UNIVERSITE PIERRE & MARIE CURIE
LA SCIENCE A PARIS

TELECOM ParisTech

Robustesse des systèmes de classification automatique des signaux audio-fréquences aux effets sonores

Master Acoustique, Traitement du signal et Informatique Appliqués à la Musique

Maxime Lardeur

Directeurs de stage : Slim Essid - Gaël Richard
Coordinateur : Carlos Agon

Université Pierre et Marie Curie - 2007/2008

TELECOM ParisTech
37, rue Dareau
75014 Paris

Remerciements

Je tiens à remercier, dans un premier temps, toute l'équipe pédagogique du master ATIAM, pour avoir assuré cette formation dans des conditions exceptionnelles.

Mes plus vifs remerciements vont à mon directeur de stage Slim Essid pour sa confiance, son soutien et sa disponibilité - je pense, entre autre, aux nombreuses heures de discussions toujours riches en idées.

Je remercie également Gaël Richard, responsable de l'équipe, Cyril Joder, doctorant, Mounira Maazaoui, co-stagiaire ainsi que toute l'équipe audio du département TSI pour leur acceuil, leur bienveillance, et la bonne humeur qui reigne dans l'équipe.

Merci à Nicolas Bedène, ingénieur du son, et aux membres du forum ingenieurduson.com pour leurs précieux conseils sur les effets utilisés en musique live et studio.

Enfin, je ne remercierai jamais assez Manue pour sa patience et son aide, ainsi que ma famille et mes amis pour leur soutien.

Table des matières

Introduction

Contexte de l'étude

Indexation et classification

Depuis maintenant quelques années, grâce notamment à l'évolution des technologies de l'internet, l'utilisateur a accès à d'importantes quantités de données multimédia. La taille de ces bases de données rend leur navigation fastidieuse pour l'utilisateur n'ayant pas d'informations précises sur ce qu'il recherche. C'est en partant de ce constat qu'est venue la nécessité de décrire automatiquement les données multimédia d'une manière plus proche de leur contenu. Ce constat a conduit à l'émergence d'une nouvelle discipline scientifique qu'est l'indexation automatique des contenus multimédia. Nous centrerons notre travail autour de cette problématique appliquée plus spécifiquement aux signaux audiofréquences.

L'indexation des signaux audio consiste à extraire d'un enregistrement une représentation symbolique qui permet de le décrire efficacement. Dans le cas de la musique, des descripteurs de haut niveau tels que le tempo, le genre, la mélodie et même la partition musicale vont trouver des applications dans les bases de données sonores avec *la recherche par le contenu*. La *recherche par similarité* quant à elle vise à trouver des données se rapprochant d'un exemple de référence selon des critères prédéfinis. Ce type de recherche est mise en place, par exemple, dans les sites qui génèrent des listes de lecture de morceaux de musique à partir d'un artiste ou d'un morceau de référence.

La mise en oeuvre de l'indexation va consister à organiser, catégoriser, classer les données en fonction des descripteurs choisis (tempo, genre...) ce qui nous permet de l'assimiler à des *problèmes de classification automatique*.

La plupart des systèmes de classification audio sont très "fragiles" lors du passage des conditions de laboratoires aux conditions réelles car ils opèrent sous les différentes paramètres pour lesquels l'apprentissage à été réalisé. Notre étude portera sur cet aspect de robustesse des systèmes de classification audio.

Cette recherche de robustesse a été appliquée au cas d'étude de la reconnaissance automatique des instruments de musiques tout en gardant l'idée d'une généralisation à d'autres problèmes de classification audio.

Classification automatique

La classification automatique vise à étiqueter des objets suivant la *classe* auxquels ils appartiennent. Les objets étudiés ici sont des extraits d'enregistrements de musique joués par un instrument seul, et les classes sont les différents instruments de musique présent dans ces enregistrements.

Les systèmes de classification comportent deux étapes (cf. figure 1) :

– l'*apprentissage*, qui va s'efforcer de trouver la description de l'espace des observations qui traduit le mieux l'association avec les classes correspondantes.
– le *test*, qui permet d'évaluer les performances du système de classification.

La phase d'apprentissage peut se décomposer en trois étapes :

– l'extraction de descripteurs à partir des données audio de la base d'apprentissage. Chaque descripteur caractérise une certaine propriété du signal susceptible de participer à la description des classes. L'ensemble des descripteurs est donné dans la partie 2.2.1.

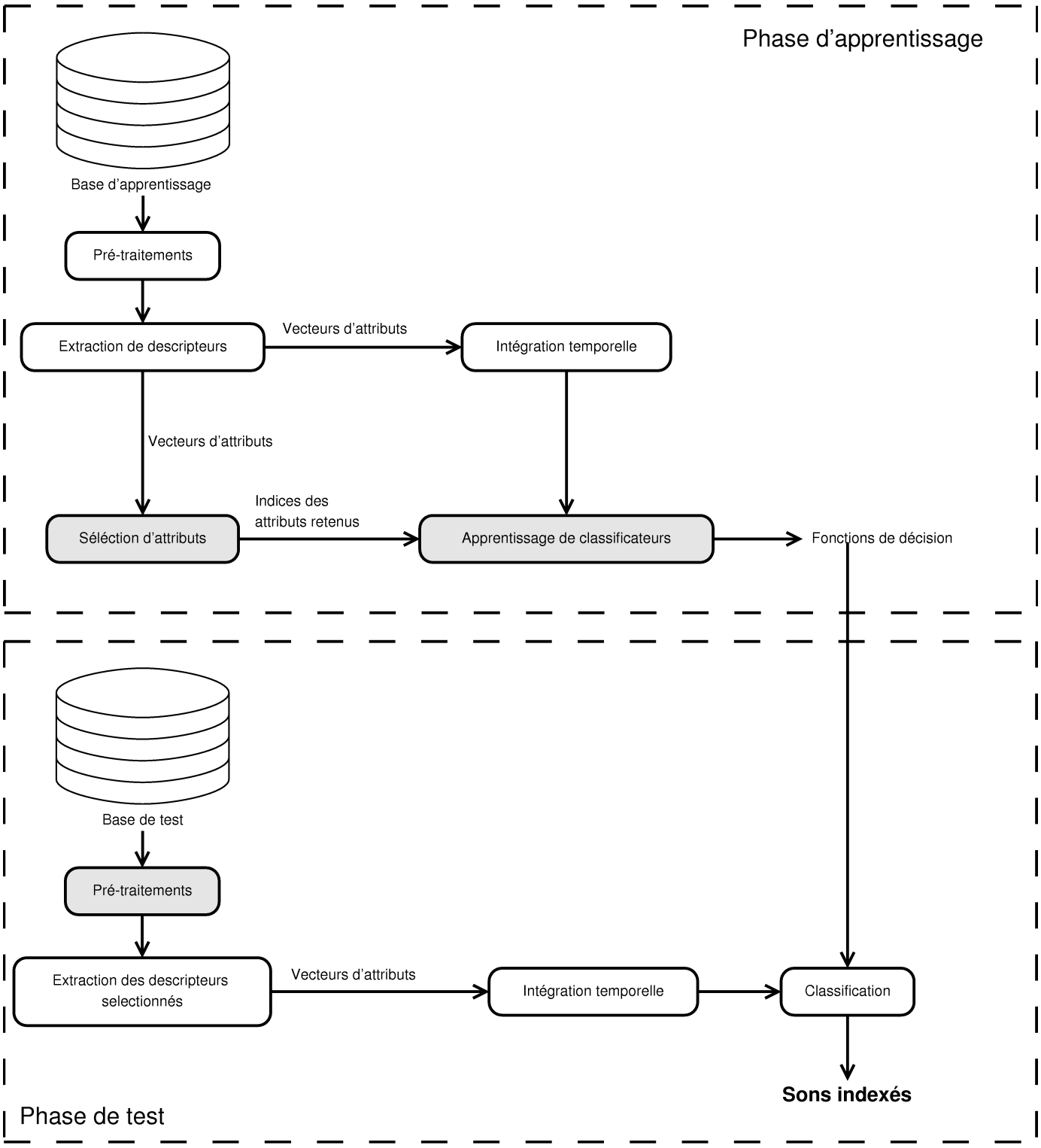

FIG. 1: Schéma d'un système de classification audio. En fond gris les étapes du processus sur lesquelles l'étude s'est portée.

– la sélection d'attributs va mesurer, selon des critères que nous définirons au 2.2.3, la pertinence des descripteurs extraits des données afin de n'en retenir qu'un sous-ensemble efficace. Les attributs, sous ensemble de descripteurs ou parfois de coefficients de descripteurs, permettent alors, pour des coûts de calcul moindres, d'obtenir des performances similaires, voir meilleures que si la totalité des descripteurs était utilisée.
– l'apprentissage[1] consiste alors à déterminer *des fonctions de décision* qui se basent sur les vecteurs d'attributs précédemment calculés de la base d'apprentissage. Ce sont les fonctions de décision qui permettront d'associer une classe à un nouveau vecteur d'attributs.

La phase de test revient alors à extraire les descripteurs qui ont été retenu lors de l'étape de sélection des attributs, et à leur associer une classe suivant les valeurs obtenues par les fonctions de décisions apprises.

Problématique : Robustesse des systèmes de classification audio

La recherche de robustesse consiste à rendre un système aussi performant lorsqu'il est testé sur des données réelles ou perturbées que sur les données de laboratoire. L'objectif principal est d'améliorer les performances du système dans sa globalité pour des données hétérogènes. Nous définirons la robustesse d'un système comme la stabilité de ses performances face à de petites perturbations dans les données. Dans les systèmes de classification audio, ces perturbations font référence à tous les traitements, volontaires ou non, que va subir le signal.

Rapporté au processus de réalisation d'une musique, le passage d'une onde acoustique à un enregistrement numérique fait intervenir un grand nombre de traitements dont les paramètres sont plus ou moins connus. Ainsi, si certains de ces paramètres sont inhérents à la production du son : la technique et le style de jeu des musiciens, la qualité de manufacture des instruments..., d'autres sont propres aux conditions d'enregistrements : prise de son, pré et post traitements, mixage, conditions acoustiques de la salle...

Notre étude portera sur les traitements se rapportant aux conditions d'enregistrement, considérés comme les perturbations face auxquelles notre système devra être robuste.

Dans l'ensemble des traitements, nous ferons la différence entre le terme d'*effet* qui est une modification en 'surface' du signal (par l'utilisation de filtres) et le terme de *transformation* qui est une modification plus profonde (par exemple le codage à bas débit).

État de l'art

Il existe très peu de travaux sur la robustesse des systèmes de classification des signaux musicaux. La parole, quant à elle, a fait l'objet de nombreuses études avec le développement de méthodes de normalisation des données hétérogènes. On trouve ainsi pour la reconnaissance du locuteur des méthodes de compensation de la variabilité spectrale [42], mais également, du côté de la transcription automatique de la parole, qui cherche plutôt à s'abstraire du locuteur, des méthodes de normalisation du conduit vocal, et des conditions acoustiques [14, 44, 12, 1].

La robustesse a été également étudiée dans les systèmes de classification et de segmentation des signaux entre bruit, parole et musique [32].

La recherche de descripteurs pertinents en vue de la reconnaissance des instruments de musique a fait l'objet de nombreux travaux [39], cependant très peu traitent de leur robustesse aux traitements.

Plusieurs méthodes d'optimisation des classificateurs de type machine à vecteur support (SVM) ont été développées. Certaines d'entre elles s'intéressent aux exemples d'apprentissage aberrants que l'on nomme "outliers" [36, 51], la fusion des décisions sur les SVM multi-classes [6], enfin d'autres proposent d'incorporer les invariants engendrés par les transformations. Ainsi, les trois principales approches d'optimisation des SVM par la prise en compte des invariants sont : les SVM virtuelles [50, 43], modification ou construction d'un noyau plus adapté [18, 43] ou encore la normalisation des vecteurs de l'espace des attributs [15, 43].

[1] du moins dans le cas supervisé où les classes possibles sont supposées connus à l'avance

Reconnaissance automatique des instruments de musique

La reconnaissance des instruments de musique équivaut à une tâche de reconnaissance du timbre. Pour éviter d'éventuelles mécompréhension sur la définitions du timbre, nous préciserons simplement que nous entendons le timbre au sens causal du terme. Il ne s'agit donc pas de différencier deux instruments de même nature (ex : deux clarinettes entre elles), tâche déjà difficile pour l'expert et quasiment impossible pour le non expert, mais de reconnaître la nature de l'instrument (ex : 'c'est une clarinette qui joue').

La reconnaissance automatique des instruments de musique est un sujet étudié depuis une quinzaine d'années et les méthodes générales n'ont que très peu évoluées. Les recherches se sont plutôt portées sur l'amélioration des différentes étapes du processus comme le choix de descripteurs et de classificateurs, c'est-à-dire comment mieux "apprendre".

Différentes stratégies de classification ont été expérimentées. Si dans les premiers temps les réseaux de neurones [20, 26, 23] ou l'algorithme des k plus proches voisins [21, 7, 34] ont été fréquemment utilisés, des méthodes probabilistes telles que des modèles gaussiens [7], de mélanges de gausiennes [3, 23] ou de chaînes de Markov cachées [28, 8] ont été explorées. Cependant les récents travaux montrent une grande adoption des machines à vecteurs supports [9, 33, 47] dans les problèmes de classification, notamment grâce à leur grand pouvoir de généralisation.

Contributions

L'objectif de notre travail est de contribuer à améliorer la robustesse des systèmes de classification audio, appliqués à la reconnaissance automatique des instruments de musiques. Nous espérons tirer profit de l'effort considérable qui a été consacré dans les travaux précédents sur la reconnaissance des instruments de musique d'une part et sur la robustesse d'autre systèmes de classifications audio, notamment dans le domaine de la parole. Après une investigation des traitements les plus représentatifs du processus de réalisation, nous avons été amenés à étudier leur impact sur chacune des étapes du système de classification. Une phase d'analyse des données a permis le développement d'outils graphiques et d'écoute qui faciliteront les travaux futurs de la classifications. Une stratégie globale de robustesse a été élaborée et mise en place sur des données réelles puis validée sur une base indépendante. Elle comprend :

- Une méthode de construction d'ensembles d'attributs robustes
- Une optimisation des SVM par incorporation des invariants suivant l'approche des SVM virtuelles

Organisation du document

Dans un premier temps, nous définirons les traitements retenus comme pertinents dans l'étude de la robustesse puis nous décrirons plus en détails le système de classification de référence tout en introduisant les bases théoriques nécessaires. Nous exposerons par la suite les dégradations sur ce système apportées par les traitements précédemment choisis. Puis, nous présenterons l'approche suivie dans l'amélioration du système de référence en s'appuyant sur l'analyse des résultats expérimentaux à travers des outils réalisés dans ce but. Enfin, nous nous intéresserons à la mise en oeuvre d'une stratégie de robustesse aux effets, suivis des résultats obtenus ainsi qu'à la généralisation de notre travail.

Chapitre 1

Effets et transformations étudiés

D'une manière générale, les paramètres d'enregistrement sont difficilement quantifiables directement depuis le signal. L'application d'effets trouve alors son intérêt dans la simulation de ses conditions d'enregistrement dans l'optique de construire une base d'apprentissage aux conditions d'enregistrement hétérogènes. Dans ce chapitre, nous discuterons du choix des traitements étudiés puis nous les présenterons un à un. Enfin un tableau récapitulera les différentes paramétrisations de ces traitements.

1.1 Choix des effets

Nous avons d'abord établi un inventaire des effets existants, non pas avec l'objectif d'être exhaustif mais afin d'obtenir un ensemble d'effets représentatifs. Puis, avec le concours d'un ingénieur du son, il a été convenu d'une liste d'effets incontournablement utilisés en live ou en studio. Enfin, le choix de la paramétrisation des effets a été fait de telle sorte qu'une fois appliquée, la modification soit perceptible tout en restant "réaliste", c'est-à-dire sans changer radicalement le timbre de l'instrument.

A la liste des traitement étudiés, nous avons ajouté les transformations suivantes, de par leur utilisation très répandue :

– codage type mp3
– le filtrage passe-bas dont le but est de simuler un sous échantillonnage.

Ces traitements ne seront pas considérés comme des effets mais comme des transformations. Celles-ci vont conduire à une dégradation du matériel sonore due à une perte d'informations plus ou moins importante.

Afin de garder un maximum de contrôle sur les effets appliqués, il a été décidé d'implémenter la plupart d'entre eux. Leur mise en oeuvre et les choix d'implémentation sont résumés dans le tableau 1.1. Les principes fonctionnels et les détails d'implémentation sont disponibles en annexe B.

effets/transformations	mise en oeuvre	référence
réverbération	logiciel sox-14.0	[10]
compression	implémentation Matlab	[53]
égalisation	implémentation Matlab	[52]
codage type mp3	LAME 32bits version 3.97	[25]
sous-échantillonnage	implémentation Matlab d'un filtre passe bas	

TAB. 1.1: Choix d'implémentation des effets et transformations

1.2 La réverbération

Généralités

La réverbération (reverb), encore appelée "effet de salle", est probablement un des effets les plus utilisés en musique [35]. L'effet de salle reproduit artificiellement l'acoustique d'une salle qui résulte principalement des

nombreuses réflexions des ondes sonores sur les parois.

L'effet de salle est principalement utilisé dans le post-traitement des enregistrements pour lesquels le microphone est situé proche d'un instrument ou du chanteur. L'acoustique d'une salle, comme par exemple une salle de concert ou une église, est appliquée au signal direct, encore appelé le son "sec". En termes de traitement du signal, l'effet de salle revient à effectuer la convolution du signal avec la réponse impulsionnelle de la salle.

La réponse impulsionnelle entre deux points d'une salle peut être schématisée comme le montre la figure 1.1 . Elle est composée du son direct, des réflexions précoces et des réflexions tardives. Les réflexions précoces auront une grande influence sur l'intelligibilité de la voix et la clarté des instruments. Le nombre des réflexions précoces augmente continuellement avec le temps et conduit à un signal aléatoire selon une décroissance exponentielle que l'on appelle "réverbération tardive". Le temps de réverbération est alors défini comme le temps que met la pression acoustique pour décroître de 60dB. Ces caratéristiques dépendent de la géométrie de la salle ainsi que de la nature des matériaux des surfaces. Enfin, au cours des différentes réflexions, le sons va subir un filtrage fréquentiel lui donnant une "couleur".

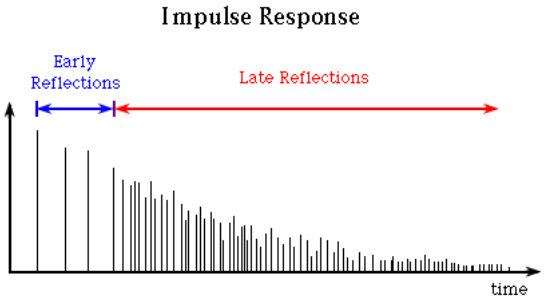

FIG. 1.1: Réponse impulsionnelle schématique d'une salle.

Paramétrisation

Après plusieurs écoutes de différents paramétrisations, nous avons choisi de garder celle par défaut, qui nous semblait donner le rendu le plus réaliste.

Utilisation : `reverb [-w|-wet-only] [reverberance (50%) [HF-damping (50%) [room-scale (100%) [stereo-depth (100%) [pre-delay (0ms) [wet-gain (0dB)]]]]]]`

Commande utilisée : sox *input output* reverb vol -3dB 50 50 100

1.3 La compression

Généralités

Les compresseurs sont utilisés pour réduire la dynamique du signal à l'enregistrement. Les silences ou les faibles niveaux du signal ne sont pas modifiés tandis que les hauts niveaux sont réduits en fonction d'une courbe statique. Il en résulte une différence amoindrie entre les niveaux forts et faibles permettant d'élever le niveau global du signal et de le rendre plus fort.

Une des utilisations les plus fréquentes de la compression est l'augmentation du sustain d'un instrument. Par exemple, après qu'une corde de guitare soit pincée, l'excitation va graduellement s'amortir. L'ajout d'une compression va prévenir d'un changement du niveau de l'instrument après avoir été excité, ce qui est perçu comme une augmentation du sustain ou un lissage de l'instrument.

Il pourait être intéressant d'augmenter le sustain autant que possible. Cependant une compression trop forte peu dénaturer le son d'un instrument. En effet, au cours du processus, on élimine la dynamique de jeu de telle sorte qu'il devient difficile de marquer les accents et les phrases. De plus, il est admis que l'attaque est un facteur très important à la reconnaissance d'un instrument. Un réglage extrême de la compression (hard limiter) va donc produire des sons qui n'appartiennent pas au répertoire de l'instrument.

Paramétrisation

Les capacités de perception humaine d'une variation de l'intensité sont beaucoup plus faible que lors d'une variation de hauteur. C'est pourquoi les paramètres de configuration (voir tableau 1.2) du compresseur utilisé peuvent paraître élevés.

Paramètre du compresseur	Valeur
threshold	0,5
compressor ratio	10
attack time	0,001
release time	1

TAB. 1.2: Paramètres de la configuration du compresseur.

1.4 L'égalisation

Généralités

Quelque soit son type (semi-paramétrique, paramétrique, graphique...), quelque soit sa forme (intégré à la console, en rack), quelque soit la technologie employée (à lampes, à transistors, numérique...), l'égaliseur remplit invariablement la même fonction : celle d'atténuer ou d'amplifier certaines des fréquences d'un signal, ce qui revient au final à modifier son timbre. Les applications qui découlent de cette fonction sont diverses et variées : on trouve d'un côté celles qui visent à solutionner des problèmes, et de l'autre celles qui poursuivent un but artistique, créatif.

