Louis-Lumière
école nationale supérieure

Memoire de fin d'études - Master Son

Vers une classification automatique des sons dans la post-production audiovisuelle.

Ajustement du modèle BEATs vers l'ontologie du monteur son.

Léo-Polde POULALION

Directeur interne : Laurent MILLOT
Directeur externe : Modan TAILLEUR
Référent académique universitaire : Laurent MILLOT
Coordinateur des mémoires : Corsin VOGEL
Examinateur : Thibault NOIROT

Paris, 2025

Mes remerciements s'adressent tout d'abord à mes directeurs de mémoire, sans qui ce travail n'aurait pas été rendu possible, Laurent Millot pour l'énergie consacrée aux multiples relectures, Modan Tailleur pour son aide avisée et Corsin Vogel pour sa rigueur méthodologique.

Je remercie également Rodrigo Sacic d'avoir pris le temps de m'expliquer son métier afin de cerner les usages et les limites de mon travail. Aussi, Steve Pecile pour m'avoir gracieusement prêté une licence *SoundminerProV6* afin de mener à bien ma partie pratique. Florent Fajole pour les ressources bibliographiques et sa fiabilité.

Ma gratitude envers l'équipe pédagogique de Louis-Lumière :
Merci à Franck Jouanny pour avoir tenu la barre du radeau section Son. Merci Alan d'être aussi génial et Sylvain, pour avoir été plus qu'un professeur de son.

Je remercie les membres du collectif *ill* dont le talent n'a de cesse de m'impressionner, pour les moments partagés. Également Séverin, pour les discussions éclairées, Mickaël pour tout ce que tu incarnes et l'ensemble de mes camarades de Son 3 pour m'avoir supporté au quotidien.

Je ne pourrais poursuivre, sans remercier Ilan, Loïs et le *GAEC Le Mas* pour m'avoir ouvert leur porte quand j'en avais le plus besoin.

Enfin, j'ai une pensée pour ma famille, je remercie mon frère sans trop savoir pourquoi, mon père pour avoir cru en moi et ma mère d'avoir toujours été là.

Ce mémoire signe aussi la fin de ces trois années à Louis-Lumière, trois années dont je ne garderai, en mon cœur et pour toujours, que de bons souvenirs. À tous ceux que j'oublie, Merci.

Résumé

Avec les récents progrès en apprentissage automatique et en traitement des signaux audio, de nouveaux outils émergent pour assister les professionnels de la *post-production* audiovisuelle. Nous traiterons en particulier du cas du monteur son et de la gestion de sa sonothèque, le rangement d'autant de données est d'une grande importance pour choisir au mieux les sons qui servent la narration. En ce sens, l'indexation - phase durant laquelle les fichiers sont annotés et organisés dans la sonothèque - est nécessaire à une navigation fluide et n'est pas à prendre à la légère.

Depuis quelques années, la proposition de la norme UCS (Universal Category System), induit une classification des sources sonores et une convention de nommage adoptée par une majorité de professionnels de l'audio. Dans ce travail, nous nous demanderons s'il est possible d'automatiser une partie du processus d'indexation, en utilisant un algorithme pour classer les sons dans des catégories UCS.

Pour explorer cette problématique, nous avons choisi d'ajuster les connaissances du classifieur audio BEATs pour l'adapter à une classification selon l'ontologie UCS. À cette fin, un jeu de données spécifique a été construit à partir de sons de nourriture et de cuisine. Le modèle ajusté atteint une précision de **87,65%** sur le jeu d'évaluation. Ces résultats suggèrent la faisabilité d'un outil d'aide à l'indexation basé sur cette approche, et dont les performances pourraient s'étendre à une classification plus détaillée.

Mots-clés : Ontologie du sonore, Sémantique du sonore, Classification des sons environnementaux, Post-production audiovisuelle, Écoute causale, UCS, Audioset, Apprentissage automatique, Architecture Transformeur, BEATs

Abstract

With recent advances in machine learning and audio signal processing, new tools are emerging to assist professionals working in audiovisual post-production. This study focuses specifically on the role of the sound editor and the management of their sound library. Organizing such a large volume of data is crucial to selecting sounds that best serve the narrative. In this context, the indexing phase — during which audio files are labeled and organized — is essential to ensure fluid and efficient navigation within the library and should not be taken lightly.

In recent years, the Universal Category System (UCS) has been proposed as a standardized framework for sound source classification and naming conventions, and has since been adopted by a majority of audio professionals. In this work, we investigate whether part of the indexing process can be automated, using an algorithm to classify sounds into UCS categories.

To explore this question, we chose to fine-tune the BEATs audio classifier so it could operate within the UCS ontology. For this purpose, a specific dataset was constructed using sounds related to food and cooking. The adapted model achieved an accuracy of **87.65%** on the evaluation set. These results suggest the feasibility of a UCS-based indexing support tool, with potential to scale up to finer-grained classification tasks.

Keywords : Sound ontology, Sound semantics, Classification of environmental sounds, Audiovisual post-production, Causal listening, UCS, Audioset, Machine learning, Transformer architecture, BEATs

Table des matières

Table des figures

Introduction.

“ La catégorisation n’est pas une question à prendre à la légère. Il n’y a rien de plus fondamental que la catégorisation pour notre pensée. ”[1]

- George Lakoff (1986) -

Le procédé de catégorisation qu’il soit conscient ou inconscient, est tout à fait naturel, il concerne notre faculté de ranger les objets dans des boîtes et ces boîtes dans des plus grandes, de sorte qu’à la fin nous ayons construit une hiérarchisation des choses de la vie. Ce processus de classification est essentiel et même souhaitable puisqu’il permet une meilleure compréhension de notre environnement. Lorsque nous devons faire un choix, nous nous basons sur nos expériences de vies passées à partir desquelles nous avons appris à tirer des conclusions. Le principe *d’inférence*[2] permet ensuite de généraliser les expériences personnelles à une situation particulière, ce qui nous oriente sur la voie à suivre. C’est par *l’apprentissage* et la confrontation à de nouvelles expériences que nous apprenons à vivre (heureux ?).

Si le cerveau humain est capable de catégoriser toutes sortes d’objets et situations, nous nous restreindrons, dans ce mémoire, au domaine du son, et plus particulièrement des sons environnementaux, ceux-ci désignent l’ensemble des sonorités susceptibles d’êtres rencontrées au quotidien, qu’elles soient d’origine humaine ou naturelle. La présente recherche s’intéressera expressément à la perception du monteur son dans la post-production audiovisuelle. Lorsqu’il travaille, le *monteur son* peut-être amené à explorer des banques de sons afin d’en intégrer des extraits dans son logiciel de montage. Des logiciels de gestion de *sonothèques* comme *Soundly* ou *Soundminer*, mettent à disposition du monteur son pléthore d’outils pour filtrer et faciliter ces recherche, qui ne sont plus basées uniquement sur les noms de fichiers, si bien que l’utilisation de mots-clés descripteurs s’est au fil des ans imposée comme une normalité.

Le problème principal de la recherche par mots-clés a longtemps été l’absence de normes globales et internationales encadrant cette pratique. En France, des tenta-

1. "Categorization is not a matter to be taken lightly. There is nothing more basic than categorization to our thought"
2. cf. Glossaire D.1

tives d'uniformisation du processus d'*indexation*[3] en sonothèque ont été proposées : Théo Serror (2018) dans son mémoire de Master de l'ENS Louis-Lumière porte son attention sur la nomenclature et la proximité sémantique entre différents mots-clés ; Jean-Michel Denizard (2017) cible les caractéristiques utiles au monteur son en restreignant les descripteurs à un dictionnaire limité de 2500 mots, son idée est intégrée au projet de recherche *Sons de France*[4] porté par le CNRS. Toutefois, aucune solution n'avait été adoptée à l'échelle internationale avant 2020 et la création de l'UCS (*Universal Category System*)[5] par Tim Nielsen, Justin Drury et Kai Paquin. Une des particularités de la norme UCS et de classer les sons selon 82 catégories, elles-même subdivisées en 752 sous-catégories. Cette proposition semble résoudre une partie des problématiques liées à l'indexation des sons et se trouve aujourd'hui majoritairement adoptée par les professionnels de l'audio.

Néanmoins, la norme UCS est parfois jugée trop lourde et la phase d'annotation peut-être fastidieuse, aussi, la proposition d'une indexation automatique serait souhaitable. Cette catégorisation automatique permettrait aux monteurs sons de bénéficier de l'organisation claire de l'UCS tout en s'épargnant une partie du processus d'indexation. Au vu des progrès grandissant de l'*apprentissage automatique*[6], il n'est pas déraisonnable de penser qu'un algorithme *classifieur*[7] puisse répartir des fichiers audios parmi ces 752 classes. Malheureusement, la recherche à ce sujet, dans une perspective audiovisuelle du moins, trouve peu de ressources, quand, en parallèle, les outils de classification de la recherche académique continuent de devenir de plus en plus performants. Fort de ce constat, ce mémoire est une tentative de synthèse des problématiques propres à nos métiers, et une sensibilisation aux sujets de l'apprentissage automatique, se plaçant donc à la croisée de deux domaines, qui, bien que voisins, interagissent encore peu.

Dans une première partie, nous dresserons le portrait des différentes lectures du sonore. Les classifications utilisées par le monteur son pour qu'il puisse naviguer rapidement dans sa sonothèque, en particulier le cas de l'UCS, que nous mettrons en parallèle d'une autre hiérarchisation, moins spécialisée, à vocation plus polyvalente. Nous poursuivrons en décrivant les bases de l'apprentissage automatique ainsi que son formalisme mathématique, avant de définir le concept de jeu de données à travers l'exemple d'Audioset. Enfin, nous nous attacherons à décrire le classifieur audio révélant les meilleures performances en classification des sons environnementaux à

3. cf. Glossaire D.1
4. https://sonsdefrance.com/#/home
5. https://universalcategorysystem.com/
6. L'apprentissage automatique est le concept qui se cache derrière le terme un peu fourre-tout d'*intelligence artificielle*
7. cf. Glossaire D.1

ce jour : Le modèle BEATs.

Ce dernier est le socle sur lequel repose notre partie pratique de mémoire que nous détaillerons dans un deuxième chapitre, nous justifierons alors notre démarche et expliquerons la méthode employée pour ajuster le modèle BEATs aux besoins propres à la condition du monteur son.

Enfin, Une troisième partie sera consacrée aux performances chiffrées de notre algorithme. Nous discuterons alors des limites de notre recherche et des ouvertures sur une éventuelle poursuite de ce travail.

Chapitre 1

État de l'art de la classification des sons et de l'apprentissage automatique.

1.1 À la recherche d'une ontologie des sons.

Dans cette première partie, nous étudierons les différentes hiérarchies des sons, des lectures polysémiques, propres aux artisans sonores du milieu audiovisuel, à celles plus généralistes, utilisées dans la recherche académique. Nous illustrerons notre propos à travers deux études de cas largement acceptées que sont les ontologies [1] UCS (destinée aux professionnels de l'audiovisuel) et Audioset (utilisée en recherche et traitement du signal), chacune prenant part dans des sphères professionnelles distinctes.

Avant de rentrer dans les détails de notre étude de cas, nous nous permettons un bref rappel sur le concept d'ontologie.

Tout objet au sens large est associé à un ensemble de descriptions qui forment sa *conceptualisation*. D'après Gruber (1993), " *Une ontologie est une spécification explicite d'une conceptualisation. [...] Ce qui est important c'est dans quel but est créé l'ontologie.* " Si la *conceptualisation* est générale et polysémique, l'*ontologie* doit permettre une hiérarchie des objets au regard d'une caractérisation monosémique motivée par un objectif clair. De même, cette hiérarchisation permet de définir *relation d'ordre* (*qui est au-dessus de qui et selon quels critères ?*). Cette *relation d'ordre* nous permet de dire que $9 > 7$ si on se base sur le critère de la valeur numérique. En revanche, si on considère le critère de la préférence, la relation d'ordre change, et on a alors $7 > 9$, le 7 étant le chiffre préféré à l'échelle mondiale, ce dernier étant associé à la chance et la prospérité (Bellos, 2014). C'est la même chose pour

1. cf. Glossaire D.1

les sons, ils seront ordonnés différemment suivant les caractéristiques considérées.

Nous invitons également notre lectorat à prendre connaissance des concepts théoriques relatifs à *l'Écoute*, rappelés dans les Annexes A. Ces connaissances, bien que nécessaires à la compréhension de cette synthèse, ne s'intègrent pas complètement dans le problème que nous souhaitons traiter.

1.1.1 Le monteur son et sa sonothèque.

Le cœur de ce mémoire réside dans l'optimisation de l'espace de travail du monteur son pour l'image animée. Nous utiliserons, pour illustrer nos propos, des exemples concrets, lesquels seront souvent présentés en anglais, les sonothèques [2] professionnelles utilisant cette langue.

Le monteur son est un professionnel du secteur de l'audiovisuel, il intervient dans la *post-production* [3], c'est-à-dire la phase de travail qui vient après l'enregistrement. Il peut exercer dans les différents domaines que sont la radio, le cinéma, le clip et plus généralement toutes formes artistiques qui implique un travail sonore. Dans le processus créatif du monteur son décrit par Delplancq (2009), la première étape est de " *choisir les sons du film* ", de décider d'une sorte d'identité sonore du film et de mettre en réserve ces sons dans ce qu'on appelle couramment un "chutier" [4].

Pour rechercher ces sons, le monteur peut piocher à trois endroits ; il va généralement regarder en priorité les sons issus du tournage, les *sons seuls* enregistrés par le chef opérateur du son. Ensuite, il va chercher dans sa sonothèque personnelle ou des sonothèques commerciales. Enfin, s'il n'a pas trouvé de sonorités qui servaient sa vision artistique, il peut lui-même enregistrer des effets sonores, ou faire appel à un bruiteur professionnel.

On s'intéresse particulièrement ici à la phase de recherche en sonothèque. En fonction de la scène, le monteur peut se placer dans différents modes d'écoute (détaillés dans les Annexes A.1.4), pour cerner différentes caractéristiques du son.

A Le son, un matériau polysémique.

> *Afin de conférer à une sonorité quelconque tous les moyens d'être découverte puis exploitée au sein d'un projet filmique dont les subtilités restent encore inconnues au moment de l'indexation [5], il aurait été envisageable [...] de s'efforcer à qualifier cette sonorité en l'abordant à travers toutes les écoutes possibles et imaginables (causale, réduite,*

2. Une sonothèque est une collection de sons.
3. cf. Glossaire D.1
4. cf. glossaire D.1
5. cf. glossaire D.1

morphologique, musicale, émotionnelle, symbolique, etc.).

- Jean-Michel Denizard (2017) -

La citation ci-dessus renforce cette idée que le son peut avoir plusieurs significations à la fois, le contexte dans lequel il est entendu influence la perception de celui-ci. Un des enjeux du monteur son est de faire passer par le son l'intention du réalisateur, en ce sens, multiplier les descripteurs du son, permet de cerner, sans même avoir besoin de l'écouter, un certain nombre de symboles véhiculés par ce son. Néanmoins, il est très chronophage (et sans doute pas nécessaire) d'être exhaustif sur la description de chaque sonorité, lesquelles auraient sûrement une liste bien trop longue pour être interprétable (Adjiman, 2015).

Jean-Michel Denizard (2017) propose donc d'indexer les sons dans les *" contextes habituels d'usages cinématographiques "*, il parvient à identifier plusieurs de ces contextes qu'il implémente au sein du projet de recherche CNRS *Sons de France*[6]. Au-delà des aspects purement techniques du fichier tels que le taux de quantification, la fréquence d'échantillonnage, le format de prise de son ou le type de microphone utilisé, qui peuvent certes renseigner au monteur sur la qualité d'un son, Denizard identifie trois grilles de lectures susceptibles d'intéresser le monteur son : Le *décor*, le *profil acoustique* et les *éléments sonores*.

Le décor qualifie le lieu de la prise de son, le contexte, si les microphones étaient placés en intérieur ou en extérieur et le cadre général de la prise de son (centre-ville, élevage ovin, autoroute, *etc*).

Ensuite, le *profil acoustique*, c'est-à-dire les caractéristiques physiques/acoustiques du son, regroupe des qualifications comme la dynamique, la densité, ou encore le plan sonore du son (notons que cette dernière est une des principales caractéristiques du choix d'un son pour le monteur Selim Azzazi, d'après Jeanne Delplancq (2009)).

Enfin, viennent les *éléments sonores*, il en existe deux types : *causaux* et *narratifs*. Les premiers décrivent les sources causales au sein de l'enregistrement, les seconds sont des descripteurs émotionnels ou esthétiques. Comme nous l'avons déjà dit dans le paragraphe sur l'écoute affective A.1.2, une *classification objective* basée sur ces critères émotionnels est impossible, chacun ayant une perception différente.

Un résultat majeur des expériences de Jean-Michel Denizard (2017) est l'identification des deux conclusions suivantes :

1. **Les sons à haute incertitude causale sont regroupés en fonctions de leur caractéristiques acoustiques.**

2. **Les sons à faible incertitude causale sont regroupés en rapport avec la source considérée.**

6. https://sonsdefrance.com/#/home

De même (Saada, 2017), dans son mémoire de Master, souligne que *" plus nous cherchons à être précis dans la description de notre perception d'un son, plus nos verbalisations gagnent en subjectivité "*. En d'autres terme, il faut choisir entre un langage général, voire imprécis, mais largement compréhensible et un langage précis mais incompréhensible.

Au regard de ces conclusions, il paraît pertinent d'**organiser les sonothèques en priorité selon des critères de causalité sonore** (cf. Annexe A.1.4 car la causalité est un langage à la fois précis et compréhensible). En revanche, pour les sonorités dont la source est ambiguë, il est d'autant plus nécessaire d'attribuer aux sons des adjectifs de description acoustiques.

> *" Une telle stratégie permettrait à la fois de restreindre le champ des possibles en se limitant à l'utilisation de qualificatifs en lien avec les quelques usages principaux dont pourrait faire l'objet le son considéré, tout en rejoignant l'un de nos objectifs qui était de dépasser la seule approche causale. "*
>
> - Jean Michel Denizard (2017) -

Propos que l'on peut mettre en regard de ce que nous dit Rodrigo Sacic, sonothécaire[7] chez HAL lorsque nous l'avons interviewé. Pour une banque de son commerciale à thème contenant uniquement des drones[8], il paraît en effet peu utile de mentionner la source d'un tel son, puisque la matière sonore a été tellement transformée que la source du son n'est généralement plus identifiable *in fine*. Au lieu de quoi il sera peut-être plus adéquat de définir des critères propres à une *écoute réduite* (voir Annexes A.1.1), c'est-à-dire des critères de forme, de couleur, de hauteur, de texture ou d'autres. Des critères de matières tels que *métallique* ou *sous-marin*, ou des critères de hauteur comme *grave*, *sombre* ou *brillant* se révèlent, pour ce cas précis, bien plus appropriés. Le thème de la polysémie[9] des sons, notamment au cinéma, est un sujet à garder en tête mais que nous ne développerons pas ici. Nous nous focaliserons uniquement sur l'aspect causal des sons et renvoyons à des lectures approfondies sur la polysémie sonore pour ceux qui souhaiteraient explorer le sujet : (Adjiman, 2015), (Denizard, 2017) et (Delplancq, 2009), par exemple.

Dès lors, nous voyons que la recherche d'une catégorisation universelle, polysémique et concise, est ambitieuse. Il est ainsi nécessaire pour chaque monteur son de connaître sa sonothèque, même si des outils facilitant l'ergonomie peuvent être imaginés. Chaque monteur son choisit la façon dont il *" range son grenier "* pour reprendre (Schaeffer, 1966)(cf. Annexe A.1.1). Bien sûr, la méthode de travail,

7. cf. Glossaire D.1
8. cf. Glossaire D.1
9. cf. Glossaire D.1

les spécialités et bien d'autres facteurs propres à chaque monteur, conditionnent la manière dont ils vont classer leurs sons.

Néanmoins, nous pouvons tout de même identifier certaines conditions plus ou moins communes. Les sons de sonothèques sont généralement *isolés* du reste des sources afin de permettre un *layering*[10]. En effet, si dans une station de travail audio-numérique *(STAN)*, il est facile de mixer des sons ensemble, il est en revanche beaucoup plus délicat d'isoler un élément sonore d'un enregistrement dans lequel auraient été mixées plusieurs sources lors de la prise de son. De ce fait, les monteurs son préfèrent généralement la polyvalence au réalisme puisque leur tâche est en quelque sorte de *" recréer le réel "*. En ce sens, pour les effets sonores, les monteurs sons préfèrent généralement des fichiers son mentionnant une source unique.

