, nous étudions le comportement des descripteurs en fonction du type d'intégration précoce, et de la durée de cette intégration.

5.1. Introduction

Après s'être focalisés sur la construction de la vérité-terrain au chapitre 4, nous nous concentrons ici sur la question de la représentation du signal. Nous avons vu au chapitre 2 que la description du morceau par des caractéristiques appropriées est une question cruciale pour la classification automatique, et pour le tagging en particulier. Il est en effet capital pour l'algorithme de classification, de travailler sur une représentation des morceaux qui fasse sens pour lui, et qui soit informative et discriminative vis à vis du critère à classifier.

Une grande partie des descripteurs usuels peuvent être rangés sur trois axes de description complémentaires : rythmique, tonale, et timbrale (*cf.* sous-section 2.2.1 p. 14). Sur ces trois axes cependant, les descriptions ne sont pas toutes homogènes : elles se distinguent par leur nature, leur horizon de description, leur niveau d'abstraction (*cf.* section 1.3)... Selon le niveau d'abstraction, la nature des descriptions change beaucoup. Des informations de bas niveau, proches du signal, donnent une description concrète et souvent fiable, mais pas toujours porteuses de beaucoup de sens. Tandis que les descriptions de haut niveau, qui demandent davantage d'abstraction par rapport à la réalité physique, sont plus aisément interprétables d'un point de vue perceptuel ou musical. De même, un descripteur basé sur des fenêtres

d'analyse courtes va bien saisir les phénomènes instantanés, tandis que des fenêtres de plus longue durée captureront plus facilement les variations lentes.

Nous proposons donc d'explorer plusieurs descripteurs en essayant de couvrir différents niveaux d'abstraction et différentes durées de description.

5.2. Couvrir différents niveaux d'abstraction

Comme nous l'avons rappelé dans l'introduction de ce chapitre, la musique comporte trois principaux aspects sur lesquels on peut ranger les descripteurs : timbre, harmonie et rythme.

La Figure 5.1 montre plusieurs descripteurs, positionnés sur les trois axes de description allant du plus bas au plus haut niveau d'abstraction. Au plus bas niveau se trouve, naturellement, le signal sans transformation. On peut constater que les MFCC et chroma, descripteurs très courants (*cf.* chapitre 2), se positionnent à un niveau assez bas. La plupart des descripteurs usuels sont en fait de bas ou moyen niveau [FLTZ11], ce qui implique un très large fossé sémantique à franchir pour obtenir le niveau des tags (très haut niveau). Afin de mieux couvrir ces axes sémantiques de description, on va chercher des représentations de plus haut niveau.

5.2.1. Timbre

La majorité des descriptions audio utilisées pour la classification sont basées sur le timbre [BCL10, TKT10, SMD10, KS10]. Les descripteurs timbraux classiques (*cf.* chapitre 2) sont de plutôt bas niveau. Cette position est illustrée dans la Figure 5.1, où les MFCC sont placés assez près du niveau « signal ». Les autres descripteurs usuels sont souvent de niveau similaire, voire inférieur.

5.2.1.1. Descripteurs liés à la psychoacoustique

L'utilité de descripteurs plus proches de la perception humaine a été pointée depuis de nombreuses années pour la classification audio. Ces considérations sont issues,

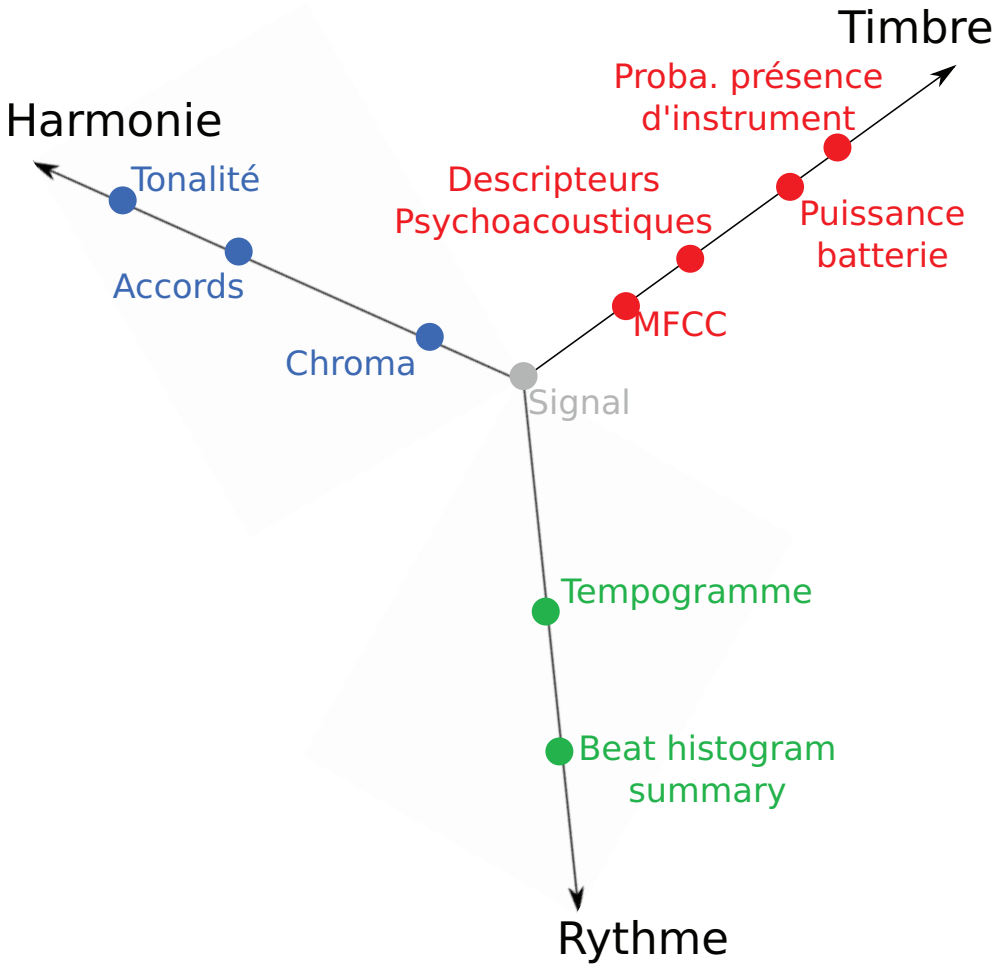


Figure 5.1.: Différents descripteurs positionnés en fonction de leur axe de description et de leur niveau d'abstraction (le niveau s'élève à mesure que l'on s'éloigne du centre).

Descripteur	Dimension	Signification
Centroïde spectral	1	Fréquence moyenne de l'énergie du spectre
Loudness	1	Intensité sonore perçue
Sharpness	2	Contenu en hautes fréquences
Largeur timbrale	1	Platitude de la fonction de Loudness
Volume	1	Taille perçue du son
Dissonance spectrale	2	Rugosité des composantes spectrales
Dissonance tonale	2	Rugosité des composantes tonales
Tonalité pure	1	Audibilité des tons spectraux
Tonalité complexe	1	Audibilité des tons virtuels
Multiplicité	1	Nombre de tons perçus

Tableau 5.1.: Ensemble de descripteurs liés à la psychoacoustique.

entre autres, de la recherche pour l'analyse automatique de scènes sonores[Ell96], et pour l'analyse de parole [RJ93]. L'échelle de Mel elle-même, utilisée par les MFCC, est motivée par la psychoacoustique [RJ93]. En musique, les études sur le timbre des instruments utilisent des descripteurs psychoacoustiques, mais certains d'entre eux sont réservés à la description de notes de musique isolées [KMW94, PMH00]. Pour la classification audio et musique, certains descripteurs motivés par la perception humaine ont été proposés durant la décennie précédente [MB03, Pee04, LR05, MNR08]. Un ensemble plutôt complet et spécifique à la musique enregistrée a été proposé dans les travaux de Y. Yang [YLC06, YLSC08].

Ces descripteurs sont présentés dans le Tableau 5.1, ils peuvent être extraits à l'aide du logiciel Psysound [1]. On peut voir que certains descripteurs sont plutôt communs (centroïde spectral, loudness), mais la plupart sont rares, voire inédits pour une tâche de classification. Ils représentent des caractéristiques comme la taille perçue du son (volume) ou le nombre de tons perçus (multiplicité).

À la fin du tableau sont présentés quelques descripteurs liés à l'aspect tonal de la musique. Cependant, ils ne représentent pas tout à fait le contenu du morceau en terme de notes : ils informent plutôt sur la perception des sonorités. C'est pourquoi nous gardons ces éléments dans la section « timbre », même s'ils possèdent une connexion avec l'aspect « harmonie ».

Cet ensemble de descripteurs, grâce à ses connexions avec la psychoacoustique, s'approche donc davantage de la perception humaine, comme nous le représentons sur

1. `http ://www.psysound.org/`

la Figure 5.1.

5.2.1.2. Puissance de la batterie

La piste de batterie est une information de relativement haut niveau puisqu'elle dépend des causes qui ont produit le signal. Elle ne se trouve donc pas tout à fait dans le signal (à moins de l'avoir tatoué), et nécessite donc d'être estimée. On peut supposer que la puissance relative de cette piste peut être utile pour estimer certaines caractéristiques d'un morceau, telles que le genre ou l'émotion.

À cet effet, nous avons utilisé un algorithme de séparation de la batterie [LBR11], qui calcule deux signaux séparés pour chaque morceau. Cet algorithme tourne rapidement et son code est disponible gratuitement en ligne[2]. La puissance relative des percussions, par rapport aux autres composantes est d'abord calculée sur des trames de 200 ms, puis on garde comme descripteur la moyenne et la variance de cette valeur sur la durée observée.

5.2.1.3. Probabilité de présence d'instruments

L'instrumentation est une information de haut niveau : il ne s'agit pas d'une transformation du signal mais d'une estimation d'informations associées à ses causes. La présence de certains instruments peut être corrélée au genre musical, ou d'autres catégories de tags.

C'est pourquoi nous avons représenté l'instrumentation par un descripteur. Celui-ci est déduit grâce à un système de reconnaissance automatique des instruments. Nous avons au préalable construit des classifieurs SVM simples, à partir d'une base de performances solo. Les instruments sont : contrebasse, percussions, guitare, piano et voix. Les descripteurs utilisés par les SVM sont : MFCC, Intensités des signaux de sous-bandes en octaves (OBSI) [Ess05], coefficients *Line Spectral Frequency*[3]. Puis, le descripteur final est formé par la probabilité de présence de chaque instrument sur la durée de signal observée.

2. `http ://perso.telecom-paristech.fr/~liutkus/`
3. Ces descripteurs peuvent être extraits facilement avec le logiciel Yaafe. Pour plus de détails : `http ://yaafe.sourceforge.net/`

5.2.2. Harmonie

Si les chroma semblent être la norme pour décrire l'harmonie, le type d'accord est moins souvent utilisé, du moins pour le tagging. Cette description du contenu harmonique est de plus haut niveau car un accord est un concept plus aisément maniable par un humain.

Nous utilisons une technique de reconnaissance automatique d'accords [Oud10] pour extraire le type d'accord instantané, parmi les 24 accords majeurs et mineurs. Notre descripteur sera l'histogramme des accords rencontrés sur la durée de description.

5.2.3. Rythme

Pour le rythme, nous utilisons une représentation de mi-niveau, et une de haut niveau : respectivement le tempogramme cyclique et le *Beat Histogram Summary* (*cf.* chapitre 2, page 18). Notre BHS est construit en utilisant une détection d'onset proposée par Alonso *et al.* [ARD05], différente de [TC02].

Afin de couvrir également un niveau d'abstraction plus bas, nous avons tenté d'utiliser directement la fonction de densité de la technique de détection d'*onset* comme descripteur. Ce type de fonction donne instantanément la probabilité d'être sur le début d'une note. Cependant, ce descripteur ne s'est pas montré suffisamment efficace pour la classification, et nous avons abandonné son utilisation.

5.2.4. Tests de performance

Les descripteurs précédemment cités dans ce chapitre sont évalués sur une tâche de tagging automatique.

Nous entraînons donc des classifieurs sur les données de CAL500, et tentons de prédire leur présence ou absence [4]. Les différents descripteurs sont intégrés sur toute la durée du signal, par moyenne et variance, puis utilisés pour entraîner un algorithme Adaboost, avec 100 itérations.

4. Pour une description détaillée des données et du processus d'évaluation, *cf.* chapitre 2.

Descripteur	MAP (en %)	AROC (en %)
MFCC	44,4	63,3
Descripteurs psychoacoustiques	47,1	64,9
Probabilité d'instruments	43,4	62,2
Puissance de la batterie	41,1	59,6
Chroma	41,5	59,7
Accords	36,6	53,0
Tempogramme cyclique	36,1	53,8
Beat Histogram Summary	36,5	53,2
Chance	33,5	49,1

Tableau 5.2.: Performances des descripteurs du signal, intégrés sur toute sa durée.

Les performances des descripteurs, en termes de MAP et d'AROC, sont présentées dans le Tableau 5.2. Nos descripteurs y sont comparés aux MFCC et à un descripteur généré aléatoirement (« Chance », sur la dernière ligne du tableau).

On voit tout d'abord que les descripteurs psychoacoustiques constituent la meilleure représentation basée sur le timbre, pour une dimensionnalité identique à celle des MFCC. Il semble en effet qu'ils donnent une description plus variée donc plus complète du signal. On constate également que la puissance de la batterie fonctionne étonnamment bien, au vu de sa dimensionnalité (2 dimensions : moyenne, variance). En effet, bien que ce soit le descripteur timbral le moins bon, il soutient la comparaison avec tous les autres. L'intervalle de confiance à 95% calculé sur l'ensemble des tags indique que l'AROC est contenue dans [58,9 ; 61,5]. Cette mesure confirme que la puissance de la batterie est significativement bien au dessus des performances du hasard, donc que ce descripteur est réellement utile.

On peut d'ailleurs le constater également dans la Figure 5.2, qui représente l'AROC obtenue avec les descripteurs, groupée par catégories de tag. On y observe que l'AROC correspondant à la puissance de la batterie est presque toujours supérieure au hasard théorique 0,5 (sauf pour les tags de genre, où elle est de 0,49). Notamment, la puissance de la batterie semble très utile pour estimer le contexte d'écoute idéal (catégorie « Utilisation »). Cela peut s'expliquer, par exemple, par le fait qu'un morceau avec une rythmique forte sera plus adapté qu'une autre pour une écoute en milieu bruyant comme lors d'une fête, pour faire danser les invités.

La probabilité de présence des instruments semble fonctionner plutôt bien, compte tenu de la simplicité des classifieurs utilisés pour estimer ce descripteur. Cependant,

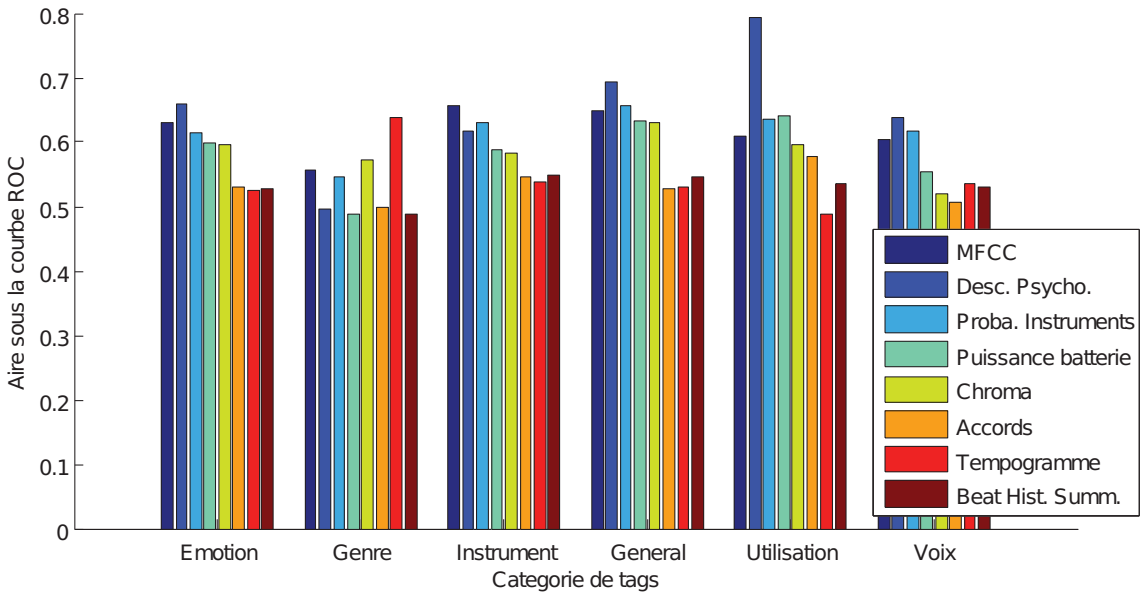


Figure 5.2.: AROC obtenue avec les descripteurs de signal utilisés, groupée par catégorie de tags.

au vu de la Figure 5.2, il peut sembler surprenant que les MFCC fonctionnent mieux pour les tags d'instruments. Cela peut s'expliquer par le fait que ce descripteur soit appris sur des données très différentes de la base CAL500, et sur d'autres instruments. Par contre, la probabilité des instruments est meilleure que les MFCC pour les trois dernières catégories. En effet, ces dernières catégories sont probablement les moins directement corrélées au signal audio, ce qui peut expliquer qu'elles soient mieux estimées par un descripteur de plus haut niveau d'abstraction.

Concernant l'aspect tonal, si les accords semblent systématiquement en-dessous des chroma, ces derniers affichent des performances tout à fait honnêtes.

Pour une description du rythme, aucun des deux descripteurs ne semble se dégager. Pour la performance globale, les deux mesures sont contradictoires. Parmi les différentes catégories de tags, la différence en AROC ne semble pas significative, sauf pour les tags de genre, où le tempogramme cyclique est la meilleure de toutes représentations évaluées. En effet, le rythme est un élément très important pour distinguer les différents genres.

On voit donc globalement que ces représentations sont très différentes les unes des autres, et que leur niveau d'abstraction change leur nature descriptive, au moins pour les représentations basées sur le timbre.