Voici quelques illustrations de ces deux types d'applications :

- Un sonorisateur se servira d'un égaliseur pour compenser les défauts d'une salle en appliquant une courbe inverse à sa réponse fréquentielle. En studio, l'égalisation sera utilisée pour "modeler" un timbre, en changer la couleur dans un but artistique : renforcer le coup d'archet d'un violon, conférer plus de corps à une caisse claire, plus d'attaque à des toms, de brillance à une voix...
- Une autre utilisation consiste à faire en sorte que les instruments, au mixage, se mélangent harmonieusement et ne se perturbent pas les uns les autres, n'empiètent pas sur leurs territoires spectraux respectifs. Dans un registre encore différent, à la prise de son, on aura parfois besoin de compenser les défauts d'un micro ou de sa position (l'effet de proximité, par exemple, qui induit quasi-systématiquement un excès de basses), d'atténuer du souffle, etc.

Au mastering, on égalisera l'ensemble d'un mixage pour l'équilibrer - pallier l'absence de telles ou telles fréquences ou au contraire, en enlever d'autres pour gommer certains excès - mais aussi pour le rendre conforme à des critères d'écoute ou de diffusion...

Que ce soit sur scène ou en studio, l'égaliseur est un élément incontournable.

Paramétrisation

Dans un souci de coller à la réalité, le choix du paramétrisation de l'égaliseur a été calqué sur les 'presets' (pré-configurations) les plus couramment utilisés dans les lecteurs multimédia tels que *Winamp*® ou *VLC*®. Par la suite, afin de faciliter l'analyse, nous avons été amenés à ne garder qu'une seule configuration dont chacune des valeurs de gain par bande de fréquences est multipliées par un coefficient (voir tableau 1.3). Les courbes de gains en fonction de la fréquence sont également données figure 1.2.

Gain (dB) - fc (Hz)	coef	31,5	63	125	250	500	1000	2000	4000	8000	16000
EQ2std	2	8	8	4,8	-5,4	-8	-3,4	4	8,8	11,2	11,2
EQ1std	1	4	4	2,4	-2,7	-4	-1,7	2	4,4	5,6	5,6
EQ-1std	-1	-4	-4	-2,4	2,7	4	1,7	-2	-4,4	-5,6	-5,6
EQ-2std	-2	-8	-8	-4,8	5,4	8	3,4	-4	-8,8	-11,2	-11,2

TAB. 1.3: Coefficients du gain par fréquences centrales de l'égaliseur graphique

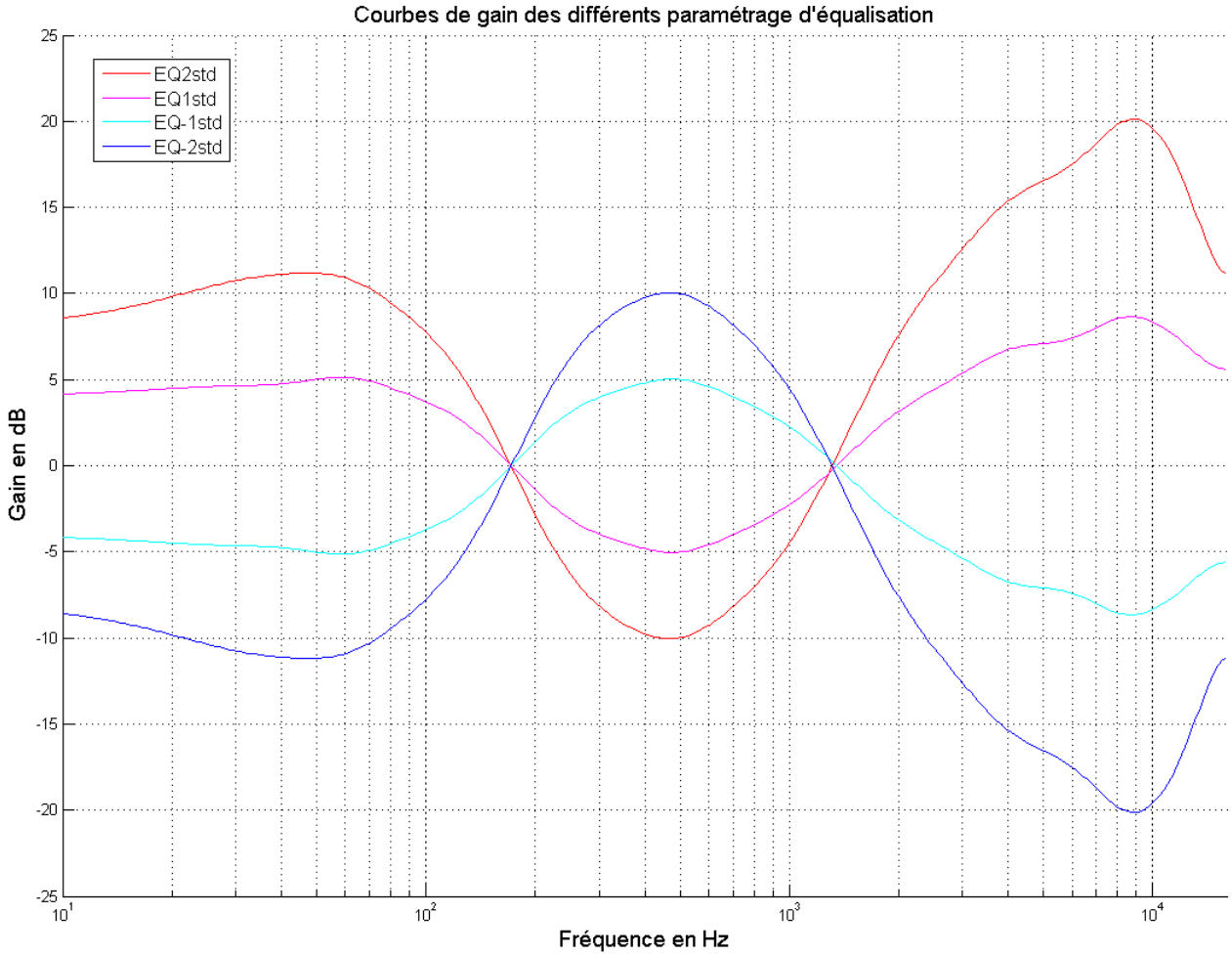

FIG. 1.2: Courbes de gain/fréquence des différents égaliseurs. Deux configurations augmentent les basses et hautes fréquences et les deux autres sont leur inverse.

1.5 Le codage avec modèle psycho-acoustique (type mp3)

Généralités

Le format mpeg I layer3 est devenu le format de compression audio le plus populaire et le plus utilisé. Il permet une réduction approximative de 1 :4 à 1 :12. La réduction de taille facilite le transfert sur les réseaux et le stockage de données musicales sur un support numérique, tel qu'un disque dur ou une mémoire flash.

Le grand principe de la compression audio : "Ne jamais transmettre ce que l'on ne peut pas entendre." Le mp3 est fondé sur le principe du "Codage perceptuel". Cela consiste à réduire au maximum la quantité d'informations nécessaires à la perception intégrale du son par l'oreille humaine. D'un point de vue strictement technique, il s'agit d'un procédé destructif ; mais cette perte est quasiment imperceptible car elle est fondée sur les limites connues du système auditif humain. Le décodeur est bien moins complexe car son seul travail est de reconstruire le signal audio à partir des composantes codées [4]. C'est pourquoi on s'intéressera uniquement au processus de codage.

La norme MPEG [45] fournit un schéma fonctionnel dont nous expliquerons les principales étapes [38, 37]. Quant à la différence de qualité entre codeur, elle résidera en majeure partie dans la "finesse" de son modèle psycho-acoustique. Nous avons choisi d'utiliser le logiciel libre LAME 32bits version 3.97 pour l'encodage et le décodage qui se base sur le modèle psycho-acoutique GPSYCHO.

Paramétrisations

Le codage et à été réalisé avec deux configurations de taux de compression((en) birate) différents :
– 32 $kbit.s^{-1}$
– 64 $kbit.s^{-1}$
Commandes passées au logiciel :
lame -m m -b *bitrate* –resample 32 *fichierSource fichierDestination*
lame –decode *fichierSource fichierDestination*

1.6 Le sous-échantillonnage

Généralités

Le sous-échantillonnage n'est pas un effet en soi mais une transformation qui vise à réduire le nombre d'échantillons servant à coder le signal audio. Au cours de ce traitement, il est nécessaire de filtrer passe-bas le signal audio à la fréquence de Nyquist ($fc = \frac{fe}{2}$) afin d'éviter les phénomènes de repliement spectral.

Paramétrisations

La valeur de la fréquence de coupure du filtre passe-bas a été fixée à $fc = 8kHz$. Ceci nous permet de simuler un sous-échantillonnage (fréquence d'échantillonnage $fe = 16kHz$) d'un facteur deux par rapport à nos signaux de référence $fe = 32kHz$.

1.7 Récapitulatif des effets et de leur paramétrisation

Label	Description	Parameters	Value	Unit
ref	sons de référence	-	-	-
comp	compression	threshold	0,5	-
		compressor ratio	10	-
		attack time	0,001	s
		release time	1	s
reverb	réverberation	reverberance	50	%
		HF-damping	50	%
		room-scale	100	%
		pre-delay	0	ms
		wet-gain	0	db
low8k	low-pass filter	fréquence de coupure	8000	Hz
32kbps.mp3	MP3 codage/décodage	bitrate	32	kb/s
64kbps.mp3	MP3 codage/décodage	bitrate	64	kb/s
EQ2std	égaliseur graphique	Coefficient	2	-
EQ1std	égaliseur graphique	Coefficient	1	-
EQ-1std	égaliseur graphique	Coefficient	-1	-
EQ-2std	égaliseur graphique	Coefficient	-2	-

TAB. 1.4: Récapitulatif des traitements condidérés avec leur paramètrisation

Chapitre 2

Bases algorithmiques et théoriques de l'étude

Ce chapitre traite des différentes étapes du système de classification étudié. Ainsi, après avoir exposé les pré-traitements et l'extraction des descripteurs, nous présenterons les machines à vecteurs supports (SVM) qui sont les classificateurs spécifiques du système, nous définirons également ce que sont les vecteurs supports avant de conclure sur les méthodes de recherche et d'incorporation des invariants aux SVM.

2.1 Pré-traitement et segmentation des signaux

2.1.1 Fréquence d'échantillonnage et normalisation

Les signaux traités ont été sous-échantillonnés à 32 kHz pour permettre de réduire la complexité des calculs (par rapport à la fréquence d'échantillonnage initiale de 44.1 kHz) tout en conservant une bonne qualité audio. On préserve le contenu spectral jusqu'à 16 kHz, ce qui est suffisant pour la reconnaissance des instruments [34].

Pour limiter les effets des différences entre les sources (concernant notamment les niveaux sonores), les signaux sont normalisés, de façon à ce que la moyenne soit nulle et que le maximum de la valeur absolue soit unitaire. La version normalisée $\hat{s}$ du signal s'obtient en faisant :

1. $\tilde{s}(n) = s(n) - \frac{1}{L}\sum_{k=0}^{L-1} s(k)$ où L est la longueur du signal.
2. $\hat{s}(n) = \frac{\tilde{s}(n)}{\max_n |\tilde{s}(n)|}$

2.1.2 Fenêtres d'analyse

Les descripteurs traduisent des propriétés du signal, comme le contenu fréquentiel qui nécessitent un nombre minimum d'échantillons pour être calculés de manière fiable. Néanmoins, ces propriétés peuvent changer rapidement. Il est donc nécessaire de calculer ces attributs sur des *fenêtres d'analyse* (*trames*) qui soient assez longues pour rendre compte de ces caractéristiques, mais assez courtes pour témoigner de leurs variations.

Les fenêtres d'analyse choisies sont de $N = 1024$ échantillons, soit 32 ms. Cette longueur de fenêtre réalise un bon compromis entre résolution temporelle et fréquentielle. Les descripteurs spectraux sont alors calculés grâce à une *transformée de Fourier à court terme* correspondant à ces fenêtres d'analyse. La fenêtre de pondération utilisée est une fenêtre de Hamming.

Pour certains descripteurs représentant des phénomènes de durée supérieur à la durée de stationnarité (comme par exemple le *vibrato*), le calcul est effectué sur des fenêtres de taille $N_l = 30N$, soit 960 ms.

Pour un traitement simple des séquences d'observation des descripteurs, on produit un *vecteur d'observations* contenant les valeurs de tous les descripteurs pour chaque fenêtre d'analyse courte (32 ms). Les valeurs des descripteurs calculées sur des longues fenêtres sont alors reproduites dans chaque vecteur d'observation « inclus » dans cette fenêtre longue.

2.1.3 Détection des segments de silence

Dans les enregistrements, on a considéré seulement les parties non silencieuses. Pour cela, les silences ont été détectés grâce à une approche simple. Sont considérés comme fenêtres de silence :

- les fenêtres présentant une amplitude maximale (en valeur absolue) mille fois plus petite (soit 30 décibels de moins) que l'amplitude maximum globale ;
- les fenêtres présentant une valeur d'amplitude constante ;
- les successions de moins de 15 fenêtres non silencieuses entre deux segments de silence.

Ces critères peuvent paraître rudimentaires mais sont suffisants pour nos signaux car ceux-ci ont été enregistrés dans des conditions idéales (en studio, avec un bruit de fond minimal).

2.1.4 Intégration temporelle

Définition

La plupart des systèmes de classification des signaux musicaux exploitent les observations des descripteurs acoustiques calculés sur des fenêtres temporelles de durée fixe (32ms avec un pas d'avancement de 16ms) en supposant l'indépendance de ces observations. Une décision est prise sur chaque fenêtre d'analyse. Cela ne rend pas compte de la dynamique temporelle des signaux audio. Or, d'après l'étude de Cyril Joder, la prise en compte de l'évolution de ces descripteurs permet des meilleures performances de reconnaissance.

Notre système comprend une phase d'intégration temporelle précoce qui consiste à extraire d'une séquence de descripteurs « instantanés » (calculés sur des petites trames temporelles) un vecteur d'attributs représentant une plus grande échelle de temps. Cela permet alors dans le même temps de prendre en compte l'évolution des descripteurs, et de diminuer l'influence des *outliers* (valeurs aberrantes). De manière pratique, cela permet aussi de combiner des descripteurs calculés sur des horizons temporels différents. De plus, ces méthodes diminuent le nombre d'observations à classifier et donc engendrent une diminution en complexité.

Paramètres utilisés

A la lumière des résultats de [**?**], nous avons réalisé une intégration temporelle précoce en moyennant les descripteurs sur 20 trames avec un pas d'avancement de 10 trames. Les nouvelles trames d'analyse ont donc une longueur de $16 * (20 + 1) = 336ms$.

Moyenne

Une séquence de valeurs de chaque descripteur est « remplacée » par la moyenne empirique de ces valeurs. Pour un attribut x_i, on calcule la moyenne $\mu_{i,f}$ sur la fenêtre de texture f de la façon suivante :

$$\mu_{i,f} = \frac{1}{l} \sum_{k=k_1^f}^{k_1^f+l} x_i[k]$$

On a donc :

$$\mathbf{X}_f = \mu_f$$

où

$$\mu = \left[\mu_1, \ldots, \mu_d\right]$$

2.2 Extraction des descripteurs

2.2.1 Généralités

La classification est basée sur une paramétrisation adéquate du contenu à traiter. La forme d'onde ne permettant pas de distinguer les instruments, il est nécessaire d'extraire du signal des données pertinentes pour la

discrimination entre les classes considérées. Ces données sont appelées descripteurs. Comme un grand nombre de descripteur de haut niveau, "l'instrument qui joue actuellement" va être une combinaison, à priori inconnue, d'un ensemble de descripteurs de plus bas niveau. Ce sont dans notre cas des valeurs scalaires numériques. On parle alors d'attributs (features), décrivant certaines caractéristiques du signal qui ont pour la plupart une interprétation physique. Il n'existe pas à ce jour un ensemble de descripteurs bien défini qui permette d'obtenir systématiquement des performances satisfaisantes dans des applications d'indexation audio. C'est pourquoi nous faisons appel à un grand nombre de descripteurs, qui représentent diverses propriétés du signal. Elles peuvent être de type spectral, comme l'énergie de certaines bandes de fréquences, temporel comme le taux de passage par zéro, ou encore cesptral avec les MFCCs et enfin psycho-acoustique. Nous présenterons l'ensemble des descripteurs utilisés dont les formulations sont disponibles en annexe.

2.2.2 Descripteurs pour la classification audio

Les descripteurs considérés dans ce travail sont un sous-ensemble de ceux utilisés dans la thèse de Slim Essid [9], dont la plupart sont issus de l'article [39]. Pour des descriptions plus détaillées, nous invitons le lecteur à consulter ces références. Sauf mention du contraire, les attributs sont calculés sur des fenêtres courtes (32 ms).

Descripteurs cepstraux

- Les 11 premiers coefficients MFCC calculés à partir de 30 sous-bandes MEL (à l'exception de c0).
- Les coefficients MFCC calculés à partir de 11 sous-bandes MEL (à l'exception de c0).

Descripteurs spectraux

- Les quatre premiers moments statistiques spectraux (centroïde spectral, largeur, asymétrie et platitude spectrales), ainsi que leurs 2 premières dérivées temporelles. Ces dérivées sont calculées en utilisant une approximation polynomiale d'ordre 2 de l'évolution des descripteurs.
- La platitude spectrale est le rapport entre moyenne géométrique et moyenne arithmétique du spectre dans un ensemble de sous-bandes. Ces valeurs sont calculées selon le standard MPEG-7, sur 23 sous-bandes.
- L'irrégularité spectrale est la dérivée spectrale d'une transformée à Q constant du signal avec une résolution d'un tiers d'octave.
- Un ensemble d'autres descripteurs de la forme spectrale : la fréquence de coupure à 99% ainsi que les pente, décroissance, flux et platitude du spectre.
- Les coefficients du filtre issus d'une analyse Auto-Régressive à l'ordre 2 du signal (à l'exception de la constante 1)

Descripteurs temporels

- Le taux de passage par zéro calculé sur les deux longueurs de fenêtres.
- Les attributs de modulation d'amplitude sur les bandes 4-8 Hz et 10-40 Hz, extraits sur les fenêtres longues.
- Les coefficients d'Autocorrelation soit les 49 premiers coefficients de la transformée inverse du périodogramme du signal.
- Les quatre premiers moments statistiques temporels extraits sur des fenêtres longues et leur deux premières dérivées .
- Les quatre premiers moments statistiques de l'enveloppe d'amplitude extraits sur des fenêtres longues et les deux premières dérivées. L'enveloppe est calculée comme le module du signal analytique.

Descripteurs perceptuels

- La loudness spécifique relative (définie par bande critique).
- La sharpness avec les deux premières dérivées temporelles.

Descripteurs Autres

- Les descripteurs d'intensité de signaux en sous-bandes d'octaves et du rapport d'énergie entre une sous-bande et la précédente.
- Un ensemble de 28 coefficients issus d'une transformée en ondelettes de Daubechies (voir [29])

Les descripteurs sont regroupés par paquets. Ces paquets et leurs abréviations sont récapitulés dans le tableau 2.1. Au total, 401 descripteurs ont été considérés.

Description des paquets d'attributs	Taille	Synopis
$C3 = [C31, ..., C311] + \delta + \delta^2$	33	MFCC à partir de 11 sous-bandes MEL
$CP3 = [CP31, ..., CP311] + \delta + \delta^2$	13	MFCC à partir de 30 sous-bandes MEL
$CQ1, (\delta, \delta^2)CQ1$	27	Coefficients cesptraux à partir d'une CQT avec résolution d'une octave
$CQ2(\delta, \delta^2)CQ2$	30	Coefficients cesptraux à partir d'une CQT avec résolution d'une demi-octave
$CQ3(\delta, \delta^2)CQ3$	30	Coefficients cesptraux à partir d'une CQT avec résolution d'un tiers d'octave
$CQ4(\delta, \delta^2)CQ4$	30	Coefficients cesptraux à partir d'une CQT avec résolution d'un quart d'octave
$Sx = [Sc, Sw, Sa, Sf] + \delta + \delta^2$	12	Moments spectraux et dérivées temporelles
$Si = [Si1, ..., Si21]$	21	Irrégularité spectrale
$FcSsARSdSvSo$	7	Fréquence de coupure, pente spectrale, coefficients AR, décroissance, flux, platitude du spectre
$SCF = [SCF1, ..., SCF23]$	23	Spectral Crest Factor
$ASF = [ASF1, ..., ASF23]$	23	Platitude spectrale (MPEG-7)
$AM[AM1, ..., AM8]$	8	Paramétres de modulation d'amplitude
$AC = [AC1, ..., AC49]$	49	Coefficients d'Autocorrelation
$ZlZ = [ZCR, lZCR]$	2	Taux de passage par 0 sur des fenêtres courtes et longues
H	15	Intensité de signaux en sous-bandes d'octaves
$Tx = [Tc, Tw, Ta, Tf] + \delta + \delta^2$	12	Moments temporels et dérivées à partir de fenetres courtes
$Tl = [Tc, Tw, Ta, Tf] + \delta + \delta^2$	12	Moments temporels et dérivées à partir de fenetres longues
$Lx = [Ld1, ..., Ld14, Sh, Sp] + \delta + \delta^2$	26	Loudness et dérivées temporelles, sharpness et largeur perceptuelle et dérivées temporelles
$DWCH = [dwchM, dwchSP, dwchSK, dwchK]$	28	Coefficients de transformée en ondelettes

TAB. 2.1: Descripteurs utilisés dans cette étude, soit au total 401 attributs.