Nous n'avons pu nous empêcher d'utiliser le terme de sonothèque pour parler de classification des sons, cependant nous n'avons pas pris le temps de le définir proprement.

> *" Sonothèque, n.f : Lieu où l'on archive des enregistrements de bruits et de divers effets sonores dans le but d'une réutilisation cinématographique, radiophonique ou télévisée. "* [11]

Si la définition fait référence à un lieu (sous-entendu) physique, c'est qu'à l'ère de l'analogique, les sons étaient enregistrés sur supports physiques et étaient concrètement rangés dans des bobines sur des étagères. Une organisation certaine était nécessaire pour se retrouver dans ce qui représentait parfois plusieurs salles remplies de bobines. Aujourd'hui, on désigne par sonothèque, non pas un lieu physique, mais plutôt une collection virtuelle d'une dizaine de téraoctets de données stockées dans deux ou trois disques durs. L'organisation de ces données reste néanmoins un enjeu majeur pour le professionnel du son, dans ce qui suit, nous parcourerrons comment les ingénieurs se sont attelés à ranger efficacement les sons au fil des évolutions technologiques.

B La sonothèque à l'ère de l'analogique.

Dans un article de *La revue du son*, Abraham Moles (1959) décrit le principe d'une sonothèque et la classification de celle-ci. Évidemment, il s'agit ici de rangement de bobines, donc du matériel, à l'époque où les métadonnées [12] n'existaient pas encore. Le principe est le même qu'une bibliothèque. Les enregistrements sonores sont collés bout à bout sur une bobine, et les bobines rangées dans des étagères,

10. cf. Glossaire D.1
11. Définition du Centre nationale des ressources textuelles et linguistiques
12. cf. glossaire D.1

Moles souligne l'intérêt de séparer les extraits sonores grâce à des bouts de scotch de couleurs sur les bobines. *" La séparation de chacun de ceux-ci (les objets sonores) par une bande de 20 cm d'amorce rose, chaque dizaine par une bande d'amorce bleu, etc, permettant le repérage rapide au déroulement. "* (Moles, 1959).

L'auteur propose de ranger les sons à l'aide d'un système de cartes perforées, chaque carte perforée étant associée à un objet sonore de la sonothèque. Chaque carte est constituée de 2 parties, une partie *en clair*, et l'autre partie *en binaire*.

- La partie en clair permet d'écrire des informations en toutes lettres concernant l'objet sonore, l'auteur, la date de prise de son, des informations de hauteur, de timbre, de niveau, des remarques, etc. La partie en clair est en plus annotée d'une *côte de localisation*, qui localise *" 1) la salle de la sonothèque dans laquelle se trouve la bobine considérée, 2) l'armoire correspondante, 3) le rayon de cette armoire, 4) l'emplacement de la bobine sur le rayon, et 5) le numéro de l'objet sonore sur la bobine. "*

- La partie en binaire de la carte consiste en la binarisation d'un certain nombre de critères (généralement jusqu'à une centaine par carte), un trou représente un "oui", une absence de trou un "non" pour le critère considéré.

> *" La multiplication des adjectifs permet toujours de réduire l'ensemble de la description à une série d'adjectifs binaires (oui ou non) au prix d'un allongement de la liste de ceux-ci et tout l'art de la catégorisation des objets sonores se ramène à effectuer cette "réduction au binaire" de la façon la plus simple et la plus économique possible. "*
>
> - Abraham Moles (1959) -

Ces critères binaires sont répartis en 3 catégories :

1. Des caractères *généraux* : nom de l'auteur, qualité, durée de l'enregistrement. présents pour tous les objets de la sonothèque.

2. Des caractères *spécifiques* : servant à préciser le portrait du son considéré, il s'agit de critères relatifs au timbre apparent (grave, aigu, mince ou épais, brillant (riche en raies) ou pur etc), à la complexité, au rythme ou à l'évolution [13] (son *crescendo*, *decrescendo*, fréquence qui monte ou qui descend etc).

3. Des caractères *constructifs* préparant à la manière dont ils pourront être utilisés. On peut ici considérer des caractères d'ordre psychologique, ou relatifs à l'évocation qu'ils donnent à l'auditeur (bruit d'eau, bruit industriel, bruit de chemin de fer, etc)

13. A. Moles ayant travaillé avec Schaeffer sur la morphologie des objets musicaux, il reprend sa terminologie.

Les cartes sont ensuite rangées dans un classeur et on peut faire une recherche par critère, une machine faisant le tri ressortira toutes les cartes vérifiant les critères renseignés. Et on n'aura plus qu'à aller les chercher dans les étagères indiquées sur la partie claire des cartes, grâce à la côte de localisation.

Le passage au numérique a permis l'ajout de champs descripteurs encapsulés au sein du fichier. L'idée de rajouter des champs descripteurs est plutôt ancienne, il y a pour le CD audio, la norme CD text et ID3 pour le mp3. On a pris l'habitude de nommer *métadonnées* ces champs descripteurs, nous verrons dans le paragraphe suivant comment ces métadonnées peuvent être utilisées pour aider le monteur son à ranger sa sonothèque.

C Les métadonnées.

Les métadonnées sont littéralement des données de données, elles permettent de lier des données audio avec des descripteurs textuels permettant une meilleure compréhension de ce que contient le fichier. Si ces métadonnées ont d'abord été introduites pour être lues par les machines, avec des informations concernant la fréquence d'échantillonnage, le taux de quantification, *etc*, elles ont vite permis l'intégration d'informations à destination des professionnels de l'audiovisuel pour fluidifier le *workflow*[14].

Le secteur de l'audiovisuel s'est assez vite emparé de cette technologie, notamment pour assurer un passage plus continu du tournage à la postproduction. Dès 1997, AATON et SADIE s'associent sous la supervision de l'EBU *(European Broadcast Union)* pour créer une nouvelle tranche de métadonnées propre à l'encapsulation *WAV*. Cette démarche donnera naissance à la tranche *BEXT (broadcast extension)* dont une description est présentée dans le tableau 1.1.

Le champ "description" contient en fait la quasi-totalité des données qui sont utiles au monteur son durant la phase de conformation (scène, prise, notes, nom des pistes, *etc*). L'arrivée de cette tranche de métadonnées a permis un gain de temps considérable tout en limitant la rupture du passage entre la prise de son et le montage. Le chef opérateur sur le tournage peut transmettre ses questionnements, ses intentions de manière plus directe au monteur son (Laurin, 2018).

En 2003, AATON intègre l'écriture de la tranche *BEXT* au sein de son nouvel enregistreur dédié au tournage cinéma. Cette nouveauté fera le succès du *Cantar X* qui deviendra, dès lors, une référence. Une poursuite de cette idée est la création d'une tranche au format XML reprenant certaines idées du *BEXT* tout en laissant plus de liberté à l'utilisateur, le iXML s'étend par ailleurs à d'autres formats que le WAVE.

14. cf. glossaire D.1

Tableau 1.1 – Champs descripteurs de la tranche BEXT (Broadcast Extension)

Nom	**Définition**
Description	Ce champ est souvent employé par les fabricants pour stocker des informations complémentaires (Numéro de piste, Nombre d'images par seconde, etc.)
Originator	Nom du producteur de l'enregistrement. Généralement celui du fabricant de l'enregistreur.
OriginatorReference	Identifiant attribué par le producteur de l'enregistrement.
OriginationDate	Date de l'enregistrement au format aaaa-mm-jj
OriginationTime	Heure de l'enregistrement au format hh :mm :ss
TimeReference	Valeur appelée *Sample Count Since Midnight*. Il s'agit du nombre de samples passés depuis minuit au moment du début de l'enregistrement. Cette valeur permet, pour une fréquence d'échantillonnage et un nombre d'images par seconde donné, de retrouver le Time Code horaire du début de l'enregistrement au sample près.
— *À partir de la version 1* —	
Version	Version du standard auquel correspond le fichier. Peut être 0, 1 ou 2.
UMID	UMID (*Unique Material IDentifier*) tel que défini par la SMPTE.
— *À partir de la version 2* —	
LoudnessValue	Valeur du Loudness intégré en LUFS (multipliée par 100)
LoudnessRange	Valeur du Loudness Range en LU (multipliée par 100)
MaxTruePeakLevel	Valeur maximum de crête réelle (True Peak) en dBTP (multipliée par 100)
MaxMomentaryLoudness	Valeur maximum du Loudness momentané en LUFS (multipliée par 100)
MaxShortTermLoudness	Valeur maximum du Loudness Short-Term en LUFS (multipliée par 100)
— *Toutes versions* —	
Reserved	Espace réservé pour un éventuel usage dans de futures versions.
CodingHistory	Historique des codages apportés au flux audio. Le format de ce champ est détaillé dans la recommandation R-98 de l'UER.

Le MPEG-7 est une norme proposée par le *Fraunhofer Institute for Integrated Circuits* au début des années 2000 pour structurer les médias audiovisuels. Cette norme fut initialement introduite en vue d'une utilisation pour l'annotation visant à recenser des métadonnées de production comme l'auteur, le titre, la date, *etc.* Elle permet aussi une indexation selon des caractères physiques du fichier, telles que la résolution, la fréquence d'échantillonnage ou le niveau global. Enfin le MPEG-7 permet de mettre en relation des concepts sémantiques avec une vidéo ou un son (par exemple le descripteur *chat* dans les métadonnées d'une vidéo montrant des chats).

Bien que le MPEG-7 soit une forme de métadonnées qui eut un sens dans l'industrie audiovisuelle, notamment pour des questions de production, elle ne constitue pas une classification des sons qui puisse satisfaire le monteur son. Les améliorations en terme d'ergonomie de travail ont principalement été apportées par les logiciels de gestion de sonothèques. Les nouveaux gestionnaires de sonothèques numériques comme *Soundly*[15] ou *Soundminer*[16] permettent un contrôle et une flexibilité sur les métadonnées des fichiers audios. Ces logiciels utilisent une manière propriétaire pour écrire les métadonnées au sein du fichier. Cependant ils sont capables de lire les formats de métadonnées publics usuels comme le *iXML* ou le *BEXT* et par ailleurs, lorsque l'utilisateur modifie les métadonnées d'un son dans le gestionnaire, le logiciel s'efforce de copier un maximum des informations au format propriétaire vers les formats publics pour assurer une compatibilité maximale entre les logiciels.

Dans le reste de ce mémoire nous nous baserons essentiellement sur *Soundminer* puisque c'est le logiciel qui nous semble à ce jour le plus complet pour la lecture et l'écriture de métadonnées dans un cadre audiovisuel. On y retrouve les champs de données suivants : *Description*, *keywords* (mot-clés), *FX Name* (Nom de l'effet), *Track Title* (Titre de la piste), *Manufacturer* (Nom du créateur) *Location* (Localisation), *Microphone* (Microphone) parmi tout un tas d'autres détaillés dans le tableau 34 en annexe, que nous appellerons "format de métadonnées *Soundminer*".

Enfin, les ontologies pour la classification du multi-média et en particulier la classification des sons se sont également multipliées. Nous verrons dans le paragraphe suivant une organisation des sons jugée pertinente et aboutie pour le montage son dans la *post-production* audiovisuelle.

D La norme UCS (*Universal Category System*) : une ontologie par et pour les professionnels de l'audio.

La norme UCS (*Universal Category System*, Système de Catégorisation Universel) désigne à la fois une catégorisation des sons (se voulant universelle) et une

15. https://getsoundly.com/
16. https://store.soundminer.com/pages/about-soundminer-inc

convention de nommage, ceci visant à fluidifier le travail de post-production dans le milieu de l'audiovisuel.

Une nomenclature stricte ...

L'UCS conseille sur la manière dont les fichiers doivent être nommés, et la norme va en fait encore plus loin puisqu'elle dicte des règles pour une partie des métadonnées que l'on retrouve dans Soundminer. Pour le champ de Description par exemple, il doit s'organiser de la manière suivante : TITLE -subtitle- Prop, Verb, DescriptiveTerms.

- TITLE (TITRE) : nom du matériau ou de la source sonore ;
- Subtitle (Sous-titre) : quand il y a plusieurs sons partageant le même titre ;
- Prop (Accessoire) : plus de détails sur le matériau ;
- Verb (Verbe) : qualifie l'action performée dans le son.
- Exemple :
  BASEBALL BAT - Drop - Wood, Drop on conrete
  BASEBALL BAT - Hit - Metal - several hits
  (BATTE DE BASEBALL - Tombe,
  BATTE DE BASEBALL - Coup)

Cette méthode, bien qu'elle ne soit basée sur aucune étude scientifique quantitative ou sondage à grande échelle, a été élaborée par un conciliabule [17] de professionnels de l'audio. Nous avons donc des raisons de penser qu'elle est une bonne manière de remplir les champs descripteurs.

Du reste, nous renvoyons à la lecture du fichier au format PDF de Kai Pacquin (2020) détaillant la nomenclature et la conformation des métadonnées.

... Et une ontologie ordonnée

L'ontologie UCS se structure en 82 catégories principales, elles-mêmes subdivisées en 752 sous-catégories [18]. Son organisation repose sur une approche sémantique spécifiquement conçue pour répondre aux besoins des monteurs son. Elle regroupe ainsi des sons partageant des causes d'émission similaires, bien que, d'un point de

17. *Andrew Quinn, Andy Martin, Arnoud Traa, Cédric Chatty, Chris Battaglia, Jeff Davis, Justin M. Davey, Michal Fojcik, Mikkel Nielsen, Paul Poduska, Roy Waldspurger, Théo Serror, Tim Farrell, and Tristan Horton.*

18. Un tableau de toutes les catégories et leur description sur le drive de l'UCS : https://docs.google.com/spreadsheets/d/1gHv0oT86GX0kZHaWk8Nqi3BNLnB8gb_B/edit?gid=1520020613#gid=1520020613

vue purement sonore – dans une perspective d'*écoute réduite* – les éléments appartenant à une même catégorie puissent être très différents les uns des autres. Cette classification repose principalement sur une logique de regroupement par la cause de production des sons, s'inscrivant ainsi dans une *écoute causale*.

Un exemple concret illustrant cette organisation peut être observé dans la catégorie *FOOD & DRINKS/Cooking*, qui rassemble des sons liés à l'univers de la cuisine. Cette catégorie inclut aussi bien des enregistrements de friture que des sons de découpe de légumes. Bien que ces deux actions relèvent d'une même activité sémantique – la préparation culinaire – elles présentent des caractéristiques acoustiques fondamentalement distinctes. En effet, la découpe de légumes se manifeste par une succession de sons percussifs, tandis que la friture génère un son continu, stationnaire sur une large bande de fréquence. Malgré ces différences sonores, ces deux types de sons sont classés ensemble en raison de leur proximité sémantique d'émission, illustrant ainsi l'approche causale adoptée par l'ontologie UCS.

Cause réelle, Cause figurée.

La *cause réelle* pour Michel Chion (2010) est la source véritable du son que l'on entend, alors que la *cause figurée* (ou *cause attribuée*) est celle perçue par l'auditeur, cette dernière est subjective, dépend du contexte, etc, alors que la première est objective, factuelle. En montage audiovisuel, une des premières sources de *design sonore*[19] consiste à utiliser la cause réelle d'un son et de l'attribuer à autre chose. Dans ce cadre, l'interaction entre le son et l'image modifie la perception de l'événement sonore, un phénomène que l'auteur désigne sous le terme de *synchrèse*. Le bruitage en est une illustration concrète : le craquement de légumes déchirés va devenir le son de ruptures d'os et de cartilages dans une scène de violence présentée à l'écran. L'image guide ainsi l'auditeur dans une reconfiguration cognitive du son, lui attribuant une *cause perçue* distincte de sa *cause réelle*.

Cette distinction soulève la question de l'organisation des sonothèques et des bases de données sonores : doivent-elles être classées en fonction des causes réelles des sons ou bien selon leurs causes perçues ? Si on regarde par exemple la catégorie UCS "BATEAUX", on retrouve les sous-catégories "MECHANISMES", "VAPEURS" ou "PORTES". La classification UCS semble ainsi privilégier une approche basée sur les causes réelles, suivant ainsi les recommendations de Denizard (2017).

La catégorie "BRUITAGE"[20] (*"FOLEY"*), repose précisément sur ce principe de *synchrèse*, en associant un son à une source visuelle alternative. Dans cette perspective, il semble justifié d'exclure la catégorie *Foley* d'une ontologie de classification

19. cf. Glossaire D.1
20. cf. Glossaire D.1

basée uniquement sur le sonore, car elle repose sur un principe d'assignation arbitraire plutôt que sur une relation intrinsèque entre le son et sa cause réelle.

Quand les sons ne sont pas classés selon leur cause.

Il y a d'autres catégories de l'UCS qui ne respectent pas cette organisation par *sources sonores*. C'est par exemple le cas de la catégorie *"ARCHIVED"*, Cette catégorie a été pensée à l'instar d'un dossier "utilitaire" dans une sonothèque, il regroupe les précédents travaux du monteur son, mais aussi les signaux de tests, ou les réponses impulsionnelles. Il ne s'agit pas d'une catégorie regroupant des sources semblables ou présentant des caractéristiques acoustiques similaires.

De même, les fichiers classés dans *"DESIGNED"* sont les sonorités créées ou modifiées qui ne rentrent dans aucune des autres catégories, il peut s'agir de textures cinématographiques. Cette classe regroupe les sons comme les *"effets sonores style trailer"*, c'est-à-dire les *"BOOM"*[21], les *"WHOOSH"*[21] ou les *"RISERS"*[21] ainsi que les textures sonores cinématographiques comme les *"DRONES"*[21] ou les *"RUMBLES"*[21] .

Le propre des ambiances sonores est qu'elles témoignent d'un regroupement de plusieurs sources sonores formant un tout, c'est-à-dire un ensemble d'éléments plus petits témoignant de l'atmosphère d'un lieu précis à un moment précis. La norme UCS, que nous considérons informée au sujet des enjeux de l'industrie audiovisuelle, définit une ambiance comme suit : *" Il s'agit généralement de l'enregistrement d'un lieu spécifique qui contient plusieurs éléments "*.

Les deux catégories *"ARCHIVED"* et *"DESIGNED"* appliquent une typologie qui n'est pas celle centrée sur la cause des sons, nous jugeons donc également peu pertinent de traiter de ces classes dans la suite de ce travail. Pareillement, nous écarterons le cas des *"AMBIANCES"* qui contiennent plusieurs sources de son, nous nous attachons à traiter ici un problème de classification nécessitant une *unique description*, ce qui est le cas de la plupart des effets sonores présents en sonothèques.

La catégorisation UCS est une ontologie précise et détaillée, il existe un tableau regroupant l'ensemble des 82 catégories et 752 sous-catégories, une brève description de la catégorie et surtout un nombre exhaustif d'exemples de sources sonores supposées appartenir à chaque catégorie ; la figure 1 illustre cette idée.

De ce fait, si un monteur son a un doute quant à où il doit ranger un son, il peut simplement effectuer une recherche textuelle (ctrl+F/cmd+F) et chercher le nom de la source qui lui pose problème. Viennent alors des recommandations discutables, que seul le monteur lui-même peut juger pertinentes ou non. Un exemple de recommandation questionnable est le fait que les alarmes de véhicules sont censées

21. cf. Glossaire

MACHINES	AMUSEMENT	Apparatus, Bumper, Carousel, Carrousel, Cars, Coaster, Contraption, Device, Drop,
MACHINES	ANTIQUE	Abacus, Adding, Antique, Apparatus, Arcade, Automaton, Babbage, Butter, Cash,
MACHINES	APPLIANCE	Air, Apparatus, Blender, Bread, Can, Cleaner, Coffee, Conditioner, Contraption,
MACHINES	CONSTRUCTION	Apparatus, Cement, Chipper, Compactor, Concrete, Contraption, Crane, Device,
MACHINES	ELEVATOR	Apparatus, Contraption, Device, Dumbwaiter, Dumbwaiters, Freight, Hoist, Lift,
MACHINES	ESCALATOR	Apparatus, Contraption, Device, Machinery, Moving, Stairs, Travelator, Walkway,
MACHINES	FAN	Air, Apparatus, Bladeless, Blower, Box, Ceiling, Circulator, Contraption, Device,
MACHINES	GARDEN	Apparatus, Blower, Chainsaw, Contraption, Cultivator, Device, Eater, Edger, Hedge,
MACHINES	GYM	Apparatus, Bike, Bowflex, Cardio, Climber, Contraption, Device, Elliptical, Exercise,
MACHINES	HITECH	3D, 7, Apparatus, Arm, Bond, CNC, Contraption, Cutter, Device, Digital, Gadget,
MACHINES	HVAC	Air, Aircon, Apparatus, Baseboard, Boiler, Chiller, Climate, Conditioner, Conditioning,
MACHINES	INDUSTRIAL	Apparatus, Assembly, Auto, Automation, Contraption, Conveyor, Cutting, Device, Die,
MACHINES	MECHANISM	7, Apparatus, Bond, Box, Contraption, Device, Gadget, Gizmo, Goldberg, James,
MACHINES	MEDICAL	Apparatus, Blood, Cat, Centrifuge, Concentrator, Contraption, CT, Defibrillator, Dental,
MACHINES	MISC	Apparatus, Contraption, Device, Machinery, Miscellaneous
MACHINES	OFFICE	Apparatus, Binding, Business, Clock, Contraption, Copier, Copy, Cutter, Device,
MACHINES	PUMP	Aerator, Air, Apparatus, Backflow, Blower, Boilers, Centrifugal, Compressor, Contraption,

FIGURE 1 – Un certain nombre d'exemples de sources sonores pour la catégorie MACHINES.