5.3. Importance de l'intégration temporelle précoce

En général pour une tâche de classification, la majorité des descripteurs issus du signal sont extraits sur des trames d'une durée relativement courte, où le signal est considéré comme stationnaire. On obtient donc une suite de vecteurs de description, mais l'algorithme doit prendre une seule décision finale pour chaque morceau. Il faut donc recourir à l'intégration temporelle, soit précoce (au niveau de la description), soit tardive (intégrer les décisions). Souvent, ces deux techniques sont utilisées conjointement [BCE+06, JER09, BMEM10] : on procède d'abord à une agrégation des trames consécutives, sur une durée de quelques secondes, puis les décisions du classifieur sur ces fenêtres agrégées sont résumées sur tout le morceau. Les fenêtres d'intégration précoce sont souvent appelées « fenêtres de texture » (*texture windows*) [TC02], tandis que les fenêtres sur lesquelles on intègre les décisions sont logiquement appelées « fenêtres de décision » (dans notre cas, la fenêtre de décision dure tout le morceau).

5.3.1. Pourquoi une intégration précoce ?

L'intégration précoce comporte de nombreuses utilités : par exemple, elle permet de réduire le bruit des descripteurs, de diminuer le nombre d'exemples d'apprentissage (et ainsi accélérer le temps de calcul) [MALH07, JER09]. L'intégration précoce constitue également une manière simple d'utiliser en même temps des descripteurs extraits sur des trames de tailles différentes (en les ramenant à la même durée de description pour pouvoir les concaténer).

Une autre utilité de l'intégration précoce est la possibilité qu'elle offre de décrire facilement de plus longues portions de signal, et ainsi d'observer des phénomènes à plus long terme, éventuellement basés sur les variations du descripteur intégré (*cf.* sous-section 2.2.3, page 20).

5.3.2. Étude sur la méthode d'intégration

L'intégration temporelle précoce a ici pour finalité de résumer un certain nombre de trames d'analyse. Ce résumé peut être réalisé de nombreuses manières. La plus

commune est l'intégration par moyenne des trames, souvent accompagnée de la variance [BCE+06, BMEM10, BBLP10]. Plusieurs autres techniques sont rapportées dans la sous-section 2.2.3, page 20. Nous étudions ici deux méthodes plus originales : des sacs de mots basés sur un clustering, et une intégration par Modèle de Markov Caché (*Hidden Markov Model*, HMM).

5.3.2.1. Intégration par moyenne, variance

L'intégration par moyenne et variance est la plus simple et la plus souvent rencontrée. C'est d'ailleurs pour cette raison que nous avons choisi cette méthode pour les tests de la section précédente.

Il s'agit donc de prendre, pour chaque coefficient du descripteur, sa moyenne et sa variance sur les trames résumées. Cela revient à représenter les données par une gaussienne multi-dimensionnelle, de covariance diagonale. Une variante consiste à utiliser un GMM au lieu d'une gaussienne simple [BTYL09]. Mais nous nous en tiendrons ici à une simple moyenne et variance.

5.3.2.2. Sac de mots

Le principe du sac de mots consiste à représenter un document (ou seulement une portion) par le nombre d'occurrences qu'il contient des mots d'un dictionnaire.

Dans notre cas, le dictionnaire du monde correspond à des groupes de trames (*clusters*) [Pee13]. Ces derniers résultent d'un partitionnement des exemples d'apprentissage par l'algorithme des k-moyennes (K-*means*) [Bis06] sur l'ensemble des trames de descripteurs issues des données d'apprentissage. Tous les vecteurs sont donc assignés à une partition. Ensuite, chaque fenêtre de texture est représentée par l'histogramme des partitions associés aux trames qu'elle contient.

Lors de la phase de test, les clusters sont déjà construits, et il suffit d'assigner les observations au cluster qui se trouve le plus près.

Cette méthode d'intégration présente les descripteurs dans un formalisme différent, et permet, par une sorte de quantification, de réduire l'effet de leur bruit.

5.3.2.3. Modèle de Markov Caché

Les modèles de markov cachés sont très utiles pour représenter des séries temporelles. Ils sont utilisés avec succès depuis longtemps dans l'analyse de parole [RJ93], mais aussi plus récemment dans la musique, pour faire de l'alignement audio sur partition [Con06, MO09], ou pour segmenter les morceaux [Rap99, LSC06].

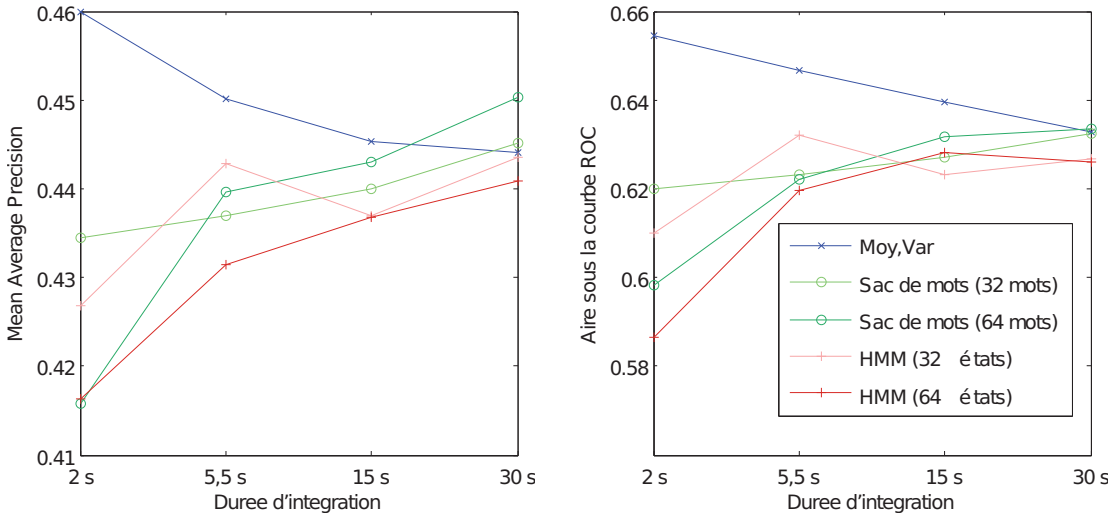
Ainsi, poursuivant une idée développée dans [LSC06] pour la segmentation, nous utilisons pour le tagging une représentation intégrée par HMM. Dans cette représentation, la séquence des trames de descripteur est vue comme la réalisation d'un HMM ergodique, avec des probabilités gaussiennes. Ce HMM est estimé en utilisant comme exemples toutes les séquences des données d'apprentissage. Ensuite, pour représenter un morceau, on décode sa séquence d'états correspondante, et la description finale sera l'histogramme de ces états décodés dans chaque fenêtre de texture.

Cette représentation permet de prendre en compte l'ordre des trames d'analyse, et la dynamique temporelle des données (*cf.* sous-section 2.2.3, page 20).

5.3.2.4. Tests de performance

Afin d'observer le comportement des différentes méthodes d'intégration, nous les utilisons sur des MFCC, en prenant diverses durées d'intégration. On commence par calculer des MFCC sur des trames de 23 ms, sur lesquelles le signal peut être considéré stationnaire. Les trames ont un recouvrement de 50%. L'intégration par moyenne est directement calculée sur ces trames. Pour les deux autres méthodes, avant de calculer les sacs de mots et le HMM, les trames sont pré-intégrées sur un huitième de seconde. Cela permet de réduire un peu le bruit, et de diminuer le nombre d'exemples à traiter pour accélérer les calculs.

Les descriptions seront calculées avec plusieurs durées d'intégration : 2 s (durée d'une mesure à 120 BPM), 5,5 s, 15 s, ainsi que sur la totalité du signal (30 s dans notre base), avec 50% de recouvrement. Les trois premières échelles ont un espacement logarithmique similaire. Une expérience préliminaire a indiqué que des échelles en dessous de 2s étaient moins utiles pour les données utilisées.



(a) Mean Average Precision **(b)** Aire sous la courbe ROC

Figure 5.3.: Résultats obtenus sur les MFCC avec les différentes méthodes d'intégration, en fonction de la durée de la fenêtre de texture.

Les performances obtenues sont présentées dans la Figure 5.3. On peut y constater deux tendances contradictoires. D'une part l'intégration par moyenne et variance apparaît tout à fait appropriée sur de courtes durées, sans doute grâce à sa capacité à représenter très précisément les valeurs résumées lorsqu'elles sont peu nombreuses. D'autre part, les représentations à base d'histogrammes semblent bénéficier d'une durée d'intégration plus longue.

Les HMM nécessitent en général une quantité de données confortable pour être appris correctement. Ils souffrent peut-être ici d'un ensemble d'apprentissage trop restreint. En outre, les histogrammes obtenus possèdent beaucoup d'états acoustiques, et sont de ce fait très variables donc difficiles à analyser par un classifieur. Cette variabilité se stabilise un peu lorsqu'ils couvrent de longues durées. Leurs performances sont tout de même encourageantes car elles deviennent comparables à celles des autres méthodes sur 30 s.

La description par sac de mots, plus simple que les HMM, fonctionne bien sur CAL500, et semble peut-être même légèrement meilleure que la moyenne et variance sur 30 s, même si cette différence n'est pas significative.

5.4. Influence de l'échelle de description

Dans la section précédente, nous avons observé le comportement des classifieurs en faisant varier non seulement la méthode d'intégration, mais également l'horizon de description. Nous avons ainsi pu constater que ce comportement changeait selon la durée utilisée. C'est pourquoi nous poursuivons ici l'étude de l'influence de l'échelle de description, l'étendant sur les représentations décrites dans la section 5.2.

En effet, l'horizon de description change la nature de ce qui est décrit. On peut ainsi décrire des portions de signal très instantanées, ou au contraire des phénomènes à plus long terme. Les phénomènes capturés peuvent alors être différents.

Dans cette étude, on calcule donc les descripteurs sur différentes durées. Puisqu'aucune des méthodes d'intégration testées ne se démarque des autres, nous choisissons d'utiliser l'intégration par moyenne et variance pour obtenir les différentes durées de description. En outre, une première étude réalisée sur des MFCC a montré que l'on obtenait des performances similaires en calculant les descripteurs directement sur la durée de description, et en les obtenant par intégration de trames plus courtes. C'est pourquoi la plupart des descripteurs seront amenés à l'échelle désirée par intégration temporelle (à l'exception du tempogramme cyclique, calculé sur la durée finale, et des accords, intégrés dans un histogramme). Certains descripteurs ne sont pas calculés à toutes les échelles, comme le tempogramme cyclique, prévu pour des portions de signal en général de plus de $2\,s$, ou le Beat Histogram Summary, censé résumer un beat histogram, et dont nous n'avons gardé que l'échelle $30\,s$.

Résultats

Les résultats de cette expérience se trouvent dans la Figure 5.4 : à gauche se trouve la MAP, et à droite l'AROC. On y retrouve, pour les MFCC, le même trajet en bleu que dans la Figure 5.3, puisqu'on utilise la même intégration par moyenne et variance. Ce trajet, qui descend quand l'échelle s'allonge, suit une tendance inverse de celui des descripteurs psychoacoustiques (en vert). Ces derniers semblent en effet profiter d'une plus grande quantité de signal à exploiter, sans doute car certains de leurs concepts portent davantage de sens. La puissance de la batterie semble mieux fonctionner sur de petites durées. Cela est probablement dû à sa simplicité, qui implique

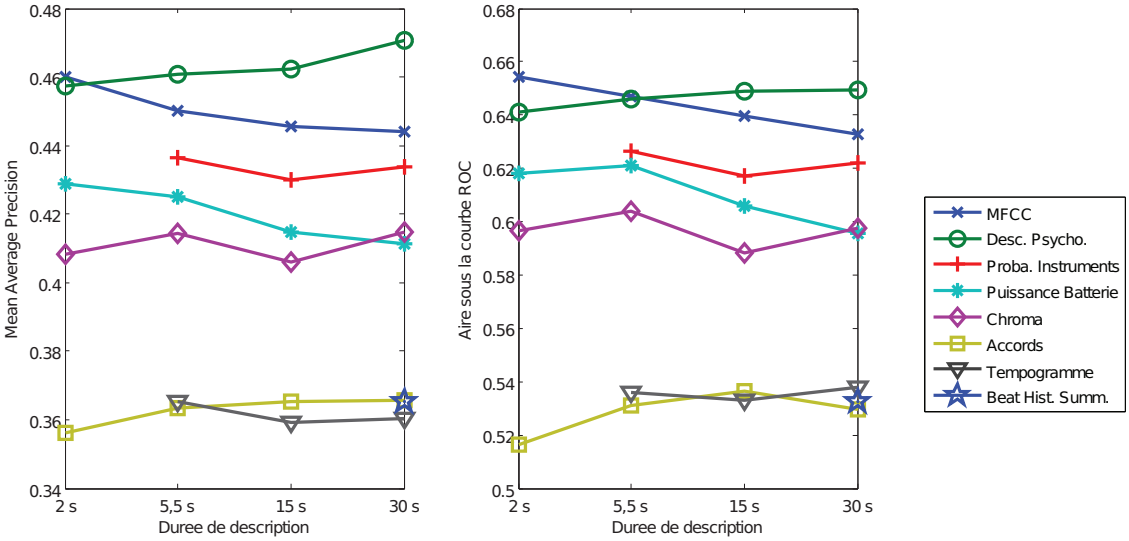


Figure 5.4.: MAP et AROC obtenues sur les différents descripteurs, avec plusieurs durées de description.

que l'algorithme d'apprentissage bénéficie d'un plus grand nombre d'exemples, sur les petites échelles, pour dégager du sens.

Concernant les descripteurs tonaux, les chroma sont meilleurs sur des durées de 5,5 s et de 30 s. La plus grande échelle correspond à un résumé de tout l'extrait sonore, davantage lié à la tonalité du morceau, tandis que 5,5 s est une description plus locale. Ces deux descriptions semblent donc bien se compléter. Les accords par contre, sont clairement plus performants sur des durées plus longues. Cela est probablement dû à leur nature d'histogramme, qui devient de moins en moins creux à mesure qu'on allonge le signal pris en compte. Les descriptions plus longues sont donc plus riches.

Le tempogramme, en revanche, ne semble pas montrer d'échelle plus appropriée. Les variations avec l'échelle sont très fines et ne semblent pas significatives, surtout pour l'AROC.

5.5. Conclusion

Au cours de ce chapitre, nous avons étudié l'aspect descriptif de la classification audio pour le tagging automatique. Nous avons exploré des descripteurs inédits ou

relativement nouveaux, afin d'avoir une bonne couverture des différents niveaux d'abstraction sur les trois aspects de description. Mais les différences entre les descriptions se marquent également par la méthode d'intégration précoce, et par la durée de description. Nous avons notamment observé que les descripteurs n'ont pas le même comportement selon leur durée de description, et que leur durée optimale est spécifique à chaque descripteur.

6. Décrire un morceau sur plusieurs échelles temporelles

Résumé Ce chapitre explore la complémentarité des descriptions calculées avec des fenêtres d'analyse de longueurs différentes. Nous y proposons un nouvel algorithme pour l'exploitation conjointe et la fusion tardive de descriptions extraites à différentes échelles. Puis nous étudions les avantages, en termes de performances, apportés par l'analyse de plusieurs échelles simultanément.

6.1. Introduction

Dans la section 5.4 du chapitre 5, nous avons pu constater qu'un même descripteur n'exprimait pas forcément la même information en fonction de la durée de la fenêtre d'analyse sur laquelle il était extrait du signal. En effet, des descripteurs extraits sur plusieurs secondes capturent des évènements à plus long terme que sur quelques millisecondes. Les phénomènes capturés à une échelle donnée ne le sont donc pas forcément à un autre horizon de description.

Or les concepts qui sous-tendent chaque tag peuvent être portés par des propriétés du signal ayant des dynamiques temporelles différentes. Partant de ce constat, on peut imaginer que différentes versions d'un même descripteur, prises à des échelles différentes, peuvent s'enrichir mutuellement malgré leur redondance. Une fusion de ces représentations pourrait ainsi permettre d'observer des phénomènes survenant sur des durées plus diverses.

La fusion de descripteurs la plus souvent réalisée est la concaténation des vecteurs de description. Cependant, cette fusion suppose des représentations extraites à partir

des mêmes fenêtres d'analyse. Elle ne peut pas être mise en place directement pour des descripteurs désynchronisés. C'est pourquoi la fusion de différentes échelles est souvent difficile à mettre en œuvre.

Nous proposons dans ce chapitre un algorithme permettant de fusionner ces descriptions du signal. Cet algorithme est testé sur deux ensembles de données distincts pour montrer l'intérêt de cette fusion.

6.2. Travaux pré-existants sur la fusion multi-échelles

La fusion de caractéristiques du signal observables à différentes échelles est un problème difficile pour la classification audio. Cela explique probablement sa relative rareté dans ce domaine, par rapport au traitement de l'image [Taa03]. Plusieurs techniques de fusion sont présentées au chapitre 2 (section 2.4, p. 28), mais elles nécessitent toutes de synchroniser les données (descripteurs ou décisions), c'est à dire de les ramener à la même granularité avant leur fusion.

L'intégration temporelle précoce (*cf.* section 5.3) peut être utilisée pour ramener toutes les descriptions à la même échelle. Il suffit pour cela d'intégrer les descripteurs extraits de trames courtes, en calant la fenêtre de texture sur la durée de la fenêtre d'analyse la plus longue. En synchronisant ainsi tous les descripteurs, on peut alors les concaténer ; mais la précision temporelle des descripteurs à court-terme est réduite. Des phénomènes évoluant à haute fréquence peuvent donc être perdus, en raison de l'effet de filtre passe-bas de l'intégration. On peut réduire un peu cet effet en calculant plusieurs statistiques sur la fenêtre d'intégration. Par exemple, dans [HBE12], on propose de calculer sur chaque fenêtre : la moyenne, la variance, le maximum et le minimum pour chaque coefficient. Il est également possible de sur-échantillonner les observations des échelles les plus longues.

Au-delà de l'intégration, certaines études proposent d'utiliser des fenêtres de différentes longueurs, tout en alignant leurs centres. C'est une première approche pour avoir des fenêtres de tailles différentes, même si cela nécessite un pas d'avancement constant. Comme on peut le voir dans la Figure 6.1, le pas d'avancement commun a de bonnes chances d'être inapproprié pour certaines échelles. En effet, les petites échelles (en bas) peuvent exclure certaines parties du signal, tandis que les longues

fenêtres peuvent se recouvrir beaucoup, conduisant à une représentation très redondante. Mais cette première approche a l'avantage d'être plutôt simple à mettre en place.