2.2.3 Sélection automatique des attributs

Introduction

Dans la plupart des problèmes de classification, un nombre important d'attributs potentiellement utiles peut être exploré. Ce nombre , dans plusieurs cas d'application, atteint les quelques centaines, voire quelques milliers (en particulier, dans le domaine de la bio-informatique). En réalité, un nombre de descripteurs aussi grand impose une charge de calcul et de stockage qui peut se révéler prohibitive. De plus, avec la plupart des classificateurs, une dimension trop élevée conduit à de moins bonnes performances de généralisation car il devient plus difficile de modéliser l'espace des attributs. Enfin, certains descripteurs peuvent être bruités (par manque de robustesse de leur extraction), ou tout simplement non pertinents pour la tâche considérée.

L'objet de la sélection d'attributs est de produire à partir des D variables initialement considérées, un sous-ensemble "optimal" de d attributs (généralement $d << D$). Il s'agit là d'une problématique de recherche qui suscite depuis une dizaine d'années un intérêt croissant de la part de la communauté de l'apprentissage artificiel" [22, 2, 49, 30, 16].

Quelques travaux sur la reconnaissance des instruments de musique ont eu recours à la sélection automatique des attributs [11, 7, 40, 41].

Nous commençons par une description des pré-traitements effectués sur les données (préalablement à la sélection). Puis nous présenterons l'algorithme que nous avons utilisé, choisi comme étant le plus pertinent du point de vue de la robustesse.

Normalisation des données

Les valeurs de plusieurs attributs, notamment issus de descripteurs de nature physique différente, présentent souvent des dynamiques assez hétérogènes. A titre d'exemple, les variables mesurant la variation d'attributs sur des trames successives (dérivées temporelles) présentent des valeurs très petites par rapport aux valeurs intra-trames. Les attributs possédant des valeurs plus grandes risquent alors d'avoir une influence plus importante sur le comportement des différents traitements à suivre (sélection, transformation, classification), même si cela ne reflète pas forcément leur pertinence pour la tâche envisagée.

Afin de contourner ce problème, on fait classiquement appel à des techniques de normalisation permettant d'uniformiser les dynamiques des différentes variables. Cette normalisation est réalisée de façon linéaire en exploitant les estimations empiriques (à partir de l'ensemble d'apprentissage) des moyennes et des variances des attributs [49] définies pour le j-ème attribut et pour l exemples par :

$$\mu_j = \frac{1}{l}\sum_{k=1}^{l} x_{k,j} \ , \ 1 \leq j \leq D \tag{2.1}$$

$$\sigma_j^2 = \frac{1}{l-1}\sum_{k=1}^{l}(x_{k,j}-\mu_j)^2. \tag{2.2}$$

La normalisation que nous désignons par "normalisation $\mu\sigma$" consiste alors à prendre

$$\hat{x}_{k,j} = \frac{x_{k,j}-\mu_j}{\sigma_j}, \tag{2.3}$$

ce qui a pour effet d'assurer que les attributs normalisés possèdent une moyenne nulle et une variance unitaire.

Inertia Ratio Maximization (IRM)

Cette approche du type *filter* a été proposée et utilisée avec succès pour la reconnaissance automatique des instruments de musique [41]. Il s'agit d'un algorithme itératif dans lequel, à chaque itération k, un sous-ensemble $\mathcal{S}_{d_k}$ de $d_k = k$ attributs est construit en incluant un attribut supplémentaire au sous-ensemble précédemment sélectionné $\mathcal{S}_{d_k-1}$. A l'itération d, $d_k = d$, et le nombre d'attributs ciblé est atteint.

Soient Q le nombre de classes, l_q le nombre de vecteurs d'attributs (vecteurs d'apprentissage) associés à la classe Ω_q et l le nombre total de vecteurs d'apprentissage ($l = \sum_{q=1}^{Q} l_q$).

Soit $\mathbf{x}_{i_q,d_k}$ le i_q-ème vecteur d'attributs de la classe Ω_q (contenant les d_k attributs sélectionnés à l'itération k), et soit μ_{q,d_k}, respectivement μ_{d_k}, le vecteur de moyenne des exemples $(\mathbf{x}_{i_q,d_k})_{1\leq i_q \leq l_q}$, respectivement le vecteur de moyenne de tous les exemples $(\mathbf{x}_{i_q,d_k})_{1\leq i_q\leq l_q;\ 1\leq q\leq Q}$.

Les attributs sont sélectionnés en se basant sur le rapport r_{d_k} entre l'inertie inter-classes B_{d_k} et le "rayon moyen" de la dispersion intra-classe[1] R_{d_k}, défini par :

$$r_{d_k} = \frac{B_{d_k}}{R_{d_k}} = \frac{\sum_{q=1}^{Q}\frac{l_q}{l}\left\|\mu_{d_k,q}-\mu_{d_k}\right\|}{\sum_{q=1}^{Q}\left(\frac{1}{l_q}\sum_{i_q=1}^{l_q}\left\|\mathbf{x}_{d_k,i_q}-\mu_{d_k,q}\right\|\right)} \tag{2.4}$$

Le principe est inspiré de l'Analyse Linéaire Discriminante. L'idée est de sélectionner les attributs qui permettent une bonne séparation entre classes (décrite par B_{d_k}) tout en minimisant la dispersion intra-classe (décrite par R_{d_k}). Par conséquent, chaque attribut supplémentaire sélectionné doit réaliser le maximum du rapport r_{d_k}.

Nous nous contentons de ce critère, sans effectuer d'étape d'orthogonalisation des vecteurs d'attributs, tout en sachant qu'il peut conduire à une certaine redondance des attributs sélectionnés. Cette redondance nous permet de rendre le modèle d'apprentissage plus robuste aux effets sonores.

[1] Il s'agit là d'une variation sur l'algorithme proposé initialement par G.Peeters.

2.3 Classification par SVM

2.3.1 Principes de décision

Décision bayésienne

On s'intéresse dans ce travail à un problème de classification supervisée, c'est-à-dire que l'on doit organiser des données en *classes* (ici, des instruments de musiques) connues d'avance.

On se donne un ensemble de Q classes $\{\Omega_q\}_{1\leq q\leq Q}$. On suppose connus :
- la *probabilité à priori* $P(\Omega_q)$ de chaque classe Ω_q,
- la *densité de probabilités conditionnelles* $p(\mathbf{x}|\Omega_q)$ décrivant la distribution du vecteur des attributs pour chaque classe Ω_q,

On déduit alors la *règle de décision bayésienne* minimisant la probabilité d'erreur sachant l'observation $\mathbf{x}$, qui associe à cette observation la classe Ω_{q_0} telle que :

$$q_0 = \arg\max_{1\leq i\leq Q} P(\Omega_q|\mathbf{x})$$

En supposant, comme dans cette étude, que les classes sont équiprobables, on obtient une règle simplifiée :

$$q_0 = \arg\max_{1\leq i\leq Q} P(\mathbf{x}|\Omega_q)$$

Fusion de décisions binaire

Pour cette étude, nous avons utilisé un schéma de classification décomposant le problème de classification à Q classes en problèmes bi-classes « un contre un » [**?**]. Il s'agit de considérer tous les couples de classes $\{\Omega_p,\Omega_q\}_{1\leq p<q\leq Q}$ (au nombre de $\frac{Q(Q-1)}{2}$) en construisant les classificateurs $\mathcal{C}_{p,q}$ permettant de discriminer Ω_p et Ω_q. Ce schéma s'impose pour l'utilisation de classificateurs qui sont binaires, comme les SVM, que nous présenterons en 2.3.2.

La décision finale nécessite donc une stratégie de fusion des décisions prises par tous les classificateurs. La stratégie adoptée est celle de Hastie et Tibshirani [17], qui permet d'obtenir des estimations des probabilités $P(\Omega_q|\mathbf{x})$.

2.3.2 Les machines à vecteurs supports (SVM)

On présente dans cette partie les Machines à Vecteurs Supports (SVM) qui sont les classificateurs utilisés dans notre système. Ces classificateurs bi-classes sont connus pour offrir de bonnes capacités de généralisation, même lorsque la dimension des vecteurs qu'ils doivent classifier est grande. Les SVM ont l'avantage d'être *discriminatives*, par opposition aux approches dites *génératives* comme les mélanges de gaussiennes, qui présupposent une structure particulières (souvent mal justifiées) des distributions de probabilités des données. La librairie Matlab LIBSVM [5] est utilisée comme classificateur à SVM.

Principe des SVM linéaires

Le principe des SVM est de séparer de manière optimale deux classes par un hyperplan dans l'espace des attributs. On note $\{\mathbf{x}_i\}_{i=1\ldots l}$ dans $\mathbb{R}^d$ un ensemble de l vecteurs de d attributs appartenant à deux classes différentes Ω_1 et Ω_{-1}. On note $\{y_i\}_{i=1\ldots l}$ dans $\{-1,+1\}$ les « étiquettes » contenant l'information de classe.

Si les données sont linéairement séparables, on peut déterminer un hyperplan $\mathcal{H}$ défini par :

$$\mathcal{H} : \mathbf{w}\cdot\mathbf{x}+b=0\ ,\ \mathbf{w}\in\mathbb{R}^d,\ b\in\mathbb{R}$$

tels que :

$$\begin{array}{rcll} \mathbf{w}\cdot\mathbf{x_i}+b & \geq & +1 & \text{si } y_i=+1 \\ \mathbf{w}\cdot\mathbf{x_i}+b & \leq & -1 & \text{si } y_i=-1 \end{array} \tag{2.5}$$

Ces deux équations peuvent se combiner pour donner :

$$y_i(\mathbf{w}\cdot\mathbf{x_i}+b)\geq 1$$

L'hyperplan optimal cherché est celui maximisant la marge entre les observations correspondant aux deux classes (illustration 2.1). Il est donné par la solution du problème d'optimisation :

$$\left\{\begin{array}{ll}\text{minimiser} & r(\mathbf{w})=\frac{1}{2}||\mathbf{w}||^2\ ;\ \mathbf{w}\in\mathbb{R}^d,\ b\in\mathbb{R}\\ \text{sous les contraintes} & y_i(\mathbf{w}\cdot\mathbf{x_i}+b)\geq 1\ ,\ \forall i=1\ldots l\end{array}\right. \tag{2.6}$$

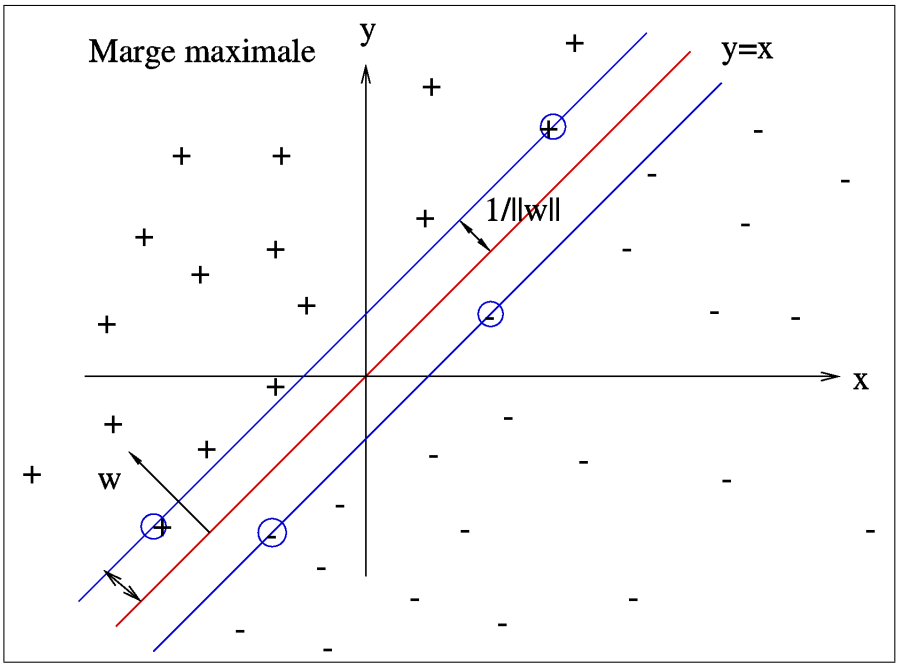

FIG. 2.1: Hyperplan optimal et marge d'un classificateur SVM

Calcul des SVM

Le problème (2.6) est un problème d'*optimisation sous contraintes* qui est résolu en introduisant des *multiplicateurs de Lagrange* $(\alpha_i)_{1\leq i\leq l}$ et un *Lagrangien* :

$$L(\mathbf{w},b,\alpha)=\frac{1}{2}||\mathbf{w}||^2-\sum_{i=1}^{l}\alpha_i[y_i(\mathbf{w}.\mathbf{x}_i+b)-1], \tag{2.7}$$

où

$$\alpha=[\alpha_1\ldots\alpha_l]^T$$

Le Lagrangien L doit être minimisé par rapport aux variables dites *primales* $\mathbf{w}$, b, et maximisé par rapport aux *variables duales* α_i : ce sont les conditions de Karush-Kuhn-Tucker (KKT) [43]. Après résolution, on retrouve que l'hyperplan optimal ne dépend que des n_s vecteurs supports du problème :

$$\mathbf{w}=\sum_{i=1}^{n_s}\alpha_i y_i\mathbf{x}_i, \tag{2.8}$$

et la fonction de décision est définie par le signe de :

$$f(\mathbf{x})=\mathbf{w}.\mathbf{x}+b=\sum_{i=1}^{n_s}\alpha_i y_i(\mathbf{x}.\mathbf{x}_i)+b. \tag{2.9}$$

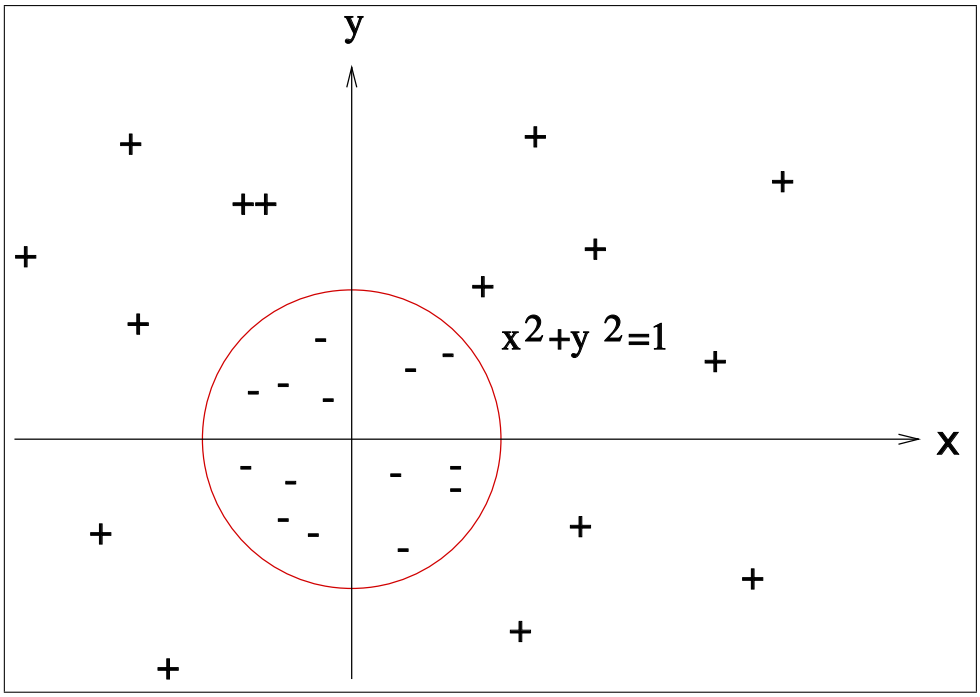

FIG. 2.2: Exemple simple de transformation : le problème n'est pas linéairement séparable en coordonnées cartésiennes, par contre en coordonnées polaires, le problème devient linéaire

SVM à *marge souple*

Dans le cas où les données ne sont pas linéairement séparables, le problème ci-dessus n'a pas de solution. Un remède consiste alors à rendre les contraintes moins rigides en introduisant des variables d'écart positives ξ_i. Les contraintes deviennent alors :

$$\forall i = 1 \dots l \begin{cases} y_i(\mathbf{w} \cdot \mathbf{x_i} + b) & \geq & 1 - \xi_i \\ \xi_i & \geq & 0 \end{cases}$$

Pour parer à ce problème, on change la fonction objectif du problème d'optimisation. La fonction à minimiser devient donc :

$$r(\mathbf{w}, \xi) = \frac{1}{2}||\mathbf{w}||^2 + C(\sum_{i=1}^{l} \xi_i)$$

Le paramètre $C > 0$ permet de régler le compromis entre la maximisation de la marge et la minimisation des erreurs de classification sur l'ensemble d'apprentissage. On parle alors de classificateurs à marge souple.

SVM non linéaires

Pour permettre aux SVM de considérer des frontières de décision non planes entre les classes, on peut transporter les données de l'espace de départ $\mathbb{R}^d$ dans un espace de Hilbert $\mathbb{E}$ de dimension supérieure, voire infinie. Pour une application $\Phi : \mathbb{R}^d \rightarrow \mathbb{E}$, on calcule une SVM linéaire pour séparer les données $\Phi(\mathbf{x}_i)$. La frontière obtenue produit des surfaces de décisions non planes dans $\mathbb{R}^d$ (voir 2.2).

Cette procédure s'effectue de manière efficace par une astuce que l'on appelle le *kernel trick* : en effet, le calcul des SVM fait intervenir les $\Phi(\mathbf{x}_i)$ uniquement sous forme de produits scalaires. On fait donc appel à une fonction $k(\mathbf{x}_i, \mathbf{x}_j)$ telle que :

$$k(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$$

Une telle fonction est appelée fonction *noyau*. La connaissance d'une telle fonction permet de calculer les SVM sans avoir à s'intéresser à la fonction Φ.

Paramètres utilisés

Dans cette étude, nous avons utilisé des SVM non linéaires exploitant un noyau gaussien (ou encore RBF pour *Radial Basis Fonction*), défini par :

$$k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(- \frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{2d\sigma^2} \right)$$

Pour exploiter le principe de fusion des décisions binaires évoqué en 2.3.1, on doit avoir une estimation de la probabilité conditionnelle d'appartenance à une classe des deux classes considérées, sachant les données. On note $f_i = k(\mathbf{w} \cdot \mathbf{x_i}) + b$ la sortie du classificateur binaire. La probabilité *a posteriori* $P(y = 1|f)$ est alors estimée par :

$$P(y = 1|f) = \frac{1}{1 + \exp(Af + B)}$$

où A et B sont des paramètres qui sont déterminés à partir des résultats de la classification sur un ensemble de développement. En raison de la taille relativement réduite de notre base de données, nous avons utilisé pour cela l'ensemble d'apprentissage.

2.4 Les vecteurs supports (SV)

2.4.1 Généralités

Les vecteurs qui se trouvent sur les marges d'un classificateur sont appelés les vecteurs supports (SV). Le problème ne dépend en fait que de ces points particuliers au sens ou si tous les autres points sont éliminés, la solution du problème reste la même.

Les vecteurs supports sont par définition situés aux "frontières" entre deux classes, définissant des surfaces de séparabilité dans l'espace des attributs. L'idée est d'étudier l'évolution de ces surfaces en fonction des traitements appliqués, ce qui revient à étudier l'évolution des vecteurs support de notre système de référence. Nous porterons donc un grand intérêt sur ces vecteurs au cours de l'analyse du problème.

2.4.2 Stabilité

Dans notre travail d'analyse des vecteurs d'observation présenté plus tard, nous allons introduire la notion de stabilité des vecteurs support aux traitements. Nous en donnons ici la définition.

Définition

Soient :
- E l'ensemble des vecteurs d'observation
- SV_E l'ensemble des vecteurs supports d'un apprentissage sur l'ensemble E
- Γ une fonction de $E \longrightarrow E$ représentant un des traitements utilisés

$$\forall\, X \in SV_E,\ X \textit{ est stable par } \Gamma \iff \Gamma(X) \in SV_{\Gamma(E)}$$

Exprimé autrement, un vecteur support X est stable par un traitement Γ si et seulement si lors d'un apprentissage sur une base transformée par Γ, le vecteur X est toujours vecteur support de la solution du problème transformé.

2.5 Recherche d'invariants

2.5.1 Connaissances préalables

D'après Schölkopf et Smola [43, 27], afin d'obtenir de meilleurs résultats de classification, les connaissances à priori d'un problème, que nous nommerons *connaissances préalables*, doivent être incorporées au

processus d'apprentissage. On entend par connaissances préalables, toutes informations complémentaires aux simples exemples de la base d'apprentissage. Sachant toutefois que plus la base d'apprentissage est grande, plus les informations d'invariants de la fonction de décision seront déjà implicitement incluses dans les observations d'une même classe.

2.5.2 Incorporer les invariants aux SVM

Il existe principalement trois manières d'incorporer ces connaissances en fonction de leur degré de formalisation 2.3. Ainsi on aura par ordre de complexité :

- Dans le premier cas, les connaissances sont utilisées pour générer des exemples d'apprentissages artificiels, nommés "exemples virtuels", en transformant les exemples d'apprentissage eux-mêmes. La SVM va alors automatiquement apprendre les invariants
- Dans le second cas, l'algorithme d'apprentissage est lui-même modifié. Une fonction de contrainte amène la SVM à construire une fonction de décision contenant les invariants désirées [46], ce qui revient à construire un noyau adapté aux données.
- Enfin, dans le troisième cas, l'invariant est introduit par projection des vecteurs d'attribut dans un espace plus approprié. La représentation des données peut également être changée en utilisant une mesure de distance modifiée entre vecteurs.