être rangées dans VEHICULES-ALARMS, plutôt que dans une catégorie ALARMS-VEHICULES, qui aurait pu exister.

E Conclusion partielle.

Nous avons parcouru certaines normes et moyens pour structurer une sonothèque. Cependant cette liste ne peut être exhaustive et on est en droit de se demander s'il n'existe pas autant de manières de ranger que de professionnels. Par exemple, le monteur son Selim Azzazi organise sa sonothèque, non pas en fonction de sources ou des *caractéristiques acoustiques* des sons, mais, chronologiquement par film, puis par scène au sein de chaque film.

Néanmoins, au vu des arguments avancés en faveur d'une *hiérarchisation causale* et des témoignages récoltés au cours de discussions informelles par les professionnels de la post-production audiovisuelle, nous jugeons que l'ontologie UCS est pertinente pour ce domaine. C'est donc sur cet outil que nous construirons notre méthodologie.

1.1.2 La recherche académique et la classification des sons environnementaux (CSE).

L'étude d'une hiérarchisation des sons n'intéresse pas que les professionnels de l'audiovisuel, dans cette partie nous détaillerons une approche académique de la classification des sons environnementaux (CSE). L'objectif de la CSE est classifier (généralement en fonction de la source des sons) toute ou partie des sons environnementaux automatiquement à l'aide d'algorithmes.

Présentons d'abord une introduction de ce que l'on considère généralement par "sons environnementaux" : il s'agit de l'ensemble des sons susceptibles d'exister dans un environnement donné. Les sons environnementaux ont un support matériel, une

source physique, contrairement aux sons de synthèse. Les sons environnementaux diffèrent aussi du signal de la *parole* dans la mesure où ils ne véhiculent pas une *information sémantique explicite et absolue*. Pour ces raisons, nous considérerons ici que musique et parole ne font pas partie des sons environnementaux, à moins que ceux-ci soient perçus comme faisant partie d'un tout formant un environnement sonore, par exemple la musique de rue ou les discussions *inintelligibles* provenant d'une terrasse de bistrot.

La tâche de classification des sons environnementaux est plus complexe que les traitements concernant le signal musical ou la parole (Mushtaq *et al.*, 2021). Il y a deux principales causes qui rendent ce problème difficile à appréhender.

La première c'est l'aspect non statique [22] et dépourvu de structures apparentes composant un son environnemental (Bansal & Garg, 2022). À l'inverse, la musique est un signal ordonné, un morceau est généralement divisé en couplets/refrains le tout formant une structure. Au sein même de ces couplets/refrains, les mesures structurent les temps en fonction de la métrique du morceau et l'harmonie structure les notes qui sonnent "juste" de celles qui sonnent "faux".

Pour ce qui est de la voix, les mots peuvent être séparés en phonèmes qui sont les blocs atomiques (insécables) du langage et l'enchaînement des phonèmes dans une phrase obéit à certaines règles. C'est d'ailleurs en se basant sur ces règles que fonctionnent les modèles de langage génératifs (*large-language models (LLM)*) comme Chat-GPT-4o ou Deepseek-r1. Ces derniers génèrent des *token* (c'est-à-dire un mot ou un bout de mot) les uns à la suite des autres. À partir des précédents *tokens*, le modèle génère celui qui a la plus grande probabilité d'apparition et ainsi de suite, de sorte qu'*in fine* il ait rédigé un paragraphe bien construit. Évidemment, on ne peut pas appliquer ces prédictions aux sons environnementaux, ces derniers étant dépourvus de structure logique apparente.

De même Bansal et Garg (2022) souligne que la grande variabilité acoustique des sons ne facilite pas l'apprentissage automatique. En effet, la CSE concerne l'étude de tous les sons qui ne sont pas de la musique ou de la parole, il en existe beaucoup. Entraîner un algorithme capable de discriminer l'ensemble de ces sons demande à la fois une grande quantité de données, mais également une grande diversité.

La CSE est un champ de recherche plutôt récent puisqu'il se développe à la fin du *XXème* siècle. Ce champ de la recherche vise à classifier de manière automatique, *i.e.* avec des machines ou algorithmes, le large panel des sons qui existent dans la nature. Au départ, ces classifieurs automatiques suivaient des règles de probabilités, les concepts mathématiques sous-jacents aux premiers modèles étaient généralement

22. Un son statique serait, au sens Schaefferien, un son avec un entretien passif, par exemple un sinus de laboratoire est un son statique, mais le même sinus entrecoupé de silences est non statique.

des chaînes de Markov cachées (Couvreur *et al.*, 1998) ou des algorithmes des k-ièmes plus proches voisins (Sawhney & Maes, 1997). À partir des années 2000 se développent des approches basées sur l'algèbre et l'optimisation de fonctions à plusieurs variables, les machines à vecteurs de support en sont un exemple (L. Chen *et al.*, 2006). Enfin, aux alentours de 2015, l'émergence des réseaux de neurones basés sur l'apprentissage profond (*deep learning*), amène de nouvelles possibilités. On a alors d'abord vu naître des modèles spécialisés dans la reconnaissance de sons bien particuliers. De nombreux travaux concernant le confort d'écoute en milieu urbain ont été menés (Salamon *et al.*, 2014)(Bello *et al.*, 2018), si bien qu'encore aujourd'hui une quantité significative des jeux d'entraînement pour la tâche de CSE ne concernent que les sons rencontrés en milieu urbain. Depuis maintenant une dizaine d'années, la CSE a le vent en poupe, poussée par des nécessités grandissantes comme les secteurs de la domotique, de la surveillance de masse, ou de la robotique (Bansal & Garg, 2022) (Chandrakala & Jayalakshmi, 2019). Néanmoins, l'application de la CSE au domaine de l'audiovisuel reste peu explorée à notre connaissance.

En définitive, on peut résumer les motivations de la communauté à la **recherche d'un passage automatique d'une écoute réduite centrée sur des caractéristiques physiques à une étude causale relative à la source d'un son.** La partie suivante sera consacrée aux principes de base de l'apprentissage automatique afin de cerner comment les algorithmes sont capables d'effectuer cette tâche de classification de manière autonome.

1.2 Apprentissage automatique et jeux de données.

Dans un l'article fondateur de l'apprentissage automatique, Alan Turing (1950) pose les bases de l'intelligence artificielle (I.A.) en théorisant ce qu'il appelle les *"Learning machines"*. Turing montre qu'il est impossible de coder à la main un programme assez long pour simuler une intelligence de niveau humain. Cependant, il n'exclut pas l'existence d'une telle machine. Il imagine un protocole en trois étapes.

1. Écrire un programme simulant l'esprit d'un enfant : *“ un cahier avec peu de mécanismes et beaucoup de pages vierges ”*[23]
2. Construire un apprentissage en montrant à la machine un certain nombre de règles ou d'exemples.
3. Corriger les erreurs du programme par un intervenant humain.

La partie à coder à la main, le cahier vierge, devient alors réalisable.

23. "Presumably the child brain is something like a notebook [...] Rather little mechanism, and lots of blank sheets." A. Turing

1.2.1 Qu'est ce que l'apprentissage ?

Avant d'aller dans le détail, il semble important d'expliquer les grands principes de l'apprentissage automatique. Le vocabulaire étant spécifique, nous avons de manière quasi-systématique, associé les termes anglo-saxons afin de permettre au lecteur une poursuite vers des recherches complémentaires. Nous aborderons dans ce préambule les notions d'apprentissage, de jeu de données et de généralisation.

A Plusieurs manières d'apprendre.

Les trois types d'apprentissages que nous verrons tout de suite font partie de l'*apprentissage statistique*, méthode la plus utilisée à ce jour [24]. Le processus d'apprentissage se fait sur des données qui servent d'exemples, la machine ne connaît pas les règles qui régissent la résolution du problème mais parvient à le résoudre en généralisant à partir de son expérience acquise par l'exemple. C'est le principe d'*inférence* [25] qui permet la généralisation d'une proposition à une population à partir de propositions tenues pour vraies sur un échantillon de cette population.

L'apprentissage supervisé est basé sur des données ordonnées et labellisées qui servent à entraîner l'algorithme. Ces données ont des caractéristiques *(features)* et une ou plusieurs étiquettes *(labels)*. Un label est une métadonnée, une étiquette attachée à chacun des éléments qui permet de les discriminer ou de les regrouper. On parle d'apprentissage supervisé dès lors que la sortie pour une entrée est connue, l'algorithme doit alors comparer sa prédiction basée sur les caractéristiques de l'entrée et la vérité terrain *(ground truth)*. La vérité terrain correspond à la performance maximale théorique de l'algorithme. Le calcul de cette différence se fait à l'aide d'une *fonction perte* parfois appelée *fonction coût* ou *fonction d'erreur*, dont la valeur est rétropropagée [26] à travers le modèle pour mettre à jour les paramètres numériques le constituant. Un algorithme qui a appris de manière supervisée effectue une tâche de *régression* pour les valeurs continues et une *classification* pour les valeurs discrètes.

Dans l'exemple 2, on considère un *espace des caractéristiques* [27] à deux dimensions et deux classes à séparer, les triangles et les ronds. On observe qu'au sein de la même classe il peut y avoir de fortes variances en termes de caractéristiques (variation inter-classe). La frontière entre les différentes classes est appelée *séparatrice* ou *frontière de décision*.

Un algorithme qui ne voit que des données non labellisées apprend de manière *non supervisée*. Le but principal de l'apprentissage non supervisé est d'identifier au

24. Cédric Villani dans : https://www.youtube.com/watch?v=zxBCYeLYauc&t=397s
25. cf. Glossaire D.1
26. cf. Glossaire D.1
27. cf. Glossaire D.1

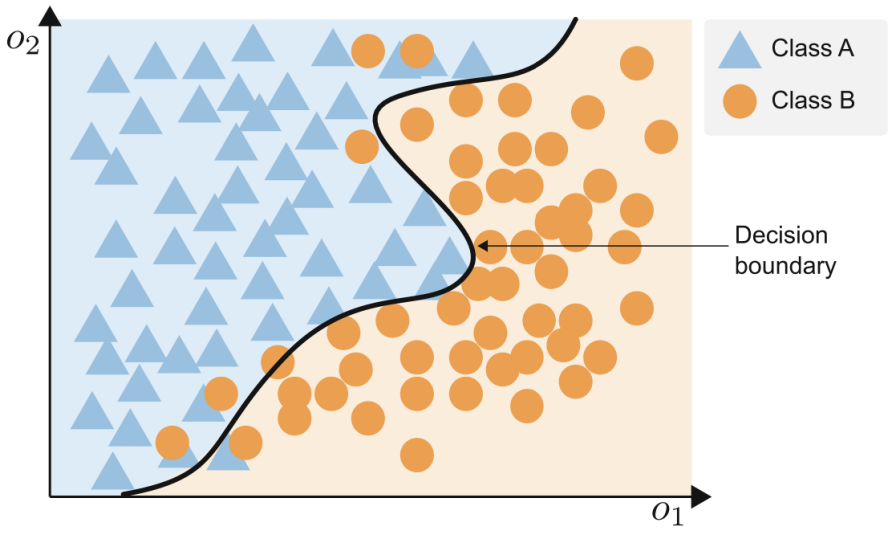


FIGURE 2 – Frontière entre deux classes A et B (VIRTANEN *et al.*, 2018).

sein des données des structures fondamentales ou des motifs qui se répètent. L'algorithme est donc capable de réaliser un *regroupement*[28], c'est-à-dire de rapprocher des entrées similaires dans son *espace des caractéristiques* ou *espace latent*, mais il n'est pas capable de donner un nom à ces groupes.

L'Apprentissage semi-supervisé est un mélange des deux méthodes précédentes, l'algorithme apprend avec une petite partie de données annotées et une grande partie de données non annotées. Ce type d'apprentissage nécessite beaucoup moins d'intervention humaine et a l'avantage de se baser en grande majorité sur des données non labellisées, ces dernières étant plus accessibles que les données labellisées. *L'apprentissage par transfert* permet ensuite de transférer les connaissances du modèles pré-entraîné (par le biais d'un apprentissage supervisé) vers une situation spécifique (PAN & YANG, 2010).

B Formalisme mathématique.

Considérons une fonction de plusieurs variables $\hat{f}(x_1, \ldots, x_n)$ construite par un algorithme au cours de son apprentissage, et posons la fonction vraie $f(x_1, \ldots, x_n)$. f est la vérité terrain (*ground truth*), en opposition à $\hat{f}$ qui est la prédiction de l'algorithme. Le but de l'apprentissage est de faire tendre $\hat{f}$ vers la vérité f, de sorte que cela revient à minimiser la quantité L dans l'équation 1.1.

$$f(x_1, \ldots, x_n) = \hat{f}(x_1, \ldots, x_n) + L(x_1, \ldots, x_n) \tag{1.1}$$

28. cf. Glossaire D.1

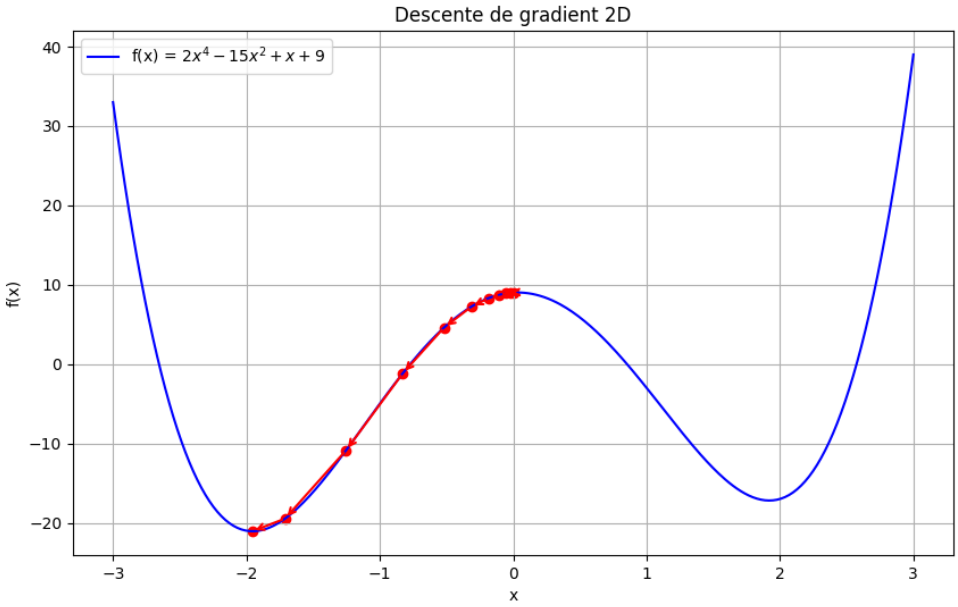


FIGURE 3 – Descente de gradient en 1D

On appelle L la fonction coût [29] (*loss-function*) en anglais. On remarque donc que si L tend vers 0, alors la prédiction de l'algorithme tend vers la vérité, *i.e.* :

$$\lim_{L \to 0} \hat{f} = f$$

C'est un problème d'optimisation mathématique, le but de l'entraînement est donc de minimiser la fonction coût. Par ailleurs, les quantités $(x_1, \ldots, x_n)$ sont appelées les *caractéristiques* des données, et sont les variables de la fonction prédiction.

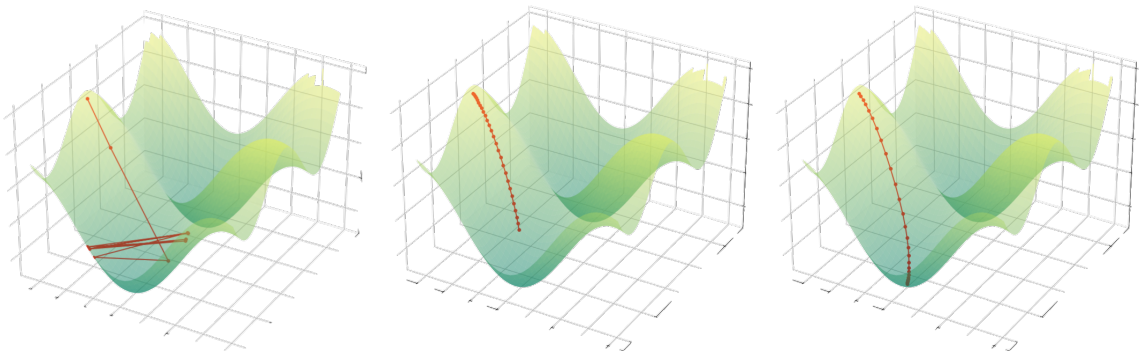
Pour clarifier, considérons un exemple à une seule caractéristique, la fonction coût ne dépend que d'une seule variable et peut donc être représentée sur un graphe 2D. Une des manières de trouver un minimum local d'une fonction continue, est d'analyser la tangente à la courbe, c'est-à-dire la dérivée de la fonction, et de se déplacer dans la direction selon laquelle la tangente est minimale. Pour une fonction coût à plusieurs variables (ce qui sera systématiquement le cas) on s'intéressera donc à ses dérivées partielles, autrement dit au *gradient* de cette fonction.

L'algorithme de la *descente de gradient* est une méthode d'optimisation mathématique applicable à toute fonction différentiable [30]. C'est un processus itératif qui calcule le gradient en un point de la fonction coût, puis se déplace dans la direction de la pente la plus fortement négative d'une quantité lr, calcule à nouveau le gradient en ce point, se déplace encore selon la direction de plus forte pente, et ainsi de

29. cf. Glossaire D.1
30. cf. Glossaire D.1

suite jusqu'à trouver un extremum local.

La quantité *lr* est le *pas d'apprentissage*[31] ou *learning rate* en anglais. Un pas d'apprentissage trop grand 4a ne permet pas de trouver le minimum de la fonction, l'algorithme oscille autour du minimum sans jamais l'atteindre. Au contraire, un pas d'apprentissage trop petit 4b ne parvient pas à atteindre le minimum avant la fin des itérations. La figure 4c permet d'atteindre l'optimum local en un nombre raisonnable d'itérations.



(a) pas trop grand (b) pas trop petit (c) pas de taille optimale

FIGURE 4 – Illustration d'une descente de gradient avec trois valeurs de pas d'apprentissage différentes.

Quand les poids du modèle sont mis à jour, on dit qu'on a fait une *itération*. Cependant il est assez rare que l'on calcule l'erreur après chaque exemple, procéder de cette manière est souvent trop chaotique. Au lieu de quoi on préfère recalculer l'erreur sur plusieurs exemples formant un *lot*[32], ce qui moyenne l'erreur et permet généralement un meilleur apprentissage. Ces lots, *batchs* en anglais, peuvent avoir différentes tailles, des valeurs génériques sont 16, 32, 64 ou 128. De fait, on applique la formule 1.2, où B_k est la taille du batch en nombre d'exemples.

$$\hat{\nabla} L = \frac{1}{B_k} \sum_{n=1}^{B_k} \nabla L \tag{1.2}$$

L'algorithme parcourt ainsi l'entièreté de son jeu d'entraînement en mettant à jour ses paramètres tous les B_k exemples. Une *époque* (*epoch*) a été réalisée lorsque l'algorithme a parcouru l'ensemble de son jeu de données. L'apprentissage se fait sur plusieurs époques afin de consolider les connaissances du modèle, puisqu'après tout : " *La répétition est à la base de la pédagogie* "[33]

31. cf. Glossaire D.1
32. cf. Glossaire : Taille de lots D.1
33. cit. Pascal Spitz, professeur à l'ENS Louis Lumière et ingénieur du son.