Dans [ML11], les auteurs utilisent ce type de découpage pour proposer une visualisation originale du contenu d'un morceau. Leur méthode utilise plusieurs horizons d'observation pour représenter les changements à plus ou moins long terme au cours du morceau. Dans [MSS06], la description du signal est calculée à partir d'un banc de filtres à Q constant. Les filtres n'ont alors en général pas le même support temporel. Cependant, une fois que le vecteur de description est calculé, aucune information n'est conservée à propos du support temporel. Une autre étude propose d'utiliser des fenêtres de tailles différentes pour comparer le signal, autour d'un instant donné, à la fois à des composantes de GMM et à des Modèles de Textures Dynamiques [ECL11]. En effet, la vraisemblance d'un modèle gaussien peut être calculée à partir d'une seule trame de signal, tandis qu'une texture dynamique est davantage conçue pour analyser une séquence de trames.

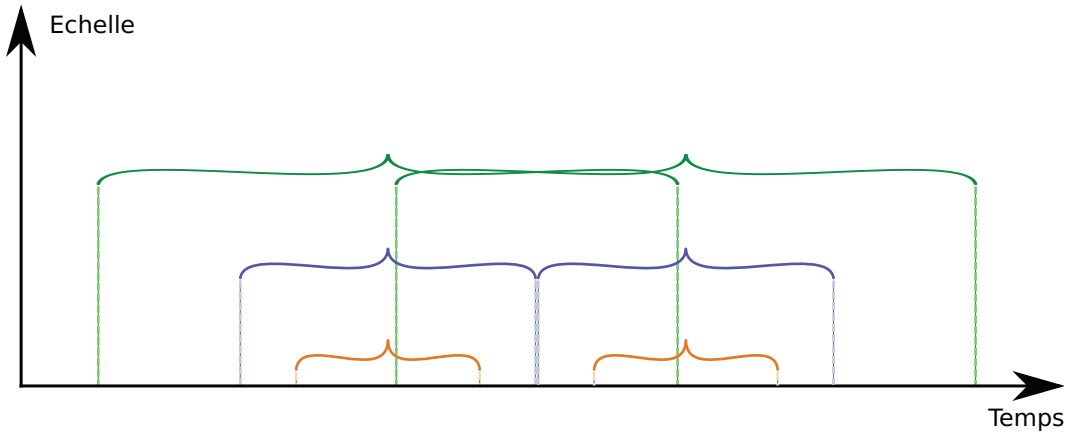


Figure 6.1.: Un découpage multi-échelles du signal, avec des fenêtres d'analyse alignées.

On trouve également depuis peu dans la littérature des techniques plus complexes faisant appel à des fenêtres d'analyse de tailles différentes. Dans [SMD10], les auteurs utilisent des MFCC calculés à différentes échelles pour la reconnaissance d'instruments. Une fusion d'échelles est réalisée à l'aide d'une Transformée en cosinus discrète. On peut également citer une étude de Hamel *et al.*, qui utilise des descripteurs de multiples échelles avec des réseaux de neurones [HLBE11].

La plupart des travaux cités dans cette section fusionnent les échelles au niveau

de la description, et donc avant la phase d'apprentissage. Ce dernier n'a donc pas connaissance de la différente nature des informations qu'il utilise.

6.3. Algorithme de boosting pour l'analyse multi-échelles

Nous proposons d'utiliser la technique du boosting pour fusionner un nombre quelconque d'échelles au niveau de la décision. Cette adaptation de l'algorithme Adaboost s'appuie sur celle présentée en section 3.5 du chapitre 3, pour la fusion de classifieurs. Effectuer une fusion tardive permet au modèle d'apprentissage de traiter différemment les échelles.

6.3.1. Plage de décision

Dans cet algorithme, on utilise un classifieur faible $\mathcal{H}_s$ à chaque échelle s utilisée. Ce classifieur utilisera uniquement les descripteurs calculés sur des trames de L_s échantillons.

À chaque itération, l'algorithme de boosting doit sélectionner le classifieur faible donnant le taux d'erreur pondérée le plus faible. Si l'on veut effectuer une comparaison équitable entre les classifieurs faibles, il faut que l'on juge leur performance sur les mêmes exemples. Par conséquent, ils doivent baser leurs décisions sur les mêmes segments sonores. Puisque les trames des différents classifieurs ne décrivent pas les mêmes portions de signal, il faut fixer la durée de décision pour toutes les échelles d'analyse.

C'est à cet effet que nous introduisons la notion de *plage de décision*. Ces plages représentent la durée de signal sur laquelle les décisions des classifieurs faibles sont prises. Dans la Figure 6.2, on voit comment une plage P_i, en gris, inclut les trames des différentes échelles. Chaque exemple $\mathbf{x}_{i,s}^n$ est un vecteur de description, où s est l'échelle temporelle, i est l'index de la plage de décision englobante et n est l'index de la trame décrite à l'intérieur de la plage de décision. On a donc :

$$P_i = \left\{ \mathbf{x}_{j,s}^n \right\}_{j=i} \tag{6.1}$$

On considère qu'une trame appartient à la plage P_i si son centre est inclus dans les limites temporelles de P_i.

D'un point de vue formel, on peut rapprocher cette configuration de l'algorithme de fusion décrit en pages 44 et 45, si l'on considère que ce sont maintenant ces plages de décision qui constituent nos exemples d'apprentissage.

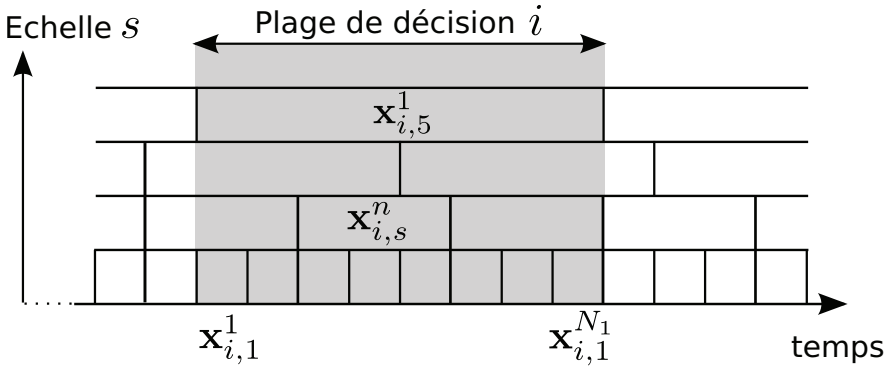


Figure 6.2.: Une plage de décision (en gris), couvrant un nombre différent de trames sur différentes échelles.

6.3.2. Cœur de l'algorithme

L'algorithme pour la classification multi-échelles est présenté dans l'Algorithme 6.1, page 80.

Les données sont donc les vecteurs de description $\mathbf{x}_{i,s}^{n}$, avec leurs labels y_i, $1 \leq i \leq I$ (où I est le nombre total de plages de décision dans l'ensemble des données d'apprentissage). Les labels ne dépendent ni de s ni de n, mais uniquement du morceau dans lequel la plage de décision P_i est inscrite, puisque dans notre configuration, un tag s'applique à un morceau entier [1].

Chacune des plages reçoit un poids associé $w_{r,i}$, qui représente, comme dans Adaboost, l'attention portée à cette plage à l'itération courante r. Au départ, tous les poids sont égaux.

Au début de chaque itération r, les poids $w_{r,i}$ sont normalisés. Puis les classifieurs faibles $h_{r,s}$ sont entraînés en utilisant les poids obtenus. On calcule alors, pour chaque

1. L'algorithme présenté ici pourrait tout à fait être utilisé pour un problème de segmentation, où les annotations ne s'appliquent qu'à des portions de morceaux. Cependant, dans un souci de simplicité des notations et de cohérence, nous décrivons ici le cas du tagging automatique.

Algorithme 6.1 Adaboost adapté pour la fusion multi-échelles de classifieurs.

Paramètre: Exemples annotés pour toutes les échelles $(\mathbf{x}_{i,s}^n, y_i)$,
$1 \leq i \leq I, \quad 1 \leq s \leq S, \quad 1 \leq n \leq N_s$

Paramètre: Classifieurs faibles $\mathcal{H}_s$

$w_{1,i} \longleftarrow \frac{1}{I}$

pour $r = 1, \ldots, R$ **faire**

$w_{r,i} \longleftarrow \frac{w_{r,i}}{\sum_{j=1}^{I} w_{r,j}}$ *// Normaliser les poids*

Entraîner les classifieurs $h_{r,s}$ avec les modèles $\mathcal{H}_s$ et les poids $w_{r,i}$

// Décisions de $h_{r,s}$ sur les plages i

$d_{r,s,i} = \text{sign}\left(\sum_{n=1}^{N_s} h_{r,s}(\mathbf{x}_{i,s}^n)\right)$,

// Calculer le taux d'erreur pondérée

$\epsilon_{r,s} \longleftarrow \sum_i w_{r,i} \mathbb{1}_{d_{r,s,i}=y_i}$

// Meilleure échelle

$\hat{s}_r \longleftarrow \text{argmin}_s \, \epsilon_{r,s}$

$\epsilon_r \longleftarrow \epsilon_{r,\hat{s}_r}$

$h_r \longleftarrow \sum_n h_{r,\hat{s}_r}$

// Coefficient associé à h_r

$\alpha_r \longleftarrow \log \frac{1}{\beta_r}$, où $\beta_r = \frac{\epsilon_r}{1-\epsilon_r}$ *// Mettre à jour le poids des exemples*

pour chaque plage P_i **faire**

// tester si $d_{r,\hat{s}_r,i} = y_i$

si $\mathbf{x}_i$ bien classifié **alors**

$w_{r+1,i} \longleftarrow w_{r,i} \, \beta_r$

sinon

$w_{r+1,i} \longleftarrow w_{r,i}$

fin si

fin pour

fin pour

Sortie: $H((x)) = \sum_r \alpha_r h_r((x))$

échelle, la décision du classifieur faible sur les plages de décision. Pour une plage P_i, elle résulte d'un vote majoritaire sur les trames appartenant à P_i. Ces décisions permettent de calculer un taux d'erreur pour chaque échelle. L'échelle $\hat{s}_r$ qui présente le taux d'erreur le plus bas est alors sélectionnée pour le classifieur fort final, avec le coefficient associé α_r. Ensuite, le poids des plages de décision bien classifiées à cette itération, est diminué.

La sortie de l'algorithme $H(\mathbf{x})$ s'utilise de la manière suivante. Lorsque l'on classifie une plage P_i, on prend une décision pour chaque élément r, en appliquant h_r aux observations de l'échelle correspondante $\mathbf{x}^n_{i,\hat{s}_r}$. Puis $H(\mathbf{x}_i)$ est une somme pondérée des décisions $h_r(\mathbf{x}_i)$. Pour finir, la décision globale sur un morceau m est basée sur une intégration tardive classique de toutes les plages de décision incluses dans m. Elle est prise par moyenne seuillée des $H(\mathbf{x}_i)$:

$$D_m = \begin{cases} 1 & \text{si} \quad \text{mean}_{P_i \in m}\, H(\mathbf{x}_i) > t \\ -1 & \text{sinon} \end{cases} \tag{6.2}$$

Le seuil final t est similaire à celui d'Adaboost. Il est en théorie de 0 mais on peut le faire varier pour changer la balance entre les morceaux classifiés positifs et négatifs. En diminuant t, l'algorithme classifiera probablement davantage de morceaux comme positifs. La classification de ces derniers sera moins fiable mais le rappel sera sûrement meilleur (*cf.* Appendice A).

6.4. Deux expériences pour l'évaluation

Nous avons effectué plusieurs expériences pour montrer l'intérêt d'avoir de multiples horizons de description. Afin d'observer uniquement la complémentarité des différentes durées d'analyse, nous avons gardé la même description pour toutes les échelles.

Une expérience de multi-tagging a donc été menée sur CAL500. Mais pour valider la méthode sur une tâche plus corrélée au contenu audio, nous avons d'abord testé notre algorithme sur une tâche de reconnaissance des instruments de musique, avec une base dédiée.

6.4.1. Reconnaissance des instruments de musique

6.4.1.1. Description de l'expérience

Le problème de la reconnaissance des instruments de musique présente les avantages d'être bien défini et fortement relié au contenu audio.

Nous avons donc réalisé une expérience sur une base constituée d'un ensemble de performances solo d'instruments acoustiques. Les œuvres sont issues du répertoire classique et incluent alternativement : du piano, de la guitare, du violon, du violoncelle, du basson et du hautbois. L'ensemble est ainsi composé de 73 performances (31 pour l'apprentissage, et 42 pour le test), pour une durée totale de 449 minutes. Pour chaque instrument, on dispose de 28 à 39 minutes de musique pour l'entraînement, et entre 22 et 64 minutes pour l'ensemble de test.

Puisque la tâche et les données sont différentes de notre base de multi-tagging, les descripteurs utilisés ne sont pas les mêmes. Ceux-ci résultent d'une sélection de 30 attributs sélectionnés par la méthode de Maximisation du Rapport d'Inertie [PR03], appliquée sur un ensemble de descripteurs cepstraux, spectraux, perceptuels et temporels utilisés auparavant sur les mêmes données [Lar08].

Ces descripteurs sont extraits à quatre échelles différentes, plus courtes que pour les données de multi-tagging. Ce choix est motivé par deux raisons. La première est que, comme nous l'avons déjà précisé, la reconnaissance d'instruments de musique est une tâche très corrélée au contenu audio. On va donc chercher à faire une analyse plus précise en capturant des phénomènes transitoires, ou du moins assez courts. La deuxième raison est que des performances solo d'instruments apparaissent comme des données moins complexes que des morceaux pop, ce qui réduit potentiellement le bruit dans les descripteurs extraits. Ces derniers bénéficieront donc moins d'une intégration précoce. L'échelle la plus courte (s_1) correspond à une fenêtre d'analyse de $L_1 = 320$ ms, ce qui représente environ la durée d'une croche à 90 BPM. Les trois autres échelles (s_2, s_3 et s_4) sont obtenues par intégration temporelle sur des fenêtres de $2L_1$, $4L_1$ et $8L_1$.

Chacun des exemples est annoté avec l'un des six instruments. Ce problème multi-classes est décomposé en six problèmes bi-classes, selon une approche un-contre-tous.

Échelle	Taux de reconnaissance (en %)
s_1	59,8
s_2	53,0
s_3	62,9
s_4	44,2
Multi-échelles	64,5

Tableau 6.1.: Performance des systèmes mono-échelle et du système multi-échelle sur la base de reconnaissance des instruments.

Lors de la phase de test, toutes les décisions sont intégrées sur l'échelle la plus longue $8L_1 = 2{,}6$ s, puis l'instrument le plus probable est choisi.

Ces prédictions permettent de calculer un taux de bonne reconnaissance :

$$R = \text{mean}_i \, \mathbb{1}_{H(\mathbf{x}_i)=y_i} \tag{6.3}$$

6.4.1.2. Résultats

Pour cette expérience, on entraîne le système de classification multi-échelles et les quatre systèmes mono-échelles sur les données d'apprentissage, avec 500 itérations de boosting [2]. Les taux de bonne reconnaissance obtenus sur l'ensemble de test se trouvent dans le Tableau 6.1. On y voit clairement que le système multi-échelles donne le meilleur taux de reconnaissance. La redondance entre les informations des échelles limite probablement le gain de performance apporté par leur exploitation conjointe. La différence entre la performance du classifieur multi-échelles et celle du système à l'échelle s_3 a été testée statistiquement. Ainsi, le test de McNemar a donné une valeur p de 0,003, ce qui signifie que l'on est sûr à 99,7% que cette différence est significative (*cf.* Appendice B).

Les descripteurs sélectionnés au fil des itérations sont très différents d'un instrument à l'autre. Il y a par contre une tendance commune dans le classifieur multi-échelles : les échelles les plus souvent sélectionnées sont la plus courte et la plus longue (s_1 et s_4). De manière surprenante, on remarque que ces deux échelles ne correspondent pas aux systèmes mono-échelle les plus performants. Cela peut s'expliquer par le fait que

2. Sur ce type de tâche, très corrélée à l'audio, il est profitable de pousser le nombre d'itérations, afin d'obtenir un apprentissage plus précis. Les données sont moins bruitées que pour une tâche de tagging, et le risque de sur-apprentissage est donc moindre.

s_1 donne la description la plus temporellement précise, tandis que s_4 a des facilités à prendre des décisions sur des plages de 2,6 s, puisque c'est son propre horizon de description. On peut aussi penser que ces deux échelles sont les plus différentes, donc les plus complémentaires, avec une redondance limitée. Le plus important est que ces sélections indiquent que l'information apportée par l'ensemble des échelles réunies, est structurellement différente d'une seule échelle.

En regardant plus précisément les détails des résultats, on s'aperçoit que le système multi-échelles n'est pas toujours le meilleur pour tous les instruments. Mais sa performance varie moins selon les instruments. Cela indique que l'approche multi-échelles donne les meilleures performances car elle est plus flexible, et peut se focaliser sur la représentation la plus appropriée.

6.4.2. Multi-tagging

Pour l'expérience de multi-tagging, nous avons choisi d'utiliser les descripteurs suivants : MFCC, chroma, description psychoacoustique, taux de passage par zéro, diffusion, asymétrie et « kurtosis » spectraux [PGS+11]. Les échelles correspondent à des horizons de description de 2 s, 3,3 s, 5,5 s, 9 s et 15 s. Ces échelles sont choisies pour avoir un espacement logarithmique constant entre deux longueurs consécutives. Certaines sont déjà utilisées dans la section 5.4 du chapitre 5 mais nous avons choisi d'en ajouter d'autres pour couvrir un plus grand nombre d'échelles différentes.

On entraîne donc les classifieurs correspondant à chaque échelle, ainsi qu'un classifieur multi-échelles, sur ces données avec 100 itérations de boosting. Les résultats de ces expériences en termes de MAP et d'AROC sont reportés dans le Tableau 6.2. Les meilleures MAP et AROC sont obtenues par le système multi-échelles. Ici encore, il paraît évident qu'il existe une certaine redondance entre les descriptions aux différentes échelles, ce qui restreint le gain apporté par la fusion de ces informations.

Malgré cela, la significativité de la différence entre ce système et le meilleur système mono-échelle a été vérifiée par un test de Student par séries appariées avec validation croisée (*cf.* Appendice B). Ce test indique avec 99% de certitude que la différence est significative. L'analyse du signal gagne donc à être menée conjointement sur plusieurs échelles.