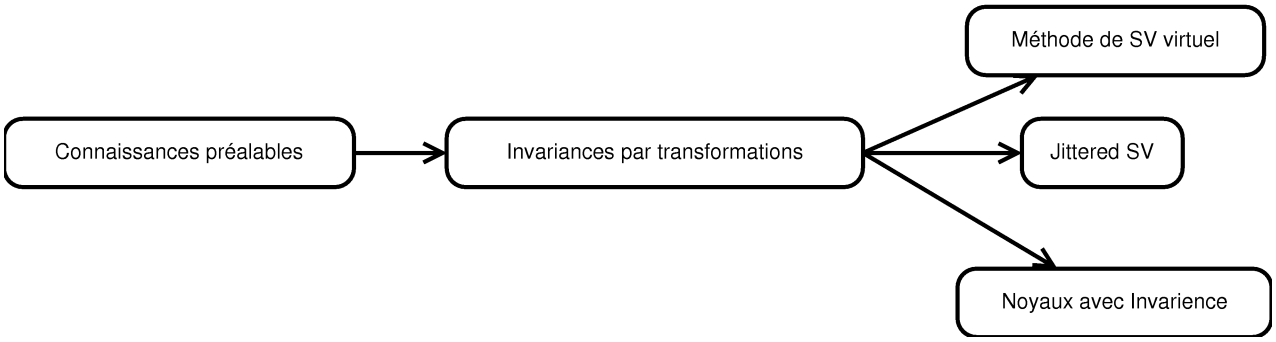

FIG. 2.3: Schéma d'incorporation des connaissances préalables aux SVM.

2.5.3 Acquisition des connaissances préalables

Très peu d'informations préalables sont disponibles dans le problème de reconnaissance des instruments de musique. Nous partons du simple fait que : "La reconnaissance d'un instrument est toujours possible malgré l'ajout de certains effets".

Nous tenterons d'acquérir d'autres connaissances d'un degré 'supérieur' par :

- une approche analytique : Elle consiste à partir des équations des effets de mettre en avant des transformations sur le calcul des descripteurs. C'est certainement la méthode qui permet d'apporter le plus de connaissances préalables. Cependant deux obstacles majeurs se sont posés dans notre cas : le premier est que les effets utilisés nécessitent un paramétrage important et la possibilité de généralisation est faible. Le second est que, supposant être arrivé à mettre en évidence des invariants dans le calcul d'un ou plusieurs descripteurs, il devient très difficile d'intégrer celles-ci suite à l'étape de sélection des attributs.
- la visualisation des vecteurs d'attribut ou d'un sous ensemble d'attribut, afin d'observer les éventuelles transformations, dans l'espace des attributs, induites par les effets.
- l'écoute comparative des vecteurs supports stables et des vecteurs support introduits, avec et sans ajout d'effets.
- une analyse statistique des vecteurs d'attributs.

Chapitre 3

Évaluation des dégradations apportées par les traitements

Dans ce chapitre, nous tentons de mesurer la stabilité des performances du système lors de l'introduction des perturbations sur les données de tests et d'apprentissage afin d'évaluer les dégradations apportées par chacun des traitements. Après une présentation de notre base de données, nous donnerons les résultats obtenu sur le système de références. Nous expliquerons également sur quels critères sont évaluées les performances. Puis nous comparerons celles-ci obtenues avec des données modifiées.

3.1 Base de données

La collecte de phrases musicales mono-instrumentales s'avère être une tâche ardue et longue. Nous avons eu à notre disposition la base de données constituée au cours de la thèse de Slim Essid. Nous utilisons uniquement le *corpus mono-instrumental* de la base qui se compose principalement d'extraits d'enregistrements numériques de musique classique, de jazz ou de supports pédagogiques, ainsi que quelques pièces de solo provenant de la base RWC.[1]

Les extraits sont encodés en mono au format PCM à une fréquence d'échantillonnage de 32kHz. Afin de simplifier l'étude et de limiter la charge de calcul, les expériences ont été menées sur un sous ensemble de 6 classes d'instruments (tableau 3.1) puis nous avons divisé ce sous-ensemble en trois bases : la base d'apprentissage, la base de test et la base pour la validation.

Nous disposons ainsi d'une base de référence dont les conditions d'enregistrement et de réalisation sont différentes d'une source à l'autre avec une séparation totale entre les sources de la base d'apprentissage et celles de la base de test.

Instrument	Code
piano	Pn
guitare acoustique	Gt
basson	Bo
hautbois	Ob
violoncelle	Co
violon	Vl

TAB. 3.1: Instruments utilisés et codes associés

[1] Base de sons musicaux conçu par des chercheurs japonais pour des travaux d'indexation audio [13].

Instruments	Sources d'apprentissage	App.	Sources test	Test	Sources validation	Valid.
Pn	6	22' 54"	6	19' 36"	6	15'38"
Gt	6	22' 54"	6	19' 36"	6	15'38"
Bo	6	22' 54"	5	19' 36"	5	15'38"
Ob	6	22' 54"	7	19' 36"	7	15'38"
Co	4	22' 54"	7	19' 36"	6	15'38"
Vl	8	22' 54"	7	19' 36"	12	15'38"

TAB. 3.2: Base de sons mono-instrumentaux avec le nombre de sources disponibles et la durée des enregistrements effectivement utilisée.

3.2 Résultats de référence

La base de référence est celle décrite au 3.1 et les résultats suivants sont ceux obtenus avec le système de classification précédemment défini.

3.2.1 Matrices de confusion

La matrice de confusion va donner le nombre d'exemples de la classe qui ont été reconnue comme correcte et le cas contraire avec quel instrument ils ont été confondu.

ref	Pn	Gt	Bo	Ob	Co	Vl	total
Pn	3545	77	2	1	46	3	3674
Gt	954	1760	21	0	867	72	3674
Bo	697	174	2659	82	18	44	3674
Ob	31	0	22	3313	155	153	3674
Co	651	412	0	65	1624	922	3674
Vl	1	0	18	58	550	3047	3674

TAB. 3.3: Matrice de confusion de référence X. A lire : *La classe de la ligne i à été reconnue $X_{i,j}$ fois comme la classe de la colonne j.*

ref	Pn	Gt	Bo	Ob	Co	Vl
Pn	**96.5**	2.1	0.1	0.0	1.3	0.1
Gt	26.0	**47.9**	0.6	0.0	23.6	2.0
Bo	19.0	4.7	**72.4**	2.2	0.5	1.2
Ob	0.8	0.0	0.6	**90.2**	4.2	4.2
Co	17.7	11.2	0.0	1.8	**44.2**	25.1
Vl	0.0	0.0	0.5	1.6	15.0	**82.9**

TAB. 3.4: Matrice de confusion traduite en pourcentage de reconnaissance.

3.2.2 Score global

Le score global est calculé à partir de la matrice de confusion en pourcentage, comme la moyenne des scores sur la diagonale (nous donnons également l'écart type). En effet, la diagonale nous donne le pourcentage d'exemples bien classés pour une classe donnée.

- Score global : 72.3
- Écart type : 20.0

3.3 Apport de l'utilisation d'un noyau

Comme expliqué au 2.3.2, très peu de problèmes de classification réels ne trouvent des solutions linéaires sans l'introduction d'erreurs. L'utilisation de noyau qui est une généralisation de l'approche linéaire permet une optimisation des fonctions de décision. Cependant le score obtenu avec un tel type de classificateur dépend fortement des paramètres du noyau. C'est pourquoi nous incluons dans notre processus une étape de recherche des paramètres optimaux que nous décrivons ci-dessous. D'autres méthodes, plus élaborées sont proposées dans les articles [48, 24].

3.3.1 Recherche des paramètres optimaux par validation croisée

Il est important de rappeler que nous utilisons un noyau gaussien dont le paramètre principal est σ (Voir 2.3.2). A cela s'ajoute le paramètre C qui permet de contrôler le compromis entre nombre d'erreurs de classement, et la largeur de la marge. Afin de trouver les paramètres optimaux, une des méthode classique est de procéder à une évaluation par validation croisée. Cette méthode consiste à effectuer l'apprentissage et le test sur des sous-ensembles de la base d'apprentissage que l'on croise entre eux deux à deux. A chaque test de validation, les paramètres du noyau changent suivant une grille prédéfinie des valeurs les plus probables. Enfin les paramètres qui ont obtenu le plus grand score en moyenne sur les tests croisés, sont gardés pour effectuer le réel apprentissage. De part la taille réduite des sous-ensembles le temps de calcul est largement diminué par rapport à une classification complète.

3.3.2 Résultats avec noyau

L'amélioration significative apportée par l'utilisation d'un noyau nous montre 3.5 la non linéarité du problème de reconnaissance des instruments de musique. Dorénavant, tous les résultats montrés seront calculés sur des SVM-RBF.

Train :REF	Score SVM linéaire	écart type	Score SVM RBF	écart type
REF	72.3	20.0	75.3	18.3

TAB. 3.5: Paramètres sélectionnés $\sigma = 2.8$ et $C = 15$

3.4 Influence des effets sur le test

Cette expérience est réalisé dans le but d'évaluer la dégradation des performances engendrée par des *données de test transformées*. Des bases de tests sont classée une à une afin de voir la détérioration respective des effets. Cependant il faut garder à l'esprit que l'objectif est de constituer une base de test globale hétérogène.

Dans le tableau 3.6, les résultats sont donnés par effet pour chacune des classes en gardant uniquement les scores de la diagonale de la matrice de confusion en pourcentage.

Pour pouvoir comparer les résultats, nous calculons l'intervalle de confiance sur les scores comme :

$$[\bar{x} - \sqrt{\sigma^2/N}; \bar{x} + \sqrt{\sigma^2/N}]$$

Avec $\bar{x}$ la moyenne et σ l'écart type.

Dans l'analyse des résultats, nous considérerons donc comme significatif un écart supérieur à 0.6% sur le score global.

Il est dès à présent possible d'établir un classement des effets qui apportent une dégradation des performances du système. Ainsi nous observons que :

- le codage à 32kbps fait perdre plus de 10% des performances, avec une quasi incapacité à reconnaître les enregistrements de Co modifiés,
- le sous-échantillonnage, les égaliseurs EQ2std, EQ1std puis la reverb apportent des dégradation raisonnable,

Train : REF	Pn	Gt	Bo	Ob	Co	Vl	score	écart type
ref	97.4	50.4	77.3	89.5	47.1	90.0	75.3	19.7
32kbps	98.9	47.3	68.4	93.3	15.3	64.4	64.6	28.1
64kbps	97.6	52.7	78.8	88.3	50.0	87.5	75.8	18.1
low8k	98.4	49.9	74.2	93.1	32.4	75.3	70.6	23.1
reverb	97.9	49.7	76.2	86.0	42.1	87.7	73.3	20.5
comp	97.2	51.4	80.8	89.8	48.3	86.9	75.7	18.9
EQ2std	95.2	28.3	69.9	84.7	61.4	84.0	70.6	21.8
EQ1std	96.4	38.3	75.6	85.7	61.9	84.6	73.8	19.0
EQ-1std	95.9	52.0	81.8	85.7	62.9	87.7	**77.7**	**15.2**
EQ-2std	95.4	56.5	83.1	85.5	58.5	89.7	**78.1**	**15.1**
Moyenne							73.6	19.8

TAB. 3.6: Matrice des diagonales de confusion de la classification des bases de tests déformées sur les classificateurs de références.

- la compression et le codage à un débit de $64 kbit.s^{-1}$ sont "transparents" sur notre système,
- il est surprenant de constater que les égaliseurs EQ-1std et EQ-2std améliorent significativement les scores tout en diminuant l'écart type. Ils agissent ainsi comme un pré-traitement “normalisant“ les données entre elles.

3.5 Influence des effets sur l'apprentissage

Nous nous intéressons ici à observer d'un coté l'impact des effets sur le processus d'apprentissage (sélection des attributs (voir tableau B.1 en annexe), paramètres optimaux du noyau) et d'un autre coté l'évolution des performances. Enfin nous comparerons les résultats obtenus au 3.4 (bases de tests modifiées sur apprentissage de référence) avec ceux où l'apprentissage et le test ont subi le même traitement.

	ref	comp	32kbps	64kbps	low8k	reverb	EQ2std	EQ1std	EQ-1std	EQ-2std
C	15	15	15	10	20	10	10	10	20	20
sigma	2.9	2.8	2.0	2.0	3.5	3.0	3.0	2.0	3.3	4.0
nb attr. rang différent	0	2	19	2	23	20	20	21	22	17
nb attr. introduits	0	0	3	1	3	1	6	5	1	1
score ref	75.3	*74.8*	*73.8*	75.8	*72.8*	**80.1**	**80.2**	**76.4**	*72.5*	*72.0*

TAB. 3.7: Résultats des différentes étapes de l'apprentissage et classification sur la base de test de référence

Il est intéressant de constater que certains effets, tels que la reverb et EQ2std, amènent un très net gain des performances sur le test de référence. L'apprentissage sur des bases transformées montre la limite des descripteurs utilisés. En effet, la sélection d'attributs va être différente selon la transformation appliquée. La comparaison des résultats est alors difficile car les classificateurs sont calculés sur des ensembles d'attributs différents. Dans le chapitre 4.1.2 nous proposerons une méthode permettant d'obtenir un ensemble d'attributs robustes.

Test/Train	ref	same fx
ref	75.3	75.3
32kbps	64.6	73.6
64kbps	75.8	76.2
low8k	70.6	74.3
reverb	73.3	78.5
comp	75.7	75.2
EQ2std	70.6	80.3
EQ1std	73.8	77.3
EQ-1std	77.7	74.9
EQ-2std	78.1	75.3
moyenne	73.6	76.1

TAB. 3.8: Résultats de l'apprentissage sur la base de référence en comparaison avec l'apprentissage ayant subi les même modifications que le test.

Afin de mesurer l'apport potentiel de l'incorporation des effets dans notre système, chacunes des bases modifiées ont été classifiées avec des fonctions de décisions calculées sur des bases d'apprentissage modifiée par le même traitement. Par comparaison des résultats obtenus, résumés dans le tableau 3.8, il semblerait qu'à l'exception de EQ-1std et EQ-2std, l'apprentissage sur une base transformée au même titre que les observations de la base de test permette d'avoir des performances bien meilleures que les classificateurs de références. De plus mis a part 32kbps et low8k, qui sont n'oublions pas des transformations amenant une grande perte d'information, nous obtenons des scores au moins équivalents à la référence de 75.3%.

Toutefois, même si nous avons trouvé un moyen de "reconnaître" les données transformées au moins aussi bien que les données de référence notre but est de rendre le système robuste à des données hétérogènes. Or cette méthode ne peut être appliquée que dans le cas où la totalité de la base de donnée a subi uniformement une seule dégradation et connue d'avance.

3.6 Constitution d'un sous-ensemble d'effets

Suite aux résultats précédents, nous avons choisi d'une part de considérer à part les transformations de codage à très bas débit et le sous-échantillonnage comme des traitements qui nécessitent des classificateurs dédiés, d'autre part que les effets de compression étudiés et de codage à débit supérieur ou égale à $64kbit.s^{-1}$ n'ont pas d'impact significatif sur les performances du système. Ce qui nous amène, par la suite, à nous concentrer uniquement sur un sous ensemble d'effets que l'on nommera SUB-FX comprenant : la réverbération et toutes les configurations d'égalisations.

$$SUB-FX = \{reverb,\ EQ2std,\ EQ1std,\ EQ-1std,\ EQ-2std\}$$

Chapitre 4

Approches suivies dans l'amélioration du système de référence

Après avoir constitué un sous-ensemble d'effets pertinents, nous présenterons dans une première section une méthode qui vise à construire un ensemble d'attributs robustes. Puis nous exposerons la démarche suivie dans la recherche des invariants passant par la mise en place des techniques d'analyses des vecteurs d'attributs et des vecteurs supports, présentées au 2.5.

4.1 Robustesse des attributs

Nous avons vu au 2.2.1 l'importance d'avoir des attributs efficaces dans le système de classification. Il s'agit ici de mettre la robustesse des attributs ainsi que l'algorithme de sélection à l'épreuve des traitements. Ainsi un attribut sera considéré d'autant plus robuste que son rang de sélection est élevé *en moyenne* au cours du processus de sélection.

4.1.1 Attributs sélectionnés sur des bases modifiées

Afin de mesurer la robustesse des attributs, nous calculons la moyenne de leur rang obtenu dans chacune des bases d'apprentissages modifiées (se référer au tableau en annexe B.1).

Après analyse de ce tableau, il est intéressant de remarquer que d'une part les ensembles sélectionnés pour *32kbps* et *low8k* font apparaître deux attributs qui ne sont sélectionnés sur aucune autre bases d'apprentissage et d'autre part que *comp* et *64kbps* n'ont que très peu de différences avec la référence du point de vue de la sélection d'attributs. Ceci est à rapprocher des résultats de la section 3.5 et permet de renforcer le choix d'un sous ensemble d'effets (section 3.6).

Enfin, les quatre premiers attributs ont été sélectionnés pour chacune des bases transformées en gardant le même rang. Ces attributs ont donc une grande séparabilité des classes et sont en l'occurrence robustes à l'ensemble des effets appliqués.

En annexe nous donnons le classement des attributs par rang moyen sur toute les bases B.2. Dans ce tableau, les attributs qui n'était pas sélectionnés dans le système de référence sont en gras et ceux dont le rang est modifié sont en italique.

4.1.2 Constitution d'un ensemble d'attributs robuste

La stratégie adoptée consiste à garder les attributs en fonction de leur rang moyen (indicateur de leur robustesse) pour constituer un ensemble robuste d'attributs. La figure 4.1 représente pour chaque attribut le rang obtenu sur les différentes bases. Par la suite, nous ne garderons que les trente premiers attributs classés suivant leur rang moyen, uniquement pour les bases modifiées par le sous ensemble d'effets SUB-FX (annexe tableau B.3). Par rapport à l'ensemble de référence, seul un attribut a été supprimé au profit d'un autre. Il

s'agit du 17ème coefficient de la loudness, remplacé par le premier coefficient de la platitude du spectre. Nous noterons l'ensemble de ces attributs **FEAT-SUB-FX**.

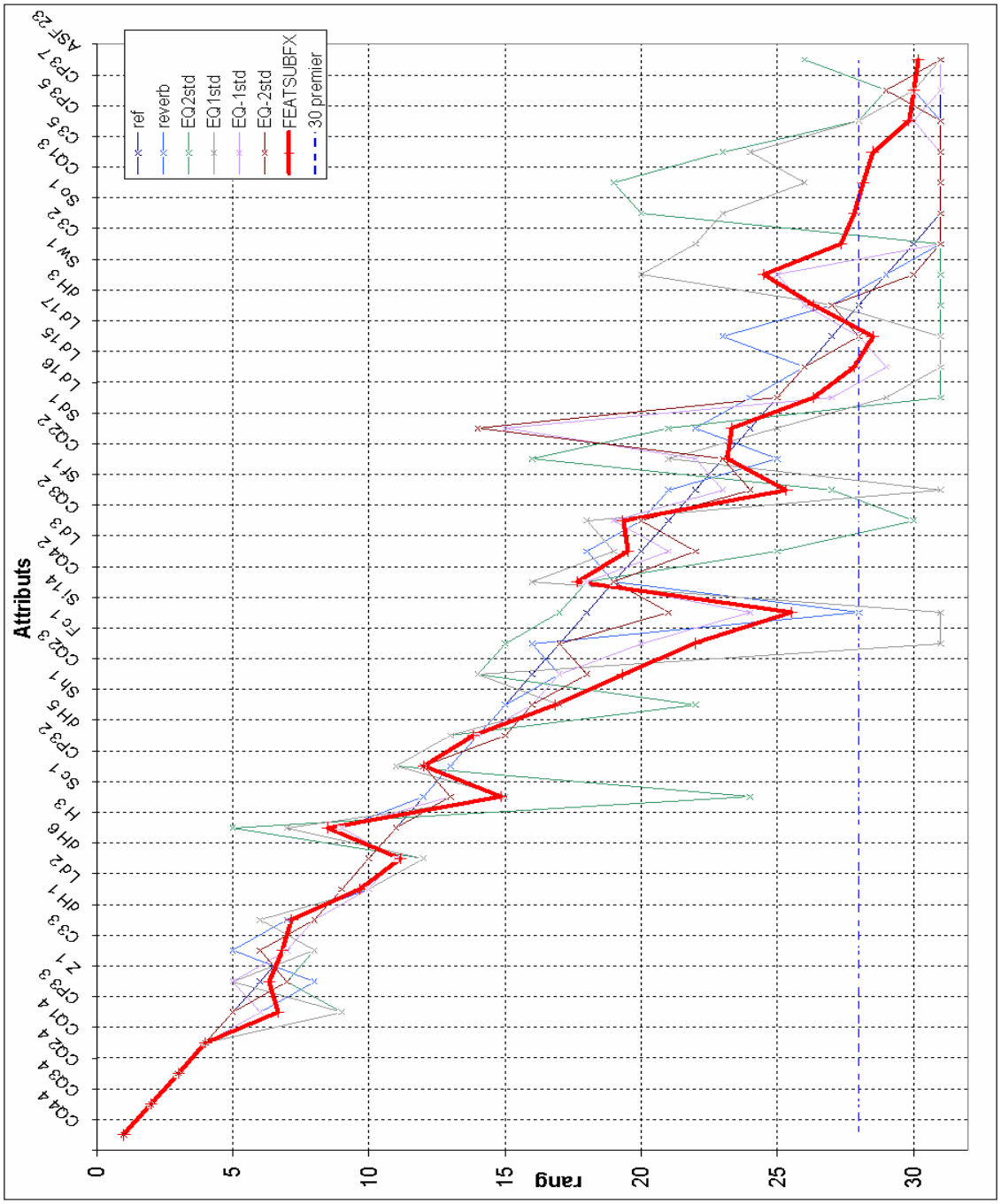

FIG. 4.1: Rang des attributs sur les différentes bases de SUB-FX avec en rouge la moyenne. Le rang des attributs non sélectionnés a été mis arbitrairement à 31.