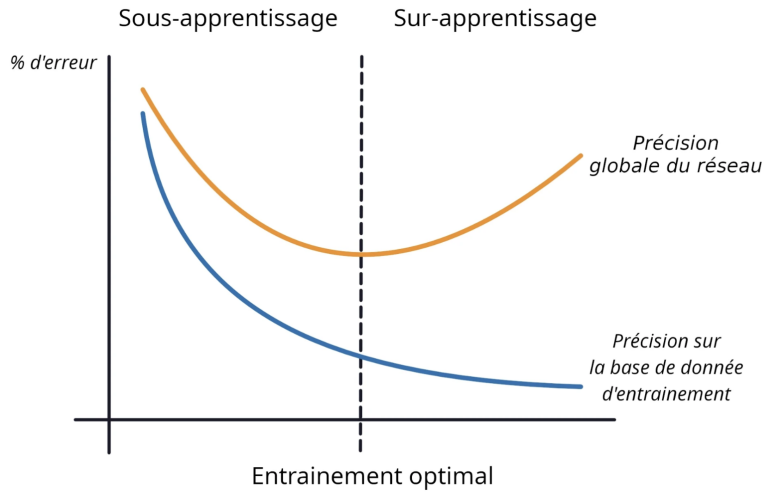


FIGURE 5 – Pourcentage d'erreur sur le jeu d'entraînement (en bleu) et le jeu d'évaluation (en orange).

1.2.2 Jeux de données (*Datasets*).

A Jeu d'entraînement, jeu de validation, jeu d'évaluation.

Quand on parle de jeux de données, on parle en fait de trois choses. Pour pouvoir contrôler l'apprentissage et les performances d'un algorithme il est nécessaire de séparer le jeu de données en :

1. Un *jeu d'entraînement* servant à l'ajustement des paramètres du modèle, c'est à partir de la différence entre les labels du jeu d'entraînement et la sortie de l'algorithme que l'erreur de la fonction coût est rétropropagée dans le réseau.

2. Un *jeu de validation* qui sert à contrôler que le modèle "apprenne bien". Pendant la phase d'apprentissage, une petite partie des données (le jeu de validation) est utilisée pour s'assurer que l'algorithme généralise bien à des données qu'il n'a jamais vues et qu'il n'est pas en phase de *surinterprétation* (cf. figure 5). La surinterprétation est la partie à droite du point d'abscisse correspondant à "l'entraînement optimal". La précision de l'algorithme augmente sur le jeu d'entraînement, mais la précision globale du réseau diminue.

3. Un *jeu d'évaluation* servant à évaluer les performances du modèle final. Une fois que la phase d'entraînement est terminée, l'ensemble des paramètres du modèles sont stockés dans un fichier *checkpoint*. Le modèle est ensuite évalué sur des données qu'il n'a jamais vu afin de mesurer ses performances.

Une répartition standard des données consiste à séparer l'ensemble des données en jeux de proportions 60%, 20% et 20% respectivement pour le jeu d'entraînement, le jeu de validation et le jeu d'évaluation. La même donnée ne peut pas être présente

à la fois dans plusieurs jeux. Il est donc très important de bien séparer les données dès le début.

B Les jeux de données existants pour la CSE.

Il existe nombre de jeux de données pour la tâche de classification des sons environnementaux. La plateforme DCASE [34] rassemble une grande partie des jeux de données *open-source*. Nous en présentons ici quelques uns.

- *UrbanSound8k* (Salamon *et al.*, 2014) : Ce jeu de données spécialisé dans les sons de villes contient 8732 fichiers de moins de 4 secondes et s'accompagne d'une hiérarchisation divisée en 10 classes. (Climatiseur, klaxon, enfants jouants, aboiement de chien, perceuse, moteurs, coups de feux, marteau-piqueur, sirène et musique de rue).

- *ESC-50* (Piczak, 2015) : Ce jeu de données est un exemple en terme de rigueur et d'uniformisation, il comprend 2000 sons de 5 secondes chacun répartis en 50 classes (40 sons par classe). Les 50 classes sont elles mêmes regroupées par 10 pour former les 5 catégories majeures suivante : Animaux, Paysages naturels et sons d'eau, sons humains (pas de parole), Sons domestiques/intérieurs, Bruits urbains/extérieurs.
- *FreeField1010* (Stowell & Plumbley, 2013) : Un jeu de données collecté à partir du projet collaboratif *Freesound* [35] qui regroupe plus de 600 000 fichiers sons libres de droits enregistrés ou synthétisées à travers le monde entier. *FreeField1010* est un jeu de données contenant 7690 audios de longueurs variables répartis selon les 7 classes suivantes : Oiseaux, Ville, Nature, Humains (pas de parole), Voix, Train, Eau. Il est destiné à l'analyse des paysages sonores et des prises de son en extérieur, ce qui fait de lui une ressource de choix dans le sens où elle intègre une approche audiovisuelle. Toutefois, la segmentation proposée ici ne convient pas aux besoins du monteur son, elle est trop peu précise. En effet, il existe bon nombre de paysages sonores extérieurs et il est évident qu'un désert ne sonne pas comme un bord de mer, et qu'une cascade en montagne ne ressemble à l'écoulement tranquille d'un ruisseau. Or, au sein de *FreeField1010*, toutes ces ambiances seraient regroupés dans la catégorie Nature.

Comme nous l'avons montré ci-dessus, la plupart de ces jeux de données sont spécialisés et ne peuvent donc pas répondre à la question d'une *classification générale*

34. https://dcase.community/
35. https://freesound.org/

des sons environnementaux. Dans le paragraphe suivant, nous nous intéresserons à un jeu de données couvrant la quasi totalité des sons environnementaux.

C Audioset : Le plus large jeu de données pour la classification des sons environnementaux.

Audioset (Gemmeke *et al.*, 2017) est une classification de la quasi-totalité des sons environnementaux développée et maintenue par Google. À l'instar des arbres phylogénétiques du vivant, cette ontologie mentionne des parents, des enfants et un certain nombre de ramifications. L'organisation des sons se fait selon 768 concepts parmi lesquels on dénombre 527 classes finales. Il est assez difficile de représenter toutes les liaisons regroupant ces concepts, ce faisant nous n'illustrerons, par la suite, que certaines parties de l'ontologie qui nous semblent pertinentes. Nous invitons par ailleurs notre lecteur à visiter le site d'Audioset [36], qui présente l'ontologie de manière détaillée et des exemples sonores pour chaque catégories de sons.

Il y a cependant un certain nombre de limites propres à la typologie Audioset, en vue d'une utilisation par le monteur son. D'un point de vue mathématique, l'ontologie respecte une structure dite de graphe orienté acyclique (GOA), et ceci a plusieurs conséquences. Premièrement, c'est une ontologie à *profondeur variable*, c'est-à-dire que les classes finales ont un nombre variables d'ancêtres. Certaines ont une généalogie courte : *Speech* → *Whispering*, d'autres ont des arrières-grands-parents : *Music* → *Musical instrument* → *Keyboard instrument* → *Piano*. Ensuite, c'est une catégorisation à héritage multiple, c'est-à-dire qu'il existe plusieurs chemins dans le graphe pour arriver à la même classe de fin, par exemple : *Bicycle Bell* peut appartenir à la fois à *Bell*, *Alarm* et à *Vehicle* (6).

Enfin, le but de cette ontologie n'est pas clairement identifié, en effet si on peut d'abord penser que les sons sont regroupés par sources, on trouve dans Audioset, des classes comme la division en *moods* musicaux selon la taxonomie de Gaver développée dans le paragraphe Annexe (A.1.2), ou encore les *Onomatopoeias* (onomatopées) qui sont des classes dont le nom imite les sonorités appartenant à la classe (*"Crack", "Hiss", "Hum", "Boing", "Zing", etc*). On perd dans ce cas la source causale du son en faveur d'un critère morphologique. En ce sens, cette ontologie n'applique pas de manière stricte l'approche causale détaillée dans le paragraphe D. Par exemple, la classe de sortie *"Hiss"* (Sifflement) [37] pourra indiquer au choix de la vapeur d'eau, un serpent ou un chat, cf figure 7. Le même label peut représenter plusieurs sources causales ce qui est en rupture avec les recommandations décrites précédemment.

Une ontologie, ou plutôt les labels permettant de construire celle-ci, devrait ré-

36. https://research.google.com/audioset/ontology/index.html
37. Définie dans l'ontologie Audioset par : Un son fricatif, comme un chat qui donne un avertissement.

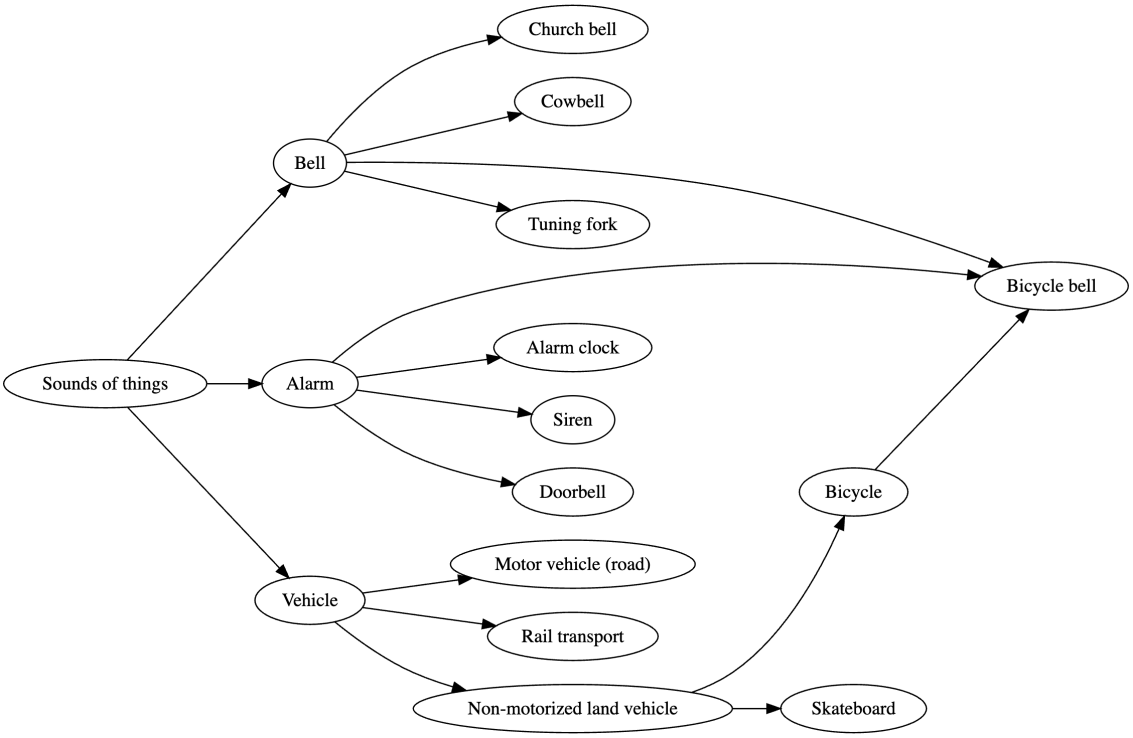


FIGURE 6 – Les différents chemins du graphe orienté de l'ontologie Audioset menant à la catégorie *Bicycle Bell*.

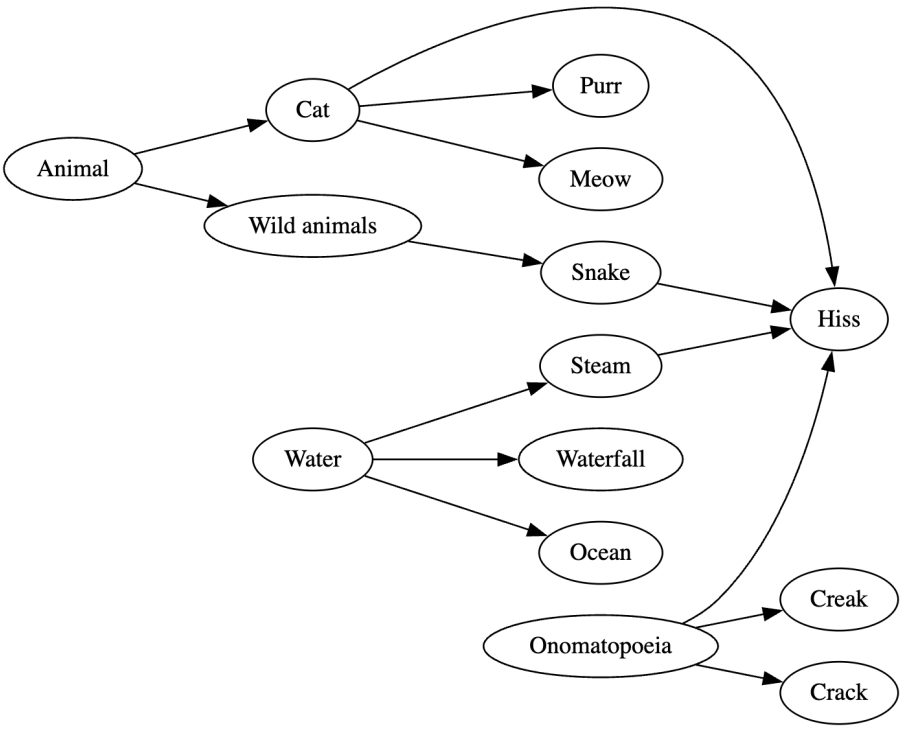


FIGURE 7 – Différents chemins menant à la classe *"Hiss"* dans l'ontologie Audioset

pondre aux deux critères que sont la *représentation* et la *non-ambiguité* (Virtanen *et al.*, 2018). Un label doit être un bon descripteur de l'évènement sonore (représentation) et avoir une correspondance 1 pour 1 claire avec le type de son qu'il décrit (non-ambiguïté). Il est dès lors évident que l'ontologie Audioset n'est pas idéale pour une application intégrale et exhaustive au métier du montage son puisqu'à un label, elle associe plusieurs types de sons de natures parfois très différentes (figure 7). Aujourd'hui on retrouve, dans la CSE, des applications variées utilisant toutes l'ontologie Audioset, la plus détaillée à ce jour.

Ce qui fait avant tout la force d'AudioSet (Gemmeke *et al.*, 2017), c'est qu'il s'agit, à l'heure actuelle, du jeu de données le plus grand pour la tâche de classification de sons environnants. Ce *dataset* contient plus de 5600 heures de médias extraits de 2 084 320 vidéos YouTube différentes. Comme le montre le tableau 1.2, la durée en minutes des éléments d'Audioset est au moins 100 fois plus grande que celle des autres jeux de données. Le tableau 1.2, mentionne en colonne 2, le type

Jeu de données	**Type**	**Classes**	**Fichiers**	**Durée (min)**
Sons environnementaux				
ESC-10	col	10	400	33
ESC-50	col	50	2000	166
NYU Urban Sound8K	col	10	8732	525
CHIME-Home	rec	7	6137	409
Freefield1010	col	7	7690	1282
CICESE Sound Events	col	20	1367	92
AudioSet	col	527	>2M	>340k
Scènes acoustiques				
Dares G1	rec	761	3214	123
DCASE 2013 Office Live	rec	16	320	19
DCASE 2013 Office Synthetic	syn	16	320	19
TUT Sound Events 2016	rec	18	954	78
TUT Sound Events 2017	rec	6	729	92
NYU Urban Sound	col	10	3075	1620
TU Dortmund Multichannel	rec	15	1170	585

Tableau 1.2 – Comparaison des datasets pour les sons environnementaux et les événements sonores (Virtanen *et al.*, 2018).

d'acquisition des données : *collectées*, *enregistrées*, ou *synthétisées*. Il est important de noter que pour des données collectées, les enregistrements existent déjà et sont simplement regroupés au sein d'un dataset. Ceci limite le "coût" de la donnée et permet d'avoir accès à des plus grandes quantités de données. On remarque en effet que la majorité des *datasets* collectés, sont plus grands que ceux enregistrés.

Audioset, est un jeu de données collecté, ce qui signifie que les sons n'ont pas été produits dans le but précis de constituer un *dataset*. Ceci a deux conséquences, la

première est positive, le jeu de données est énorme et a pu être constitué à moindre coût, des annotateurs se sont occupés de mettre des labels sur les quelques deux millions d'extraits sonores, plutôt que de les enregistrer, ce qui aurait été beaucoup plus demandant en temps et en ressources matérielles. La deuxième conséquence est plus discutable, elle concerne l'isolement des sources traitées. Bien souvent, les sons environnementaux se superposent à d'autres sources sonores, car les extraits proviennent de vidéos, pour l'immense majorité amateures. De ce fait un *bruit de pas* se superpose à la *voix* d'un guide touristique largement au premier plan sonore, ou alors, une *fermeture de placard* se distingue à travers une *musique* extradiégétique[38] 15 dB plus forte ...

1.2.3 Principe de généralisation.

Nous venons d'identifier que, dans le cas d'un apprentissage statistique, un algorithme apprend à travers des exemples constituant un *jeu de données (dataset)*. Il paraît évident que, même si le jeu de données est gigantesque, il ne pourra jamais représenter la totalité des situations auxquelles l'algorithme pourra être confronté. Dès lors, un des buts fondamentaux de l'apprentissage machine est la *généralisation* à des données inconnues. Une bonne machine est celle qui est capable de généraliser ce qu'elle a appris et de l'appliquer à des entrées qu'on ne lui a jamais présentée. Si calculer l'erreur de généralisation est un problème mathématique difficile, il est néanmoins possible de l'estimer en effectuant une phase de test de notre algorithme sur des données qu'il n'a jamais vues. Le jeu de données est alors séparé en deux sous-ensembles, un *ensemble d'apprentissage* et un *ensemble d'évaluation*. La performance de l'algorithme peut-être évaluée sur ces deux ensembles, le but étant de maximiser la précision de la machine sur les deux.

Le phénomène de *surinterprétation (overfitting)*[39] a lieu lorsqu'un algorithme performe très bien sur un jeu d'entraînement mais a du mal à généraliser ses résultats sur des données qu'il n'a jamais vues. Il apprend en fait des particularités des données de son jeu d'entraînement plutôt que d'apprendre des caractéristiques générales inhérentes à chaque classe. C'est la raison pour laquelle dans la figure 2, la séparatrice ne sépare pas 100 % des triangles et des ronds. L'algorithme dans ce cas ne se base pas sur des détails propres à son jeu d'entraînement mais sur des caractéristiques plus globales propres à chaque classe. Des techniques comme le *dropout* (mise à 0 de certains poids de manière aléatoire) servent à empêcher la surinterprétation lors de la phase d'apprentissage.

Au contraire, si le modèle n'est pas assez complexe, *i.e.* s'il ne possède pas assez

38. cf. Glossaire D.1

39. On trouve aussi le terme de "sur-apprentissage" dans la littérature française mais le mot est horriblement mal choisi.

de paramètres, il ne sera pas capable de représenter toutes les subtilités qui séparent les classes et sera donc en cas de *sous-interprétation (underfitting)*.

Les concepts de base de l'apprentissage automatique ayant été abordés, nous détaillerons dans la partie suivante l'étude approfondie d'un classifieur audio algorithmique.

1.3 Étude du classifieur audio BEATs.

Avant d'entamer la lecture de ce chapitre, nous invitons toute personne non familière avec les architectures basiques de réseaux de neurones à prendre connaissance des outils détaillés dans l'Annexe B.

Dans cette section, nous regarderons en détail le fonctionnement du classifieur audio BEATs (S. Chen *et al.*, 2023), il s'agit à ce jour et à notre connaissance du modèle offrant les meilleures performances en CSE, atteignant une précision de 98,1% sur le jeu de données ESC-50 et une précision moyenne (mean-average precision) de 0,506 sur Audioset.

L'architecture neuronale de BEATs est constituée de transformeurs. Les transformeurs sont des combinaisons de plusieurs blocs de traitement entraînables, chaque bloc ayant un rôle spécifique, et sont, le plus généralement, organisés de la manière suggérée en figure 8. Nous détaillerons dans la suite de ce paragraphe l'utilité et le fonctionnement des blocs principaux.

1.3.1 Architecture générale d'un réseau transformer.

A La discrétisation (*Tokenizing*).

Pour comprendre ce qu'est un *tokenizer*, il faut d'abord comprendre ce qu'est un *token*, dit symbole (parfois jetons) en français. Ces symboles sont des petites briques de données, qui, combinées ensemble et mis les uns à la suite des autres, forment des structures plus complexes. On prend souvent l'exemple des larges modèles de langage (LLM) qui décomposent des phrases complexes en mots ou morceaux de mots. Ces *tokens* sont regroupés dans un *codebook*, un répertoire qui les recense. Pour faire simple, prenons l'exemple du LLM, le *tokenizer* décompose un message d'entrée en une succession de symboles parmi ceux contenus dans son dictionnaire. Ces *tokens* correspondent à la forme de données comprises et utilisées par le transformeur.