Échelle	MAP (en %)	AROC (en %)
$s_1 = 2$ s	43,2	64,1
$s_2 = 3{,}3$ s	44,2	65,2
$s_3 = 5{,}5$ s	44,8	65,8
$s_4 = 9$ s	45,6	66,7
$s_5 = 15$ s	45,7	66,4
Multi-échelles	46,6	67,1

Tableau 6.2.: Performance des systèmes mono-échelle et du système multi-échelles sur CAL500.

6.5. Conclusion

Dans ce chapitre, nous avons proposé un nouvel algorithme de boosting permettant de fusionner pour l'apprentissage des informations hétérogènes. Les informations utilisées peuvent être extraites du signal sur différentes fenêtres d'analyse. L'originalité de l'approche proposée ici tient au fait que la fusion ait lieu au niveau de la décision, ce qui permet au classifieur d'avoir connaissance des échelles utilisées.

Bien qu'il soit possible, avec cet algorithme, d'utiliser des descripteurs spécifiques à chaque échelle, nous avons ici utilisé les mêmes descripteurs à toutes les échelles. En effet, nous avions montré au chapitre 5 que les représentations du signal n'étaient pas équivalentes selon leur horizon de description. Les expériences de ce chapitre ont maintenant prouvé que malgré leur évidente redondance, plusieurs versions d'un même descripteur calculées sur des fenêtres différentes présentaient une certaine complémentarité les unes avec les autres.

L'algorithme proposé ici permet également de fusionner des descriptions calculées différemment selon les échelles. Nous allons expérimenter cette possibilité dans le chapitre 7.

7. Données collaboratives et fusion multi-niveaux

Résumé Dans ce chapitre, nous explorons l'utilité pour le tagging automatique de diverses informations pouvant être trouvées automatiquement sur Internet. Les descriptions que nous en tirons seront évaluées et comparées individuellement, puis ensemble. Pour finir, nous utiliserons la fusion multi-échelles décrite au chapitre 6 pour intégrer ces descriptions à un système plus complet incluant aussi des informations de bas et moyen niveaux, issues du signal audio.

7.1. Introduction

Nous avons présenté au chapitre 5 des descriptions musicales calculées à partir du signal. Jusqu'au milieu des années 2000, dans le domaine du MIR, il était courant de considérer que le signal audio était la seule source de données exploitables dont on disposait pour réaliser une description par le contenu. Depuis, la musique (tout comme la vidéo et les images) se consomme de plus en plus fréquemment en ligne, proposée par des services gardant trace du comportement des utilisateurs, et les incitant à donner un retour sur leur expérience. Le site Last.fm [1] propose ce type de service : il utilise les statistiques d'écoute des utilisateurs pour leur créer des recommandations musicales, et leur permet également de taguer des morceaux ou d'indiquer s'ils les aiment.

Au delà de ces services, des outils ont émergé, permettant à des passionnés de musique ou d'autres sujets, de contribuer à des encyclopédies, des bases de données

1. `http ://www.lastfm.fr`

AIR French Band* – Moon Safari

Label: Source – 7243 8 44978 2 8, Source – 724384497828, Virgin – CDV 2848
Format: CD, Album
Country: Europe
Released: 1998
Genre: Electronic
Style: Downtempo, Ambient

more images

Tracklist Hide Credits

1	La Femme D'Argent Backing Vocals – J-B Dunckel*, N. Godin* Handclaps – Caroline L.*, J-B Dunckel*, N. Godin*, All* Organ – Eric Regert Tambourine – N. Godin* Written-By – J-B Dunckel*, N. Godin*	7:09
2	Sexy Boy Drums – Marlon* Vocals – J-B Dunckel* Vocals, Electric Guitar, Talkbox, Electronics [Syrinx], Synthesizer [Korg Ms20, Moog] – N. Godin* Written-By – J-B Dunckel*, N. Godin*	4:57
3	All I Need Acoustic Guitar – P. Woodcock* Electric Piano [Wurlitzer] – J-B Dunckel* Synthesizer [Korg Ms20, Solina String Ensemble], Drums, Organ – N. Godin* Vocals – Beth Hirsch Written-By – Beth Hirsch	4:27
4	Kelly, Watch The Stars! Drums – Marlon* Vocoder, Glockenspiel, Clavinet, Synthesizer [Minimoog], Handclaps – J-B Dunckel* Vocoder, Synthesizer [Moog Bass, Casiotone], Handclaps, Glockenspiel – N. Godin* Written-By – J-B Dunckel*, N. Godin*	3:46
5	Talisman Arranged By [Strings] – David Whitaker Drums – Marlon* Electric Piano [Wurlitzer] – N. Godin* Synthesizer [Minimoog] – J-B Dunckel* Written-By – J-B Dunckel*, N. Godin*	4:16
6	Remember Talkbox, Vocoder, Synthesizer [Roland String Ensemble], Tambourine, Electric Guitar – N. Godin* Vocoder, Backing Vocals, Electric Piano [Wurlitzer] – J-B Dunckel* Written-By – J-B Dunckel*, J-J Perrey*, N. Godin*	2:34
7	You Make It Easy Arranged By [Strings] – David Whitaker Percussion, Glockenspiel, Harmonica, Synthesizer [Moog], Handclaps – N. Godin*	4:01

Figure 7.1.: Extrait de la page de l'album « Moon Safari », sur Discogs.

ou des catalogues collaboratifs. L'exemple le plus connu, Wikipédia[2], est bien sûr une source d'informations très riche. Mais des sites plus spécialisés proposent des quantités considérables de références, renseignées très précisément et parfois avec de nombreux détails. Ces données sont organisées de manière appropriée pour leur exploitation automatique, et facilement récupérables grâce à des interfaces de programmation (API) spécifiques. Un site très connu de ce type est IMDb[3] pour le cinéma, mais on peut citer aussi MusicBrainz[4] ou Discogs[5] pour la musique. La Figure 7.1 illustre la précision des informations que l'on peut parfois trouver sur ce type de sites.

Les descriptions que l'on peut obtenir en exploitant ces données collaboratives sont

2. `http ://www.wikipedia.org`
3. `http ://www.imdb.com`
4. `http ://musicbrainz.org`
5. `http ://www.discogs.com`

souvent très différentes de celles extraites du signal. Notamment, puisqu'elles sont générées à la base par des humains, la description qu'elles donnent est plus proche de la perception humaine que celles que l'on obtient à partir du signal audio. Ces représentations sont donc en général de beaucoup plus haut niveau d'abstraction. C'est pour cette différence qu'il est souhaitable de les exploiter en même temps que les informations de contenu audio. En effet, des informations aussi différentes ont de bonnes chances de s'enrichir mutuellement.

Après avoir décrit et évalué quelques exemples de données collaboratives extraites pour le tagging automatique, nous utiliserons la technique présentée au chapitre 6 pour les fusionner avec les données de contenu.

7.2. Descripteurs issus du contexte éditorial et social

Les descripteurs éditoriaux et sociaux (parfois qualifiés de « contextuels » puisqu'ils proviennent de données trouvées autour du morceau) utilisent des données récupérées sur Internet. Ils peuvent tous être construits automatiquement avec pour seules données de départ le nom de l'artiste et le titre du morceau.

7.2.1. Tags utilisateurs

Les tags donnés par les utilisateurs donnent une description haut-niveau de la réaction des auditeurs à la musique écoutée. Il est possible de les utiliser directement pour les mêmes applications que les tags automatiques. Cette méthode d'annotation permet d'obtenir une grande quantité de données avec une fiabilité tout à fait correcte. Cependant, les tags sociaux comportent plusieurs problèmes pour l'indexation (également évoqués p. 30). Premièrement, le vocabulaire choisi par les utilisateurs n'est pas toujours approprié, avec des tags peu exploitables, comme *Seen_live* ou *Favorites*. De plus, le vocabulaire n'étant pas structuré, un même tag ainsi peut se retrouver écrit de plusieurs manières différentes, parfois avec des fautes d'orthographe (*classic*, *classical*, *clasical*). Le troisième défaut tient au fait que, comme le remarque [TBL08], si un tag n'est pas associé à un morceau par les utilisateurs, on ne peut pas forcément en déduire qu'il n'est pas approprié. Cela peut simplement

Rufus Wainwright : « Cigarettes and chocolate milk »		Spice Girls : « Stop »	
Tag	Pertinence	Tag	Pertinence
Chamber pop	1,0	Teen pop	1,0
Singer-songwriter	0,84	Dance pop	0,99
Cabaret	0,63	Rock	0,73
Piano	0,48	Pop	0,70
Soundtrack	0,48	Ballad	0,68
Pop	0,46	Europop	0,59
Folk	0,44	Alternative rock	0,49
Vocal	0,41	Female	0,48
Romantic	0,38	Sexy	0,46
Male vocalist	0,38	Group	0,46

Tableau 7.1.: Les dix premiers tags Echo Nest pour deux morceaux de CAL500.

signifier que ce tag est négligé par les utilisateurs, ou que le morceau est peu populaire, donc avec peu d'annotations. C'est pour cette dernière raison que ces tags sont qualifiés de « faibles ».

Malgré cela, si l'on cherche à construire automatiquement des tags plus fiables et mieux structurés, les tags utilisateurs peuvent être utilisés comme des sources de description tout à fait utiles à la classification [LSSH09, MCJT12]. C'est pourquoi nous collectons des tags donnés par The Echo Nest [6]. Ce service donne un ensemble de labels pour chaque morceau, avec une pertinence relative ($0 \leq p \leq 1$). Un exemple est donné dans le Tableau 7.1 sur deux morceaux de CAL500. Afin de limiter la parcimonie de cette description, nous gardons uniquement les 20 tags les plus fréquents sur les morceaux que nous avons. Tous les tags sont représentés par leur pertinence p.

7.2.2. Paroles

Les paroles constituent un aspect de description très lié à la sémantique. Cette modalité a été parfois utilisée pour la reconnaissance d'émotions [KS10], mais les travaux qui l'utilisent pour d'autres catégories de classification restent rares. Quelques études de R. Mayer et R. Neumayer tentent de reconnaître le genre musical en utilisant des descripteurs venant des paroles [NR07, MNR08, MN09]. Mais ces descripteurs sont

6. `http ://www.echonest.com/`

davantage liés à la structure des textes qu'à leur sens ou leur thème (nombre d'occurrences de chaque mot, nombre de lettres par mot, ...). On peut donc considérer qu'ils expriment des informations de bas niveau d'abstraction sur les textes.

Mais les paroles d'une chanson sont par nature censées apporter du sens pour l'auditeur, donc exprimer un haut niveau d'abstraction. C'est pourquoi nous proposons d'utiliser une représentation davantage liée au sens des paroles.

7.2.2.1. Thèmes des paroles

Après avoir récupéré les paroles des morceaux sur ChartLyrics[7], on construit une représentation basée sur des « thèmes latents ».

On commence par calculer une matrice M de dimension $N_{morceaux} \times N_{mots}$, indiquant la *term frequency-inverse document frequency* (TF-IDF) de chaque mot pour chaque morceau. La TF-IDF prend en compte la fréquence globale des mots, de manière à représenter la fréquence relative d'un mot dans un morceau particulier, par rapport aux autres morceaux. Le vocabulaire est construit sur l'ensemble des morceaux (en excluant les *mots vides*[8]).

Puis, on va chercher à transformer cette matrice pour faire émerger des groupes de mots ayant des thèmes communs. Cette transformation est réalisée classiquement par une Analyse sémantique latente (LSA) [LGH08, YLC+08]. Nous faisons émerger ces thèmes latents au moyen d'une Factorisation en Matrices Non-négatives (NMF) [SL01]. Cette technique entretient des relations avec la LSA [GG05].

La NMF consiste à approximer la matrice non-négative M en un produit de deux matrices non-négatives P et T pour provoquer l'émergence des thèmes :

$$M \approx P \cdot T \tag{7.1}$$

La première matrice P (de dimension $N_{morceaux} \times N_{thèmes}$) donne la pertinence de chaque thème pour chaque morceau. La matrice T (de dimension $N_{thèmes} \times N_{mots}$) donne la contribution de chaque mot à chaque thème. En principe, $N_{thèmes}$ est choisi

7. `http ://www.chartlyrics.com/`

8. Les mots vides sont des mots trop communs pour être pris en compte dans ce type d'analyse. En français, les mots « le », « de » ou « ces » sont des exemples de mots vides.

bien inférieur à N_{mots} et $N_{morceaux}$, de sorte que les matrices P et T soient beaucoup plus petites que M.

Les lignes de P constituent notre description des thèmes pour chaque morceau.

7.2.2.2. Émotion des paroles

Les thèmes peuvent être informatifs pour l'apprentissage automatique, mais les paroles ont aussi de bonnes chances d'évoquer des émotions particulières.

Pour construire cette représentation de haut niveau d'abstraction, nous avons utilisé un corpus, construit par Bradley & Lang [BL99], qui place 2 477 mots anglais communs dans un espace émotionnel à trois dimensions[9] (*valence*, *excitation* et *domination*[10]). Chaque mot m du texte est alors remplacé par sa valeur émotionnelle, entre 0 et 10, trouvée dans le dictionnaire D : $v = D(m)$. Quand on ne trouve pas d'entrée qui correspond exactement dans le dictionnaire, on utilise le « stemmer de Lancaster[11] » (*Lancaster stemmer*) [Pai90] pour tenter de faire correspondre la racine du mot avec la racine d'une entrée : $v = D_{racine}(\text{racine}(m))$.

Pour finir, on représente chaque mot par la moyenne et la variance des trois dimensions émotionnelles, pour tous les mots trouvés dans D ou D_{racine}.

Pour illustrer cette représentation, le Tableau 7.2 (p. 93) donne les valeurs d'émotion des dix mots les plus fréquents dans deux morceaux de CAL500 : « Fly me to the Moon » de Frank Sinatra, et « He War » de Cat Power. On constate, pour commencer, que la majorité des mots ne sont pas trouvés dans le dictionnaire. Cependant, une partie de ces mots non trouvés semblent dépourvus d'un sens émotionnel identifiable et non ambigu. Les mots suivis d'une astérisque ont été trouvés dans le dictionnaire après racinisation. Les valeurs globales pour tout le morceau sont rapportées dans la dernière ligne du tableau.

9. Un nouveau corpus a été proposé très récemment, contenant 13 915 mots, et dont l'utilisation pourrait donner une représentation plus précise de l'émotion des paroles [WKB13].

10. La **valence** est le degré de plaisir évoqué par le mot en question, l'**excitation** désigne l'intensité émotionnelle, et la **domination** place le mot sur un axe allant de la soumission à la domination.

11. Le « *stemming* » (ou « *racinisation* » en français) est une opération consistant à calculer une base partagée par les différents mots qui dérivent d'une même racine. Par exemple, en anglais, les mots *« love »*, « *loving* » et « *lover* » peuvent être réduits à la même racine « *lov* ».

Frank Sinatra : « Fly me to the Moon »				Cat Power : « He War »			
Mot	Valence	Excitation	Domination	Mot	Valence	Excitation	Domination
words	/	/	/	*hey*	/	/	/
me	8,06	5,97	7,88	*war*	2,08	7,49	4,50
let	/	/	/	*will**	6,60	5,30	6,00
fill	/	/	/	*kill**	1,89	7,86	4,54
heart	7,39	6,34	5,49	*back*	/	/	/
song	7,10	6,07	5,85	*run**	5,67	4,76	5,47
sing	6,77	5,73	5,37	*know*	6,93	5,77	6,90
forever	/	/	/	*never*	/	/	/
long	/	/	/	*meant*	/	/	/
worship	/	/	/	*needle*	/	/	/
Moyenne (variance)	7,02 (0,62)	5,42 (0,99)	5,57 (0,57)	Moyenne (variance)	3,88 (1,97)	6,23 (1,35)	4,88 (0,88)

Tableau 7.2.: Émotion des dix mots les plus fréquents pour deux morceaux de CAL500, et valeurs globales sur tout le morceau.

Les moyennes semblent cohérentes par rapport à l'état d'esprit des morceaux. On constate par ailleurs une variance élevée pour la valence et l'excitation de « He War ». Ceci est dû à de nombreux mots qui ne portent pas l'émotion globale de ce morceau (dans le tableau, on peut voir *will*, *run*, et *know*).

7.2.3. Image de la pochette du disque

Tous les enregistrements commerciaux possèdent une image associée pour leur pochette. L'image est, en principe, en rapport avec la musique qu'elle illustre. Les informations qu'apporte la pochette sont en principe très éloignées de celles du signal.

C'est pourquoi nous avons récupéré des images de pochettes sur Discogs et Last.fm. Ces images sont représentées par de nombreux descripteurs MPEG-7 [12] [SS02]. Nous ajoutons à ces descripteurs un histogramme de couleurs et une estimation du nombre de visages visibles [DNBL08].

7.2.4. Décennie de sortie

Il est commun d'estimer que les créateurs d'une œuvre sont influencés par leur époque et son contexte. On peut donc penser que cette information peut aider à la classification automatique de la musique.

On récupère alors l'année de sortie des morceaux, grâce à Echonest. Même si elle peut être informative, l'année de sortie apparaît comme un descripteur très bruité pour beaucoup de tags. Cette année donc est quantifiée à la décennie, afin de réduire le bruit du descripteur.

7.2.5. Tests de performance

Les performances moyennes des descripteurs contextuels sur CAL500 sont reportées dans le Tableau 7.3. On peut voir que les tags utilisateurs constituent la représentation la plus informative. On constate que tous les descripteurs affichent des

12. Ces descripteurs sont : *scalable color*, *dominant color*, *color layout*, *color structure*, *homogeneous texture*, *edge histogram*, *contour shape* et *region shape*.

Descripteur	MAP (en %)	AROC (en %)
Tags utilisateurs	46,0	65,0
Paroles 16 thèmes	38,6	56,3
Paroles 32 thèmes	38,1	55,9
Émotion paroles	36,5	54,4
Descripteurs d'image	35,2	51,8
Décennie de sortie	36,3	53,2

Tableau 7.3.: Performances des descripteurs contextuels sur CAL500.