Remarques :

L'algorithme d'apprentissage implémenté dans libSVM ne tient pas compte de l'ordre des attributs.

Nous nous limitons volontairement à 30 attributs cependant la puissance de calcul des machines, sans cesse grandissante, nous invite de plus en plus à considérer une sélection d'un grand nombre d'attributs (jusqu'à une centaine).

4.2 Analyse des vecteurs d'attributs

4.2.1 Visualisations des vecteurs d'observation

C'est dans l'idée d'observer des invariants qu'un outil de visualisation a été implémenté. Il permet de suivre l'évolution des vecteurs d'observation et en particulier des vecteurs supports, suite à leur transformation. Du au nombre important de vecteur et à leur dimension supérieure à 3, plusieurs techniques ont été utilisées afin de rendre possible la visualisation.

Diminution du nombre d'observation par quantification vectorielle

Le nombre important de vecteur d'observation va rendre trop complexe leur visualisation, notament au niveau du temps de calcul. La quantification vectorielle va effectuer une segmentation des vecteurs et les remplacer par un vecteur moyen que l'on nomme centroïde.

Diminution de la dimensionnalité par ACP

La dimension élevée des vecteurs d'observation $dim(X) = 30$ ne permet pas leur visualisation dans l'espace. L'ACP ou encore l'analyse en composante principale a pour objectif de décrire un ensemble d'observations, par de nouvelles variables en nombre réduit. Ces nouvelles variables seront des combinaisons linéaires des variables originales et porteront le nom de Composantes Principales (CP).

En général, la réduction du nombre de variables utilisées pour décrire un ensemble de données provoque une perte d'information. L'ACP procède de façon à ce que cette perte d'information soit la plus faible possible, selon un sens précis et naturel que l'on donnera au mot "information".

Soit R_x la matrice de covariance des vecteurs d'apprentissage. Dans un premier temps, une décomposition en valeurs propres de R_x est calculée :

$$R_x = V \Delta V^t \tag{4.1}$$

où Δ est la matrice des valeurs propres, supposées ordonnées par ordre décroissant et V est la matrice des vecteurs propres. La matrice $W = V^t$ est alors utilisée pour projeter les vecteurs d'observation x_i selon les trois premières composantes :

$$y_i = \mathbf{W} x_i \tag{4.2}$$

L'Analyse en Composantes principales peut donc être vue comme une technique de réduction de dimensionalité et est donc utilisée dans notre étude afin de permettre la visualisation des vecteurs dans l'espace en trois dimensions.

Observations

Choix de représentation :
- chaque couleur représente une classe
- les points de couleurs représentent un vecteur de référence
- les lignes sont les trajectoires prisent par les vecteurs après leur transformations (en noir)
- les symboles sont les différentes configurations des effets

Afin de minimiser la perte d'information engendrée, nous appliquons la PCA sur des paquets d'attributs plutôt que sur l'ensemble des attributs sélectionnés.

Liste des paquets observés :
- SX : Sc, Sw, Sf
- ZlZ : Z, lZ
- HdH
- CP3

La figure 4.2 est la visualisation du paquet des moments spectraux avec la trajectoire prise par les vecteurs lors de l'ajout d'une reverb. La distance entre les points avant et après est très faible et semble prendre des direction aléatoire. La même tendance se retrouve sur la visualisation des autres paquets.

Quant à l'égalisation 4.3, toutes les configurations sont observées sur le même graphe et les trajectoires de points sont dans l'ordre suivant : EQ-2std (△) -> EQ-1std (△) -> ref (. ou +)-> EQ1std (+) -> EQ2std (+). Avec entre parenthèse le symbole correspondant.

Il est intéressant de voir l'influence du coefficient de gain. Contrairement à la reverb, il semble qu'une réelle trajectoire complexe soit suivie par les vecteurs égalisés. Il est également fort intéressant de remarquer qu'une direction de la trajectoire permet d'éloigner les points entre classes.

Il semble qu'il n'existe pas de transformations dans l'espace des attributs (ou paquets d'attributs uniquement) assez simple pour être incorporer dans l'apprentissage des classificateurs par les méthodes de construction de noyaux ni de modification du calcul de la distance.

Cependant nous avons observé qu'il existe pour l'égalisation une direction de la trajectoire suivie par les vecteurs qui permettent d'éloigner les points entre classes ce qui devrait augmenter la séparabilité des classes entre elles. Ceci permet d'avoir une information sur les configurations d'égalisation à incorporer, susceptibles d'apporter un gain de robustesse.

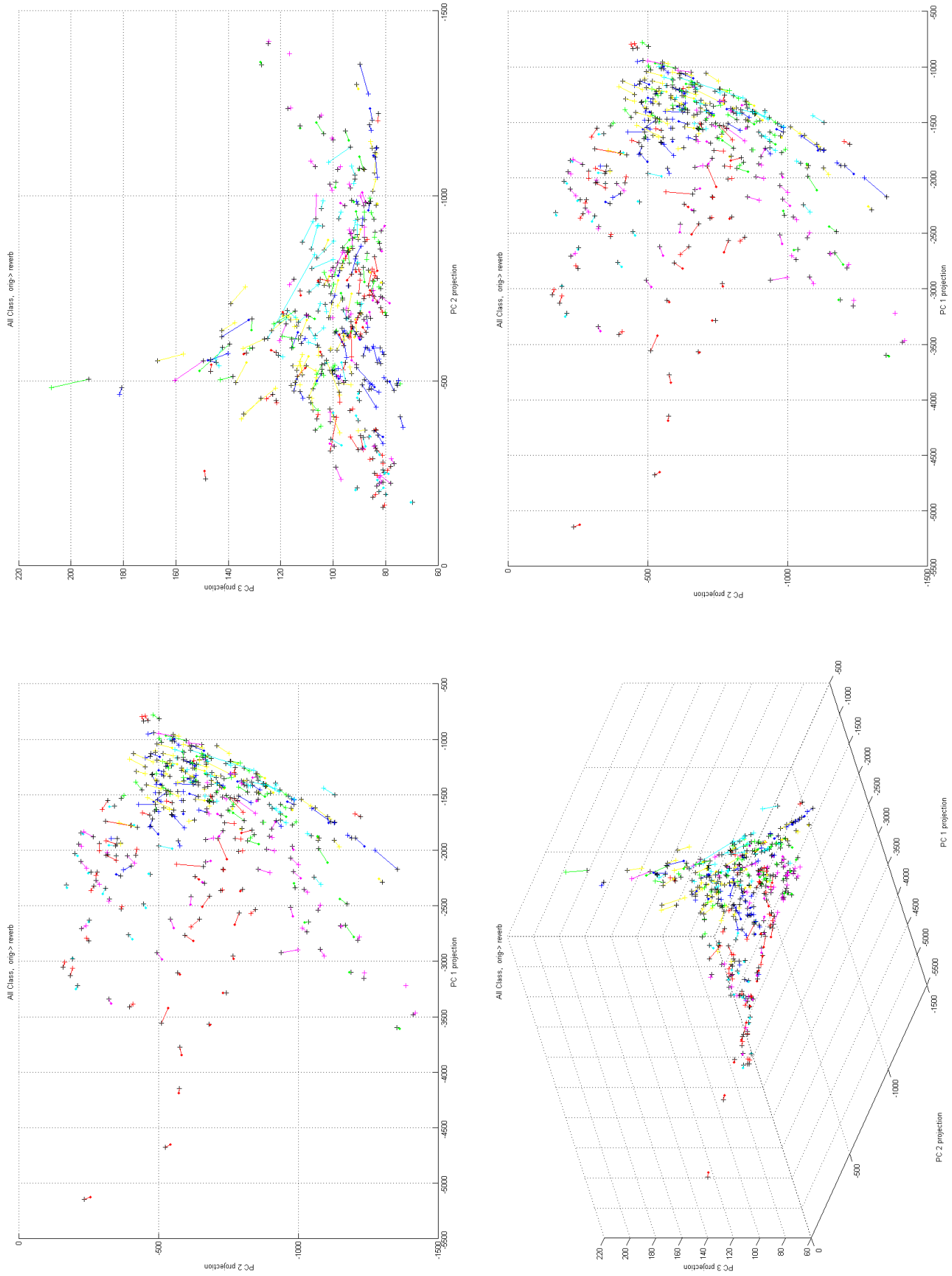

FIG. 4.2: Visualisation des trajectoires suivis par le paquet Sx modifié par la reverb

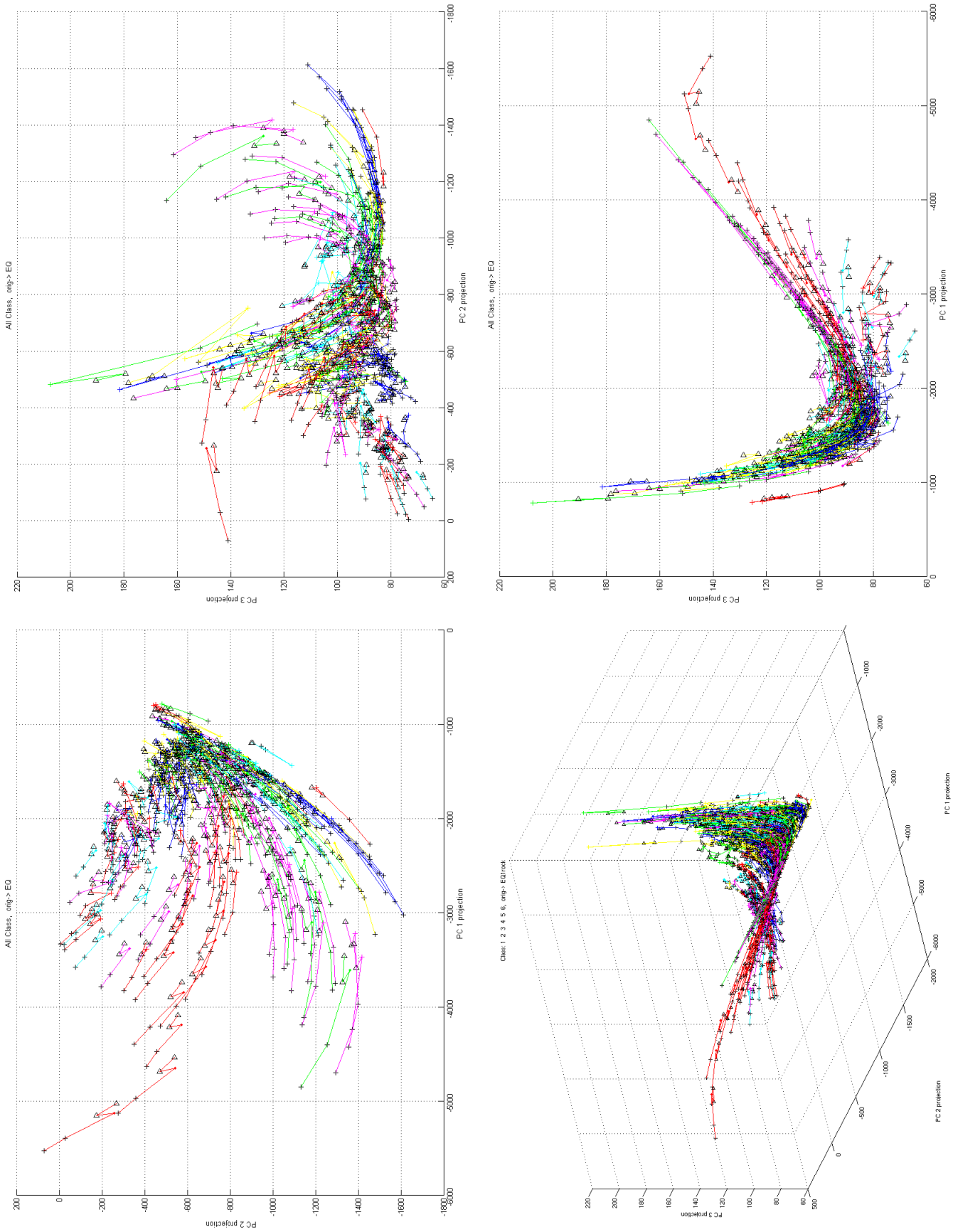

FIG. 4.3: Visualisation des trajectoires suivis par le paquet Sx modifié par l'égalisation

4.2.2 Écoute des vecteurs supports

Un outil d'écoute des vecteurs supports à été développé à partir de SCPLAB. SCPLAB est un outil d'analyse et d'écoute des signaux de parole développé pour Matlab. Il permet entre autre de :

- sélectionner, zoomer et écouter un segment du signal.
- utiliser des outils d'analyse, comme la FFT, le cesptre, calcul du pitch, le spectrogramme, etc
- compare et analyser plusieurs signaux entre eux.

Nous avons utiliser cet outil de manière détournée afin d'aligner les vecteurs supports sous le signal (voir figure 4.4). Toutes les caractéristiques de l'outil restent cependant utilisable (écoute et analyse de segments). Afin de différencier les vecteurs support entre eux et d'identifier les outliers, la hauteur de chaque histogramme représente la valeur du coefficient α 2.7 normalisé au paramètre C.

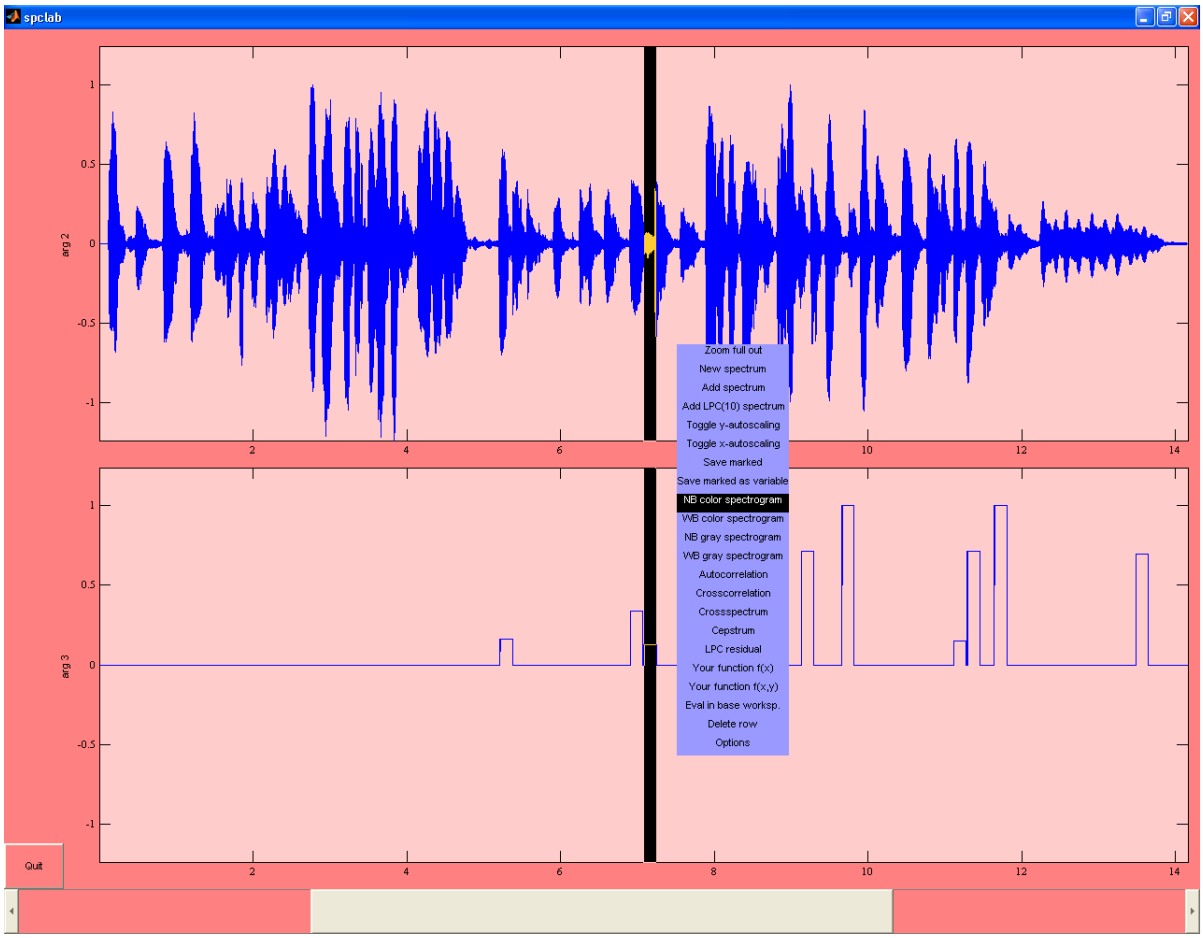

FIG. 4.4: Fenêtre de l'outil d'écoute et de visualisation temporelle des vecteurs supports . La hauteur associée à chaque vecteur support est égale à la valeur du coefficient α (voir l'équation 2.7) normalisée au paramètre C.

L'écoute comparative des vecteurs support stable et des vecteurs support introduits, avec et sans ajout d'effets est une tâche difficile car aucune étude n'existe sur les particularités de ces vecteurs. De plus le nombre important de vecteurs multiplié par le nombre d'effets rend l'écoute longue et fastidieuse. Une étude plus poussée passant par une campagne d'écoute comparative fixant certains critères permettrait de dégager des conclusions sur les spécificités des vecteurs supports.

4.2.3 Analyse statistique des vecteurs d'attributs

Une analyse statistique à été réalisée en calculant pour chaque effet, la variance moyenne intra-classes des vecteurs d'attributs 4.5. Il est d'autant plus intéressant que cette variance est petite, car c'est un indicateur de la compacité des vecteurs d'une classes.

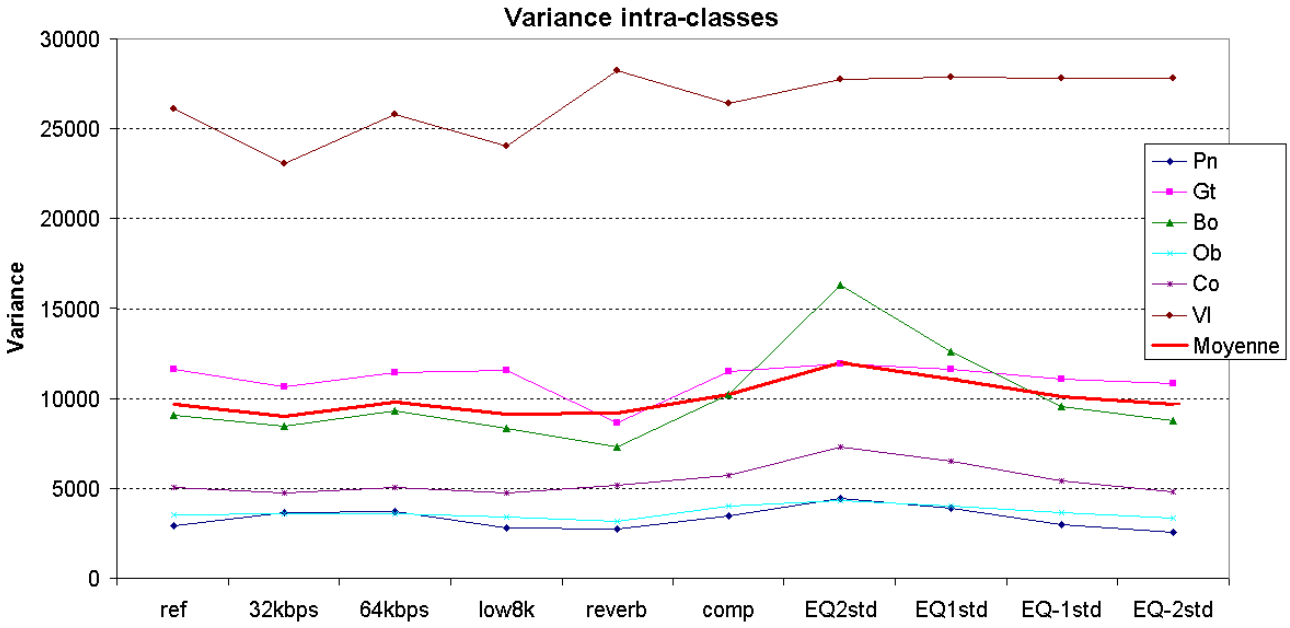

FIG. 4.5: Variance intra classes pour chacune des bases d'apprentissage modifiées

Nous calculant également pour chaque effet la variance inter-classes qui est un indicateur de la séparabilité entre les classes. Nous cherchons à augmenter cette valeur le plus possible.