Dans le cas de BEATs, les données audios sont prétraitées pour être mises sous forme de vecteurs de mels à 128 dimensions. Ces vecteurs sont ensuite regroupés par blocs de 16 appelés *patches*, c'est la phase de *tokenisation* (cf. 1.3).

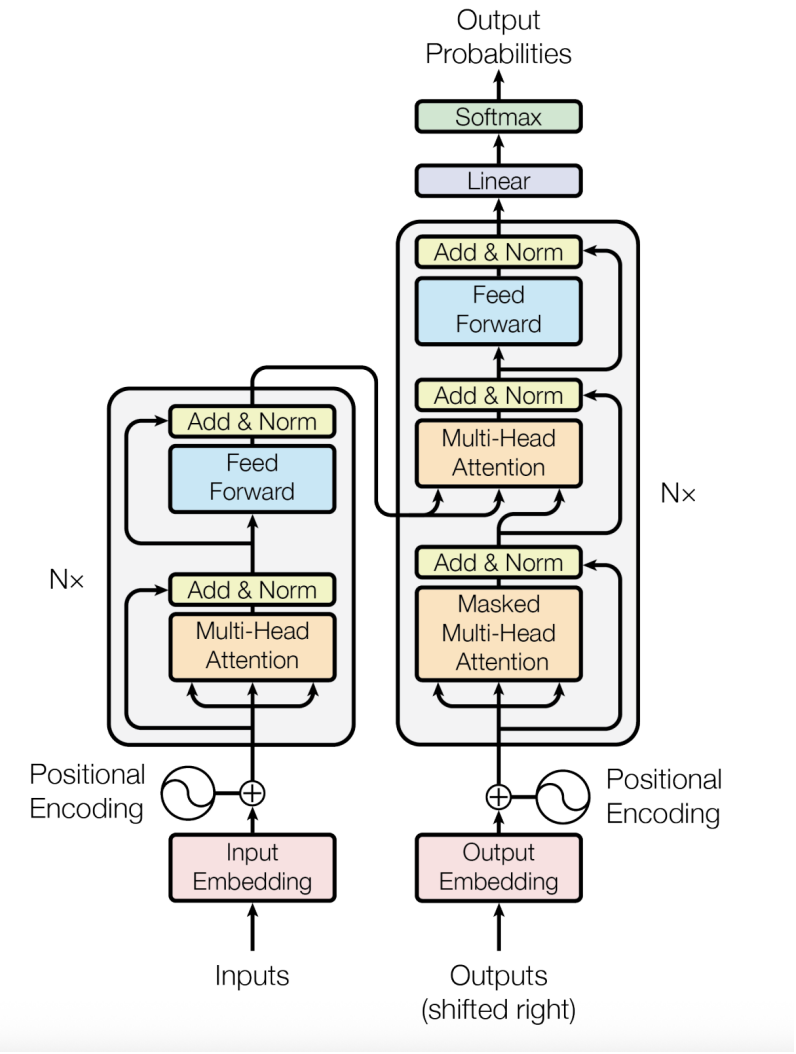


FIGURE 8 – Architecture générale d'un modèle transformer (VASWANI *et al.*, 2017).

$$\underbrace{\begin{bmatrix} x_1^{(1)} \\ x_2^{(1)} \\ \vdots \\ x_{128}^{(1)} \end{bmatrix}}_{\text{1 frame}} \xrightarrow{Tokenisation} \underbrace{\begin{bmatrix} x_1^{(1)} \\ x_2^{(1)} \\ \vdots \\ x_{128}^{(1)} \end{bmatrix} \begin{bmatrix} x_1^{(2)} \\ x_2^{(2)} \\ \vdots \\ x_{128}^{(2)} \end{bmatrix} \cdots \begin{bmatrix} x_1^{(16)} \\ x_2^{(16)} \\ \vdots \\ x_{128}^{(16)} \end{bmatrix}}_{\text{1 patch = 16 frames}} \tag{1.3}$$

B Le plongement (*Embedding*).

Ensuite, on trouve un bloc dit de plongement (*embedding*), en fait, c'est un bloc qui permet de passer d'un langage humain à un langage machine, c'est-à-dire des vecteurs de la dimension de la couche d'entrée vers la dimension d'un espace latent abstrait. Il est à noter que deux *tokens* similaires auront des vecteurs de plongement voisins, avec peu de différences numériques pour chacune de leurs dimensions. Concrètement, pour un LLM, le plongement est le moment ou les bouts de mots

sont transformés en vecteurs de grandes dimensions. Les *embeddings* de BEATs ont une dimension de 768. Pour aller plus en détail, les *tokens* = *patches* de BEATs ont une dimension de 2048, le plongement consiste en la projection linéaire d'un espace de dimension 2048 vers un espace de dimension 768.

C Codage de la position.

À la différence des réseaux récursifs (*recursive neural network* RNN) qui traitent l'information par blocs en répétant le même schéma à chaque itération, les transformeurs, eux, reçoivent tous les symboles d'un seul coup. Il devient alors impératif d'encoder une information de position temporelle, qui jouera un rôle important dans le procédé d'attention que nous détaillerons plus tard. La première chose à faire consiste à donner un indice à chaque *token*, $\{0, 1, 2 \dots\}$, il existe ensuite plusieurs manières d'encoder le position des *tokens* dans un fichier, qu'il s'agisse d'un texte, d'un son ou d'une image.

L'encodage de position sinusoïdal

Une des formes les plus triviales d'encodage de position proposée par Vaswani *et al.* (2017) consiste à construire un vecteur position PE_p de la taille de l'espace de plongement d_{model}.

p désigne alors la position du *token* dans la représentation de départ et i un entier tel que $i \in \{0, 1, 2, \dots, \frac{d\mathrm{model}}{2} - 1\}$.

$$\mathrm{PE}p[2i] = \sin\left(\frac{p}{10000^{2i/d\mathrm{model}}}\right), \quad \mathrm{PE}p[2i+1] = \cos\left(\frac{p}{10000^{2i/d\mathrm{model}}}\right) \tag{1.4}$$

Le vecteur de position PE_p est construit en itérant sur l'indice de fréquence i. *L'encodage positionnel sinusoïdal* projette chaque position p dans un espace vectoriel réel de dimension d_{model}, où chaque paire de dimensions correspond respectivement à la partie réelle et imaginaire d'une exponentielle complexe $e^{i\omega p}$, avec différentes fréquences ω.

On rappelle :

$$e^{i\omega p} = \cos(\omega p) + i\sin(\omega p)$$

On peut donc interpréter cet espace comme une base réelle associée à un sous-espace de Fourier, ce qui permet de définir un produit scalaire, une norme euclidienne, et donc une mesure de distance entre positions (figure 9).

Cet encodage des positions a plusieurs avantages, d'abord il permet de projeter une grande quantité d'informations sur une plage $[-1, 1]$ (l'espace image des fonctions cosinus et sinus). Par ailleurs, il permet de mettre à distance deux symboles éloignés, tout en rapprochant deux symboles proches (cf. figure 9).

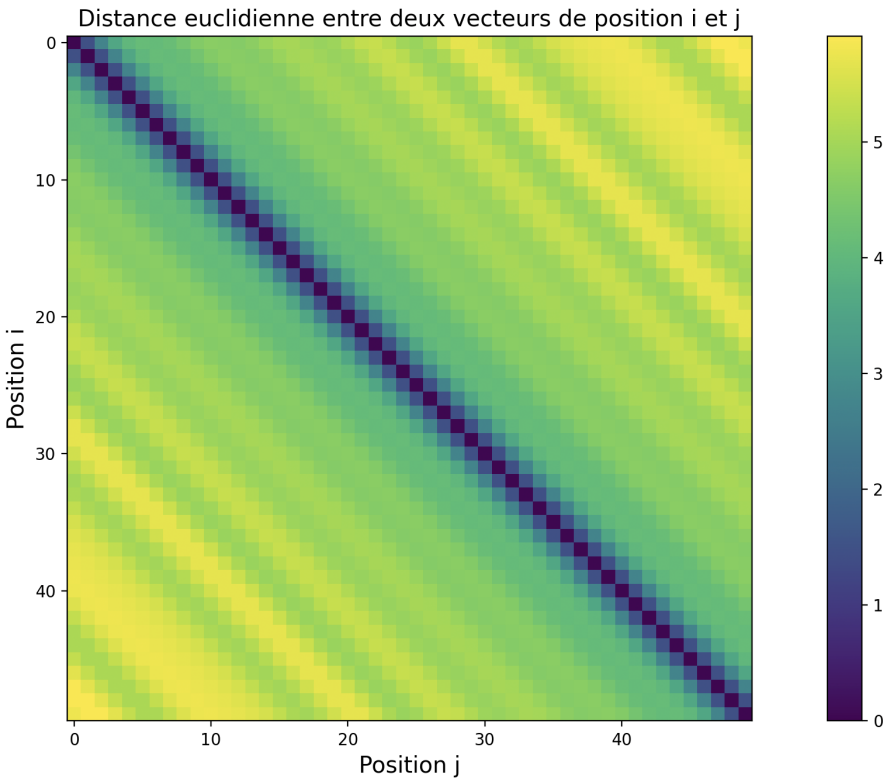


FIGURE 9 – Valeur de la norme $\mathcal{L}_2$ entre deux vecteurs positions.

Enfin, ce vecteur position est tout simplement ajouté au vecteur d'entrée.

$$e_t = x_t + \mathrm{PE}_t \tag{1.5}$$

e_t est ce qu'on appelle le vecteur de plongement (*embedding vector*), il contient à la fois les caractéristiques acoustiques extraites et l'information de position.

Néanmoins, il y a trois inconvénients majeurs à cette méthode qui nous empêchent de l'appliquer au cas du son.

1. D'abord, ce codage de position n'est pas *entraînable*, il est fixé et n'est pas capable de s'adapter en fonction des données d'entrée. Il est en effet très probable que des données aussi variées que l'ensemble des sons environnementaux, nécessitent des distances variables entre les jetons d'entrée. Pour prendre un exemple parlant, un moteur reste un moteur, qu'il tourne à 800 tr/min ou 6000 tr/min. Alors l'espace temporel entre deux *tokens* d'entrée identiques (considérons le régime du moteur cyclique et permanent) n'est pas le même selon que le moteur tourne à fréquence haute ou fréquence basse. Néanmoins les deux sons sont produits par le même objet. Cela donne une bonne intuition de pourquoi il est préférable d'avoir des encodages de positions entraînables.

2. Deuxièmement, l'encodage sinusoïdal donne une position *absolue* à chaque jeton d'entrée. Les éléments à la fin et au début du fichier sont traités différemment par les couches du transformeur. Il y a donc un biais qui pourrait amener à favoriser ou défavoriser un *pattern* audio selon qu'il soit au début ou à la fin de la séquence d'entrée. Or, il n'y a pas de raisons de procéder comme tel pour le son, et plus particulièrement pour l'analyse des sources acoustiques, ces dernières demeurant invariantes par translation dans le temps (cf. B).

3. Ce qui s'avère plus pertinent pour apporter du *contexte* aux sons, c'est de relier un *token* à ses plus proches voisins, de sorte que l'information de contexte soit uniquement *relative*.

Encodage de position à convolution

Une méthode qui satisfait aux critères précédents, est *l'encodage positionnel à convolution*, c'est celui utilisé dans le modèle BEATs. Il s'agit en fait de créer un vecteur position à partir d'un réseau de neurones à convolution. Ces noyaux de convolution sont ajustés par ce qu'on appelle une *tête d'attention*[40] qui est une matrice indiquant les poids des relations entre les *tokens* en fonction de leurs positions respectives. Les paramètres des noyaux de convolutions du réseau de neurones sont la résultante d'un entraînement, le réseau apporte généralement plus d'attention aux éléments qui évoluent dans un voisinage temporel comme le montre la figure 10.

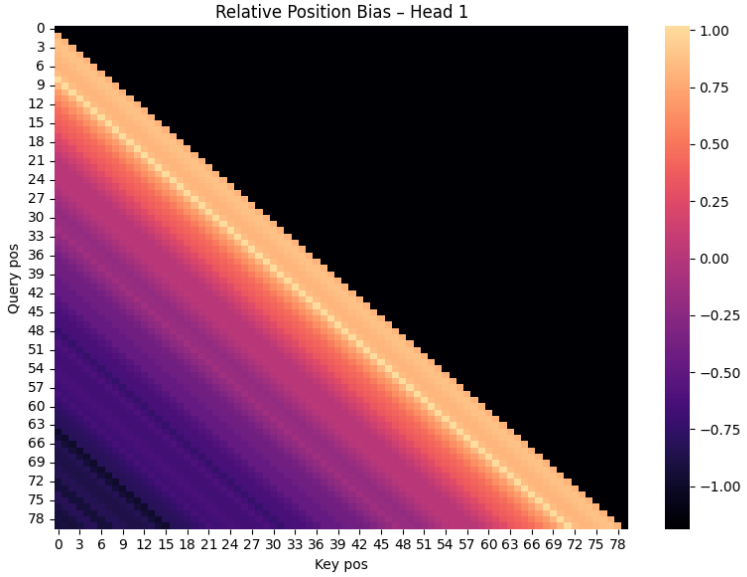


FIGURE 10 – Corrélation entre les différents symboles selon leur espacement temporel, *tête d'attention n°1 de BEATs*.

40. cf. Glossaire D.1

D Bloc transformer.

Le bloc transformeur est, à proprement parler, le bloc de traitement d'une architecture transformeur, de tels blocs sont des réseaux de neurones composés de 2 mécanismes :

1. *Le procédé d'attention* : c'est ce qui permet au transformeur d'avoir une sorte de *"mémoire"*, concrètement, il permet au réseau de donner plus ou moins de poids aux *tokens* en fonction de leur contexte. Qu'il s'agisse de texte ou bien de son, le contexte temporel joue une part importante de la compréhension. Considérons un son impulsif, comme un coup d'arme à feu, dans un milieu très réverbérant. Le bruit blanc qui suit l'impulsion ne doit pas être considéré comme la source causale d'un son mais comme la conséquence du tir dans une acoustique particulière. Et ceci change tout quand à la détermination de la source réelle du son, dans un cas la source sonore attribuée est une arme à feu, et dans l'autre, un vent ou une pluie (ou tout autre son s'apparentant à un bruit blanc). On observe donc ici l'importance du contexte temporel.

2. Le procédé normal *feedforward* : C'est le traitement à travers un réseau de neurones type perceptron multicouche [41]. Le réseau a comme données d'entrée les vecteurs de sortie du bloc d'attention.

Le cœur d'un réseau transformeur consiste en une succession de plusieurs blocs transformeurs les uns à la suite des autres, généralement au moins une dizaine ; BEATs en possède 12.

1.3.2 Architecture de BEATs.

Appliquer les modèles performants pour la parole directement au domaine des sons environnementaux donne des résultats assez peu efficaces d'après Chong *et al.* (2022).

L'approche proposée dans (S. Chen *et al.*, 2023) consiste à entraîner un réseau auto-supervisé, non pas sur des caractéristiques acoustiques, comme il est courant de le faire, mais sur des labels sémantiques, contenant plus d'informations globales sur les sons, et donc se rapprochant d'un *critère de discrimination sémantique*. L'apprentissage se concentre sur des caractéristiques *haut-niveau* plutôt que sur des caractéristiques physiques et acoustiques *bas-niveau*. Une chose intéressante à remarquer est que l'humain fonctionne aussi de cette manière, en stockant, en extrayant et en regroupant des informations de niveau sémantiques et non des caractéristiques physiques (Patterson *et al.*, 2007).

41. Cf. A pour plus de détails sur les réseaux types

BEATs est un modèle auto-supervisé, qui bénéficie de certains avantages par rapport à un modèle d'apprentissage supervisé classique. La méthode auto-supervisé a un statut un peu ambigu dans la littérature, il est parfois identifié à l'apprentissage non-supervisé et parfois vu comme une sous-partie de ce dernier. On distingue habituellement :

- *Les méthodes supervisées* : dont le jeu de données de départ est annoté avec des labels.
- *Les méthodes non supervisées* : dont le jeu de données de départ n'est pas annoté de labels.

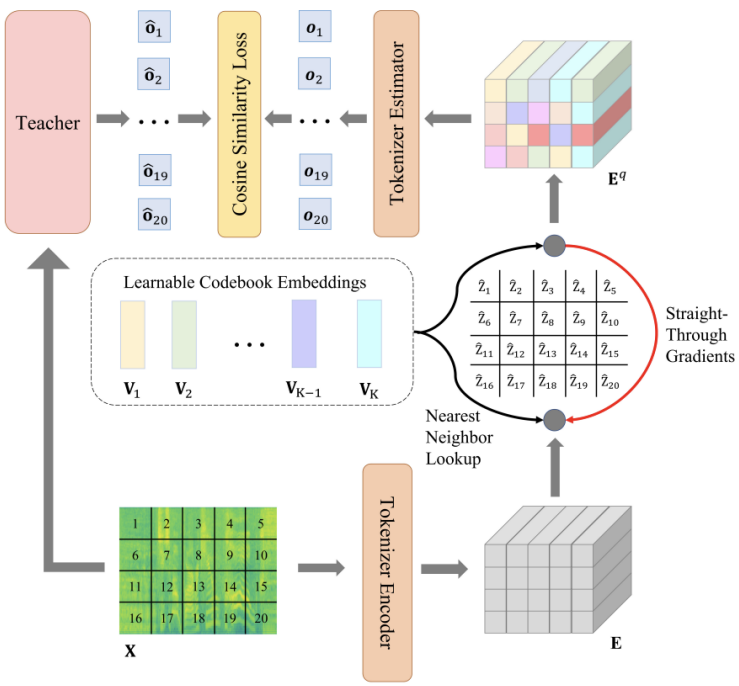
Néanmoins, au sein de ce dernier paradigme, on peut considérer un algorithme qui serait capable de poser un label sur un objet (le *tokenizer* acoustique), ceci pendant qu'un autre algorithme apprend une représentation latente du même objet (*self-supervised learning (SSL) audio model*). C'est ce principe qui est utilisé dans BEATs où le *SSL audio model* et le *tokenizer* acoustique apprennent de manière conjointe.

Commençons par détailler le fonctionnement de ces 2 algorithmes. D'abord, on trouve le *tokenizer* acoustique, c'est un Encodeur-Décodeur. L'encodeur est un réseau de neurones transformeur à 12 couches qui prend en entrée des *patchs*. Ces *patchs* correspondent à la concaténation des vecteurs de puissances spectrales sur 16 fenêtres temporelles [42] (*frames*). Ils forment la séquence de vecteurs $\mathbf{X} = \{\mathbf{x_t}\}_{t=1}^{T}$. Le rôle de l'encodeur est de transformer ces vecteurs d'entrée de dimension $16 \times 128 = 2048$ en une représentation latente de dimension 768 définie par la séquence de vecteurs $\mathbf{E} = \{\mathbf{e_t}\}_{t=1}^{T}$ (phase d'*embedding*). Ensuite, pour chaque $\mathbf{e_t}$ on lui associe un vecteur $\mathbf{v_i}$ du *codebook* $\mathbf{V} = \{\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_k}\}$ par un algorithme du plus proche voisin. Le *codebook* est une sorte de dictionnaire, une banque de vecteurs qui contient des vecteurs servant à quantifier les données (phase de *tokenisation*). On note la sortie de l'encodeur discrétisée (*quantized*) $\mathbf{E^q} = \{\mathbf{v}_{z_t}\}_{t=1}^{T}$ où $z_t = \arg\min_i ||\mathbf{e_t} - \mathbf{v_i}||_2^2$ est le numéro du vecteur dans le *codebook* le plus proche de l'entrée $\mathbf{e_t}$.

La partie décodeur utilise comme vecteur d'entrée la séquence $\mathbf{E^q}$ et en fournit une représentation discrète $\{\mathbf{o_t}\}_{t=1}^{T}$, cette sortie étant celle comparée avec la sortie du SSL audio model.

Le *self-supervised learning audio model* est basé sur une architecture de réseau du type *Visual Transformer (ViT)*, bien qu'il soit utilisé pour de l'audio, l'algorithme traite en fait des images, ou plutôt des morceaux d'images. Ces *patches* t sont la concaténation de 16 décompositions instantanées en spectrogramme de Mel $\mathbf{X}$ représentant le son. Le *ViT* prend donc en entrée une suite de vecteurs $\mathbf{x_t}$, les projette

42. Les vecteurs de puissances spectrales sont extraites sur 128 bandes spectrales, avec des fenêtres de 25 ms et un taux de recouvrement de 60%.