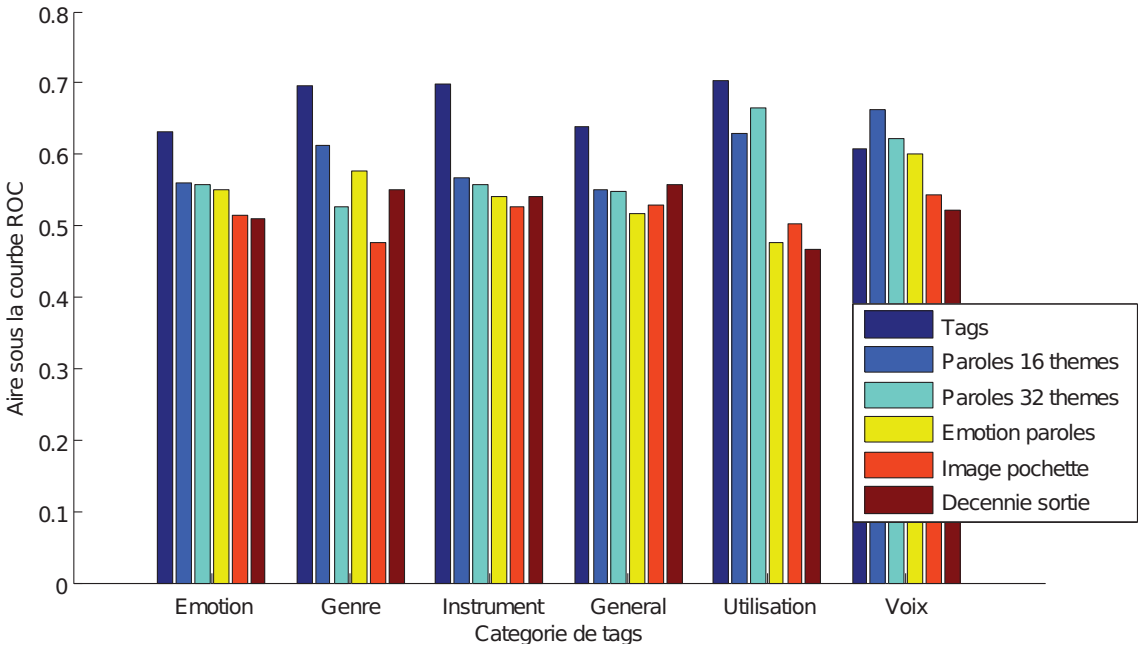


Figure 7.2.: AROC obtenue avec les descripteurs contextuels, groupée par catégorie de tags.

performances au dessus du hasard, même les descripteurs visuels, qui donnent pourtant les prédictions les moins fiables. Cela montre que des informations utiles sont effectivement contenues dans l'image de la pochette.

Dans la Figure 7.2, on peut voir l'AROC des différents descripteurs, groupés par catégories de tags. On peut constater que les tags utilisateurs sont particulièrement appropriés pour prédire les tags de *Genre*, *Instrument* et *Utilisation*. Ce sont en effet des catégories auxquelles les utilisateurs prêtent souvent attention lorsqu'ils taguent des morceaux.

Les thèmes de paroles, même s'ils donnent des performances modestes sur l'ensemble des tags, constituent la meilleure représentation sur les tags de *Voix*. Il ne semble pas

surprenant d'observer que les paroles influencent la manière de chanter. Cette représentation des paroles donne de meilleurs résultats que celle basée sur les émotions, mais celle-ci est tout de même bonne sur les tags de *Genre* et de *Voix*.

Globalement, la Figure 7.2 montre, comme au chapitre 5, que les représentations donnent des informations de différente nature, puisque leurs performances relatives ne sont pas constantes entre les tags.

7.3. Fusion multi-niveaux

7.3.1. Des représentations vivant à différentes échelles

Les descripteurs présentés dans ce chapitre sont très différents de ceux du chapitre 5, extraits du signal. Cette différence s'exprime à plusieurs points de vue. Tout d'abord, ils ouvrent l'accès à de nouvelles modalités, comme les paroles, plus difficiles à déduire du signal [MV10], ou l'image de la pochette. De plus, ces nouvelles modalités et les descriptions qui en sont faites se rapportent à un plus haut niveau d'abstraction. Ces différences laissent supposer que les descriptions du contenu seraient grandement enrichies par les descripteurs de contexte proposés dans ce chapitre.

La manière la plus simple de fusionner ces représentations est de synchroniser les fenêtres d'analyse du morceau, puis de concaténer les vecteurs de description (*cf.* section 6.2 page 76). L'intégration temporelle précoce peut permettre, à cet effet, d'obtenir toutes les représentations sur la durée d'analyse la plus longue. Or puisque les descripteurs de contexte sont la plupart du temps extraits sur tout le morceau (30s de signal, dans notre cas) ; c'est donc sur cette durée globale qu'il faudrait intégrer tous les descripteurs.

Pourtant, nous avons vu dans la section 5.4 que le comportement des descripteurs dépendait de leur durée d'observation, et que 30s n'était pas toujours la durée optimale. À ce sujet, on constate que la meilleure échelle a tendance à s'allonger quand le niveau d'abstraction s'élève. Il semble donc plus intéressant de réaliser un apprentissage qui fusionne les représentations en les laissant sur leur meilleure durée de description.

Méthode de fusion	MAP (en %)	AROC (en %)
Fusion précoce par intégration	50,1	69,0
Fusion multi-échelles	51,4	70,3

Tableau 7.4.: Performances comparées de la fusion multi-échelles par rapport à la fusion précoce par intégration, sur un jeu de descripteurs sélectionnés.

Pour toutes ces raisons, on propose d'utiliser l'algorithme décrit au chapitre 6, afin de fusionner les descriptions tout en les gardant à leur meilleure échelle.

7.3.2. Validation expérimentale

Dans cette expérience, on tente les deux approches de fusion énoncées dans la sous-section 7.3.1 : fusion précoce par intégration temporelle, et fusion semi-tardive par boosting multi-échelles.

Afin d'éviter les problèmes liés à la dimensionnalité des descripteurs, ces deux types de fusion sont appliquées, non pas à tous, mais à un ensemble de descripteurs issus du signal et de son contexte. Ceux-ci sont sélectionnés à la main pour leurs bonnes performances et leur supposée complémentarité : MFCC-moyenne/variance, puissance de la batterie, descripteurs liés à la psycho-acoustique, tags utilisateurs et thèmes des paroles.

Les résultats de cette expérience sont reportés dans le Tableau 7.4. Pour commencer, on constate que les deux systèmes de fusion donnent de bien meilleures performances que celles des descripteurs utilisés individuellement. En effet, la meilleure MAP (obtenue par les descripteurs psycho-acoustiques) était de 47,1%, tandis que la meilleure AROC (avec les tags utilisateurs) était de 65,0%. Cela montre que les descripteurs peuvent bien s'enrichir mutuellement, mais aussi que les deux systèmes de fusion permettent de bénéficier de cette complémentarité.

D'autre part, on voit clairement que la fusion multi-échelles donne de meilleures performances que la fusion précoce. Il semble donc tout à fait utile de prendre en compte les différentes échelles de description lors d'une fusion multi-niveaux.

On remarque tout de même que l'amélioration induite par l'utilisation de plusieurs échelles reste modeste. Une piste pour prolonger cet expérience serait de répliquer

chaque descripteur à toutes les échelles où il fait sens, puis de laisser le boosting sélectionner les échelles les plus utiles. En effet, l'utilité d'exploiter un même descripteur à plusieurs échelles a été démontrée au chapitre 6. En prenant tous les descripteurs que nous avons étudiés, à toutes les échelles, on aboutit à une description de plusieurs centaines d'attributs pour chaque échelle. Heureusement, la complexité du boosting de souches de décision est linéaire en nombre de dimensions (*cf.* chapitre 3). Par contre, il apparaît que l'algorithme de boosting de souches de décision ait des difficultés à gérer un grand nombre de descripteurs. En effet, la sélection d'attributs qu'il réalise est plutôt sommaire[13]. Par conséquent, l'apprentissage est détérioré en présence de nombreux attributs non pertinents. C'est pourquoi nous avons mené des travaux pour proposer une meilleure sélection d'attributs au sein de l'algorithme de boosting. Nos premières évaluations se révèlent encourageantes mais restent non concluantes, c'est pourquoi elles ne sont pas présentées ici.

7.4. Conclusion

Nous avons décrit dans ce chapitre des descripteurs nouveaux ou récents issus du contexte social. Ce dernier apparaît comme une source d'informations très riche, et de nombreuses représentations qui peuvent en découler restent encore à explorer ou à perfectionner. Nous avons ensuite utilisé l'algorithme décrit au chapitre 6 pour fusionner ces représentations avec des descriptions issues du signal, décrites au chapitre 5. Cet algorithme permet de prendre les descripteurs, issus de niveaux d'abstraction différents, à leur échelle optimale. On obtient ainsi de meilleurs résultats qu'avec une fusion précoce par intégration.

Pour de futurs travaux, le système de fusion multi-niveaux bénéficierait probablement d'une adaptation lui permettant de trouver un sous-ensemble optimal de descripteurs. Ceci permettrait d'augmenter le nombre de descripteurs utilisés, et donc de proposer des descriptions plus riches, tout en laissant l'apprentissage sélectionner les dimensions qu'il juge les plus utiles.

On peut également imaginer prendre des classifieurs faibles différents pour traiter chaque échelle. En effet, rien n'oblige notre algorithme de fusion à utiliser le même

13. À chaque itération, l'algorithme cherche le descripteur permettant la meilleure séparation des données par seuillage, mais n'observe pas en profondeur la répartition des exemples dans chaque dimension (*cf.* section 3.3 p. 40).

classifieur pour toutes les échelles, et il pourrait être intéressant de choisir le classifieur en fonction de la nature des données à traiter à chaque échelle.

8. Conclusion

Bilan des travaux

Nous avons proposé dans cette thèse des descriptions de morceaux enrichies et de nouvelles techniques, qui modifient quelque peu le schéma classique du tagging automatique. Ces modifications ont été facilitées par l'utilisation d'algorithmes de boosting, dont la malléabilité rend aisée son adaptation à de nombreuses configurations.

Tout d'abord, nous avons mis au point de nouveaux descripteurs de signal pour la classification, en tentant de mieux couvrir l'axe sémantique. Nous avons pu constater que le comportement des descripteurs extraits de l'audio était différent en fonction de la durée de signal qu'ils représentaient. Même si ce n'est pas toujours aussi simple, on observe globalement que la durée optimale de description s'allonge souvent lorsque l'on va vers les descripteurs de haut niveau. Nous avons également proposé des descripteurs construits à partir de données récupérables en ligne simplement. Ils décrivent davantage le contexte éditorial et social de chaque morceau. Ces descripteurs s'appliquent globalement à la totalité du morceau, et sont tout à fait complémentaires des descripteurs de signal.

Dans un second temps, afin de pouvoir fusionner des représentations aussi différentes, nous avons mis au point un algorithme de fusion multi-échelles, qui permet d'exploiter en même temps des descripteurs extraits sur des fenêtres d'analyse différentes, tout en prenant en compte cet asynchronisme. L'élaboration de cet algorithme a été facilitée par son appui sur le boosting, ce dernier possédant des facilités naturelles à fusionner les données. Lors de nos premiers tests, nous sommes parvenus à améliorer les performances d'un unique ensemble de descripteurs, simplement en l'exploitant à plusieurs échelles temporelles. Il est également intéressant de noter que les échelles

les plus utiles lors de la fusion multi-échelles, ne sont pas celles qui présentent la meilleure performance lorsqu'elles sont exploitées individuellement. Cela montre que la réunion de plusieurs échelles porte une information de nature différente que ces mêmes échelles prises séparément.

Mais l'algorithme de fusion multi-échelles a été conçu pour pouvoir fusionner des représentations qui existent naturellement à des échelles distinctes. C'est en effet en exploitant des descripteurs différents à chaque échelle qu'il se montre particulièrement utile. Cela permet d'optimiser toutes les descriptions en les gardant à leur échelle optimale.

Enfin, nous avons proposé un nouveau cadre d'apprentissage pour le tagging automatique. Ce cadre s'appuie sur la construction d'un nouveau type de vérité-terrain, continue et non plus binaire. Cette vérité-terrain reflète mieux les différents degrés d'association qu'il peut exister entre un morceau et un tag. Nous avons appris ces nouvelles cibles grâce à un algorithme de boosting régressif. L'expérience montre que l'information sur les incertitudes qui peuvent émerger lors de l'annotation des données, est une information utile à l'apprentissage des tags. Il est donc tout à fait conseillé de la prendre en compte.

Perspectives

Nous suggérons plusieurs pistes de recherche, ouvertes par les travaux effectués dans le cadre de cette thèse.

Perspectives générales Le boosting de souches de décision est un algorithme particulièrement peu coûteux en temps de calcul (*cf.* chapitre 3). De plus, il est possible de paralléliser leur construction. C'est pourquoi cet algorithme reste approprié, peut-être même privilégié, pour exploiter des bases de données plus grandes. L'exploitation de CAL500 a permis d'obtenir des résultats dont la fiabilité a été vérifiée statistiquement. Cependant, il peut être instructif de voir si des expériences menées sur des données plus nombreuses peuvent donner des résultats encore plus clairs. De plus, l'intégration par HMM (*cf.* section 5.3, p. 67) bénéficierait probablement d'un plus grand nombre d'exemples d'apprentissage (dans une moindre mesure, l'intégration par sacs de mots pourrait en profiter également).

Pour prolonger l'apprentissage régressif Une première piste pour améliorer l'apprentissage régressif serait d'utiliser des fonctions de coût moins convexes que l'erreur quadratique. En effet, la convexité de cette dernière la rend particulièrement sensible aux données bruitées ou aux erreurs d'annotation [LS10]. Des pertes comme celle de Huber [Hub64] semblent un bon compromis entre la précision de l'apprentissage et la robustesse à ces erreurs [HTF09]. Mais comme nous l'avons vu au chapitre 3 (section 3.4), les algorithmes utilisant d'autres fonctions de coût sont plus compliqués à mettre en place.

D'autre part, dans tous les travaux menés pour cette thèse, nous avons considéré les tags indépendamment les uns des autres. Or nous avons vu à la sous-section 2.3.1 (p. 22) qu'il était possible, et même profitable, d'exploiter les corrélations entre les tags. Ceci pourrait être effectué en menant un apprentissage de régression multiple. Ce type d'algorithme consiste à apprendre à prédire la valeur d'une variable à plusieurs dimensions. Une régression multiple permettrait donc d'apprendre en même temps l'ensemble des tags, et ainsi d'exploiter leurs dépendances. On peut, pour ce faire, adapter l'algorithme de boosting régressif en lui intégrant un coût pour la régression multiple.

À l'étape de la description des morceaux Parmi les nouveaux descripteurs étudiés, certains affichent des performances modestes, mais tout de même meilleures que le hasard. C'est le cas des probabilités de présence des instruments, des descripteurs textuels et visuels. En tant que preuve de concept, nous avons montré que les aspects du morceau représentés par ces descripteurs peuvent être informatifs pour la classification audio. Mais la construction de ces descripteurs peut être grandement affinée.

Par exemple, les probabilités de présence des instruments pourraient faire appel à des classifieurs plus complexes et réglés plus précisément. De plus, les instruments reconnus pourraient être choisis autrement, de manière à mieux représenter des instruments courants dans la pop music. Surtout, les instruments sont appris sur une base de performances solo, et il pourrait être intéressant d'apprendre à les reconnaître dans des morceaux donnant à entendre plusieurs instruments.

Les descripteurs visuels pourraient être explorés individuellement, afin de repérer les plus utiles et ainsi diminuer la dimensionnalité de la représentation. Cela réduirait

également l'effet des descripteurs bruités, ou moins corrélés aux caractéristiques qui nous intéressent, et qui peuvent détériorer la qualité d'analyse.

Par ailleurs, les données utilisateurs ne se limitent pas à des tags. D'autres informations peuvent être utiles à exploiter pour le tagging automatique, comme par exemple la popularité des morceaux, en nombre d'écoutes. Les évaluations des morceaux par les utilisateurs, très utilisées en recommandation automatique [SFHS07], pourraient également trouver une place dans un système de tagging.

Plus globalement, les descripteurs extraits du signal sont souvent prévus pour décrire des portions de signal relativement stables (les descripteurs à court-terme considèrent même le signal comme stationnaire). Cette stabilité pourrait être repérée par une fonction de nouveauté ou un système de segmentation automatique, et ainsi guider le positionnement des fenêtres d'analyse ou de texture. Des fenêtres courtes pourraient être calées sur des notes ou des temps, des fenêtres à moyen-terme correspondraient à une ou plusieurs mesures, tandis que les fenêtres les plus longues engloberaient des sections du morceau. Cela pourrait permettre d'avoir des descripteurs moins bruités et plus expressifs.

À l'étape de l'apprentissage Nous avons évoqué au chapitre 3 que le boosting d'arbres réalise une sélection embarquée de descripteurs. Cependant, cette sélection est assez rudimentaire : elle consiste simplement à sélectionner les attributs qui peuvent, par un simple seuil, séparer au mieux les exemples des deux classes. C'est pourquoi, lors de nos tentatives, l'algorithme présentait des difficultés à exploiter de nombreux descripteurs à la fois (*cf.* chapitre 7). Il pourrait être utile de réaliser une sélection préalable [PR03], mais puisque l'algorithme réalise lui-même une sélection, il serait intéressant de lui permettre de construire lui-même un ensemble d'attributs réduit, et performant pour la prédiction des tags.

Annexes

A. Métriques d'exactitude pour l'évaluation de classifieurs

Résumé Dans cette annexe, nous décrivons les principales métriques de type *accuracy* (*exactitude*) pour l'évaluation de classifieurs. Nous détaillons ainsi les calculs et l'interprétation du taux d'erreur, de la précision, du rappel, de la F-mesure de la *Mean average precision* (précision moyenne) et de l'aire sous la courbe *Receiver Operating Characteristic*.

A.1. Introduction

Pour commencer, nous allons détailler le processus même d'évaluation de l'*exactitude* des classifieurs.

Dans un cas de tagging automatique, nous cherchons à évaluer un ensemble de $|\mathcal{L}|$ classifieurs, chacun cherchant à prédire un tag l donné. Ces classifieurs sont utilisés pour estimer la présence ou non de leur tag associé, sur un ensemble de M morceaux.

L'évaluation comprend donc deux entrées :
- une matrice de prédictions, représentant les sorties des classifieurs, $\mathbf{P}$, de dimension $M \times |\mathcal{L}|$;
- une matrice d'annotations $\mathbf{A}$, binaire, et de même dimension que $\mathbf{P}$, qui décrit la vérité-terrain à laquelle on compare nos prédictions.

A.2. Métriques de récupération (*retrieval*)

Un score de récupération suppose que l'on effectue une requête sur l'ensemble des morceaux, et qu'un sous ensemble non-ordonné est retourné. Puis on mesure la

correspondance entre l'ensemble ainsi récupéré, et l'ensemble des morceaux qui auraient dû être retournés d'après la vérité-terrain. Cette configuration implique donc des prédictions binaires.