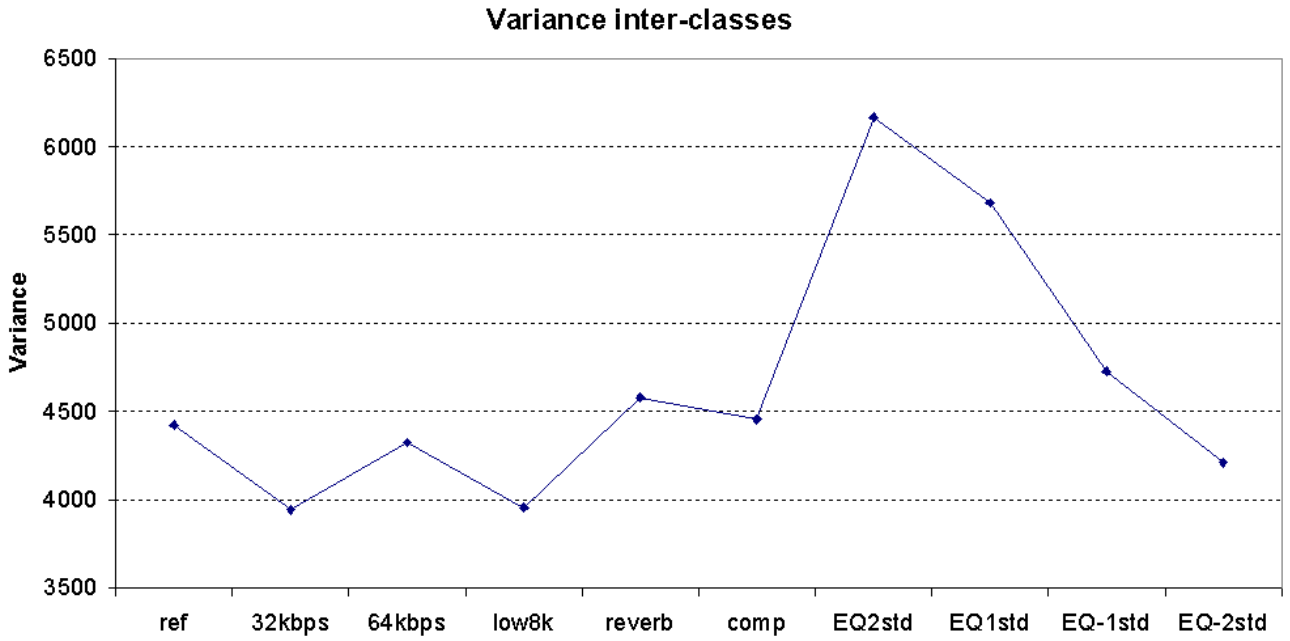

FIG. 4.6: Variance inter classes pour chacune des bases d'apprentissage modifiées

Un effet intéressant sera un compromis entre l'augmentation de la variance inter-classes et la stabilité de la variance intra-classes. On observe alors une bonne cohérence entre ces graphes, les visualisations et les résultats de classification, à savoir que les effets qui améliorent l'apprentissage sont ceux qui ont une grande variance inter-classes et inversement.

Chapitre 5

Mise en oeuvre d'une stratégie robuste

Nous avons vu au cours des chapitres précédents que l'analyse des vecteurs supports s'avère difficile cependant certains effets montrent des propriétés très intéressantes sur les performances des classificateurs. Ceci nous permet de mettre en oeuvre une stratégie d'incorporation des invariants tout en combinant les modifications du système de classification pour le rendre robuste dans sa globalité.

Après avoir montrer le gain apporté par l'ensemble d'attributs robuste construit au 4.1.2, nous exposerons l'approche des SVM virtuelles ainsi que les performances obtenues. Puis nous présenterons une méthode de normalisation des données hétérogènes pour finir sur la validation du système par les résultats de la classification de la base de validation.

5.1 Stratégie globale

Le schémas 5.1 présente les étapes du système de classification qui ont été modifiées dans la mise en oeuvre d'une stratégie robuste.

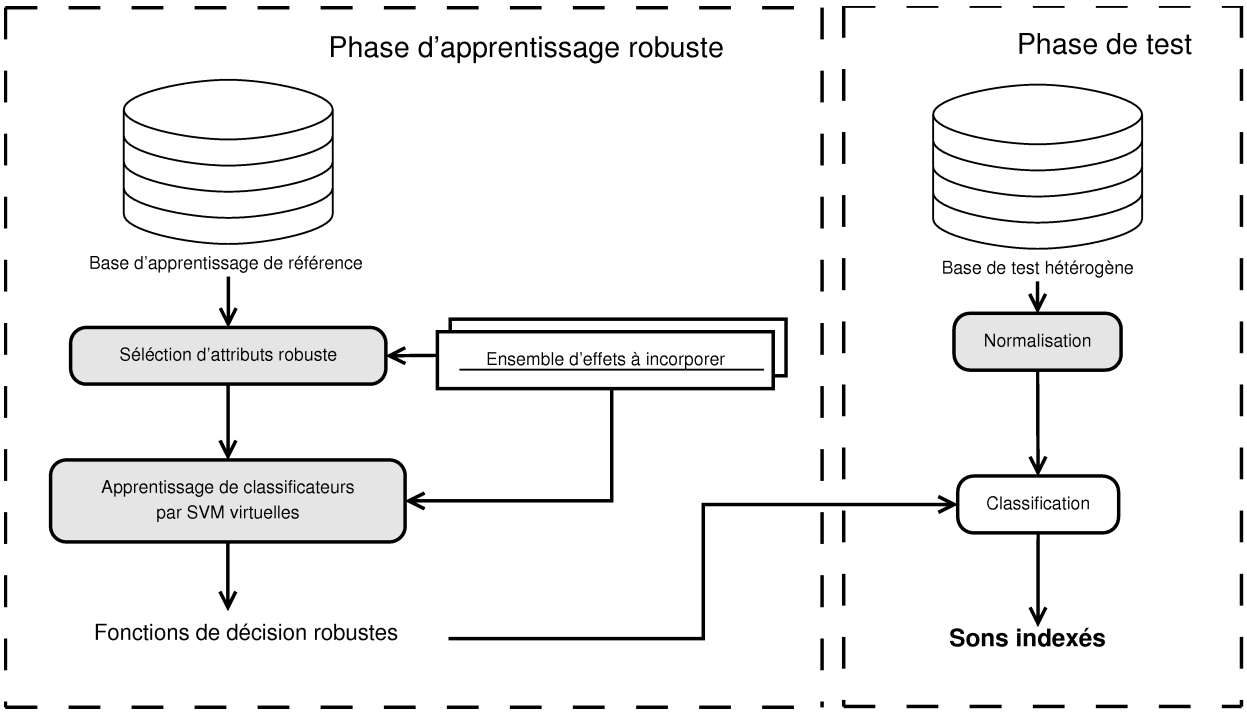

FIG. 5.1: Schéma du système de classification robuste

5.2 Résultats de classification avec l'ensemble d'attributs robustes

Le tableau suivant donne les performances du système de classification avec l'ensemble d'attributs, présenté au 4.1.2, en comparaison avec la référence.

Traindb : ref	attributs ref		attributs robuste	
	score	écart type	score	écart type
ref	**75.3**	19.7	**76.3**	20.4
reverb	73.3	20.5	73.6	22.3
EQ2std	70.6	21.8	71.2	23.6
EQ1std	73.8	19.0	75.0	21.0
EQ-1std	77.7	15.2	78.4	17.0
EQ-2std	78.1	15.1	78.7	16.6
Moyenne	74.8	18.6	**75.5**	20.2

TAB. 5.1: Comparaison des résultats de classification avec une sélection d'attributs robuste par rapport à la référence

Une nette amélioration sur le test de référence est apportée par l'apprentissage sur l'ensemble d'attributs FEAT-SUB-FX. De plus, en moyenne sur les effets, nous obtenons des performances (75,5%) sensiblement égale au résultats de référence (75,3%), ce qui est encourageant sur les résultats à venir pour la robustesse globale du système. Dorénavant tous les classificateurs seront calculés avec l'ensemble d'attribut FEAT-SUB-FX.

5.3 SVM virtuelle

5.3.1 Présentation de la méthode

L'approche par SVM virtuelles est une des méthodes introduites au 2.5. Elle vise à incorporer les invariants en générant des exemples d'apprentissages artificiels, nommés "exemples virtuels". Elle se base sur l'hypothèse selon laquelle les vecteurs support suffisent à décrire le problème de classification. Nous motiverons cette hypothèse par une analyse des vecteurs supports au 5.4.

L'apprentissage par SVM virtuelle se décompose en trois phases (voir schémas 5.2) :

- un apprentissage classique nous donne les vecteurs supports ainsi que les fonctions de décision de chaque problème bi-classes.
- la génération d'exemples virtuels par l'application des effets à incorporer sur les vecteurs supports **uniquement** (calculés à lors de la première phase).
- un second apprentissage est réalisé sur la base d'apprentissage à laquelle les exemples virtuels ont été ajoutés.

A la fin du processus, les fonctions de décision sont plus robustes aux effets incorporés.

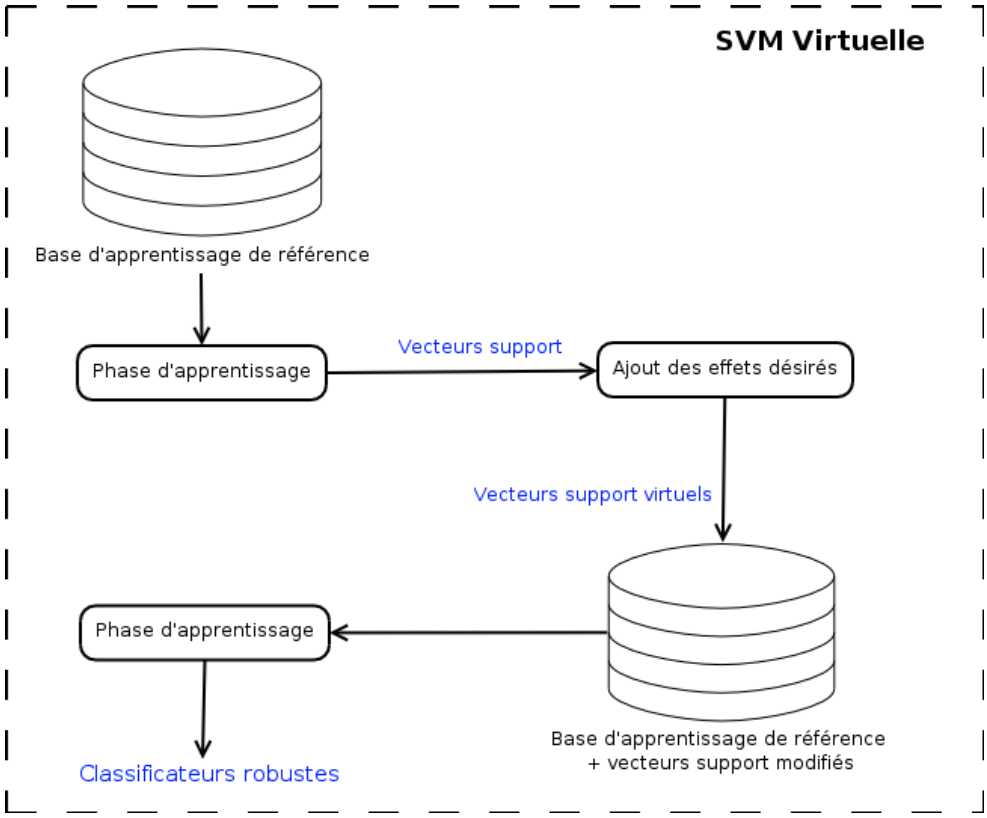

FIG. 5.2: Processus des SVM virtuelles

5.3.2 Performances des SVM virtuelles

L'approche des SVM virtuelles a été adoptée et mise en oeuvre uniquement pour les effets appartenant à SUB-FX. Les résultats sont résumés dans le tableau 5.2 et sont comparés avec ceux obtenus par le système de référence en prenant en compte l'ensemble d'attributs robustes.

Traindb : ref	ref (attributs robustes)		SVM virtuelle	
	score	écart type	score	écart type
ref	**76.3**	20.4	**78.9**	18.9
reverb	73.6	22.3	76.7	20.4
EQ2std	71.2	23.6	75.9	19.3
EQ1std	75.0	21.0	78.3	16.8
EQ-1std	78.4	17.0	80.8	20.4
EQ-2std	78.7	16.6	80.6	19.5
Moyenne	75.5	20.2	**78.5**	19.2

TAB. 5.2:

Les résultats obtenus sont bien aux delà de nos espérances car d'une part les performances sur la classification de la base de test ont augmentées de près de 2.5%, soit dans la globalité 3.5% de plus que les 75.3% du système de référence. D'autre part, les résultats en moyenne sur les bases modifiées montrent une bonne stabilité des performances du système ($78.5 \approx 78.9$).

5.3.3 Le choix des effets à incorporer

Suite aux visualisations du 4.2.1, nous testons l'hypothèse selon laquelle une des directions de la trajectoire suivie par les vecteurs supports, lors de l'ajout d'une égalisation, serait à privilégier quant à la séparabilité des

classes entre elles.

Soient les sous-ensembles d'effets suivants et les transformations associées :

- SUB-FX = $\{reverb,\ EQ2std,\ EQ1std,\ EQ-1std,\ EQ-2std\}, FX()$
- SUB-FX1 = $\{reverb,\ EQ-1std,\ EQ-2std\}, FX_1()$
- SUB-FX2 = $\{reverb,\ EQ2std,\ EQ1std\}, FX_2()$

Traindb : ref	ref + $FX(SV_{ref})$		ref + $FX_1(SV_{ref})$		ref + $FX_2(SV_{ref})$	
	score	écart type	score	écart type	score	écart type
ref	**78,9**	18,9	**75,5**	20,3	**76,7**	19,2
reverb	76,7	20,4	73,5	21,8	74,5	20,9
EQ2std	75,9	19,3	69,6	23,7	73,0	20,9
EQ1std	78,3	16,8	73,5	21,1	76,1	18,2
EQ-1std	80,8	20,4	77,2	22,1	78,9	20,7
EQ-2std	80,6	19,5	77,4	21,8	78,7	20,1
Moyenne	**78,5**	19,2	**74,4**	21,8	**76,3**	20,0

TAB. 5.3: Résultats de classificateurs dont l'apprentissage à été réalisé sur le principe des SVM virtuelles avec des ensembles d'effets à incorporer différents

Selon les résultats du tableau 5.3, il semblerait que l'incorporation des effets de SUB-FX2 amène un gain de stabilité sans pour autant réellement augmenter les performances du test de référence (76,7%). Tandis que l'ensemble SUB-FX1 n'apporte qu'une dégradation des performances. Il est très intéressant de voir que l'ajout d'un ou plusieurs effets peut ne pas amener le gain de performance espéré sans l'ajout d'effets complémentaires.

Nous pouvons alors émettre l'hypothèse suivante : lorsqu'une trajectoire, induite par l'évolution d'un paramètre d'une transformation, est "clairement visible" dans l'espace des attributs, il est alors d'un plus grand intérêt de générer des exemples virtuelles avec des paramétrisations variées voir opposées. Nous n'avons cependant aucune hypothèse quant au choix des "bornes" et de l'échantillonnage des configurations entre celles-ci.

5.4 Réduire le problème aux vecteurs supports

Cette section est une justification expérimentale de la stratégie de classification robuste élaborée par l'utilisation des SVM virtuelles.

5.4.1 Proportions de vecteurs support stables

Sachant que théoriquement, la solution des SVM ne dépend que des vecteurs supports (voir 2.4.1), il est question ici d'étudier si, après transformation, les vecteurs supports de référence restent supports dans les modèles d'apprentissage sur les bases transformées, ce qui revient à regarder leur stabilité, notion introduite au 2.4.2.

Nous effectuons le calcul pour chaque problème bi-classes, donnant ainsi une matrice des vecteurs supports stables par transformation. On réalise alors la somme du nombre de vecteurs support stables par classe que l'on compare à la somme du nombre de vecteurs support total par problème bi-classe. Nous donnons en exemple le calcul dans le cas de la reverb (voir tableau 5.4).

ref -> reverb	Pn	Gt	Bo	Ob	Co	Vl	Total
Pn	0	987	154	12	113	22	961/1328
Gt	990	0	158	3	332	10	1166/1649
Bo	173	178	0	3	14	23	309/342
Ob	10	1	3	0	12	22	31/62
Co	121	341	15	10	0	92	405/551
Vl	25	13	24	15	87	0	132/186
Total							2905/4118

TAB. 5.4: Nombre de vecteurs supports **stables** d'un apprentissage effectué sur une base réverbérée.

Remarque : Le nombre de vecteurs d'observation correspondant à un vecteur support est nettement inférieur sur le nombre total de vecteurs supports car un même vecteur d'observation peut intervenir en tant que vecteur support dans plusieurs problèmes bi-classes.

L'ensemble des données précédentes est résumé dans l'histogramme figure 5.3.

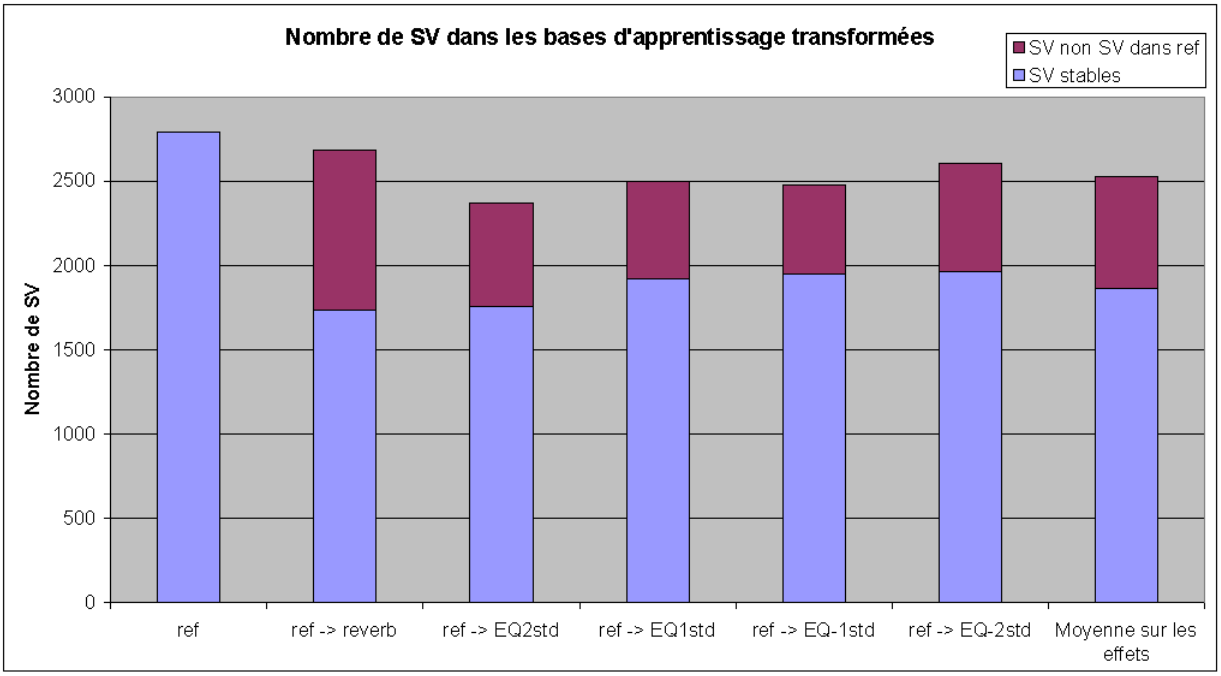

FIG. 5.3: Histogramme du nombre de vecteurs supports dans les modèles d'apprentisage sur les différentes bases.

On remarquera que pour certains effets, le nombre total de vecteurs supports est inférieur à celui de référence, traduisant une diminution de la complexité du problème. Ceci permet d'expliquer en partie l'amélioration des résultats présentés au 3.5. Le nombre important de vecteurs supports stables nous permet de justifier l'approche par SVM virtuelles.

5.4.2 Simplification

Rappelons qu'en théorie, la solution d'un problème de classification ne dépend que des vecteurs supports. Cette hypothèse forte nous permettrait de construire des bases de données uniquement constituées de vecteurs supports. Cependant, dans la pratique, il s'avère que la classification sur une base d'aprentissage ne contenant que des vecteurs supports ne donne pas d'aussi bon résultats à cause des méthodes d'optimisation contenu dans les algorithmes de calcul des SVM.

5.5 Normalisation

La plupart des méthodes de normalisation, présentées dans notre état de l'art sont difficilement transposables à notre étude car elle font références aux problèmes de classification de la parole. Elles ne sont pas toujours interprétables donc pertinents pour les instruments de musique. De plus le système en lui même permet difficilement d'introduire des changements sur le calcul des descripteurs autrement qu'en créant un nouveau descripteur, ce qui n'était pas souhaitable dans notre cas. Enfin, nous avons vu que notre système de classification comprend déjà une phase de standardisation qui normalise les vecteurs d'attributs de test (réf 2.1.1) rendant incompatible certaines méthodes de normalisation.

Nous avons déjà évoqué toutefois, par les résultats intéressants obtenus, une méthode qui consiste à appliquer systématiquement une égalisation de type *EQ-1std* ou *EQ-2std* aux données de test. Cette méthode est difficilement justifiable autrement que par les conclusions de notre analyse expérimentale. En effet, les vecteurs d'attributs semblent se "concentrer" sur des surfaces intra-classes plus petites donnant ainsi de meilleurs résultats de classification. Les capacités de généralisation de cette méthode sont à valider et la recherche d'une configuration optimale de l'effet n'a pas été effectuée.

La normalisation visant à s'abstraire des conditions d'enregistrement reste une des phases de notre système à approfondir, avec des pistes sur la modification des données de test elles-mêmes plutôt que sur la normalisation des vecteurs d'attributs.

5.6 Validation des hypothèses

Selon l'article [31], il est important de vérifier que les améliorations apportées sont toujours vraies lors d'une classification sur des données provenant de sources complètement différentes que celles de la base de test. C'est là qu'intervient la base de validation, nous permettant de mesurer les capacités de généralisation, et de vérifier certaines hypothèses sur le choix des effets (tableau 5.5).