(b) Self-Distilled Tokenizer

FIGURE 11 – Technique d'apprentissage du *Self-distilled tokenizer* (S. CHEN *et al.*, 2023).

dans un espace à 768 dimensions, et obtient alors une séquence de vecteurs intégrés (*embedded*) $\mathbf{E} = \{\mathbf{e_t}\}_{t=1}^{T}$. Ces vecteurs forment l'entrée de l'encodeur-décodeur, l'encodeur transforme les *embedded vectors* dans une représentation latente $\mathbf{R} = \{\mathbf{r_t}\}_{t=1}^{T}$ qui servira d'entrée au module décodeur. En sortie du décodeur on obtient une prédiction de label de niveau sémantique $\mathbf{\hat{o}_t}$ pour chacun des *patches* t d'entrée.

La comparaison entre les deux modèles se fait via une fonction perte au niveau des labels sémantiques. La *cross entropy function* calcule la distance entre la prédiction du *SSL model* $\mathbf{\hat{o}_t}$ et la prédiction du *tokenizer* $\mathbf{o_t}$.

1.3.3 Entraînement de l'algorithme BEATs.

L'apprentissage de BEATs se fait en trois phases distinctes : L'initialisation, le pré-apprentissage, et l'ajustement [43].

1. Initialisation : dans le cadre de BEATs, l'initialisation se fait de manière aléatoire (*cold start*), les vecteurs du *codebook* sont initialisés aléatoirement ainsi que les poids et biais du *ViT*.

2. Le pré-apprentissage : c'est la phase la plus longue et la plus complexe, comme il s'agit d'un algorithme auto-supervisé basé sur 2 réseaux de neurones, ils ne

43. cf. Glossaire D.1

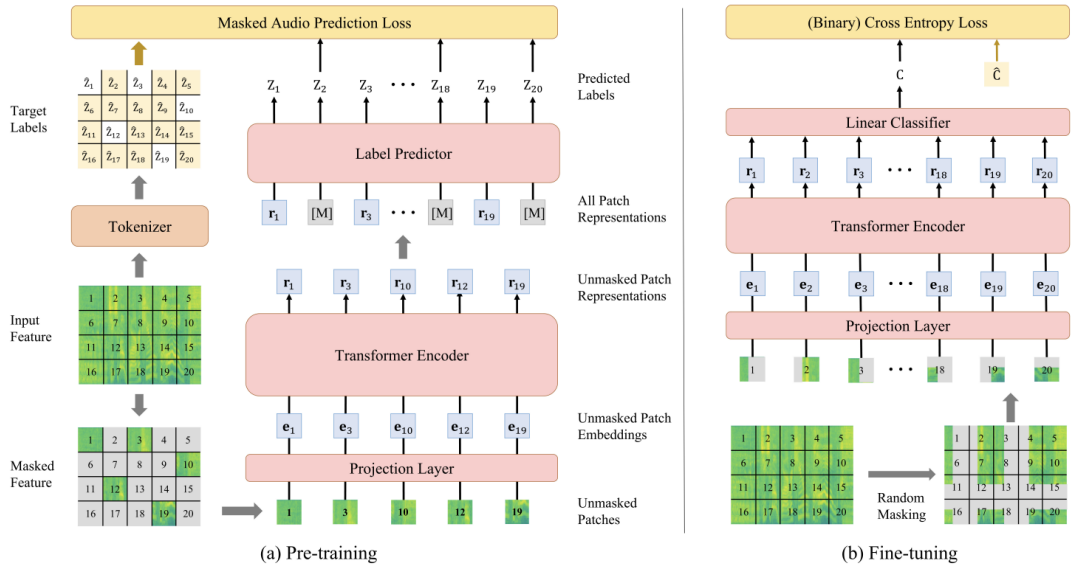


FIGURE 12 – Technique d'apprentissage du SSL audio Model (S. CHEN *et al.*, 2023).

peuvent pas apprendre "en même temps", ils apprennent de manière successive, chacun servant alternativement de "professeur" à l'autre.

Lorsque c'est le SSL qui apprend, ses poids sont ajustés grâce aux valeurs de la fonction perte qui sont rétropropagées dans le réseau. Le SSL apprend donc par rapport aux prédictions *tokenizer* acoustique qu'il considère comme la *ground-truth*. Notons que les chercheurs utilisent ici une méthode de masquage dont l'efficacité à été démontrée par HUANG *et al.* (2023). Le principe est de cacher une grande proportion (75-90%) du signal d'entrée. Un encodeur apprend une représentation compacte de la partie restante du signal. Un décodeur (modèle génératif) essaye de reconstruire le signal complet à partir de la représentation de l'encodeur comme illustré dans la figure 13.

Une fois que la fonction perte converge, le SSL et le *tokenizer* échangent leurs rôles, on entre alors dans la phase d'apprentissage du *tokenizer*. Ce dernier met à jour ses poids et biais, ainsi que les vecteurs du *codebook* par rétropropagation. Les deux étapes vont ainsi se répéter, chacun des algorithmes apprenant de l'autre successivement.

3. La phase d'ajustement : C'est l'ultime phase d'apprentissage, il faut bien comprendre qu'à ce stade, l'algorithme est capable de tâches de regroupement [44] et de séparation, il parvient à identifier de informations abstraites sans pouvoir les relier à une sémantique précise. Il est donc incapable de donner des informations compréhensibles par un humain. Le but de la phase d'ajustement est, par un apprentissage supervisé, d'insuffler à l'algorithme une sémantique par

44. cf. Glossaire D.1

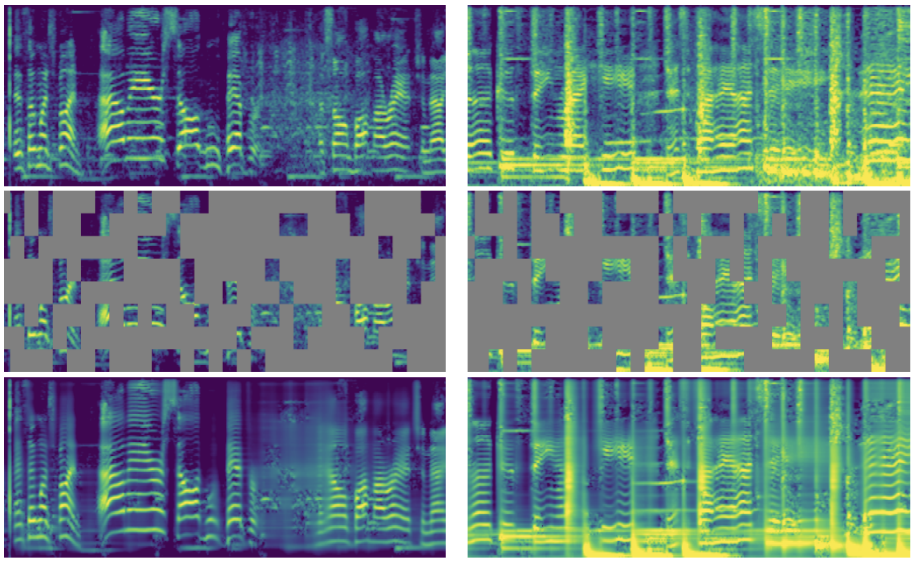


FIGURE 13 – Visualisation d'un auto-encodeur en utilisant un fort masquage aléatoire du spectrogramme. En haut le spectrogramme initial, au milieu le schéma de masquage, et en bas la visualisation de l'auto-encodeur à partir de l'entrée masquée (HUANG *et al.*, 2023).

le biais d'une ontologie. Ici, l'algorithme a été ajusté sur l'ontologie et le jeu de données Audioset, la portée sémantique des extraits audio se rattache donc essentiellement à la nature causale du son, comme nous l'avons vu dans le C.

1.3.4 Prétraitement des signaux.

Avant de commencer la phase d'apprentissage, il est important de faire du prétraitement sur les données dans le but d'améliorer l'extraction des caractéristiques acoustiques des signaux. Ce prétraitement peut servir à enrichir la diversité intra-classe du jeu de données, minimiser le bruit de fond ou apporter de la robustesse à l'algorithme en détériorant les données d'entraînement.

De manière plus pragmatique, c'est aussi le moment où on va uniformiser les sons, en terme de durée, de niveau, les mettre à la même fréquence d'échantillonnage, *etc.* Le prétraitement décrit aussi le procédé d'*extraction des caractéristiques*

A Uniformisation des données.

En ce qui concerne la phase de prétraitement de BEATs, les signaux sont d'abord uniformisés. Le niveau est la première variable à être alignée. Les données d'entrée sont amplifiés par un facteur 2^{15}. Notons qu'il s'agit de données numériques, il n'y a donc pas de saturation tant que l'on ne restreint pas les amplitudes en leur imposant une valeur maximale. Il y a une raison historique à cela. La plupart des jeux de données comme *AudioSet*, *ESC50*, *UrbanSound8k* et d'autres, sont stockés au format 16-bit, c'est-à-dire dans un intervalle entier $[\![-32768, 32767]\!]$. Ce sont ces

données qui sont utilisées pour entraîner l'algorithme. La bibliothèque python *librosa* que nous utilisons pour lire les fichiers, charge les audios en 32 bits flottants, dans un intervalle réel $[-1, 1]$. Pour ramener les valeurs numériques chargées par *librosa* sur les mêmes ordres de grandeurs que les données d'entraînement, il faut les multiplier par $32768 = 2^{15}$.

Une autre phase du prétraitement est l'uniformisation des données à une fréquence d'échantillonnage de 16 kHz. La bande passante est donc réduite à l'intervalle $[0, 8000]$ Hz. L'idée est ici d'avoir une représentation plus compacte des données, sur des sujets humains, des sources traitées avec un filtre coupe-haut, ne sont pas moins reconnues par les sujets. La conclusion de Gygi *et al.* (2004) est que l'information contenue dans le haut du spectre ne donne pas d'informations sur la source causale du son. BEATs utilise donc la partie du spectre utile à l'identification du son, tout en retirant plus 50% d'informations pas ou peu utiles à l'entraînement des neurones (la bande $[8000, 20000]$ Hz).

B Extraction des caractéristiques.

Les réseaux de neurones se basent rarement sur des représentations brutes [45] (*raw*), c'est-à-dire simplement la suite des échantillons audios ; mais plutôt sur des représentations de dimensions réduites des caractéristiques acoustiques des sons (c'est le cas des spectrogrammes, des décompositions en coefficients de Fourier, etc). Le but de cette extraction de caractéristiques est de condenser l'information utile dans un minimum d'espace tout en restant suffisamment riche pour effectuer la tâche demandée, on parle de *compression parcimonieuse*. Ceci rend la phase d'apprentissage moins gourmande en termes de calcul et permet de revoir la taille des réseaux à la baisse (Virtanen *et al.*, 2018). C'est ce qu'on appelle *l'extraction de caractéristiques à la main* (*hand-designed features*).

Si différentes manières d'extraire les caractéristiques acoustiques existent (cf. figure 14), il y a toujours trois phases qui constituent cette étape.

D'abord une étape de *blocage de la fenêtre* (*frame blocking*), où on considère une fenêtre audio *i.e.* un *buffer* [46], dont la taille est généralement comprise entre 20 ms et 60 ms. Cette fenêtre est bloquée pour qu'un traitement lui soit appliqué. Ensuite, une phase de *pondération de la fenêtre*, cela peut consister en une multiplication par une fonction de pondération de Blackmann-Harris, Hamming ou Hann, qui visent à annuler les effets de bords aux extrémités du *buffer* et limiter l'importance des lobes secondaires dans les spectres (Virtanen *et al.*, 2018). Enfin, on applique généralement une transformée en cosinus discrète (*discrete cosine transform*), une transformée de Fourier rapide (*fast fourier transform*), ou une

45. Bien que ce soit parfois le cas, cf. Annexes C.1
46. cf. Glossaire D.1

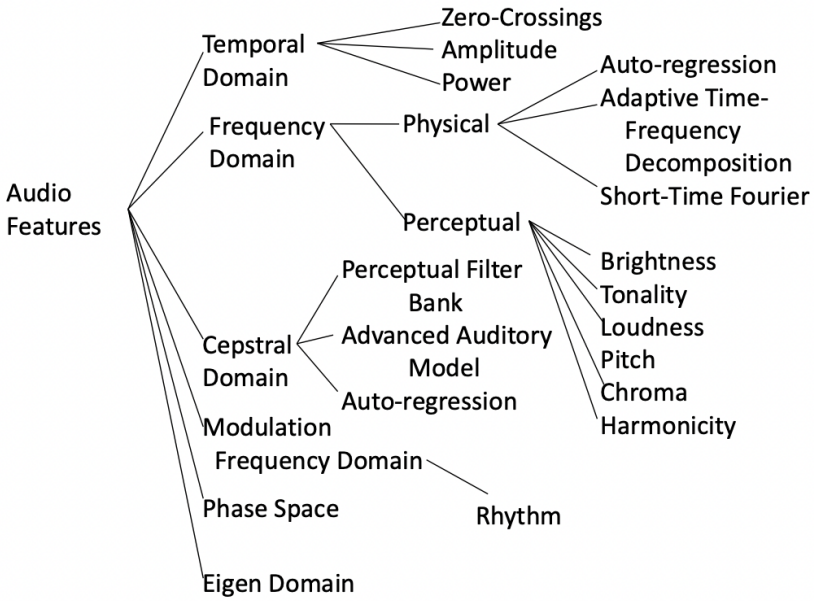


FIGURE 14 – Arbre représentant l'ensemble des *hand designed features* (CHACHADA & KUO, 2013).

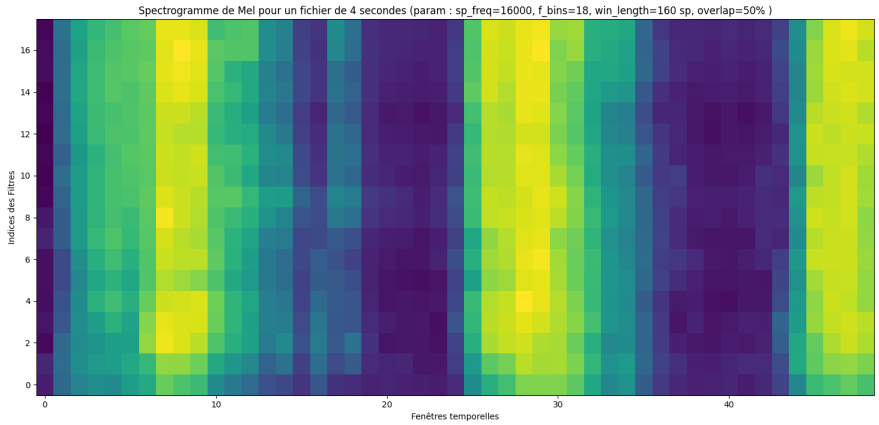
transformée en ondelette discrète (*discrete wavelet transform*) sur la fenêtre pour obtenir une *description fréquentielle* du signal dans la fenêtre. Cette succession de trois phases est effectuée pour chaque tranche du signal, et, en général, on procède à un *recouvrement "overlapping"* (superposition de fenêtres temporelles) de taux allant de 50% à 75% de la taille de la fenêtre pour essayer de conserver quelques traces de l'évolution dynamique à l'intérieur d'un buffer (VIRTANEN *et al.*, 2018).

Pour ce qui est de la classification de sources sonores, les travaux de ABAYOMI-ALLI *et al.* (2022) montrent que, la représentation basée sur le log Mel spectrogramme est une des méthode d'extraction de caractéristique les plus courantes, utilisée dans 18,6% des cas.

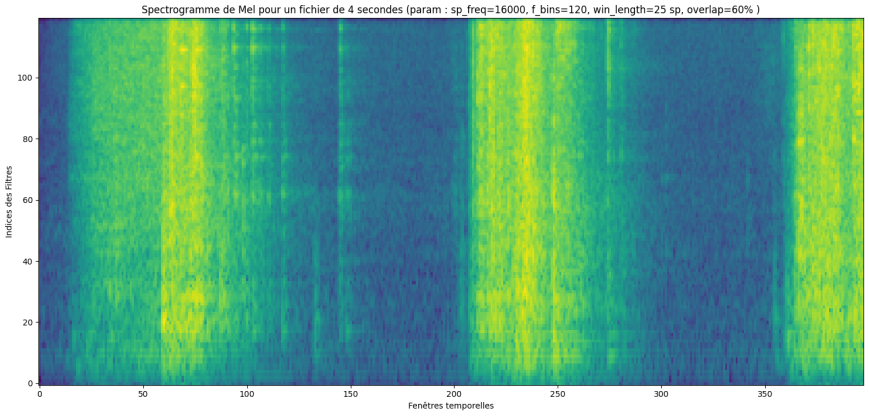
Spectrogramme de Mel

L'échelle de Mel divise les fréquences en bandes étroites espacées de manière logarithmique. Contrairement aux coefficients de Fourier qui sont répartis linéairement, les bandes de Zwicker permettent une meilleure précision dans les basses fréquences et sont plus espacées et larges dans les hautes fréquences, raison pour laquelle on dit qu'elles simulent le ressenti humain. Cette description permet par ailleurs d'utiliser moins de coefficients qu'une description de type Fourier (un nombre de fréquences égal au produit de la durée de la fenêtre en secondes par la fréquence de Nyquist ; par exemple 1323 coefficients complexes pour un buffer de 60 ms et une fréquence d'échantillonnage de 44.1 kHz).

Le signal est alors analysé par fenêtres temporelles selon le même principe qu'une



(a) Représentation d'un spectrogramme de Mel avec une basse résolution.



(b) Représentation d'un spectrogramme de Mel avec une grande résolution.

FIGURE 15 – Deux spectrogrammes de Mel avec des paramètres différents : a) 18 bandes de Mel, 160 samples de longueur de fenêtres ; b) 120 bandes de Mel, 25 samples de Longueur de fenêtre.

transformée de Fourier glissante. Néanmoins, il faut comprendre que les *coefficients de Mel* sont une mesure de la puissance du signal (somme des carrés des échantillons dans la fenêtre) par bandes de fréquences, ces bandes de fréquences sont calculées en une utilisant une banque de filtres (*filter bank* ou *f-bank*), chaque filtre étant le filtre passe-bande défini par deux fréquences successives du découpage fréquentiel en Mels. La figure 15 montre deux exemples de spectrogrammes de Mel, les niveaux forts sont en jaune et les niveaux faibles, en bleu. Notons que pour certaines applications relativement triviales, le cas 15a peut suffire, mais pour des tâches plus complexes, il est préférable d'éviter une trop forte compression des données en privilégiant des paramètres similaires à ceux de la figure 15b.

L'extraction des caractéristiques repose sur une connaissance *a priori* des signaux, mais l'apprentissage non supervisé peut-être utilisé ici pour laisser à l'algorithme le choix d'extraire les caractéristiques qui lui semblent les plus appropriées et donc d'optimiser cette extraction de caractéristiques acoustiques. Nous ne détaillerons pas ces méthodes, car elles n'interviennent pas dans la suite de notre travail, mais l'article de Xu *et al.* (2017) est une bonne lecture complémentaire à ce sujet.

BEATs utilise une extraction des caractéristiques basée sur un spectrogramme de Mel. Une fois la phase de normalisation des données d'entrée terminée, la description temps-fréquence est soumise au transformeur encodeur afin d'extraire des caractéristiques de niveau sémantique. Les *patchs* (les morceaux du mel-spectrogramme) sont associés à des vecteurs du *codebook*, puis entrés dans l'encodeur qui les transforme en vecteurs de représentations dans son espace latent. Il est important de considérer que ces vecteurs pourraient contenir de l'information très *haut niveau* (*i.e.* d'ordre sémantique ou causale) selon l'architecture théorique de BEATs, et nous testerons si c'est bien le cas après la phase d'ajustement (3.1.2).

1.3.5 Projection dans l'espace de sortie.

Les vecteurs de l'espace latent sont ensuite entrés dans la deuxième partie du transformeur, le décodeur, qui prédit les labels.

La dernière couche du transformeur possède 527 neurones pour les 527 classes de la catégorisation Audioset. La dernière étape consiste en l'application d'une fonction SoftMax (cf. Annexes A) sur tous les éléments de sortie, la classe prédite par l'algorithme est celle correspondant à la plus grande probabilité. De part l'organisation de son ontologie, Audioset nécessite souvent de regarder le top-k des classes (c'est-à-dire les k classes les plus probables pour un son en sortie d'algorithme) (Kong *et al.*, 2020). Ainsi, sur les 527 valeurs associés à un fichier audio en sortie de BEATs, il est pertinent de regarder les 5 classes les plus présentes afin d'en déduire la source véritable du son.