Dans notre cas, la requête est un tag l. La réponse est la colonne $M \times 1$ constituée des prédictions correspondant à ce tag : $\mathbf{P}_{m,l}$, $1 < m < M$. En fonction de cette réponse et de la vérité-terrain $\mathbf{A}_{m,l}$, on peut placer les documents dans une matrice de confusion. Cette matrice est un tableau de contingence, croisant deux critères : document retourné/non-retourné et document approprié/non-approprié (Tableau A.1) [HKTR04]. Ces catégories correspondent respectivement aux valeurs de $\mathbf{P}$ et $\mathbf{A}$. Les classes retourné/non-retourné sont aussi appelées positif/négatif. On a donc des documents vrais positifs (VP), des faux positifs (FP), et de même pour les négatifs.

	Retourné	Non retourné	Total
Approprié	VP	FN	N_a
Inapproprié	FP	VN	N_i
Total	N_p	N_n	M

Tableau A.1.: Matrice de confusion.

Les valeurs ainsi décrites permettent de définir toutes les mesures de retrieval. Ainsi, le **taux d'erreur** est la probabilité qu'une prédiction soit incorrecte :

$$e = \frac{VP + VN}{M} \tag{A.1}$$

La précision et le rappel sont des mesures très populaires, complémentaires l'une de l'autre. La **précision** est la proportion de vrais positifs sur le nombre de documents retournés :

$$p = \frac{VP}{N_p} \tag{A.2}$$

La précision représente donc la probabilité qu'un document retourné soit correct. Le **rappel** est la proportion des documents valides qui ont été effectivement retournés par le système :

$$r = \frac{VP}{N_a} \tag{A.3}$$

Le rappel représente donc la probabilité qu'un document approprié se retrouve dans la réponse du système.

Si la tâche évaluée est de suggérer une liste de k documents, la précision seule peut éventuellement suffire. Si au contraire, on cherche à récupérer tous les documents valides, le rappel est le plus important. Cependant, précision et rappel se complètent dans la description du comportement d'un système de recherche, et il est rare d'observer l'un sans garder un oeil sur l'autre. En général, si l'on demande au système de retourner davantage de documents (k augmente), le rappel va augmenter tandis que la précision diminuera. Cette dualité amène à envisager une nouvelle mesure qui décrit le compromis entre les deux autres.

C'est dans ce but qu'est définie la **F-mesure** :

$$f = \frac{2pr}{p + r} \tag{A.4}$$

La variation de la F-mesure en fonction de la précision et du rappel est montrée dans la Figure A.1. On peut y voir que lorsque précision et rappel présentent des valeurs bien différentes, la F-mesure est plus sensible aux variations de la mesure la plus faible.

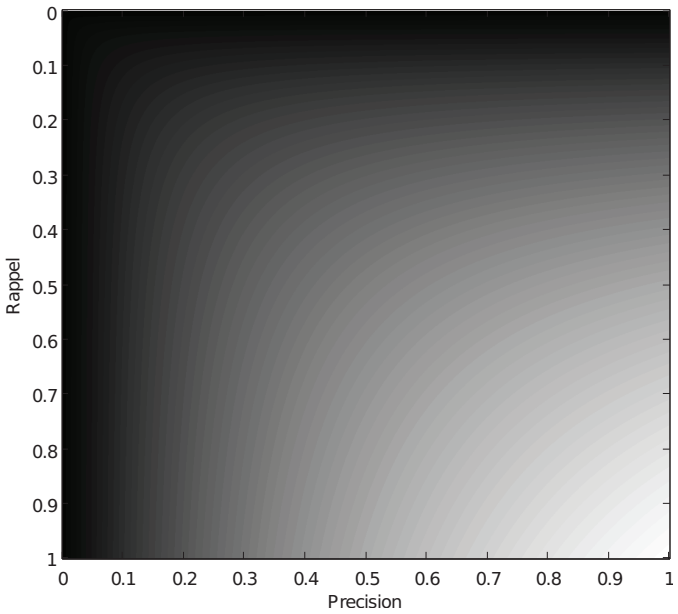


Figure A.1.: F-mesure en fonction de la précision et du rappel.

Une autre approche est souvent utilisée pour combiner la précision et le rappel : il s'agit de calculer la précision moyenne parmi les différents niveaux de rappel possibles. Cette mesure est communément appelée *Mean Average Precision* (MAP). Mais dans ce cas, le cadre devient différent puisqu'on fait varier les valeurs de la matrice de confusion. On est en fait plutôt dans un cadre de classement (*ranking)*, ce que nous décrivons dans la section suivante.

A.3. Métriques de classement

Puisque les métriques de retrieval changent en fonction du nombre de documents retournés, il peut être intéressant d'observer ces variations. En effet, la plupart des classifieurs sont capables de donner des prédictions réelles, et la taille de la réponse dépend généralement d'un paramètre très simple et empirique (taille de réponse fixe, ou seuil de détection, comme dans l'Algorithme 3.1). Le choix de ce paramètre n'est en général pas inhérent à l'algorithme de classification, ce qui nous amène à imaginer des mesures qui résument le comportement du système pour plusieurs tailles de réponse.

En pratique, les mesures de ranking permettent d'évaluer une liste de documents triés par le système, par ordre de probabilité d'association avec le tag de la requête (t). Comme nous disposons d'une vérité-terrain binaire, cela revient à observer les variations des valeurs du Tableau A.1 quand on modifie la taille de la réponse. Il s'agit alors de retourner les N_p documents les plus probables, et d'observer les changements pour plusieurs valeurs de N_p. Un système parfait placerait tous les documents réellement associés au tag t au début de la liste.

La première métrique de ranking que nous observons est la **Mean Average Precision** (MAP) [TS06]. Cette mesure est la moyenne sur tous les tags de l'*Average Precision* (AP_t). Pour calculer l'Average Precision associée à un tag l, on parcourt la liste triée des documents retournés, du plus au moins probable. A chaque fois que l'on tombe sur un document valide m (donc lorsque le rappel change), on calcule la précision p_m associée l'ensemble des documents rangés avant m (m inclus). La moyenne de ces précisions $p_{m,l}$ donne l'Average Precision. Soit $\hat{\mathbf{P}}_{s,l}$ la matrice des prédictions probabilistes, et soit $\tilde{\mathbf{A}}_{m,l}$ la liste des annotations triées par prédiction

$\hat{\mathbf{P}}_{m,l}$ décroissante. L'average precision s'écrit alors :

$$AP_l = \frac{1}{\sum_{m=1}^{k} \tilde{\mathbf{A}}_{m,l}} \sum_{m=1}^{k} \tilde{\mathbf{A}}_{m,l} p_{m,l} \ , \tag{A.5}$$

où k est le nombre de morceaux retournés, et $p_{m,l} = \frac{\sum_{i=1}^{m} \tilde{\mathbf{A}}_{i,l}}{m}$ est la précision si l'on prend les m morceaux les plus probables. La mean average precision est ensuite la moyenne des AP :

$$MAP = \text{mean}_l \ AP_l \tag{A.6}$$

Comme nous l'avons évoqué à la fin de la section A.2, la mean average precision revient à calculer la precision moyenne pour les différentes valeurs de rappel. On note que la MAP est donc sensible à la probabilité *a priori* des tags. En effet, plus un tag est présent, plus il a de chances d'avoir une MAP élevée (surtout si k est grand).

Le comportement du système de retrieval peut également être examiné avec la courbe **Receiver operating characteristic** (ROC) [DHS00]. Cette courbe représente le rappel en fonction du taux de fausse alarme. Ce dernier est défini comme la proportion parmi les documents non-valides, de ceux qui sont tout de même retournés par la requête :

$$FA = \frac{FP}{N_n} \tag{A.7}$$

Des prédictions binaires s'obtiennent par seuillage des prédictions souples. La courbe ROC se dessine en faisant varier le seuil de détection sur l'ensemble des valeurs de $\hat{\mathbf{P}}_{m,l}$. On obtient ainsi une bonne représentation de la capacité du classifieur à séparer les données entre les deux classes.

L'aire sous la courbe ROC (AROC) donne une mesure de la qualité de cette courbe [Bra97]. Sa valeur se situe entre 0 (pire cas) et 1 (détection parfaite), 0,5 étant la performance du hasard. Dans toutes nos évaluations, nous calculons cette valeur par intégration trapézoïdale (d'autres méthodes sont possibles). Il est important de noter que ce type d'interpolation conduira nécessairement à une légère sous-estimation de l'aire AUC puisque la courbe ROC est, en principe, concave.

L'AROC donne donc une idée globale de la capacité du système à séparer les exemples positifs des négatifs. Contrairement à la MAP, cette mesure n'est pas sensible à la distribution *a priori* des valeurs d'annotation.

B. Tests statistiques pour l'évaluation des prédictions

Résumé La significativité statistique d'un résultat d'expérience a parfois besoin d'être vérifiée. Cette annexe donne quelques détails sur les tests statistiques évoqués au cours de cette thèse. Ces tests sont : le test de McNemar et le test de Student par séries appariées avec validation croisée.

B.1. Introduction

Nous notons dans le chapitre 2 que le choix des données d'apprentissage est très important pour la construction d'un système de classification. En effet, le comportement du système va être très influencé par les données sur lesquelles il est appliqué. En entraînant et en évaluant des algorithmes sur un jeu de données en particulier, il est donc possible d'observer certaines différences de comportement, notamment des différences dans leurs performances, qui sont seulement dues à des spécificités des données. Lorsque des différences de performances mesurées sont ténues, il peut alors être nécessaire d'utiliser des tests statistiques pour estimer si ces différences sont statistiquement significatives.

Ces tests modélisent une mesure de performance comme un tirage d'une variable aléatoire, suivant une loi non connue d'espérance μ. Deux classifieurs c_a et c_b, possédant des performances identiques, ont donc des distributions équivalentes (notamment, $\mu_a = \mu_b$). Cette affirmation constitue l'hypothèse nulle. Toutefois, même sous cette hypothèse nulle, les performances mesurées lors d'une expérience sont rarement strictement égales. Ce que les tests statistiques cherchent à déterminer, c'est la probabilité, d'après les expériences, que les espérances μ_a et μ_b soient identiques. Cette

probabilité est appelée *valeur p*, ou *p-valeur*. On considère les espérances différentes lorsque $p \leq 0{,}05$ (parfois, $p \leq 0{,}01$), et l'hypothèse nulle est alors rejetée.

L'analyse statistique des résultats permet également de déterminer des intervalles de confiance. Ces intervalles donnent une fourchette de valeurs qui contient très probablement l'espérance de la mesure analysée (le niveau de confiance est en général de 95%).

Nous décrivons ici les tests statistiques utilisés dans les travaux de cette thèse : le test de McNemar et le test de Student.

B.2. Test de McNemar

Le test de McNemar est appliqué lorsque les données sont séparées en un unique ensemble d'apprentissage, et un ensemble de test [Eve77]. Les classifieurs c_1 et c_2 sont entraînés sur l'ensemble d'apprentissage avant d'être utilisés pour prédire les classes des éléments de l'ensemble de test. D'après les prédictions données par les classifieurs c_a et c_b sur l'ensemble de test, on construit un tableau de contingence, comptant les morceaux selon qu'ils sont bien ou mal classifiés par c_a ou c_b (Tableau B.1).

	Mal classifié par c_b	Bien classifié par c_b
Mal classifié par c_a	N_{00}	N_{01}
Bien classifié par c_a	N_{10}	N_{11}

Tableau B.1.: Tableau de contingence utilisé par le test de McNemar.

Sous l'hypothèse nulle, les deux algorithmes ont le même taux d'erreur, donc $N_{01} = N_{10}$. On calcule alors la statistique suivante :

$$\chi^2 = \frac{(|N_{01} - N_{10}| - 1)^2}{N_{01} + N_{10}} \tag{B.1}$$

Cette statistique peut être approximée par une loi du χ^2 à un degré de liberté (le terme -1 au numérateur est ajouté à la statistique standard, pour prendre en compte le fait que cette statistique est discrète alors que la loi du χ^2 est continue).

En regardant dans une table de distribution [1], on déduit que si l'hypothèse nulle est vraie, alors χ^2 a 5% de chances d'être supérieure à $\chi^2_{0,05} = 3{,}841$. Ainsi, si la valeur χ^2 calculée est supérieure, on peut en déduire que les deux classifieurs c_a et c_b ont des performances significativement différentes [2].

B.3. Test de Student par séries appariées avec validation croisée

Pour ce test [3], on effectue R apprentissages sur les mêmes données [Die98]. À chaque étape r (*fold)*, on divise les données en un ensemble d'apprentissage A_r, et un ensemble de test T_r. Les A_r forment une partition de la base de données.

On obtient ainsi, pour chaque fold, deux taux d'erreur $\epsilon_a^{(r)}$ et $\epsilon_b^{(r)}$. On suppose alors que les différences $\epsilon^{(r)} = \epsilon_a^{(r)} - \epsilon_b^{(r)}$ sont indépendantes et distribuées selon une loi normale. Si l'hypothèse nulle est vraie, l'espérance $E\left(\epsilon^{(r)}\right) = 0$. On calcule la statistique suivante :

$$t = \frac{\bar{\epsilon}\sqrt{R}}{\sqrt{\frac{1}{R-1}\sum_{r=1}^{R}\left(\epsilon^{(r)} - \bar{\epsilon}\right)^2}} \tag{B.2}$$

où $\bar{\epsilon} = \frac{1}{R}\sum_{r=1}^{R}\epsilon^{(r)}$. Sous l'hypothèse nulle, cette statistique suit une loi de Student avec $n-1$ degrés de liberté. En regardant dans une table de distribution [4], on déduit qu'avec 10 folds de validation croisée, si l'hypothèse nulle est vraie, alors t a 5% de chances d'être supérieure à $t_{0,05} = 2{,}262$. On peut donc raisonnablement la rejeter si la valeur t calculée est supérieure [5].

1. Par exemple `http ://sites.stat.psu.edu/~mga/401/tables/Chi-square-table.pdf`
2. Pour information, $\chi^2_{0,01} = 6{,}635$.
3. Aussi appelé « *test t* », ou « *t-test* » en anglais.
4. Par exemple `http ://www.sjsu.edu/faculty/gerstman/StatPrimer/t-table.pdf`
5. Pour information, $t_{0,01} = 2{,}821$.

C. Liste des tags analysés pour les tests

Tag original	Traduction en français
Emotion-Arousing/Awakening	Émotion-Excité/Éveillé
Emotion-Not-Arousing/Awakening	Émotion-Pas-Excité/Éveillé
Emotion-Not-Bizarre/Weird	Émotion-Pas-Bizarre/Étrange
Emotion-Calming/Soothing	Émotion-Calme/Apaisant
Emotion-Carefree/Lighthearted	Émotion-Insouciant/Enjoué
Emotion-Not-Carefree/Lighthearted	Émotion-Pas-Insouciant/Enjoué
Emotion-Cheerful/Festive	Émotion-Enjoué/Festif
Emotion-Not-Cheerful/Festive	Émotion-Pas-Enjoué/Festif
Emotion-Emotional/Passionate	Émotion-Émotif/Passionné
Emotion-Not-Emotional/Passionate	Émotion-Émotif/Passionné
Emotion-Exciting/Thrilling	Émotion-Passionnant/Palpitant
Emotion-Happy	Émotion-Heureux
Emotion-Laid_back/Mellow	Émotion-Tranquille/Serein
Emotion-Not-Laid_back/Mellow	Émotion-Pas-Tranquille/Serein
Emotion-Light/Playful	Émotion-Léger/Espiègle
Emotion-Not-Light/Playful	Émotion-Léger/Espiègle
Emotion-Loving/Romantic	Émotion-Affectueux/Romantique
Emotion-Not-Loving/Romantic	Émotion-Pas-Affectueux/Romantique
Emotion-Pleasant/Comfortable	Émotion-Agréable/Confortable
Emotion-Not-Pleasant/Comfortable	Émotion-Pas-Agréable/Confortable
Emotion-Positive/Optimistic	Émotion-Positif/Optimiste
Emotion-Not-Positive/Optimistic	Émotion-Pas-Positif/Optimiste
Emotion-Powerful/Strong	Émotion-Puissant/Fort

Tag original	Traduction en français
Emotion-Not-Powerful/Strong	Émotion-Pas-Puissant/Fort
Emotion-Sad	Émotion-Triste
Emotion-Tender/Soft	Émotion-Tendre/Doux
Emotion-Not-Tender/Soft	Émotion-Pas-Tendre/Doux
Emotion-Touching/Loving	Émotion-Touchant/Affectueux
Genre-Alternative	Genre-Alternatif
Genre-Classic_Rock	Genre-Rock_classique
Genre-Electronica	Genre-Électronique
Genre-Pop	Genre-Pop
Genre-Rock	Genre-Rock
Instrument-Acoustic_Guitar	Instrument-Guitare_acoustique
Instrument-Backing_Vocals	Instrument-Chœurs
Instrument-Bass	Instrument-Basse
Instrument-Drum_Set	Instrument-Batterie
Instrument-Electric_Guitar_(clean)	Instrument-Guitare_électrique_(claire)
Instrument-Electric_Guitar_(distorted)	Instrument-Guitare_électrique_(distordue)
Instrument-Female_lead_vocals	Instrument-Voix_lead_masculine
Instrument-Male_lead_vocals	Instrument-Voix_lead_féminine
Instrument-Piano	Instrument-Piano
Instrument-Synthesizer	Instrument-Synthétiseur
Song-Catchy/Memorable	Morceau-Accrocheur/Marquant
Song-Not-Catchy/Memorable	Morceau-Pas-Accrocheur/Marquant
Song-Not-Changing_energy_level	Morceau-Pas-Niveau_d'énergie_changeant
Song-Fast_tempo	Morceau-Tempo_rapide
Song-Not-Fast_tempo	Morceau-Pas-Tempo_rapide
Song-Heavy_beat	Morceau-Rythmique_puissante
Song-Not-Heavy_beat	Morceau-Pas-Rythmique_puissante
Song-High_energy	Morceau-Très_énergique
Song-Not-High_energy	Morceau-Pas-Très_énergique
Song-Texture_acoustic	Morceau-Texture_acoustique
Song-Texture_electric	Morceau-Texture_électrique
Song-Texture_synthesized	Morceau-Texture_synthétique
Song-Tonality	Morceau-Tonalité

Tag original	Traduction en français
Song-Very_danceable	Morceau-Très_dansant
Song-Not-Very_danceable	Morceau-Pas-Très_dansant
Usage-At_a_party	Utilisation-Lors_d'une_fête
Vocals-Emotional	Voix-Émouvante
Vocals-Strong	Voix-Puissante

Publications de l'auteur

Publications dans le cadre du doctorat

R. Foucard, S. Essid, M. Lagrange, and G. Richard. Multi-scale temporal fusion by boosting for music classification. *International Society for Music Information Retrieval (ISMIR)*, Miami, FL, USA, October 2011.