Traindb : ref	ref		ref + $FX(SV_{ref})$		ref + $FX_1(SV_{ref})$		ref + $FX_2(SV_{ref})$	
	score	écart type	score	écart type	score	écart type	score	écart type
ref	**77,3**	17,0	**79,2**	16,0	77,9	17,0	78,0	17,1
reverb	75,9	18,6	78,0	17,2	76,9	18,0	76,8	18,4
EQ2std	73,4	21,7	78,5	16,0	73,6	22,5	76,2	18,0
EQ1std	76,9	19,0	80,4	13,1	77,2	19,1	78,9	14,7
EQ-1std	79,7	15,0	81,3	11,5	79,8	15,3	80,6	12,2
EQ-2std	79,2	14,3	80,3	13,1	79,2	14,4	79,6	13,1
Moyenne	**77,1**	17,6	**79,6**	14,5	77,5	17,7	78,3	15,6

TAB. 5.5: Comparaison des résultats de la classification de la *base de validation* sur la référence, le classificateur robuste

Les résultats montrent que la stratégie de robustesse amène effectivement une nette amélioration tant au niveau de la stabilité des performances face à des données fortement perturbées que sur la classification de la base de référence. De plus l'hypothèse selon laquelle que certain effets n'amènent un réel gain que lorsque la paramétrisation couvre un spectre assez large est vérifiée. Enfin nous observons que l'ajout d'une égalisation, aux données de la base de validation, reste toujours très profitable sur les résultats de la classification (81.3%).

Conclusions et perspectives

Conclusions

Le travail au cours de ce stage a permis d'effectuer en premier lieu une étude des traitements couramment utilisés tout au long de la chaîne de production musicale. Puis, une seconde phase a consisté à analyser l'influence de ces traitements sur le système de classification automatique présenté au chapitre 1 sur la reconnaissance des instruments de musique. Cette analyse nous a donné les directions de l'approche suivie dans l'amélioration du système de référence. Ainsi plusieurs outils d'analyse ont été développés comme la visualisation, l'écoute et l'analyse statistique des vecteurs d'attributs et notamment des vecteurs supports. Cette seconde analyse nous a permis de développer une méthode de construction d'ensemble d'attributs robustes, d'étudier les méthodes de normalisation des données de test, enfin, par une mesure complémentaire de l'importance des vecteurs supports, d'élaborer une stratégie d'incorporation des invariants en adoptant la méthode des SVM virtuelles.

Les résultats obtenus ont été très satisfaisants car non seulement le système de référence a été rendu plus robuste par la stabilité des résultats face aux traitements mais nous obtenons de plus une amélioration des résultats de référence de l'ordre de 2,5% par la méthode des SVM virtuelles. Cette amélioration peut monter jusqu'à +5,5% si une normalisation est effectuée sur les données de test. Notons qu'une intégration temporelle tardive pourrait être réalisée pour augmenter les scores en conséquence.

La stratégie mise en oeuvre se veut être globale afin d'incorporer la notion d'invariants aux effets dans le processus de classification, depuis la sélection des attributs jusqu'à la normalisation des données de test en passant par la classification elle même.

Résumé de la méthode de robustesse :

1. Analyse préalable des traitements par les méthodes de visualisation, d'écoute, statistique...
2. Constitution d'un sous-ensemble de traitements pertinents (SUB-TR) suite à l'analyse préalable.
3. Construction d'un ensemble d'attributs robuste en moyennant les rangs obtenus par une sélection d'attributs sur les bases modifiées par (SUB-TR)
4. Apprentissage par SVM virtuelle en générant des exemples "artificiels" à partir des vecteurs supports d'une pré-classification, en appliquant les traitements de (SUB-TR).
5. Normalisation des observations à classifier.
6. Indexation des sons par les classificateurs robustes

Le problème de robustesse des systèmes de classification se réduit alors à une recherche des traitements et configurations pertinents à incorporer au système. Tout en essayant de trouver un compromis entre le nombre de traitements et la complexité engendrée par la quantité de données généré. Notre méthode est donc une approche générale de la construction de systèmes de classification audio robustes qui nécessite une adaptation au problème.

Cette adaptation sera facilitée grâce aux outils d'analyse développés. Ces outils pourront permettre une étude approfondie des vecteurs d'attributs et notamment des vecteurs supports. Suivant les résultats et les *connaissance préalables* que cette phase d'analyse aura permis de dégager, une optimisation supérieure pourra être envisagée. Comme, par exemple, une régression des trajectoires des vecteurs d'attributs, permettant de suivre non plus l'approche des SVM virtuelles mais celle de la construction de noyau adapté.

Notre approche est essentiellement basée sur les classificateurs discriminatifs et nous n'avons que très peu abordés le coté probabiliste du problème. Ainsi une analyse supplémentaire pourra être effectuée sur cet axe.

Une perspective reste ouverte également sur la recherche d'une méthode de sélection systématique des effets et surtout de leur paramétrage. Il est bien entendu que malgré notre volonté d'être le plus générique possible sur les effets choisis, nous avons pu en négliger certains, notamment par rapport à d'autre problèmes de classification.

Enfin, un travail important sera la validation de la généricité de la méthode par son application à d'autres systèmes de classification, comme la segmentation parole/musique/bruit, la reconnaissance automatique du genre...

Annexe A

Principe fonctionnel et implémentation des effets/traitements

A.1 La réverbération

Implémentation

La réverbération nécessitant une implémentation relativement conséquente, nous avons choisi d'utiliser le logiciel sox 1.14 dont l'effet de salle est basé sur l'algorithme 'freeverb' [19].

Dans son principe le plus simple, l'algorithme utilise quatre filtres de Shroeder (filtre passe-tout (en) allpass filter) en séries et huit filtres Schroeder-Moorer en parallèles (voir A.1).

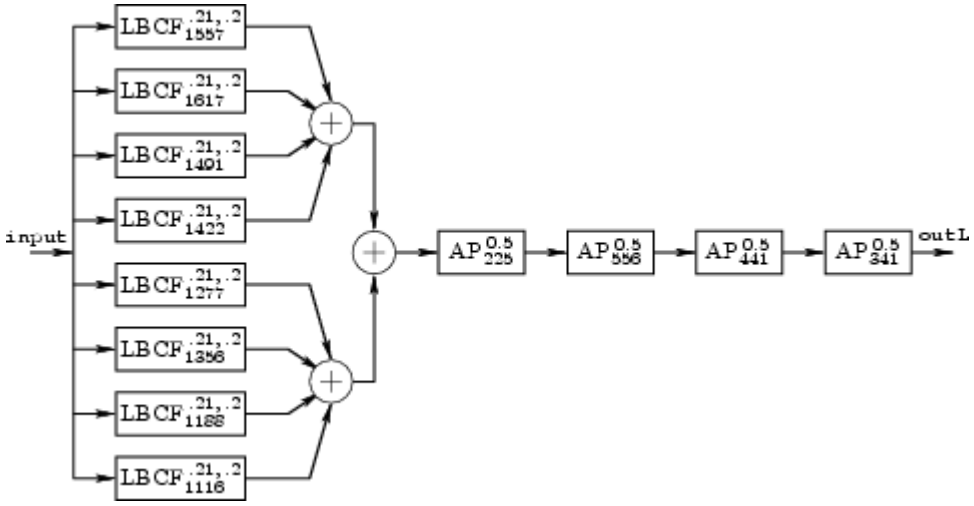

FIG. A.1: Diagramme en bloc de l'algorithme freeverb.

Les filtres Schroeder-Moorer sont des filtres en peignes à retour passe-bas ((en) lowpass-feedback-comb filter (LBCF)) et sont construits de telle sorte que la sortie de la ligne de retard est filtrée en passe-bas puis réinjectée à l'entrée de celle-ci. La fonction de transfert d'un tel filtre est :

$$H(z) = \frac{1-d}{1-dz^{-1}} \tag{A.1}$$

Lorsque le paramètre d'amortissement ((en) damping) $d = 0$, le LBCF se réduit à un simple filtre en peigne dans lequel le retour n'est pas filtré. Dans l'ensemble, la fonction de transfert d'un LBCF est :

$$LBCF_N^{f,d} \triangleq \frac{1}{1 - f\frac{1-d}{1-dz^{-1}}z^{-1}} \tag{A.2}$$

Le paramètre f est le facteur d'échelle du passe-bas ((en) feedback lowpass) et détermine principalement le temps de réverbération pour les basses fréquences. Le feedback lowpass conduit à une décroissance du temps de réverbération en fonction de la fréquence, ce se raproche le plus de la réalité.

Enfin le passe-tout noté AP_N^g est défini par :

$$AP_N^g \triangleq \frac{-g + z^{-N}}{1 - gz^{-N}} \tag{A.3}$$

A.2 La compression

Principe fonctionnel

La relation d'entrée/sortie d'un compresseur est souvent décrite par une simple courbe comme le montre la figure A.2. Elle correspond à la valeur de gain en sortie pour une valeur de gain en entrée. Pour les faibles valeurs de gain, la valeur en sortie reste inchangé (pente de la courbe = 1). A partir d'un certain seuil (T) de gain en entrée la pente de la courbe devient inférieure à 1 et est égale au facteur de compression (K).

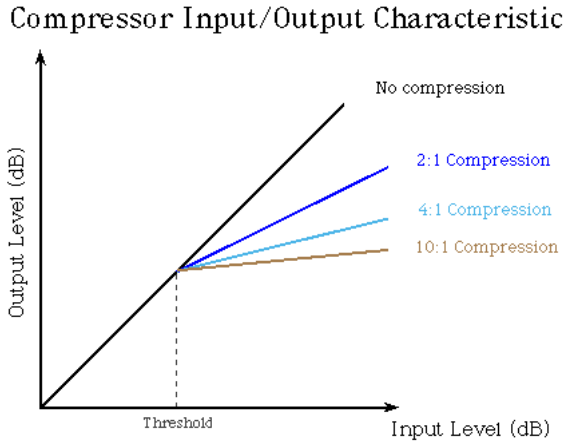

FIG. A.2: Courbe de gain entrée/sortie d'un compresseur

Pour que la compression ne soit trop brutale, un temps de transition est introduit avant que la compression ne soit complètement établie que l'on nomme temps d'attaque. Il en va de même quand le gain redescent sous le seuil avec un temps de relachement de la compression. Afin illustrer ces deux notions, la figure A.3 montre l'application d'une compression sur une sinusoïde.

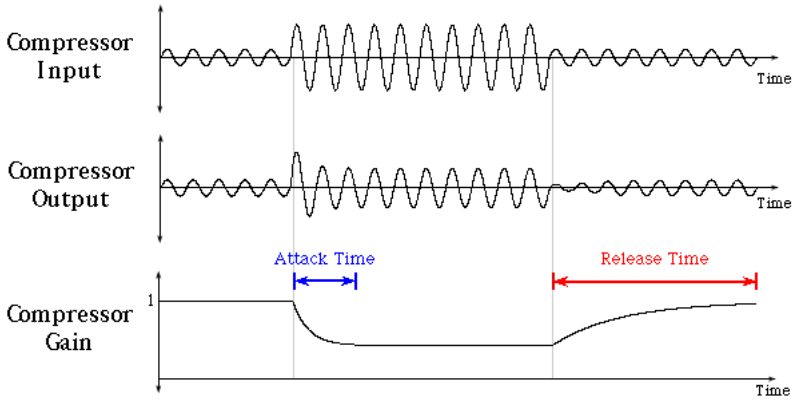

FIG. A.3: Temps d'attaque et temps de relache d'un compresseur appliqué à une sinusoïde

Implémentation

D'un point de vue implémentation, le compresseur est divisé selon les étapes suivantes :

- Détecteur RMS (Root Mean Square) par un filtrage passe-bas de la valeur absolue du signal. Le filtrage est réalisé par un filtre-α d'équation :

$$rms(n) = \alpha\, abs(x(n)) + (1-\alpha)\, rms(n-1) \tag{A.4}$$

- Calcul du nouveau gain g_{new} : lorsque celui-ci dépasse le seuil T, on applique le facteur de compression sinon $g_{new} = 1$.
- Le gain de sortie est ensuite recalculé en tenant compte du temps d'attaque α_a et temps de relâchement α_r.

$$g_{out} = \alpha_{a,r} g_{new} + (1-\alpha_{a,r}) g_{out}(n-1) \tag{A.5}$$

- Enfin le gain de sortie du compresseur est multiplié au signal d'entrée.

A.3 L'égalisation

Principe fonctionnel

Le choix de conception s'est porté sur une égalisation "graphique" dont les fréquences centrales sont par bandes d'octaves suivant le standard ISO avec un facteur de qualité Q constant. La largeur de bande Δf s'exprimant comme : $\Delta f = \frac{fc}{Q}$ celle-ci augmentera donc avec la fréquence.

Un égaliseur graphique se réduit alors à une chaîne de filtres avec des valeurs de gain par bande Enfin passe-bas et passe-haut seront des filtres "Shelving", et les passe-bandes des filtres "Peak" qui sont tout deux des filtres recursifs. A.4

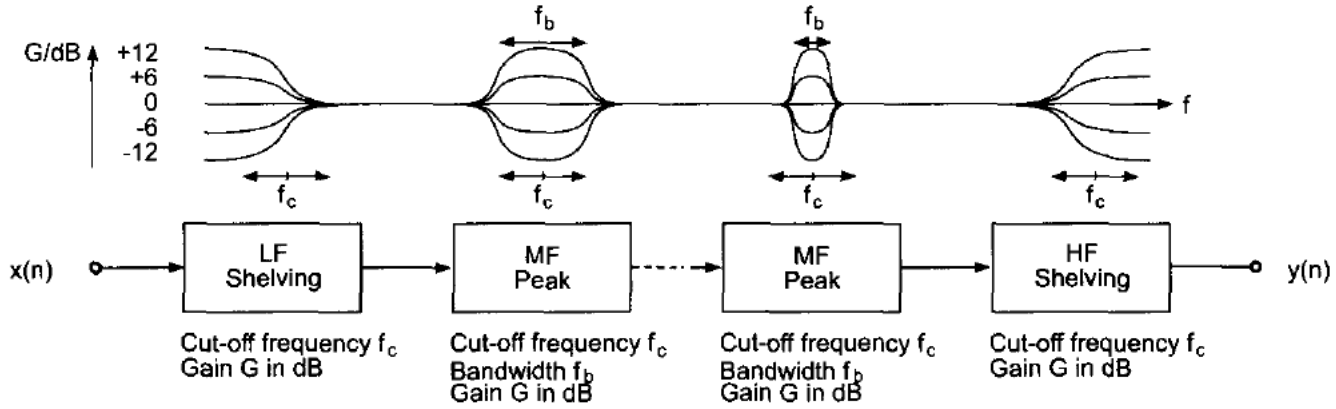

FIG. A.4: Chaine de filtre d'un égaliseur graphique.

Implémentation

L'implémentation a été réalisée à l'aide de Matlab suivant [52, 53]. Notre égaliseur est divisé en dix sous-bandes utilisant un passe-bas et un passe-haut aux extrémités ainsi que huit passe-bandes. L'approche suivie permet le calcul directe des cinq coefficient de la fonction de transfert du second ordre des différents filtres :

$$H(z) = \frac{a_0 + a_1 z^{-1} + a_2 z^{-2}}{1 + b_1 z^{-1} + b_2 z^{-2}} \tag{A.6}$$

Suivant leur nature, (passe-haut/bas, passe-bande) et la valeur du gain (positive/négative), les coefficients seront calculés à partir des expressions données dans les tableaux suivant A.5, A.6.

Avec $K = \tan(2\pi f_c / f_s)$

Avec $K = \tan(2\pi f_c / f_s)$

peak (boost $V_0 = 10^{G/20}$)				
a_0	a_1	a_2	b_1	b_2
$\frac{1+\frac{V_0}{Q_\infty}K+K^2}{1+\frac{1}{Q_\infty}K+K^2}$	$\frac{2(K^2-1)}{1+\frac{1}{Q_\infty}K+K^2}$	$\frac{1-\frac{V_0}{Q_\infty}K+K^2}{1+\frac{1}{Q_\infty}K+K^2}$	$\frac{2(K^2-1)}{1+\frac{1}{Q_\infty}K+K^2}$	$\frac{1-\frac{1}{Q_\infty}K+K^2}{1+\frac{1}{Q_\infty}K+K^2}$
peak (cut $V_0 = 10^{-G/20}$)				
a_0	a_1	a_2	b_1	b_2
$\frac{1+\frac{1}{Q_\infty}K+K^2}{1+\frac{V_0}{Q_\infty}K+K^2}$	$\frac{2(K^2-1)}{1+\frac{V_0}{Q_\infty}K+K^2}$	$\frac{1-\frac{1}{Q_\infty}K+K^2}{1+\frac{V_0}{Q_\infty}K+K^2}$	$\frac{2(K^2-1)}{1+\frac{V_0}{Q_\infty}K+K^2}$	$\frac{1-\frac{V_0}{Q_\infty}K+K^2}{1+\frac{V_0}{Q_\infty}K+K^2}$

FIG. A.5: Coefficients de la fonction de transfert du filtre Peak

low-frequency shelving (boost $V_0 = 10^{G/20}$)				
a_0	a_1	a_2	b_1	b_2
$\frac{1+\sqrt{2V_0}K+V_0K^2}{1+\sqrt{2}K+K^2}$	$\frac{2(V_0K^2-1)}{1+\sqrt{2}K+K^2}$	$\frac{1-\sqrt{2V_0}K+V_0K^2}{1+\sqrt{2}K+K^2}$	$\frac{2(K^2-1)}{1+\sqrt{2}K+K^2}$	$\frac{1-\sqrt{2}K+K^2}{1+\sqrt{2}K+K^2}$
low-frequency shelving (cut $V_0 = 10^{-G/20}$)				
a_0	a_1	a_2	b_1	b_2
$\frac{1+\sqrt{2}K+K^2}{1+\sqrt{2V_0}K+V_0K^2}$	$\frac{2(K^2-1)}{1+\sqrt{2V_0}K+V_0K^2}$	$\frac{1-\sqrt{2}K+K^2}{1+\sqrt{2V_0}K+V_0K^2}$	$\frac{2(V_0K^2-1)}{1+\sqrt{2V_0}K+V_0K^2}$	$\frac{1-\sqrt{2V_0}K+V_0K^2}{1+\sqrt{2V_0}K+V_0K^2}$
high-frequency shelving (boost $V_0 = 10^{G/20}$)				
a_0	a_1	a_2	b_1	b_2
$\frac{V_0+\sqrt{2V_0}K+K^2}{1+\sqrt{2}K+K^2}$	$\frac{2(K^2-V_0)}{1+\sqrt{2}K+K^2}$	$\frac{V_0-\sqrt{2V_0}K+K^2}{1+\sqrt{2}K+K^2}$	$\frac{2(K^2-1)}{1+\sqrt{2}K+K^2}$	$\frac{1-\sqrt{2}K+K^2}{1+\sqrt{2}K+K^2}$
high-frequency shelving (cut $V_0 = 10^{-G/20}$)				
a_0	a_1	a_2	b_1	b_2
$\frac{1+\sqrt{2}K+K^2}{V_0+\sqrt{2V_0}K+K^2}$	$\frac{2(K^2-1)}{V_0+\sqrt{2V_0}K+K^2}$	$\frac{1-\sqrt{2}K+K^2}{V_0+\sqrt{2V_0}K+K^2}$	$\frac{2(K^2/V_0-1)}{1+\sqrt{2/V_0}K+K^2/V_0}$	$\frac{1-\sqrt{2/V_0}K+K^2/V_0}{1+\sqrt{2/V_0}K+K^2/V_0}$

FIG. A.6: Coefficients de la fonction de transfert du filtre Shelving

A.4 Le codage MP3

Principe fonctionnel

Banc de filtres Les 32 filtres passe-bande de cette fonction décomposent en fréquence le signal audio en autant de sous-bandes. Les 32 filtres sont similaires et ont tous une bande passante de $fe/(2*32) = 689Hz$ pour une fréquence d'échantillonnage de fe=44.1kHz). Cette décomposition est réversible et la fonction inverse est utilisée dans le décodeur pour reconstituer le signal audio.

FFT et modèle psycho-acoustique La FFT est réalisée sur 1024 points, ce qui donne une résolution de 43Hz avec fe=44.1kHz. La fonction "Modèle psycho-acoustique" utilise le contenu spectral actuel du signal audio fourni par la FFT et le seuil d'audibilité de l'ouïe pour :

- identifier les "masqueurs"
- et en déduire la courbe du seuil de masquage

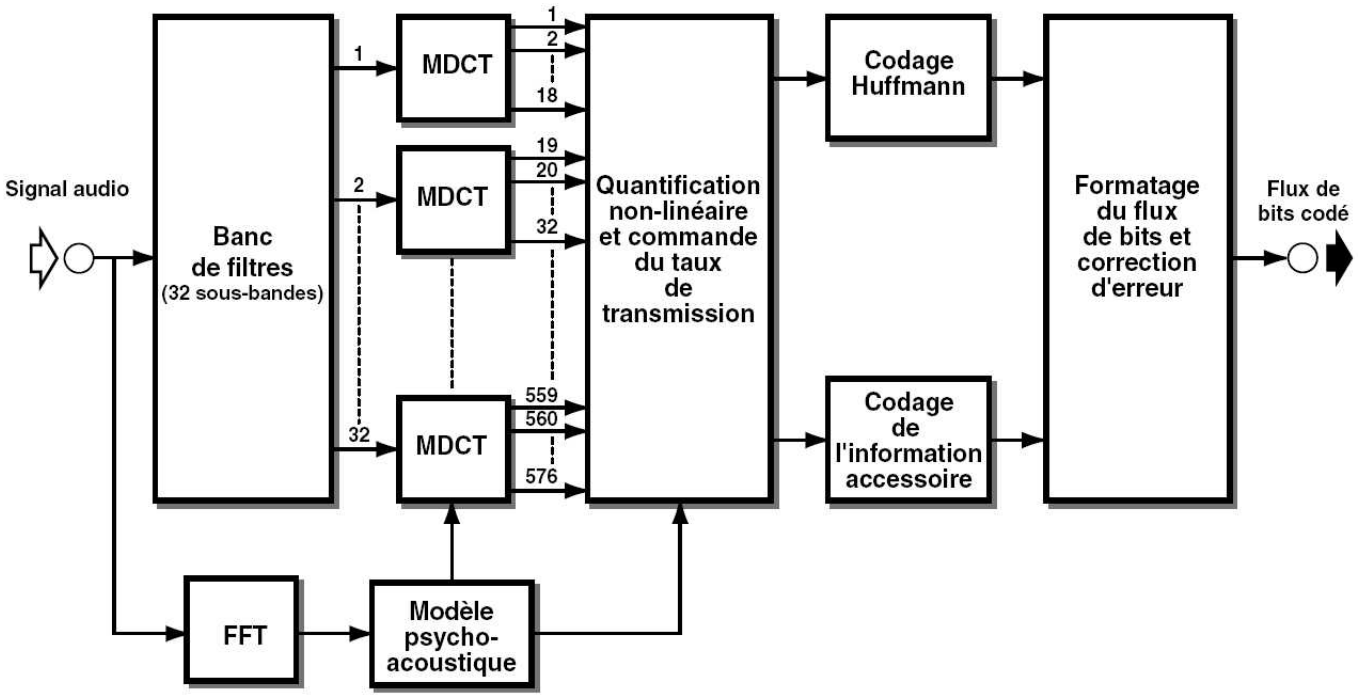

FIG. A.7:

MDCT L'acronyme MDCT signifie : "Modified Discrete Cosinus Transform". Il s'agit en fait de filtres passe bande réalisés à base d'algorithmes permettant de réduire la quantité de calculs. A ce niveau la quantité d'information reste inchangée.