1.4 Conclusion partielle.

Après avoir défini les enjeux du monteur son et détaillé un de ces outils pour organiser les sons, nous avons vu comment les algorithmes étaient capables de classifier des données audios à travers l'étude du modèle BEATs. Nous avons ensuite intuité que l'ontologie Audioset ne parvenait pas à résoudre un certain nombre de problèmes inhérents au domaine du montage son et proposerons donc, dans le chapitre suivant, une méthodologie visant à transférer les connaissances de BEATs vers l'ontologie UCS, cette dernière étant plus à même de répondre aux besoins du monteur son.

Chapitre 2

Utilisation et ajustement du modèle BEATS pour l'indexation des sons en sonothèque.

L'objectif de cette partie est de savoir s'il est possible d'élaborer un algorithme basé sur une architecture de réseau de neurones capable, à partir des caractéristiques acoustiques d'un son, de le classifier dans une ontologie basée sur la cause réelle de ce son. La question de la classification automatique de sons pour l'audiovisuel a déjà été traitée par Hadi Harb (2001), mais, comme c'était il y a plus de vingt ans, nous avons de bonnes raisons de penser que les nouveaux algorithmes de classification pourraient désormais répondre à nos attentes. Les ontologies ont elles aussi évolué, Audioset (Gemmeke *et al.*, 2017) propose un découpage complet de la quasi-totalité des sources sonores existantes.

Évidemment, les enjeux relatifs aux sonorités diffèrent entre le monteur son et l'ingénieur en traitement du signal, ils portent, à chaque objet sonore, une perception, un sens, une finalité différente. De fait, ils ne catégorisent pas les sons de la même manière et utilisent des ontologies différentes. Nous avons précédemment montré comment l'ontologie UCS répondait à la majorité des problématiques liées à la pratique professionnelle du monteur son. Notre travail permettra donc d'évaluer comment les outils de classification automatique actuels, avec leur système d'analyse propre, pourraient permettre de résoudre un certain nombre de problématiques propres aux métiers du son et de l'audiovisuel.

Dans un premier temps, nous aurons besoin de construire un jeu de données annotées en UCS. Afin de garantir un jeu de données concordant avec les données manipulées par le monteur son, nous collecterons des données audio issues de sonothèques commerciales. Ces données ont un coût, et l'entraînement d'un modèle

général (traitant l'ensemble des sons environnementaux), nécessite une grande puissance de calcul. Par conséquent, nous nous contenterons dans ce mémoire d'un jeu de données contenant 6 sous-catégories de la catégorie UCS *FOOD&DRINK*.

Dans un second temps, nous utiliserons un modèle qui satisfait l'état de l'art actuel sur la classification de sons environnementaux : l'algorithme BEATs (S. Chen *et al.*, 2023). BEATs sera utilisé pour comparer l'ontologie source Audioset et l'ontologie cible UCS afin d'établir les rapprochements éventuels entre les classes des deux ontologies.

Après avoir montré la profonde différence d'architecture entre les deux ontologies mentionnées, nous essayerons de remplacer la dernière couche de BEATs qui donne une prédiction sur les classes Audioset, par une couche effectuant une prédiction dans l'ontologie UCS. Notre partie pratique consistera en l'entraînement d'une couche contenant 6 neurones ayant pour sortie les prédictions de présence des 6 sous-catégories.

2.1 Construire un jeu de données pertinent.

D'après Virtanen *et al.* (2018) c'est l'application concrète de l'algorithme qui doit conditionner les données d'apprentissage. Dans notre cas, nous devons collecter des données sonores ayant vocation à être intégrées dans une production audiovisuelle. Aussi, nous avons constitué une base de données à partir de sons issus de sonothèques commerciales, car ces exemples sonores sont ceux qui se rapprochent le plus du type de fichiers que nous cherchons à classifier.

> *“ Un jeu de données pour la recherche est naturellement très délicat. Son contenu doit être choisi méticuleusement pour permettre une couverture suffisante des aspects intéressants, une variabilité suffisante pour la caractérisation de ces aspects, et une taille suffisante pour une modélisation robuste. ”*[1]
>
> - Annamaria Mesaros *et al.* (2018) -

Ici, trois termes sont à retenir, la *couverture*[2], la *variabilité*[3] et la *taille*[4]. Un jeu de données[5] a une bonne *couverture* s'il contient autant de catégories que nécessaire à la tâche de classification. La *variabilité* qualifie les dissemblances des différents éléments appartenant à une même classe. Un jeu de données doit faire mention, pour

1. “ A dataset for research is, naturally, very delicate. The content must be carefully selected to provide sufficient coverage of the aspects of interest, sufficient variability in characterizing these aspects, and a sufficient quantity of examples for robust modeling. ”
2. cf. Glossaire D.1
3. Idem.
4. Idem.
5. cf. Glossaire D.1

chaque classe, d'échantillons possédant différentes conditions : d'émission, d'enregistrement, d'acoustique, *etc*, afin de représenter idéalement l'ensemble des caractéristiques propres à chaque classe. Enfin, la *taille* du jeu de données est sûrement le critère le plus important. Les exemples doivent être répartis de sorte qu'il y ait un nombre suffisant d'exemples pour chaque catégorie. Plus il y a d'exemples dans une catégorie, moins la prédiction sera biaisée.

Nous décidons d'abord de nous consacrer à une petite proportion de l'ontologie UCS, nous étudierons donc la catégorie *FOOD&DRINK* et ses 6 sous-catégories : *Cooking, Glassware, Ingredients, Kitchenware, Misc* et *Tableware* dont les définitions sont données dans le tableau 2.1.

Sous catégorie	**Définition**
Cooking	Choses ouvertement liées à la cuisine, à la pâtisserie, à la préparation, au découpage.
Glassware	Mouvement de verres et de bouteilles, notamment dans un restaurant ou un bar. Enregistrements isolés de verrerie.
Ingredients	Noix, graines, céréales, etc. Les enregistrements d'aliments crus sont souvent utilisés à d'autres fins de conception, comme les enregistrements de légumes.
Kitchenware	Mouvement de casseroles et de poêles, grands bols en métal, bols de préparation.
Misc	Sons de Nourriture et Boisson qui ne vont dans aucune des autres catégories de cette liste.
Tableware	Argenterie et/ou manipulation d'assietes et bols, dans un restaurant ou un dîner à la maison.

TABLEAU 2.1 – Définitions des sous catégories de *FOOD&DRINK* dans la catégorisation UCS.

Le choix de porter notre attention sur la catégorie *FOOD&DRINK* s'est fait en utilisant deux critères principaux. Le premier est l'existence de sonothèques dédiées aux sons de cuisine et de préparation culinaire. Les vidéos courtes dédiées à la préparation culinaire étant courantes sur les réseaux sociaux, cela a motivé, de la part des industriels, la création de sonothèques regroupant une grande variété de ces sons pour faciliter le montage amateur à destination des réseaux sociaux (entre autres). Il existe donc suffisamment de données pour les *collecter* et les rassembler dans un jeu de données, sans avoir à *enregister* les sons.

Le second critère motivant notre choix est la taille de la catégorie *FOOD&DRINK*. Cette classe est d'une taille moyenne pour la catégorie UCS, elle ne possède que six sous-catégories, quand d'autres en possèdent une vingtaine, sa taille est donc ni trop grande, *i.e.* trop complexe pour entraîner un modèle avec peu de données, ni

trop petite, car permettant toutefois une classification sur un nombre significatif de classes. Par ailleurs, une particularité est que les sous-catégories de *FOOD&DRINK* sont séparées selon les deux méthodes les plus courantes au sein de l'ontologie UCS, le *timbre* et le *sens*. D'une part, on retrouve la discrimination par *timbre/matériau*, les catégories *Kitchenware, Glassware* et *Tableware* comportent les mêmes types d'actions, en l'occurrence des saisies et des impacts, mais sur des matériaux différents : du verre, de la céramique de taille variée, de l'inox, *etc.* D'autre part, on y retrouve la présence de familles à *forte diversité intra-classe*, c'est-à-dire regroupant des actions *sémantiquement proches* mais ayant des caractéristiques acoustiques très variées. C'est le cas pour la sous-catégorie *Cooking* qui regroupe des actions aussi hétérogènes que la découpe de légumes, le mixage du potage ou le battage de la crème.

Pour ces raisons, *FOOD&DRINK* nous a paru constituer un candidat idéal pour une étude préliminaire, et, en constituant un jeu de données à partir de ces sons, nous pensions pouvoir conclure quant à la capacité de BEATs à identifier non seulement des variations de timbres, mais aussi des variations relatives au sens et aux actions performées[6] dans le son, ces deux types de variations étant nécessaires si l'on veut prétendre à un outil de recherche pertinent pour le monteur son. Pour toutes les raisons évoquées ci-dessus, nous pensons pouvoir étendre les résultats présentés dans la partie 3.1 à une plus grande partie de l'ontologie UCS, sous réserve d'avoir assez de données, et une puissance de calcul suffisante pour l'entraînement.

Pour des raisons de praticité, nous optons pour un jeu de données collecté. Bien sûr, il aurait été possible d'enregistrer toutes sortes de sons relatifs à la cuisine, seulement, il faut ensuite uniformiser les données en terme de niveau, de fréquence d'échantillonnage, nommer les fichiers, renseigner leurs catégories et sous-catégories et autres.

En ce sens, il nous a paru plus réaliste d'acheter banques de sons commerciales. Le jeu de données que nous avons constitué se base sur les cinq sonothèques suivantes : *BoaSounds Household*[7], *Articulated Essential Kitchen*[8] *SSL39 HouseHold Sounds*[9], *Cooking and Food Prep - Celine Woodburn*[10] et *Kanpai - Kai Paquin*[11].

Utiliser plusieurs sonothèques faites par différentes personnes permet d'avoir, au sein de notre jeu de données, différentes habitudes d'enregistrement et de performances, ce qui maximisera la capacité de généralisation de notre algorithme.

Par ailleurs, même si l'algorithme BEATs effectue un prétraitement sur les don-

6. La performance désigne les actions réalisées pour produire la matière sonore qui est enregistrée.
7. https://justsoundeffects.com/products/household/
8. https://articulatedsounds.com/audio-royalty-free-library/sfx/essential-kitchen
9. https://www.surroundsoundlab.net/product/ssl39-everyday-household/
10. https://www.asoundeffect.com/sound-library/cooking-and-food-prep/
11. https://www.asoundeffect.com/sound-library/kanpai/

nées qu'il reçoit afin d'uniformiser le niveau, il est important d'avoir des éléments similaires en termes de niveau sonore et de fréquence d'échantillonnage. Pour le niveau sonore, une règle tacite des sonothèques commerciales consiste à normaliser les sons avant de les mettre en vente, de ce fait, les fichiers ont généralement un niveau assez fort, proche du 0 dB FS. Pour ce qui est de la fréquence d'échantillonnage, comme BEATs charge les fichiers échantillonés à 16 kHz, nous avons sous-échantillonné tous les fichiers à 16 kHz. Enfin, comme BEATs traite des signaux mononophoniques, mais que la plupart des fichiers sont des fichiers stéréophoniques, le jeu de données a été construit en conservant uniquement le premier canal de chaque fichier.

Dans son étude sur le temps d'identification de 41 signaux du quotidien, (Ballas, 1993) montre que les sujets humains mettent entre 1,253 et 6,823 secondes pour identifier la source d'un son, avec une moyenne autour de 4 secondes. De même, la plupart des jeux de données sont composés de sons durant entre 2 et 5 secondes (Piczak, 2015)(Salamon *et al.*, 2014). Aussi, nous avons fait le choix de diviser nos fichiers de sonothèques en morceaux d'une durée de 4 secondes, le jeu de données est au final constitué de 8120 audios de 4 secondes formant une durée totale d'environ 540 minutes (9 heures). Le tableau 2.2 montre la répartition des exemples de notre jeu de données au sein de chacune des sous-catégories de *FOOD&DRINK*. Notons

Sous-catégorie	**Nombre d'échantillons**	**Durée (en minutes)**
Cooking	1462	97
Glassware	4299	287
Ingredients	307	20
Kitchenware	153	10
Misc	1037	69
Tableware	862	57

Tableau 2.2 – Nombre d'échantillons sonores et minutage pour chaque sous catégories de *FOOD&DRINK* dans le jeu de données.

que certaines classes sont moins représentées que d'autres ; c'est le cas de la classe *Kitchenware* (la moins représentée) qui compte environ 30 fois moins d'exemples que la classe *Glassware* (la plus représentée). Faute d'avoir pu collecter plus de représentants de la catégorie *Kitchenware* nous devrons être vigilants quant à la distribution des données dans nos jeux d'apprentissage et d'évaluation.

A Jeu d'évaluation et jeu d'entraînement.

L'ensemble des données que nous avons collectées est appelé *jeu de données global*, au cours de notre méthode nous aurons besoin de scinder ce jeu de base en 2 sous parties que nous explicitons ci dessous.

- Le *jeu d'évaluation* a une taille correspondant à 20% de l'ensemble de notre

jeu de données et constitue les données sur lequel l'algorithme sera contrôlé au terme de son entraînement. Ce jeu de données servira notamment la partie 3 (Résultats et Discussions) de ce mémoire.

- Le *jeu d'entraînement*, lui, correspond aux *exemples* sur lesquels l'algorithme va apprendre, il est constitué d'environ 80% de la durée du jeu de donnée global.

Il est nécessaire de prendre quelques précautions lors de la construction du jeu d'évaluation. Premièrement aucun fichier ne doit être en commun entre le jeu d'entraînement et le jeu d'évaluation, en termes mathématiques, l'intersection des deux espaces doit être nulle $J_{entr} \cap J_{eval} = 0$. Deuxièmement, nous avons identifié que certaines classes sont beaucoup moins représentées que d'autres. Aussi pour s'assurer que toutes les classes soient idéalement représentées dans le jeu d'évaluation, on effectue une sélection manuelle des fichiers de sorte que chaque sous-catégorie venant des différentes sonothèque commerciale possède la même distribution dans le jeu de validation, le jeu d'entraînement et le jeu de données global. (cf. tableau 2.2).

Sous catégorie	**Taille dans le jeu d'entraînement (Méga-octets)**	**Taille dans le jeu d'évaluation (Méga-octets)**
Cooking	237,8	63
Glassware	7540	1185
Ingredients	403,3	99,3
Kitchenware	193,4	43,5
Misc	2470	526
Tableware	2240	424,4

TABLEAU 2.3 – Taille en Méga-octets de chaque sous catégories de *FOOD&DRINK* dans le jeu d'évaluation et le jeu d'entraînement.

Dans le tableau ci-dessus, on a tenté de respecter une répartition correspondant à 80% - 20% pour chaque sous-catégorie et chaque sonothèque commerciale. La constitution de ce jeu d'évaluation a nécessité de contrôler chaque son à la main, cette tâche fastidieuse pouvant difficilement être automatisée. En effet pour chaque catégorie, il faut s'assurer que chaque sonothèque soit représentée à hauteur d'environ 20%, en terme de durée mais aussi en terme du nombre de fichiers. Par ailleurs, beaucoup de sonothèques contiennent plusieurs prises d'un même objet, par exemple :

- *"FOODCook_ Food Prep Buttering Rivita_ Celine Woodburn_ Cooking and Food Prep_ 01.wav"*
- *"FOODCook_ Food Prep Buttering Rivita_ Celine Woodburn_ Cooking and Food Prep_ 02.wav"*

- *"FOODCook_ Food Prep Buttering Rivita_ Celine Woodburn_ Cooking and Food Prep_ 03.wav"*

Face à ce genre de situation, on essaye si possible, de répartir les fichiers à hauteur de 20% dans le jeu d'évaluation.

Pour toutes ces raisons, il est parfois nécessaire de faire des compromis, expliquant pourquoi pour certaines catégories dans le jeu d'évaluation, on ne trouve pas exactement 20% de la proportion totale.

2.2 Choix du modèle.

Afin de choisir au mieux le modèle de classifieur que nous allons utiliser, une bonne première idée consiste à comparer les performances des modèles faisant état de la recherche actuelle. Nous nous sommes intéressés à quatre modèles utilisant des architectures différentes : PaNNs (Kong *et al.*, 2020), PaSST (Koutini *et al.*, 2022), AST (Gong *et al.*, 2021), et BEATs (S. Chen *et al.*, 2023).

La communauté de recherche en classification des sons environnementaux possède plusieurs jeux de données à sa disposition, dont Audioset (Gemmeke *et al.*, 2017) et ESC-50 (Piczak, 2015) notamment. La grande majorité des modèles sont testés au moins sur ces deux jeux de données, le tableau 2.4 recense les différences de performances entre les quatre modèles évoqués plus haut. On remarque que le modèle

Modèle	**Audioset** (mAP)	**ESC-50** (Exactitude)
PaNNs	0,439	94,7%
PaSST	0,471	96,8%
AST	0,485	95,6%
BEATs	0,506	98,1%

Tableau 2.4 – Performances des algorithmes états de l'art sur les deux jeux de données Audioset et ESC-50.

BEATs obtient le meilleur score sur les deux jeux de données étudiés, c'est donc sur lui que notre choix s'est porté pour mener la suite de notre expérience. Nous supposons que c'est l'algorithme qui sera le mieux capable d'effectuer notre test de classification.

2.3 Tentative d'association entre les classes Audioset et UCS.

L'entièreté des algorithmes de reconnaissance de sources formant l'état de l'art actuel, AST (Gong *et al.*, 2021), PaNNs (Kong *et al.*, 2020), PaSST (Koutini *et al.*, 2022) et BEATs (S. Chen *et al.*, 2023) a été entraînée sur l'ontologie Audioset.

Dans la première partie de ce mémoire nous avons considéré l'UCS comme adaptée aux besoins du monteur son. Pour ces raisons, c'est entre ces deux ontologies que nous essaierons de faire des liens par la suite.

Nous proposons, dans cette section, un protocole expérimental permettant de comparer deux ontologies. Il s'agira d'établir une correspondance entre *l'ontologie source* Audioset, ayant servi a entraîner BEATs, et *l'ontologie cible* UCS adaptée aux réalités de l'industrie audiovisuelle.

2.3.1 Description de l'expérience.

Le réentraînement d'une couche de classification est un procédé coûteux en terme de données et de calculs, avant de parler d'apprentissage par transfert et d'ajustement, il est intéressant de questionner les liens qui existent entre les deux ontologies Audioset et UCS. Audioset est l'ontologie source, c'est la hiérarchisation sur laquelle BEATs a été entraîné. UCS est l'ontologie cible, celle que nous visons en vue d'une application au milieu de l'audiovisuel. L'ontologie Audioset étant très détaillée, notre hypothèse était qu'il pouvait y avoir des corrélations évidentes avec des classes de l'UCS et ceci même si les deux ontologies semblent tout de même assez différentes. En effet, on trouve des cas *one to many* (un vers plusieurs), la catégorie *Doors* d'Audioset possède plusieurs images dans l'ontologie UCS, ainsi que des cas *many to one* (plusieurs vers un), par exemple, les catégories *Frying* et *Chopping* ont une image unique (*FOOD&DRINK/Cooking*) en UCS.

Notre première phase de test consistait à utiliser l'algorithme BEATs tel qu'il est disponible sur le dépôt GitHub [12] et sans ajustement sur aucune données. L'algorithme a été testé sur une multitude de sons de la classe *FOOD&DRINK*, et, pour chacune des sous-catégories, nous avons noté leur répartition dans la hiérarchie Audioset. Les fichiers testés étaient des fichiers bruts extraits de sonothèques commerciales, ils étaient notamment de longueur diverses.

Les résultats présentés dans la partie 3.1.1 montrent les corrélations entre les classes des deux ontologies. Une matrice de corrélation a été construites et des graphiques en barre permettront de montrer les descripteurs Audioset les plus proches pour les 6 sous-catégories UCS : *Cooking, Glassware, Ingredients, Kitchenware, Misc, Tableware.*

Matrice de corrélation

Une *matrice de corrélation* ou *matrice de vraisemblance* permet d'établir des correspondances entre une ontologie source et une ontologie cible. C'est un outil de

12. https://github.com/microsoft/unilm/tree/master/beats

mesure couramment utilisé dans le champ de l'apprentissage automatique.