R. Foucard, S. Essid, M. Lagrange, and G. Richard. A Regressive boosting approach to automatic audio tagging based on soft annotator fusion. *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, Kyoto, Japan, March 2012.

R. Foucard, S. Essid, M. Lagrange, and G. Richard. Étiquetage automatique de musique : une approche de boosting régressif basée sur une fusion souple d'annotateurs. *Compression et Représentation des Signaux Audiovisuels (CORESA)*, Lille, France, May 2012.

R. Foucard, S. Essid, G. Richard and M. Lagrange. Exploring new features for music classification. *International Workshop on Image and Audio Analysis for Multimedia Interactive Services (Wiamis)*, Paris, France, July 2013.

Publication à l'issue du stage de master

R. Foucard, J. Durrieu, M. Lagrange, and G. Richard. Multimodal similarity between musical streams for cover version detection. *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, Dallas, TX, USA, March 2010.

Bibliographie

[ADP07] J. Aucouturier, B. Defreville, and F. Pachet. The bag-of-frames approach to audio pattern recognition : a sufficient model for urban soundscapes but not for polyphonic music. *Journal of the Acoustical Society of America (JASA)*, 122(2) : 881–91, August 2007.

[AHEK10] P. Atrey, M. Hossain, A. El Saddik, and M. Kankanhalli. Multimodal fusion for multimedia analysis : a survey. *Multimedia Systems*, 16(6) : 345–379, April 2010.

[AP08] J. Aucouturier and E. Pampalk. Introduction - From Genres to Tags : A Little Epistemology of Music Information Retrieval Research. *Journal of New Music Research*, 37(2) : 87–92, June 2008.

[APRB07] J. Aucouturier, F. Pachet, P. Roy, and A. Beurivé. Signal + Context = Better Classification. In *International Society for Music Information Retrieval (ISMIR)*, pages 425–430, Vienna, Austria, September 2007.

[ARD05] M. Alonso, G. Richard, and B. David. Extracting note onsets from musical recordings. In *IEEE International Conference on Multimedia and Expo (ICME)*, Amsterdam, Netherlands, July 2005.

[BBLP10] Y. Boureau, F. Bach, Y. LeCun, and J. Ponce. Learning mid-level features for recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, San Francisco, CA, USA, June 2010.

[BCE+06] J. Bergstra, N. Casagrande, D. Erhan, D. Eck, and B. Kégl. Aggregate features and ADABOOST for music classification. *Machine Learning*, 65(2-3) : 473–484, 2006.

[BCL10] L. Barrington, A. Chan, and G. Lanckriet. Modeling music as a dynamic texture. *IEEE Transactions on Audio, Speech and Language Processing (TASLP)*, 18(3) : 602–612, March 2010.

[BCTL07] L. Barrington, A. Chan, D. Turnbull, and G. Lanckriet. Audio Information Retrieval using Semantic Similarity. In *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, volume 2, pages II–725 – II–728, Honolulu, Hawaï, 2007.

[BHW10] A. Ben-Hur and J. Weston. A user's guide to support vector machines. *Methods in molecular biology*, 609 : 223–39, January 2010.

[Bis06] C. Bishop. *Pattern Recognition and Machine Learning*. Information science and statistics. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[BK06] J. Bergstra and B. Kégl. Meta-features and AdaBoost for music classification. In *Machine Learning Journal : Special Issue on Machine Learning in Music*, 2006.

[BL99] M. Bradley and P. Lang. Affective Norms for English Words (ANEW) : Instruction Manual and Affective Ratings. Technical report, Center for Research in Psychophysiology, University of Florida, Gainesville, FL, USA, 1999.

[BMEM10] T. Bertin-Mahieux, D. Eck, and M. Mandel. Automatic Tagging of Audio : The State-of-the-Art. In Wenwu Wang, editor, *Machine Audition : Principles, Algorithms and Systems*. IGI Publishing, 2010.

[BMEML08] T. Bertin-Mahieux, D. Eck, F. Maillet, and P. Lamere. Autotagger : a model for Predicting social tags from acoustic features on large music databases. *Journal of New Music Research*, 37(2) : 115–135, June 2008.

[BMEWL11] T. Bertin-Mahieux, D. Ellis, B. Whitman, and P. Lamere. The Million Song Dataset. In *International Society for Music Information Retrieval (ISMIR)*, Miami, FL, USA, October 2011.

[Bow09] A. Bowie. Music aesthetics and critical theory. In J. Harper-Scott and J. Samson, editors, *An Introduction to music studies*, chapter 5, pages 79–94. Cambridge University Press, 2009.

[BP05] J. Bello and J. Pickens. A Robust Mid-Level Representation for Harmonic Content in Music Signals. In *International Society for Music Information Retrieval (ISMIR)*, pages 304–311, London, UK, September 2005.

[Bra97] A. Bradley. The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7) : 1145–1159, July 1997.

[Bre98] L. Breiman. Arcing classifiers. *Annals of Statistics*, 26(3) : 801–849, June 1998.

[BTYL09] L. Barrington, D. Turnbull, M. Yazdani, and G. Lanckriet. Combining audio content and social context for semantic music discovery. In *ACM Special Interest Group on Information Retrieval (SIGIR)*, pages 387–394, Boston, MA, USA, July 2009. ACM.

[BYTL08] L. Barrington, M. Yazdani, D. Turnbull, and G. Lanckriet. Combining Feature Kernels for Semantic Music Retrieval. In *International Society for Music Information Retrieval (ISMIR)*, pages 614–619, Philadelphia, PA, USA, September 2008.

[CCH06] Ò. Celma, P. Cano, and P. Herrera. Search Sounds : An audio crawler focused on weblogs. In *International Society for Music Information Retrieval (ISMIR)*, Victoria, Canada, October 2006.

[CCL11] E. Coviello, A. Chan, and G. Lanckriet. Time Series Models for Semantic Music Annotation. *IEEE Transactions on Audio, Speech and Language Processing (TASLP)*, 19(5) : 1343–1359, July 2011.

[CCL12] E. Coviello, A. Chan, and G. Lanckriet. Multivariate autoregressive mixture models for music auto-tagging. In *International Society for Music Information Retrieval (ISMIR)*, pages 547–552, Porto, Portugal, October 2012.

[CK01] A. Clare and R. King. Knowledge discovery in multi-label phenotype data. *Principles of data mining and knowledge discovery*, 2168 : 42–53, 2001.

[Con06] A. Cont. Realtime Audio to Score Alignment for Polyphonic Music Instruments Using Sparse Non-negative constraints and Hierarchical HMMs. In *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, Toulouse, France, May 2006.

[Cou10] P. Coulangeon. Les métamorphoses de la légitimité. *Actes de la recherche en sciences sociales*, 181-182(1-2) : 88–105, 2010.

[DHS00] R. Duda, P. Hart, and D. Stork. *Pattern Classification (2nd Edition)*. Wiley-Interscience, 2nd edition, November 2000.

[Die98] T. Dietterich. Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms. *Neural Computation*, 10(7) : 1895–1923, 1998.

[DNBL08] P. Dunker, S. Nowak, A. Begau, and C. Lanz. Content-based mood classification for photos and music. In *ACM International conference on Multimedia Information Retrieval (MIR)*, page 97, Vancouver, Canada, October 2008.

[ECL11] K. Ellis, E. Coviello, and G. Lanckriet. Semantic annotation and retrieval of music using a bag of systems representation. In *International Society for Music Information Retrieval (ISMIR)*, Miami, FL, USA, October 2011.

[ELBMG07] D. Eck, P. Lamere, T. Bertin-Mahieux, and S. Green. Automatic Generation of Social Tags for Music Recommendation. In *Advances in neural information processing systems (NIPS)*, 2007.

[Ell96] D. Ellis. *Prediction-driven computational auditory scene analysis*. PhD thesis, Massachussetts Institute of Technology, 1996.

[Ess05] S. Essid. *Classification automatique des signaux audio-fréquences : reconnaissance des instruments de musique*. PhD thesis, Université Pierre et Marie Curie, 2005.

[Eve77] B. Everitt. *The Analysis of contingency tables*. Chapman and Hall, London, 1977.

[FELR11] R. Foucard, S. Essid, M. Lagrange, and G. Richard. Multi-scale temporal fusion by boosting for music classification. In *International Society for Music Information Retrieval (ISMIR)*, Miami, FL, USA, October 2011.

[FHT00] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression : a statistical view of boosting. *Annals of Statistics*, 28(2) : 337–407, 2000.

[FISS03] Y. Freund, R. Iyer, R. Schapire, and Y. Singer. An Efficient Boosting Algorithm for Combining Preferences. *Journal of Machine Learning Research*, 4 : 933–969, 2003.

[FLTZ11] Z. Fu, G. Lu, K. Ting, and D. Zhang. A Survey of Audio-Based Music Classification and Annotation. *IEEE Transactions on Multimedia*, 13(2) : 303–319, April 2011.

[Fri01] J. Friedman. Greedy function approximation : a gradient boosting machine. *Annals of Statistics*, 29(5) : 1189—-1232, 2001.

[FS96] Y. Freund and R. Schapire. Experiments with a New Boosting Algorithm. In *International Conference on Machine Learning (ICML)*, pages 148–156, Bari, Italy, July 1996.

[FS97] Y. Freund and R. Schapire. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *Journal of Computer and System Sciences*, 55(1) : 119–139, August 1997.

[FS99] Y. Freund and R. Schapire. A Short Introduction to Boosting. In *International Joint Conference on Artificial Intelligence (IJCAI)*, volume 14, pages 1401–1406, Stockholm, Sweden, July 1999.

[Fuj99] T. Fujishima. Realtime chord recognition of musical sound : A system using common lisp music. In *International Computer Music Conference (ICMC)*, pages 464–467, Beijing, China, October 1999.

[GÓ6] E. Gómez. *Tonal description of music audio signals*. PhD thesis, Music Technol. Group, Univ. Pompeu Fabra, Barcelona, Spain, 2006.

[GG05] E. Gaussier and C. Goutte. Relation between PLSA and NMF and implications. In *ACM Special Interest Group on Information Retrieval (SIGIR)*, pages 601–602, Salvador, Brazil, 2005.

[GMK10] P. Grosche, M. Müller, and F. Kurth. Cyclic Tempogram - A Mid-level Tempo Representation For Music Signals. In *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, Dallas, Texas, USA, March 2010.

[Gro12] P. Grosche. *Signal Processing Methods for Beat Tracking , Music Segmentation , and Audio Retrieval*. PhD thesis, Max-Planck-Institut für Informatik, 2012.

[HBE12] P. Hamel, Y. Bengio, and D. Eck. Building musically-relevant audio features through multiple timescale representations. In *International Society for Music Information Retrieval (ISMIR)*, pages 553–558, Porto, Portugal, October 2012.

[HCS11] X. He, D. Cai, and Y. Shao. Laplacian regularized gaussian mixture model for data clustering. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 23(9) : 1406–1418, 2011.

[HKTR04] J. Herlocker, J Konstan, L. Terveen, and J. Riedl. Evaluating collaborative filtering recommender systems. *Transactions on Information Systems*, 22(1) : 5–53, 2004.

[HLBE11] P. Hamel, S. Lemieux, Y. Bengio, and D. Eck. Temporal pooling and multiscale learning for automatic annotation and ranking of music audio. In *International Society for Music Information Retrieval (ISMIR)*, Miami, FL, USA, October 2011.

[HTF09] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer, New York, NY, USA, 3rd edition, 2009.

[Hub64] P. Huber. Robust Estimation of a Location Parameter. *Annals of Mathematical Statistics*, 35(1) : 73—-101, December 1964.

[JER09] C. Joder, S. Essid, and G. Richard. Temporal Integration for Audio Classification with Application to Musical Instrument Classification. *IEEE Transactions on Audio, Speech and Language Processing (TASLP)*, 17 : 174–186, 2009.

[JL10] I. Jhuo and D. Lee. Boosting-based multiple kernel learning for image re-ranking. In *ACM International conference on Multimedia (MM)*, pages 1159–1162, Florence, Italy, October 2010.

[Joa98] T. Joachims. Making Large-Scale SVM Learning Practical. Technical report, Universität Dortmund, Dortmund, 1998.

[KMW94] J. Krimphoff, S. McAdams, and S. Winsberg. Caractérisation du timbre des sons complexes - II. Analyses acoustiques et quantification psychophysique. *Journal de Physique*, 4(C5) : 625–628, 1994.

[KPS+08] P. Knees, T. Pohle, M. Schedl, D. Schnitzer, and K. Seyerlehner. A document-centered approach to a natural language music search engine. In *European Conference on Information Retrieval (ECIR)*, pages 627–631, Glasgow, Scotland, UK, March 2008. Springer-Verlag.

[KRG13] S. Kiranyaz, J. Raitoharju, and M. Gabbouj. Evolutionary feature synthesis for content-based audio retrieval. In *International Conference on Communications, Signal Processing, and their Applications (ICCSPA)*, pages 1–6, Sharjah, UAE, February 2013.

[KS10] Y. Kim and E. Schmidt. Music Emotion Recognition : a State of the

Art Review. In *International Society for Music Information Retrieval (ISMIR)*, pages 255–266, 2010.

[Lam08] P. Lamere. Social tagging and music information retrieval. *Journal of New Music Research*, 37(2) : 101–114, 2008.

[Lar08] M. Lardeur. *Robustesse des systèmes de classification automatique des signaux audio-fréquences aux effets sonores*. Master thesis, Université Pierre et Marie Curie, 2008.

[Law08] E. Law. The Problem of accuracy as an evaluation criterion. In *International Conference on Machine Learning (ICML)*, Helsinki, Finland, July 2008.

[LBR11] A. Liutkus, R. Badeau, and G. Richard. Gaussian Processes for Underdetermined Source Separation. *IEEE Transactions on Signal Processing*, 59(7) : 3155–3167, July 2011.

[LCB+04] G. Lanckriet, N. Cristianini, P. Bartlett, L. Ghaoui, and M. Jordan. Learning the Kernel Matrix with Semidefinite Programming. *Journal of Machine Learning Research*, 5 : 27–72, 2004.

[LGH08] C. Laurier, J. Grivolla, and P. Herrera. Multimodal Music Mood Classification Using Audio and Lyrics. In *International Conference on Machine Learning and Applications (ICMLA)*, pages 688–693, San Diego, CA, USA, December 2008.

[LK77] J. Landis and G. Koch. The measurement of observer agreement for categorical data. *Biometrics*, 33(1) : 159–174, 1977.

[Log00] B. Logan. Mel Frequency Cepstral Coefficients for Music Modeling. In *International Society for Music Information Retrieval (ISMIR)*, Plymouth, MA, USA, October 2000.

[LP08] D. Little and B. Pardo. Learning musical instruments from mixtures of audio with weak labels. In *International Society for Music Information Retrieval (ISMIR)*, Philadelphia, PA, USA, September 2008.

[LR05] T. Lidy and A. Rauber. Evaluation of Feature Extractors and Psycho-Acoustic Transformations for Music Genre Classification. In *International Society for Music Information Retrieval (ISMIR)*, London, UK, September 2005.

[LS09a] M. Levy and M. Sandler. Music information retrieval using social tags and audio. *IEEE Transactions on Multimedia*, 11(3) : 1–14, 2009.

[LS09b] N. List and H. Simon. SVM-optimization and steepest-descent line search. In *Conference on Learning Teory (COLT)*, Montreal, Canada, June 2009.

[LS10] P. Long and R. Servedio. Random classification noise defeats all convex potential boosters. *Machine Learning*, 78(3) : 287–304, December 2010.

[LSC06] M. Levy, M. Sandler, and M. Casey. Extraction of high-level musical structure from audio data and its application to thumbnail generation. In *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, volume 5, Toulouse, France, May 2006.

[LSSH09] C. Laurier, M. Sordo, J. Serrà, and P. Herrera. Music Mood Representations from Social Tags. In *International Society for Music Information Retrieval (ISMIR)*, pages 381–386, Kobe, Japan, October 2009.

[LvA09] E. Law and L. von Ahn. Input-agreement : a new mechanism for collecting data using human computation games. In *ACM Conference on Human Factors in Computing Systems (CHI)*, pages 1197–1206, Boston, MA, USA, April 2009.

[LWM+09] E. Law, K. West, M. Mandel, M. Bay, and S. Downie. Evaluation of algorithms using games : the case of music tagging. In *International Society for Music Information Retrieval (ISMIR)*, Kobe, Japan, October 2009.

[Mak75] J. Makhoul. Linear prediction : A tutorial review. *Proceedings of the IEEE*, 63(4) : 561–580, April 1975.

[MALH07] A. Meng, P. Ahrendt, J. Larsen, and Lars Kai Hansen. Temporal Feature Integration for Music Genre Classification. *IEEE Transactions on Audio, Speech and Language Processing (TASLP)*, 15(5) : 1654–1664, July 2007.

[MB03] M. Mckinney and J. Breebaart. Features for Audio and Music Classification. In *International Society for Music Information Retrieval (ISMIR)*, pages 151–158, 2003.

[MCJT12] J. Moore, S. Chen, T. Joachims, and D. Turnbull. Learning to Embed Songs and Tags for Playlist Prediction. In *International Society for*

Music Information Retrieval (ISMIR), pages 349–354, Porto, Portugal, October 2012.

[MD10] M. Mauch and S. Dixon. Simultaneous Estimation of Chords and Musical Context From Audio. *IEEE Transactions on Audio, Speech and Language Processing (TASLP)*, 18(6) : 1280–1289, August 2010.

[ME05] M. Mandel and D. Ellis. Song-Level Features and Support Vector Machines for Music Classification. In Joshua D Reiss and Geraint A Wiggins, editors, *International Society for Music Information Retrieval (ISMIR)*, pages 594–599, London, UK, September 2005.

[ME08a] M. Mandel and D. Ellis. A Web-Based Game for Collecting Music Metadata. *Journal of New Music Research*, 37(2) : 151–165, 2008.