Quantification non linéaire Cette fonction réalise en grande partie la compression du débit binaire. Le codage MPEG utilise simultanément 2 méthodes :

- La suppression complète des composantes totalement masquées. Il s'agit de celles situées sous le seuil de masquage calculé par la fonction "FFT et Modèle psycho-acoustique".
- Les autres composantes ne peuvent évidemment pas être éliminées (il n'y aurait plus de son !) mais on constate qu'un bruit plus important peut être toléré dans la bande de fréquence critique concernée. Le codeur MPEG exploite ce constat. En effet, le rapport "signal/bruit de quantification" augmente de 6dB environ par bit ajouté dans la représentation des nombres. Ainsi, si on peut augmenter le bruit de quantification tout en restant inaudible, on peut se permettre de réduire le nombre de bits représentatifs des nombres des signaux numériques. Ce traitement est effectué pour chaque bande critique en fonction du seuil de masquage fourni par la fonction "FFT et modèle psycho-acoustique" : le bruit de quantification doit toujours rester inférieur au seuil de masquage, tout en s'en rapprochant le plus possible : NMR < 0dB

Codage de Huffmann Il s'agit d'un algorithme de compression sans perte. La compression Huffman consiste à coder les données selon leur récurrence statistique. Plus la valeur à coder est courante, plus le code qui lui est associé est court. Au moment de la décompression, ces codes de longueurs variables sont confrontés à une table de correspondance qui restitue leur valeur initiale. Cette méthode de compression, qui n'est pas spécifique au MP3, assure à elle seule une compression de l'ordre de 20 à 25

A.5 Le sous-échantillonnage

Implémentation

Nous ne réalisons pas réellement un réechantillonnage mais uniquement la partie filtrage passe bas anti-aliasing en gardant la fréquence d'échantillonnage à 32kHz. Les coefficients de la fonction de transfert du filtre A.6 sont directement calculé suivant les expréssions du tableau A.1.

$K = \tan(2\pi\frac{f_{cut}}{f_s})$

b_1	b_2	b_3	a_1	a_2	a_3
$\frac{K^2}{1+\sqrt{2}K+K^2}$	$\frac{2K^2}{1+\sqrt{2}K+K^2}$	$\frac{K^2}{1+\sqrt{2}K+K^2}$	1	$\frac{2(K^2-1)}{1+\sqrt{2}K+K^2}$	$\frac{1-\sqrt{2}K+K^2)}{1+\sqrt{2}K+K^2}$

TAB. A.1: Coefficients de la fonction de transfert du filtre passe bas avec fs=fréquence d'échantillonnage et fcut=fréquence de coupure du filtre.

Annexe B

Selection des attributs

B.1 Attribut sélectionnés sur chacunes des bases

	ref	comp	32kbps	64kbps	low8k	reverb	EQ2std	EQ1std	EQ-1std	EQ-2std
1	CQ4-4	CQ4-4	CQ4-4	CQ4-4	CQ4-4	CQ4-4	CQ4-4	CQ4-4	CQ4-4	CQ4-4
2	CQ3-4	CQ3-4	CQ3-4	CQ3-4	CQ3-4	CQ3-4	CQ3-4	CQ3-4	CQ3-4	CQ3-4
3	CQ2-4	CQ2-4	CQ2-4	CQ2-4	CQ2-4	CQ2-4	CQ2-4	CQ2-4	CQ2-4	CQ2-4
4	CQ1-4	CQ1-4	CQ1-4	CQ1-4	CQ1-4	CQ1-4	CQ1-4	CQ1-4	CQ1-4	CQ1-4
5	CP3-3	CP3-3	*Z-1*	CP3-3	*Z-1*	*C3-3*	*H-3*	*Z-1*	*Z-1*	CP3-3
6	Z-1	Z-1	*Sc-1*	Z-1	*C3-3*	*CP3-3*	*dH-1*	*dH-1*	*CP3-3*	*C3-3*
7	C3-3	C3-3	C3-3	C3-3	*dH-1*	*dH-1*	*Z-1*	*H-3*	C3-3	*Z-1*
8	dH-1	dH-1	dH-1	dH-1	*CP3-3*	*Z-1*	*C3-3*	*C3-3*	dH-1	dH-1
9	Ld-2	Ld-2	*Sh-1*	Ld-2	*Sc-1*	*H-3*	*CP3-3*	*CP3-3*	*H-3*	Ld-2
10	dH-6	*H-3*	*CP3-3*	*H-3*	*Ld-2*	*Ld-2*	*Ld-2*	*Ld-2*	*Ld-2*	*H-3*
11	H-3	*dH-6*	*Ld-2*	*dH-6*	*Sh-1*	*dH-6*	*CP3-2*	*CP3-2*	*dH-6*	*dH-6*
12	Sc-1	Sc-1	*CP3-2*	Sc-1	*dH-6*	Sc-1	*dH-6*	*dH-6*	*CP3-2*	*CP3-2*
13	CP3-2	CP3-2	*H-3*	CP3-2	*H-3*	CP3-2	*dH-5*	*dH-5*	*Sc-1*	*Sc-1*
14	dH-5	dH-5	*dH-6*	dH-5	*CP3-2*	dH-5	*CQ2-3*	*CQ2-3*	dH-5	*Sd-1*
15	Sh-1	Sh-1	*dH-5*	Sh-1	*dH-5*	Sh-1	*CQ4-2*	*Sc-1*	*Sd-1*	*dH-5*
16	CQ2-3	CQ2-3	**C3-4**	CQ2-3	*Fc-1*	*Fc-1*	*Sw-1*	*CQ4-2*	*Sh-1*	*Sh-1*
17	Fc-1	Fc-1	Fc-1	Fc-1	*CQ2-3*	*CQ2-3*	*Ld-3*	*Sh-1*	*CQ2-3*	Fc-1
18	Si-14	Si-14	*Sf-1*	Si-14	*Sf-1*	*Ld-3*	*CQ3-2*	*CQ3-2*	*CQ4-2*	*CQ2-3*
19	CQ4-2	CQ4-2	*C3-2*	CQ4-2	*Si-14*	CQ4-2	**CQ1-3**	*Ld-3*	*CQ3-2*	CQ4-2
20	Ld-3	Ld-3	*Si-14*	Ld-3	*CQ4-2*	*CQ3-2*	**So-1**	*Sw-1*	*Fc-1*	*CQ3-2*
21	CQ3-2	CQ3-2	*Ld-3*	CQ3-2	**C3-4**	*Sf-1*	*C3-2*	*CQ2-2*	*Ld-3*	*Si-14*
22	Sf-1	Sf-1	*CQ4-2*	Sf-1	*Ld-3*	*Ld-16*	*Sh-1*	*C3-2*	*CQ2-2*	*Ld-3*
23	CQ2-2	CQ2-2	*CQ3-2*	CQ2-2	*CQ3-2*	*dH-3*	**C3-5**	**So-1**	*Sf-1*	CQ2-2
24	Sd-1	Sd-1	**Sa-1**	Sd-1	*C3-2*	*Ld-15*	*Sc-1*	**C3-5**	*Si-14*	*Sf-1*
25	Ld-16	Ld-16	*Sd-1*	Ld-16	*CQ2-2*	*CQ2-2*	*CQ2-2*	*Sd-1*	*Sw-1*	Ld-16
26	Ld-15	Ld-15	**CP3-5**	Ld-15	*Sd-1*	*Ld-17*	**ASF-23**	**CQ1-3**	*dH-3*	Ld-15
27	Ld-17	Ld-17	*CQ2-2*	Ld-17	**C3-5**	*Sw-1*	*dH-3*	*dH-3*	*Ld-16*	*dH-3*
28	dH-3	dH-3	dH-3	dH-3	**Sa-1**	*Si-14*	**CP3-5**	**CP3-5**	*Ld-17*	*Ld-17*
29	Sw-1	Sw-1	*Ld-16*	Sw-1	*Ld-15*	*C3-2*	**CP3-7**	*Ld-16*	*Ld-15*	**CP3-7**
30	C3-2	C3-2	*Ld-15*	**CP3-5**	*Ld-16*	**CP3-7**	*Ld-16*	**CP3-7**	**CP3-5**	*Sw-1*

TAB. B.1: Resultats de l'algorithme de sélection des attributs sur les bases d'apprentissage transformées.

B.2 Selection d'attributs robuste

Sur l'ensemble des bases d'apprentissage

rang	Attributs	Rang moyen
1	CQ4-4	1
2	CQ3-4	2
3	CQ2-4	3
4	CQ1-4	4
5	*Z-1*	6
6	*C3-3*	6.8
7	*CP3-3*	6.8
8	*dH-1*	7.4
9	*H-3*	9.7
10	*Ld-2*	9.7
11	*dH-6*	11.5
12	*CP3-2*	12.4
13	*Sc-1*	12.8
14	dH-5	14.1
15	Sh-1	15.1
16	CQ2-3	17.6
17	*CQ4-2*	18.6
18	*Fc-1*	19.9
19	*Ld-3*	20
20	*CQ3-2*	20.4
21	*Si-14*	22.8
22	Sf-1	23.2
23	CQ2-2	23.7
24	Sd-1	23.9
25	*Sw-1*	26.7
26	*Ld-16*	26.7
27	*C3-2*	26.8
28	dH-3	27.3
29	*Ld-15*	27.8
30	**C3-4**	28.5
31	*Ld-17*	28.7
32	**C3-5**	29.1
33	**So-1**	29.1
34	**CQ1-3**	29.3
35	**CP3-5**	29.7
36	**Sa-1**	30
37	**CP3-7**	30.4
38	**ASF-23**	30.5

TAB. B.2: Rang moyen des attributs sur les bases d'apprentissages

Sur les bases d'apprentissage ref et SUB-FX

Rang	Attributs	Rang moyen
1	CQ4-4	1.00
2	CQ3-4	2.00
3	CQ2-4	3.00
4	CQ1-4	4.00
5	*Z-1*	6.33
6	*CP3-3*	6.67
7	C3-3	6.83
8	dH-1	7.17
9	*H-3*	8.50
10	*Ld-2*	9.67
11	*dH-6*	11.17
12	*CP3-2*	12.00
13	*dH-5*	13.83
14	*Sc-1*	14.83
15	*CQ2-3*	16.00
16	*Sh-1*	16.83
17	*CQ4-2*	17.67
18	*CQ3-2*	19.33
19	*Ld-3*	19.50
20	*Fc-1*	22.00
21	*CQ2-2*	23.17
22	*Sd-1*	23.33
23	*Sw-1*	24.50
24	*Sf-1*	25.33
25	*Si-14*	25.50
26	*dH-3*	26.33
27	*Ld-16*	26.33
28	*C3-2*	27.33
29	**So-1**	27.83
30	*Ld-15*	27.83
31	**CQ1-3**	28.17
32	**C3-5**	28.50
33	*Ld-17*	28.50
34	**CP3-5**	29.83
35	**CP3-7**	30.00
36	**ASF-23**	30.17

TAB. B.3: Rang moyen des attributs sur les bases d'apprentissages du sous ensemble SUB-FX.

Bibliographie

[1] A. Acero and RM Stern. Robust speech recognition by normalization of the acoustic space. *Acoustics, Speech, and Signal Processing, 1991. ICASSP-91., 1991 International Conference on*, pages 893–896, 1991.

[2] Avrim L. Blum and Pat Langley. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97(1-2) :245–271, 1997.

[3] J. Brown. Computer identification of musical instruments using pattern recognition, 1997.

[4] Pramod Chandraiah. Specification and design of a MP3 audio decoder /. Master's thesis, University of California, Irvine, 2005.

[5] C. C. Chang and C. J. Lin. *LIBSVM : a library for support vector machines*. Online, 2001.

[6] Minkook Cho and Hyeyoung Park. A robust SVM design for multi-class classification. In Shichao Zhang and Ray Jarvis, editors, *Australian Conference on Artificial Intelligence*, volume 3809 of *Lecture Notes in Computer Science*, pages 1335–1338. Springer, 2005.

[7] A. Eronen. Automatic musical instrument recognition, 2001.

[8] A. Eronen. Musical instrument recognition using ica-based transform of features and discriminatively trained hmms. *Signal Processing and Its Applications, 2003. Proceedings. Seventh International Symposium on*, 2 :133–136 vol.2, July 2003.

[9] Slim Essid. *Classification automatique des signaux audio-fréquences : reconnaissance des instruments de musique*. PhD thesis, Telecom Paristech, 2005.

[10] SoX Sound eXchange. sox-14.0. `http://sox.sourceforge.net/`.

[11] Ichiro Fujinaga. Machine recognition of timbre using steady-state tone of acoustic musical instruments, March 19 1998.

[12] Diego Giuliani, Matteo Gerosa, and Fabio Brugnara. Improved automatic speech recognition through speaker normalization. *Computer Speech & Language*, 20(1) :107–123, 2006.

[13] M. Goto. Rwc music database : Popular, classical, and jazz music databases, 2002.

[14] Evandro B. Gouvea and Richard M. Stern. Speaker normalization through formant-based warping of the frequency scale. In *Proc. Eurospeech '97*, pages 1139–1142, Rhodes, Greece, September 1997.

[15] A. B. A. Graf, A. J. Smola, and S. Borer. Classification in a normalized feature space using support vector machines. *IEEE-NN*, 14 :597–605, May 2003.

[16] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3 :1157–1182, 2003.

[17] Trevor Hastie and Robert Tibshirani. Classification by pairwise coupling. In Michael I. Jordan, Michael J. Kearns, and Sara A. Solla, editors, *Advances in Neural Information Processing Systems*, volume 10. The MIT Press, 1998.

[18] Andrew Oliver Hatch. *Kernel Optimization for Support Vector Machines : Application to Speaker Verification*. PhD thesis, EECS Department, University of California, Berkeley, Dec 2006.

[19] Julius Orion Smith III. *Physical audio signal processing for virtual musical instruments and audio effects*. 2008.

[20] I. Kaminskyj and A. Materka. Automatic source identification of monophonic musical instrument sounds. In *IEEE International Conference on Neural Networks (ICNN'95)*, volume 1, pages 189–194, Perth, Western Australia, November-December 1995. IEEE.

[21] Ian Kaminskyj and Tadeusz Czaszejko. Automatic recognition of isolated monophonic musical instrument sounds using k NNC. *J. Intell. Inf. Syst*, 24(2-3) :199–221, 2005.

[22] Ron Kohavi and George John. Wrappers for feature subset selection. *Artificial Intelligence*, 97 :273–324, 1997.

[23] Bożena Kostek and Andrzej Czyżewski. Automatic recognition of musical instrument sounds - further developments. 2001. 110th Audio Eng. Soc. Convention, 110th Audio Eng. Soc. Convention.

[24] Fernando De la Torre Frade and Oriol Vinyals. Parameterized kernels for support vector machine classification. *International Conference on Computer Vision Theory and Applications*, pages 207–213, March 2007. associated project Component Analysis for Data Analysis.

[25] LAME Ain't an Mp3 Encoder. Lame 32bits version 3.97. `http://lame.sourceforge.net/`.

[26] Christopher James Langmead. Sound analysis, comparisan and modification based on a perceptual model of timbre. In *Intenational Computer Music Conference*, 1995.

[27] Quoc V. Le, Alex J. Smola, and Thomas Gärtner. Simpler knowledge-based support vector machines. In *ICML '06 : Proceedings of the 23rd international conference on Machine learning*, pages 521–528, New York, NY, USA, 2006. ACM.

[28] Jonghyun Lee and Joohwan Chun. Musical instruments recognition using hidden markov model. *Signals, Systems and Computers, 2002. Conference Record of the Thirty-Sixth Asilomar Conference on*, 1 :196–199 vol.1, Nov. 2002.

[29] Tao Li and M. Ogihara. Music genre classification with taxonomy. *Acoustics, Speech, and Signal Processing, 2005. Proceedings. (ICASSP '05). IEEE International Conference on*, 5 :v/197–v/200 Vol. 5, March 2005.

[30] Huan Liu and Hiroshi Motada, editors. *Feature extraction, construction and selection : A data mining perspective*. Kluwer Academic Publishers, Norwell, MA, 1998.

[31] Arie Livshin and Xavier Rodet. The importance of cross database evaluation in musical instrument sound classification : A critical approach. In *ISMIR*, 2003.

[32] Lie Lu, Hao Jiang, and HongJiang Zhang. A robust audio classification and segmentation method. In *MULTIMEDIA '01 : Proceedings of the ninth ACM international conference on Multimedia*, pages 203–211, New York, NY, USA, 2001. ACM.

[33] J. Marques and P. Moreno. A study of musical instrument classification using gaussian mixture models and support vector machines, 1999.

[34] Keith Dana Martin. *Sound-source recognition : a theory and computational model*. PhD thesis, Massachusetts Institute of Technology, Dept. of Electrical Engineering and Computer Science, 1999.

[35] Denis Mercier. *Le livre des techniques du son, tome 1 : Notions fondamentales*. Dunod, 2002.

[36] G. Nataneli and P. Faloutsos. Robust classification of strokes with SVM and grouping. In *Advances in Visual Computing*, pages I : 76–87, 2007.

[37] Davis Pan. A tutorial on MPEG/audio compression. *IEEE MultiMedia*, 2(2) :60–74, 1995.

[38] Davis Yen Pan. Digital audio compression. *Digital Technical Journal of Digital Equipment Corporation*, 5(2) :28–33 (or 28–40 ? ?), Spring 1993.

[39] Geoffroy Peeters. A large set of audio features for sound description (similarity and classification). Technical report, 2004.

[40] Geoffroy Peeters, Amaury La Burthe, and Xavier Rodet. Toward automatic music audio summary generation from signal analysis. In *ISMIR*, 2002.

[41] Geoffroy Peeters and Xavier Rodet. Hierarchical gaussian tree with inertia ratio maximization for the classification of large musical instruments databases. In *Proc. of the 6th Int. Conf. on Digital Audio Effects*, London, UK, 2003.

[42] D.A. Reynolds and R.C. Rose. Robust text-independent speaker identification using gaussian mixture speaker models. *Speech and Audio Processing, IEEE Transactions on*, 3(1) :72–83, Jan 1995.

[43] B. Schoelkopf and A. J. Smola. *Learning with Kernels*. The MIT Press, Cambridge, MA, 2002.

[44] Ok Keun Shin. A vector-quantizer based method of speaker normalization. In *ACIS-ICIS*, pages 402–407. IEEE Computer Society, 2005.

[45] S. Shlien. Guide to mpeg-1 audio standard. *Broadcasting, IEEE Transactions on*, pages 206–218, 1994.

[46] P. Simard, B. Victorri, Y. LeCun, and J. Denker. Tangent prop - A formalism for specifying selected invariances in an adaptive network. In J. E. Moody, S. J. Hanson, and R. P. Lippmann, editors, *Advances in Neural Information Processing Systems 4*, pages 895–903, San Mateo, CA, 1992. Morgan Kaufmann.

[47] Christian Simmermacher, Da Deng, and Stephen Cranefield. Feature analysis and classification of classical musical instruments : An empirical study. In Petra Perner, editor, *Industrial Conference on Data Mining*, volume 4065 of *Lecture Notes in Computer Science*, pages 444–458. Springer, 2006.

[48] Carl Staelin. Parameter selection for support vector machines. Technical Report HPL-2002-354R1, Hewlett Packard Laboratories, November 19 2003.

[49] S. Theodoridis and K. Koutroumbas. *Pattern Recognition, 3rd Edition*. Academic Press, 2006.

[50] Christian J. Walder and Brian C. Lovell. Homogenised virtual support vector machines. 2005.

[51] L. Xu, K. Crammer, and D. Schuurmans. Robust support vector machine training via convex outlier ablation. *Proceedings National Conference on Artificial Intelligence (AAAI-06)*, 2006.

[52] Udo Zolzer. Audio processing systems, January 03 1997.

[53] Udo Zölzer. *DAFX-Digital Audio Effects*. Wiley, pub-WILEY :adr, 2002.