[ME08b] M. Mandel and D. Ellis. Multiple-instance learning for music information retrieval. In *International Society for Music Information Retrieval (ISMIR)*, pages 577–582, Philadelphia, PA, USA, September 2008.

[MKRG12] Toni Mäkinen, Serkan Kiranyaz, J. Raitoharju, and Moncef Gabbouj. An evolutionary feature synthesis approach for content-based audio retrieval. *EURASIP Journal on Audio, Speech, and Music Processing*, 2012(1) : 23, 2012.

[ML11] M. Mauch and M. Levy. Structural Change on Multiple Time Scales as a Correlate of Musical Complexity. In *International Society for Music Information Retrieval (ISMIR)*, pages 489–494, Miami, FL, USA, October 2011.

[MN09] R. Mayer and R. Neumayer. Multi-modal Analysis of Music : A large-scale Evaluation. In Nicola Orio, Andreas Rauber, and David Rizo, editors, *WEMIS*, pages 30–35, Alicante, Spain, 2009. University of Alicante.

[MNR08] R. Mayer, R. Neumayer, and A. Rauber. Combination of audio and lyrics features for genre classification in digital audio collections. In *ACM International conference on Multimedia Information Retrieval (MIR)*, page 159, Vancouver, Canada, October 2008.

[MO09] Nicola Montecchio and Nicola Orio. A discrete filter bank approach to audio to score matching for polyphonic music. In *International Society for Music Information Retrieval (ISMIR)*, pages 495–500, Kobe, Japan, October 2009.

[Moo96] T.K. Moon. The expectation-maximization algorithm. *IEEE Signal Processing Magazine*, 13(6) : 47–60, 1996, http ://ieeexplore.ieee.org/articleDetails.jsp ?arnumber=543975.

[MSS06] N. Mesgarani, M. Slaney, and S.A. Shamma. Discrimination of speech from nonspeech based on multiscale spectro-temporal Modulations. *IEEE Transactions on Audio, Speech and Language Processing (TASLP)*, 14(3) : 920–930, May 2006.

[MV10] A. Mesaros and T. Virtanen. Automatic Recognition of Lyrics in Singing. *EURASIP*, 2010, January 2010.

[NR07] R. Neumayer and A. Rauber. Integration of text and audio features for genre classification in music information retrieval. In *European Conference on Information Retrieval (ECIR)*, volume 4425 of *Lecture Notes in Computer Science*, pages 724–727. Springer Berlin Heidelberg, 2007.

[NTTM09] S. Ness, A. Theocharis, G. Tzanetakis, and Luis G. Martins. Improving automatic music tag annotation using stacked generalization of probabilistic SVM outputs. In *ACM International conference on Multimedia (MM)*, pages 705–708, Beijing, China, 2009. ACM.

[Oud10] L. Oudre. *Template-based chord recognition from audio signals*. PhD thesis, Télécom Paristech, 2010.

[Pai90] C. Paice. A word stemmer based on the Lancaster stemming algorithm. In *ACM Special Interest Group on Information Retrieval (SIGIR)*, pages 56–61, Brussels, Belgium, September 1990.

[Pee04] G. Peeters. A large set of audio features for sound description (similarity and classification) in the CUIDADO project. Technical report, Ircam, Paris, France, 2004.

[Pee13] G. Peeters. *Indexation automatique de contenus audio musicaux*. Habilitation à diriger des recherches, Ircam, Université Paris VI, 2013.

[PGS+11] G. Peeters, B. Giordano, P. Susini, N. Misdariis, and S. McAdams. The Timbre Toolbox : extracting audio descriptors from musical signals. *Journal of the Acoustical Society of America (JASA)*, 130(5) : 2902–2916, December 2011.

[PMH00] G. Peeters, S. McAdams, and P. Herrera. Instrument Sound Descrip-

tion in the Context of MPEG-7. In *International Computer Music Conference (ICMC)*, Berlin, Germany, August 2000.

[PR03] G. Peeters and X. Rodet. Hierarchical Gaussian Tree with Inertia Ratio Maximization for the Classification of Large Musical Instruments Databases. In *International Conference on Digital Audio Effects (DAFx)*, London, UK, September 2003.

[PR07] F. Pachet and P. Roy. Exploring Billions of Audio Features. In *International Workshop on Content-Based Multimedia Indexing (CBMI)*, pages 227–235, Bordeaux, France, June 2007.

[PR09] F. Pachet and P. Roy. Analytical Features : A Knowledge-Based Approach to Audio Feature Generation. *EURASIP Journal on Audio, Speech, and Music Processing*, 2009(1) : 1–23, 2009.

[PVG+11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and É. Duchesnay. Scikit-learn : Machine Learning in Python. *The Journal of Machine Learning Research*, 12 : 2825–2830, February 2011.

[Qui93] J. Quinlan. *C4.5 : Programs for machine learning*. Morgan Kaufmann Publishers Inc., 1993.

[Rap99] C. Raphael. Automatic segmentation of acoustic musical signals using hidden Markov models. *IEEE transactions on pattern analysis and machine intelligence (TPAMI)*, 21(4) : 360–370, April 1999.

[RBCG08] A. Rakotomamonjy, F. Bach, S. Canu, and Y. Grandvalet. SimpleMKL. *Journal of Machine Learning Research*, 9 : 2491–2521, November 2008.

[RJ93] L. Rabiner and B. Juang. *Fundamentals of Speech Recognition*. Prentice Hall, Inc., Upper Saddle River, NJ, USA, April 1993.

[SCBG08] M. Sordo, Ò. Celma, M. Blech, and E. Guaus. The Quest for Musical Genres : Do the Experts and the Wisdom of Crowds Agree ? In *International Society for Music Information Retrieval (ISMIR)*, Philadelphia, PA, USA, September 2008.

[SDPS10] F. Smeraldi, M. Defoin-Platel, and M. Saqi. Handling missing features with boosting algorithms for protein-protein interaction prediction.

In *Data Integration in the Life Science*, volume 6254, pages 132–147, August 2010.

[SFHS07] J. Schafer, D. Frankowski, J. Herlocker, and S. Sen. Collaborative Filtering Recommender Systems. In P. Brusilovsky, A. Kobsa, and W. Nejdl, editors, *The Adaptive Web*, volume 4321 of *Lecture Notes in Computer Science*, chapter 9, pages 291–324. Springer, Berlin, Heidelberg, 2007.

[SL01] D. Seung and L. Lee. Algorithms for non-negative matrix factorization. *Advances in neural information processing systems (NIPS)*, 13(1) : 556–562, 2001.

[Sla02] M. Slaney. Semantic-audio retrieval. In *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, volume 4, pages IV–4108 – IV–4111, Orlando, FL, USA, May 2002.

[SLC07] M. Sordo, C. Laurier, and Ò. Celma. Annotating Music Collections : How Content-Based Similarity Helps to Propagate Labels. In *International Society for Music Information Retrieval (ISMIR)*, pages 531–534, Vienna, Austria, September 2007.

[SMD10] B. Sturm, M. Morvidone, and L. Daudet. Musical Instrument Identification using Multiscale Mel-frequency Cepstral Coefficients. In *European Signal Processing Conference (EUSIPCO)*, pages 477–481, Aalborg, Denmark, 2010.

[SS99] R. Schapire and Y. Singer. Improved Boosting Algorithms Using Confidence-rated Predictions. *Machine Learning*, 37(3) : 297–336, 1999.

[SS02] P. Salembier and T. Sikora. *Introduction to MPEG-7 : Multimedia Content Description Interface*. John Wiley & Sons, Inc., New York, NY, USA, 2002.

[SSKS12] M. Sordo, J. Serrà, G. Koduri, and X. Serra. Extracting semantic information from an online carnatic music forum. In *International Society for Music Information Retrieval (ISMIR)*, pages 355–360, Porto, Portugal, October 2012.

[SZM06] N. Scaringella, G. Zoia, and D. Mlynek. Automatic Genre Classification of Music Content. *IEEE Signal Processing Magazine*, 23(2) : 133–141, 2006.

[Taa03] B. Taar Romeny. *Front-end vision and multi-scale image analysis.* Springer, 1st edition, 2003.

[TBL06] D. Turnbull, L. Barrington, and G. Lanckriet. Modeling music and words using a multi-class naïve Bayes approach. In *International Society for Music Information Retrieval (ISMIR)*, pages 254–259, Victoria, Canada, October 2006.

[TBL08] D. Turnbull, L. Barrington, and G. Lanckriet. Five Approaches to Collecting Tags for Music. In *International Society for Music Information Retrieval (ISMIR)*, pages 225–230, Philadelphia, PA, USA, September 2008.

[TBTL08] D. Turnbull, L. Barrington, D. Torres, and G. Lanckriet. Semantic Annotation and Retrieval of Music and Sound Effects. *IEEE Transactions on Audio, Speech and Language Processing (TASLP)*, 16(2) : 467–476, February 2008.

[TC02] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, 10(5) : 293–302, 2002.

[TK07] G. Tsoumakas and I. Katakis. Multi-label classification : An overview. *International Journal of Data Warehousing and Mining (IJDWM)*, 3(3) : 1–13, September 2007.

[TKT10] D. Tingle, Y. Kim, and D. Turnbull. Exploring automatic music annotation with "acoustically-objective" tags. In *ACM International conference on Multimedia Information Retrieval (MIR)*, Philadelphia, PA, USA, March 2010.

[TLBL07] D. Turnbull, R. Liu, L. Barrington, and G. Lanckriet. A game-based approach for collecting semantic annotations of music. In *International Society for Music Information Retrieval (ISMIR)*, Vienna, Austria, September 2007.

[TS06] A. Turpin and F. Scholer. User performance versus precision measures for simple search tasks. In *ACM Special Interest Group on Information Retrieval (SIGIR)*, pages 11–18, Seattle, WA, USA, August 2006.

[VJ01] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *IEEE Conference on Computer Vision and Pat-*

tern Recognition (CVPR), volume 1, pages 511–518, Kauai, HI, USA, December 2001.

[WE04] B. Whitman and D. Ellis. Automatic Record Reviews. In *International Society for Music Information Retrieval (ISMIR)*, Barcelona, Spain, October 2004.

[WKB13] A. Warriner, V. Kuperman, and M. Brysbaert. Norms of valence, arousal, and dominance for 13,915 English lemmas. *Behavior research methods*, February 2013.

[WLMM07] C. Weihs, U. Ligges, F. Mörchen, and D. Müllensiefen. Classification in music research. *Advances in Data Analysis and Classification*, 1(3) : 255–291, November 2007.

[WSDR09] J. Weil, T. Sikora, J. Durrieu, and G. Richard. Automatic generation of lead sheets from polyphonic music signals. In *International Society for Music Information Retrieval (ISMIR)*, pages 603 – 608, Kobe, Japan, October 2009.

[YC10] Y. Yang and H. Chen. Ranking-Based Emotion Recognition for Music Organization and Retrieval. *IEEE Transactions on Audio, Speech and Language Processing (TASLP)*, (99) : 1–1, 2010.

[YH99] Y. Yohannes and J. Hoddinott. Classification and regression trees : An Introduction, 1999.

[YLC06] Y. Yang, C. Liu, and H. Chen. Music emotion classification : a fuzzy approach. In *ACM International conference on Multimedia Information Retrieval (MIR)*, pages 81–84, Santa Barbara, CA, USA, October 2006.

[YLC+08] Y. Yang, Y. Lin, H. Cheng, I. Liao, Y. Ho, and H. Chen. Toward Multi-Modal Music Emotion Classification. *Advances in Multimedia Information Processing (PCM)*, 5353 : 70–79, December 2008.

[YLLC09] Y. Yang, Y. Lin, A. Lee, and H. Chen. Improving Musical Concept Detection by Ordinal Regression and Context Fusion. In *International Society for Music Information Retrieval (ISMIR)*, pages 147–152, Kobe, Japan, October 2009.

[YLSC08] Y. Yang, Y. Lin, Y. Su, and H. Chen. A Regression Approach to Music Emotion Recognition. *IEEE Transactions on Audio, Speech and Language Processing (TASLP)*, 16(2) : 448–457, 2008.

[ZE02] Bianca Zadrozny and Charles Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *ACM International conference on knowledge discovery and data mining (SIGKDD)*, page 694, Edmonton, Canada, July 2002.

Notations

Liste des acronymes

API	Application Programming Interface : Interface de programmation
AROC	Aire sous la courbe ROC
BHS	Beat Histogram Summary : résumé d'histogramme rythmique
CART	Classification And Regresison Trees : Arbres de classification et de régression
CQT	Constant-Q Transform : Transformée à Q constant
GMM	Gaussian Mixture Model : Modèle de mélanges gaussiens
HMM	Hidden Markov Model : Modèle de Markov Caché
LSA	Latent Semantic Analysis : Analyse sémantique latente
MAP	Mean Average Precision
MFCC	Mel-Frequency Cepstral Coefficients : Coefficients cepstraux sur l'échelle de Mel
MIR	Music Information Retrieval : Extraction d'informations musicales
NMF	Non-negative Matrix Factorization : Factorisation en matrices non-négatives
ROC	Receiver Operating Characteristic
SVM	Support Vector Machine : Machine à vecteurs de support
TF-IDF	Term Frequency-Inverse Document Frequency : Fréquence du terme-Fréquence inverse de document
TFCT	Transformée de Fourier à court-terme

Liste des symboles

$\mathbb{1}$	$\mathbb{1}_{Pr} = 1$ si Pr est vraie, 0 sinon.
EM	Espérance-Maximisation
$\mathcal{H}$	Modèle pour l'apprentissage automatique
h	Classifieur automatique
l	Tag (label)
$\mathcal{L}$	Ensemble de tous les tags (labels)
m	Morceau de musique
P	Plage de décision
r	Numéro de l'itération dans un algorithme
t	Seuil de décision
w	Poids d'un exemple d'apprentissage
$\mathbf{x}$	Exemple d'apprentissage

Index

A

Ada-ABS, 47
Adaboost, 39
Aire sous la courbe ROC, 34, 111
Algorithme des k-moyennes, 68
Annotations faibles, 30, 90
Apprentissage à noyaux multiples, 28, 44, 48
Arbre de décision, 40
AROC, 34, 111

B

Bag of frames, 20
Beat Histogram Summary, 18

C

CAL500, 32
Calibration, 29
Chroma, 17
Classifieur faible, 38
Cohen, Kappa de, 55

D

Démarrage à froid, 4

E

Empilement de classifieurs, 24

F

Factorisation en Matrices Non-négatives, 91
Faible (classifieur), 38
Fenêtre de texture, 21, 67
F-mesure, 109

G

GMM, 25, 68, 77

H

Histogramme rythmique, 18
HMM, 68, 69

I

Intégration temporelle, 14

K

Kappa de Cohen, 55
K-means, 68

L

Lancaster, Stemmer de, 92
LS_Boost, 45, 57

M

Machine à vecteurs de support, 25, 28, 38, 63
MAP, 34

Matrice de confusion, 108
Maximisation du Rapport d'Inertie, 82
McNemar, Test de, 35, 83, 114
Mean Average Precision, 34, 110
Mel-frequency Cepstral Coefficients, 15
MFCC, 15
Mirex, 32
Modèle de Markov Caché, 68, 69
Modèle de mélange gaussien, 25, 68, 77
Modèle de Texture Dynamique, 21
Mot vide, 91
Moyenne de scores calibrés, 29

N
Niveau d'abstraction, 7, 59, 60
Non-negative Matrix Factorization, 91

P
Pandora, 4, 32, 34
Plage de décision, 78
Précision, 108
P-valeur, 83, 114

R
Racinisation, 92
Rappel, 108
Receiver Operating Characteristic, 34, 111

S
Sac de mots, 68
Sac de trames, 20
Souche de décision, 42
Stacked generalization, 24
Stemming, 92
Student, Test de, 35, 57, 84, 115
SVM, 25, 28, 38, 63

T
Tag, 2
Tag A Tune, 31
Taux d'erreur, 108
Tempogramme, 18
Tempogramme cyclique, 19
Test de McNemar, 35, 83, 114
Test de Student, 35, 57, 84, 115
Texture, fenêtre de, 21, 67
TF-IDF, 91

V
Valeur p, 83, 114
Validation croisée, 35

FUSION MULTI-NIVEAUX PAR BOOSTING POUR LE TAGGING AUTOMATIQUE

Rémi FOUCARD

RESUME : Les tags constituent un outil très utile pour indexer des documents multimédias.

Cette thèse de doctorat s'intéresse au tagging automatique, c'est à dire l'association automatique par un algorithme d'un ensemble de tags à chaque morceau. Nous utilisons des techniques de boosting pour réaliser un apprentissage prenant mieux en compte la richesse de l'information exprimée par la musique.

Un algorithme de boosting est proposé, afin d'utiliser conjointement des descriptions de morceaux associées à des extraits de différentes durées. Nous utilisons cet algorithme pour fusionner de nouvelles descriptions, appartenant à différents niveaux d'abstraction. Enfin, un nouveau cadre d'apprentissage est proposé pour le tagging automatique, qui prend mieux en compte les subtilités des associations entre les tags et les morceaux.

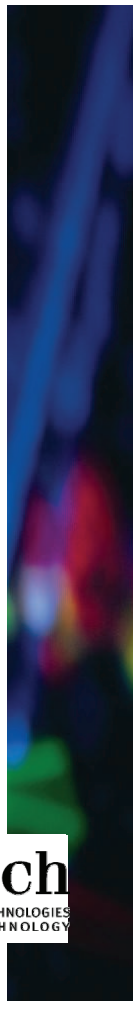
MOTS-CLEFS : Music information retrieval, Tagging automatique, Apprentissage automatique, Boosting, Fusion de classifieurs.

ABSTRACT : Tags constitute a very useful tool for multimedia document indexing.

This PhD thesis deals with automatic tagging, which consists in associating a set of tags to each song automatically, using an algorithm. We use boosting techniques to design a learning which better considers the complexity of the information expressed by music.

A boosting algorithm is proposed, which can jointly use song descriptions associated to excerpts of different durations. This algorithm is used to fuse new descriptions, which belong to different abstraction levels. Finally, a new learning framework is proposed for automatic tagging, which better leverages the subtlety of the information expressed by music.

KEY-WORDS : Music information retrieval, Automatic Tagging, Automatic learning, Boosting, Classifier fusion.



TELECOM ParisTech

EDITE

ParisTech
INSTITUT DES SCIENCES ET TECHNOLOGIES
PARIS INSTITUTE OF TECHNOLOGY