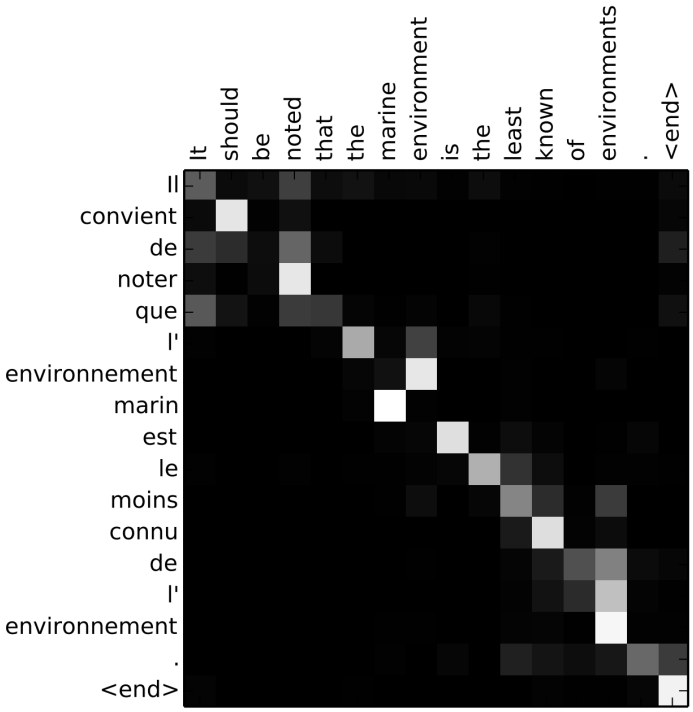


FIGURE 16 – Matrice de vraisemblance entre deux versions de la même phrase dans deux langues différentes (BAHDANAU *et al.*, 2016).

La figure 16 montre la corrélation entre la sémantique des mots d'une même phrase traduite dans deux langues différentes.

L'analyse des résultats de cette première expérience 3.1.1 montre une incompatibilité profonde entre l'ontologie source et l'ontologie cible, les corrélations entre les différentes classes sont trop faibles pour espérer en tirer une application fiable. En ce sens, nous pensons qu'il serait pertinent d'entraîner un modèle sur des données de l'ontologie cible, afin qu'il apprenne à classifier efficacement ce type d'éléments, c'est ce que nous détaillerons dans la partie suivante.

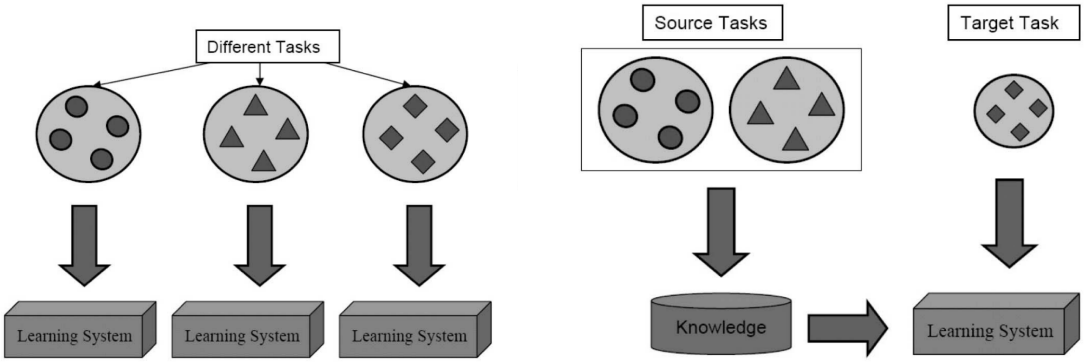
2.4 ENTRAÎNEMENT DU MODÈLE.

La méthode proposée dans cette section consiste à réentraîner la dernière couche du réseau de neurones, afin de proposer un classifieur sur 6 classes. D'abord, nous aurions pu considérer un modèle en faisant *tabula rasa*[13] et l'entraîner sur une grande base de données annotées en UCS. L'avantage de cette méthode est que l'on contrôle pleinement l'outil, de sa conception profonde aux données avec lesquelles il est entraîné. Les inconvénients relèvent d'un problème matériel : les données annotées en

13. Concept philosophique selon lequel l'esprit naît vierge et serait formé par l'expérience seule.

UCS coûtent cher, et ce genre d'entraînement nécessite des unités de calcul puissantes pour effectuer la phase d'apprentissage. BEATs a été entraîné sur plusieurs dizaines de cartes graphiques (*graphics processing unit - GPU)*, ressources matérielles auxquelles nous n'avons pas accès. En ce sens, refaire un apprentissage du modèle en partant de zéro n'était pas une solution envisageable.

Toutefois, il est commun en apprentissage automatique, de procéder à un apprentissage par transfert [14] (Pan & Yang, 2010), l'avantage d'une telle méthode est que la taille du jeu de données d'ajustement peut être revu à la baisse, comme illustré dans le cas 17b.



(a) Principe d'un apprentissage multi-tâche (*multi-task learning*)

(b) Principe d'un apprentissage par transfert (*transfer learning*)

Figure 17 – Comparaison entre l'apprentissage multi-tâche (a) et l'apprentissage par transfert (b) (Pan & Yang, 2010).

Une approche possible consiste à supprimer la dernière couche de BEATS (qui fait une classification sur 527 classes) pour la remplacer par les 752 catégories UCS (ou une portion de celles-ci), puis, à ajuster (*fine-tune*) le modèle avec des données annotées en UCS. En réalité, nous avons entraîné le modèle sur 6 des 752 classes correspondant aux catégories de notre jeu de données 2.1.

2.4.1 Architecture du réseau de neurone.

BEATS a été ajusté sur le jeu de données Audioset, ce qui concrètement se manifeste par une couche de projection dans l'espace des labels Audioset en sortie du transformeur. Comme mentionné dans la section C, il y a 527 classes de sortie possibles, la dernière couche du prédicteur possède donc 527 neurones correspondant aux classes de sorties d'Audioset.

En pratique, nous devons supprimer cette dernière couche, et la remplacer par un réseau de neurones qui fera la projection dans un nouvel espace de sortie. Le réseau que nous souhaitons entraîner est d'une architecture assez simple. Ce réseau a pour

14. cf. Glossaire D.1

but de passer de données de l'espace latent de BEATs vers une représentation de la dimension de l'espace de sortie. La dimension de l'espace de sortie dépend du nombre de classes dans l'ontologie utilisée, nous nous restreindrons à une classification en 6 catégories, comme expliqué dans le paragraphe 2.1. Notre couche de projection, que nous appellerons *tête de classification* est donc un réseau de neurones linéaire dense avec 768 neurones d'entrées (pour la dimension de l'espace latent) et 6 neurones de sorties (pour la dimension de l'espace de sortie). Mathématiquement, la tête effectue une *projection linéaire* des vecteurs d'un espace de dimension 768 vers un espace de dimension 6 (voir figure 18).

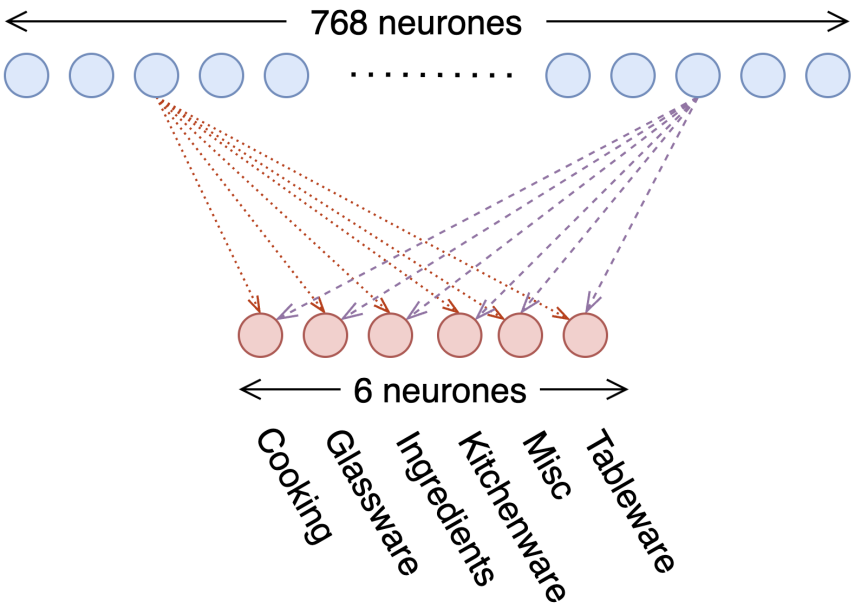


FIGURE 18 – Schéma de la tête de classification.

Il faut maintenant procéder à l'apprentissage de la tête afin d'ajuster les poids et biais de ce réseau de neurones. Se pose alors la question de l'optimisation des variables d'apprentissage, nous expliquons nos choix à ce sujet dans la partie qui suit.

2.4.2 Choix des hyperparamètres d'apprentissage.

Lorsqu'un algorithme de classification est construit, il possède un certain nombre de *paramètres* propres à son architecture, qui peuvent être optimisés pour maximiser les performances. Par exemple, le nombres de couches de neurones, le nombre de neurones par couches (cf. Annexes B.1), mais aussi le nombre de têtes d'attention ou le nombre de filtres de Mel pour extraire les caractéristiques du signal (cf. paragraphe 1.3.2 B), sont autant de variables qui peuvent être optimisées. Lors de l'entraînement, des architectures différentes seront testées et on retiendra la combinaison de paramètres qui maximisent la précision du classifieur. Cependant, dans notre cas, nous nous basons sur une architecture déjà existante et optimisée, aussi nous avons finalement peu de paramètres à tester. Ni le nombre de couches, ni le nombre de neurones ne peuvent être modifiés dans la phase d'ajustement à laquelle nous procéderons. Si les paramètres propres à l'architecture de notre modèle ne sont pas modifiables, il est en revanche possible d'influer les paramètres régissant le processus d'apprentissage lui-même. On parle alors d'*hyperparamètres*.

Un de ces hyperparamètres est le *pas d'apprentissage*[15], et il est coutume d'essayer plusieurs pas d'apprentissage lors de l'entraînement. Pour la phase d'apprentissage, nous avons testé les trois valeurs suivantes pour le pas d'apprentissage : $[10^{-2}, 10^{-3}, 10^{-4}]$, celles-ci se révélant être les valeurs les plus couramment utilisées.

De même, nous avons essayé différentes tailles de lots (*batch size*) pour observer dans quel cas l'algorithme apprend le mieux. Nous avons testé ces tailles pour des lots de 16 et 32 exemples.

Nous renvoyons à la lecture de la partie B pour la définition de ces paramètres.

Pour chaque combinaison d'hyperparamètres, il faut procéder à un nouvel entraînement. Cela représente $3 \times 2 = 6$ modèles différents à tester pour le même jeu de validation.

2.5 Méthodologie des tests.

2.5.1 Techniques de validation.

La *validation* est un moyen de contrôler l'apprentissage d'un modèle. Il existe plusieurs méthodes, parmi elles, les méthodes de validations croisées sont moins biaisées.

D'abord, définissons ce qu'est une *validation non croisée*. Cette méthode consiste simplement à séparer les données en 2 populations, un *jeu d'apprentissage training-set*, et un *jeu de validation test-set*. Le jeu d'entraînement représente au moins 60% des données totales. Dans ce cas les données ne se mélangent pas, le modèle est évalué

15. cf. Glossaire D.1

Numéro de pli	**Blocs formant le jeu d'entraînement**	**Bloc formant le jeu de validation**
1	2, 3, 4, 5, 6	1
2	3, 4, 5, 6, 1	2
3	4, 5, 6, 1, 2	3
4	5, 6, 1, 2, 3	4
5	6, 1, 2, 3, 4	5
6	1, 2, 3, 4, 5	6

TABLEAU 2.5 – Tableau expliquant une validation croisée à 6 blocs.

une seule fois sur son jeu de validation (qui fait aussi office de jeu d'évaluation dans ce cas).

Une autre forme de validation, particulièrement utile lorsqu'on possède peu de données, est la *validation croisée*. L'ensemble des données est divisé en k blocs. Un des k blocs est considéré comme le bloc d'évaluation pendant que les $k-1$ autres constituent le jeu d'entraînement. Une telle répartition est appelé un pli (*fold*) (cf. tableau 2.5). Ensuite, un score de validation correspondant à la performance de l'algorithme sur le bloc de validation courant est calculé. L'opération est répétée en utilisant un autre bloc de validation et ainsi de suite jusqu'à ce que l'ensemble des k blocs aient servi de jeu validation. La performance du modèle est alors définie en prenant la moyenne de la performance sur l'ensemble des blocs.

La *validation croisée stratifiée* (*stratified k-fold*) est une forme de validation à k-blocs dans laquelle il y a la même répartition des classes dans tous les ensembles d'apprentissage. Cela est particulièrement utile lorsqu'il y a des données moins représentées que d'autres dans un jeu de données. Le jeu de données *Urbansound8k* (SALAMON *et al.*, 2014), par exemple, est présenté initialement en 10 blocs, avec une distribution des classes identique dans chaque bloc, même si certaines classes possèdent moins de membres que d'autres.

Dans le cas d'un validation croisée, le jeu de validation et le jeu d'évaluation ne sont pas confondus. La validation sert à prévenir la surinterprétation du modèle pour savoir quand l'apprentissage doit s'arrêter. Mais, pour évaluer les performances du modèle sur des données qu'il n'a jamais vues, et ainsi vérifier sa capacité de généralisation, le modèle doit être évalué sur un jeu d'évaluation indépendant qui a lieu après l'apprentissage.

Dans le développement du script d'apprentissage, le choix a été fait d'utiliser une méthode de validation croisée stratifiée ; la principale motivation étant la faible taille de notre jeu de données et la quantité variable d'exemples pour les différentes classes (cf. Tableau 2.2). La première chose à faire est de séparer le jeu de données en deux sous-ensembles : un jeu d'entraînement/validation avec application de la

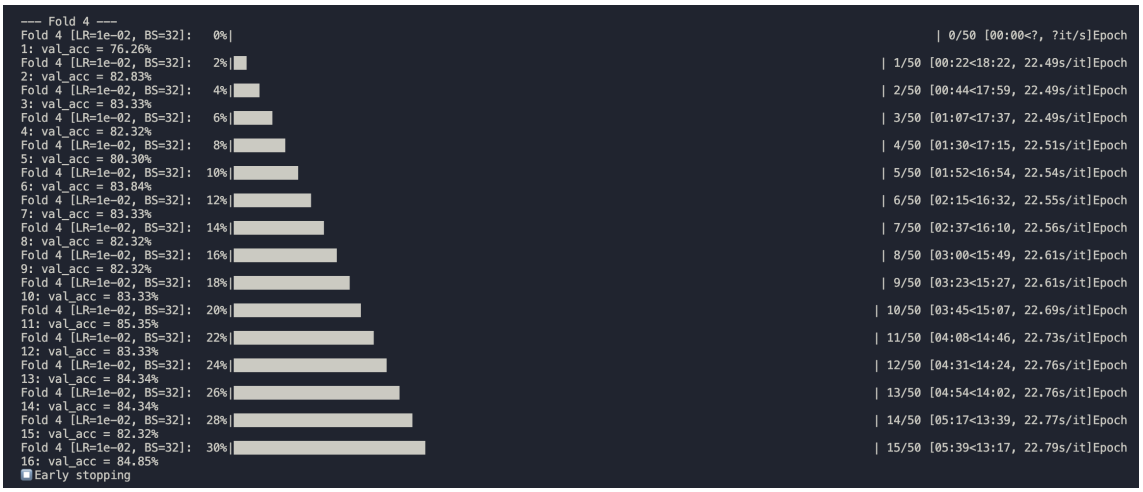


FIGURE 19 – Affichage dans la console pendant la phase d'entraînement, l'entraînement s'est arrêté à la 15ème époque (cf. B), car la tête de classification ne progressait plus, c'est l'arrêt anticipé (*early-stopping*).

validation croisée, et un jeu d'évaluation pour tester l'inférence du modèle entraîné. La répartition des données se fait selon les proportions suivantes : 80% pour le jeu d'entraînement/validation, et 20% pour les données d'évaluation. À l'instar de ce qui est fait pour *UrbanSound8k* (SALAMON *et al.*, 2014), il y a autant de plis que de neurones dans la couche de sortie. Nous avons donc effectué une validation croisée à 6-blocs et avons évalué le modèle sur un sous-ensemble dont la taille correspond 20% des données totales.

À chaque pli, une tête de classification vierge est chargée. L'initialisation se fait avec des poids et biais aléatoires. L'apprentissage se fait au maximum sur 50 époques, une époque correspond au passage de l'entièreté du jeu d'entraînement dans l'algorithme. Chaque époque se termine par un contrôle de performance sur le jeu de validation (ce dernier dépendant du pli dans lequel on est). Chaque entraînement aboutit donc au stockage des poids de 6 modèles.

Afin de ne pas perdre de temps inutile lors de l'apprentissage, nous procédons à un arrêt anticipé (*early stopping*) dès que la précision du modèle sur le jeu de validation stagne trop longtemps. L'apprentissage se fait avec une patience de 5 époques, c'est-à-dire que si la précision n'augmente pas pendant 5 époques successives, on considère que l'algorithme a atteint ses limites et qu'aller au delà ne mènerait qu'à une surinterprétation nuisible à la généralisation (cf. figure 19). On sauvegarde alors les paramètres de la couche de classification.

2.5.2 Évaluation du modèle.

A Mesures d'évaluations.

Pour le chapitre qui suit, consacré aux mesures d'évaluations utilisées pour analyser nos résultats, nous avons besoin de définir quelques notions de base. Soit c une classe, on appelle *système* le modèle et *référence* la vérité terrain, c'est-à-dire le label associé au fichier son. On définit comme suit les statistiques intermédiaires :

- Vrai positif (TP) : une prédiction correcte, le système et la référence indiquent tous deux la présence de la classe c.
- Vrai négatif (TN) : le système et la référence indiquent tous deux l'absence de la classe c.
- Faux positif ou insertion (FP) : le système indique la présence de la classe c alors que la référence indique une classe c absente.
- Faux négatif ou suppression (FN) : le système prédit une absence de la classe c alors que la référence indique la présence de la classe c.

On désigne par TP, TN, FP, FN, le nombre respectivement de vrais positifs, de vrais négatifs, de faux positifs et de faux négatifs.

- La *précision* mesure la proportion de vrais positifs parmi l'ensemble des éléments classés comme positifs.

$$P = \frac{TP}{TP + FP}$$

- Le *rappel* (ou sensibilité) mesure la proportion de vrais positifs correctement détectés parmi tous les éléments réellement positifs.

$$R = \frac{TP}{TP + FN}$$

- L'*exactitude*[16] est une mesure de la proportion d'éléments bien classés qu'ils soient positifs ou négatifs parmi l'ensemble des observations.

$$ACC = \frac{TP + TN}{TP + TN + FP + FN}$$

L'exactitude a l'avantage de donner une mesure simple de la capacité d'un système à prendre la bonne décision, c'est la métrique la plus rencontrée dans la littérature pour des problèmes de classifications de scènes sonores. Le problème

16. cf. Glossaire D.1

Tête n°	*Misc*	*Ingredients*	*Cooking*	*Tableware*	*Glassware*	*Kitchenware*
1	25.28%	32.20%	39.63%	0.00%	2.89%	0.00%
2	10.58%	31.87%	57.05%	0.00%	0.49%	0.00%
3	1.76%	81.35%	12.21%	0.00%	4.68%	0.00%
4	0.44%	11.26%	88.22%	0.00%	0.07%	0.00%
5	1.11%	8.11%	90.31%	0.00%	0.47%	0.00%
6	9.88%	64.89%	25.19%	0.00%	0.04%	0.00%
Moyenne	8.18%	38.28%	52.10%	0.00%	1.44%	0.00%

TABLEAU 2.6 – Prédiction multi-classe pour chacune des têtes de classification et valeur de la moyenne sur les 6 têtes pour un son appartenant à la catégorie *Cooking*.

de l'exactitude c'est qu'elle est influencée par l'équilibre inter-classe. Pour les classes rares (TP+FN petit), un système peut avoir une grande proportion de vrai négatifs, même s'il ne fait pas de prédiction correctes, ce qui mène paradoxalement à une grande exactitude.

Il paraît donc essentiel de rester prudent s'agissant de la mesure de performance de notre algorithme en combinant plusieurs méthodes d'évaluation.

Nous venons de présenter des mesures d'évaluations pour une classification binaire, cependant le modèle effectue une classification multi-classe. Dans ce cas, il est important de regarder les performances globales (*i.e.* sur l'ensemble des classes) de l'algorithme, ainsi que ses performances pour chaque classe.

La phase d'évaluation a lieu après l'entraînement, après le stockage des 6 têtes correspondant chacune à l'entraînement sur un des plis. Les têtes sont combinées avant d'être évaluées afin de faire une prédiction plus objective. Concrètement, on prend la probabilité pour une classe et on la moyenne avec toutes les probabilités des autres têtes (cf. tableau 2.6).

Chapitre 3

Résultats et discussions.

3.1 Résultats.

3.1.1 Analyse des prédictions audioset de BEATs, sans réentraînement du modèle.

Dans cette partie, nous montrons et discutons des corrélations supposées entre les ontologies Audioset et UCS.

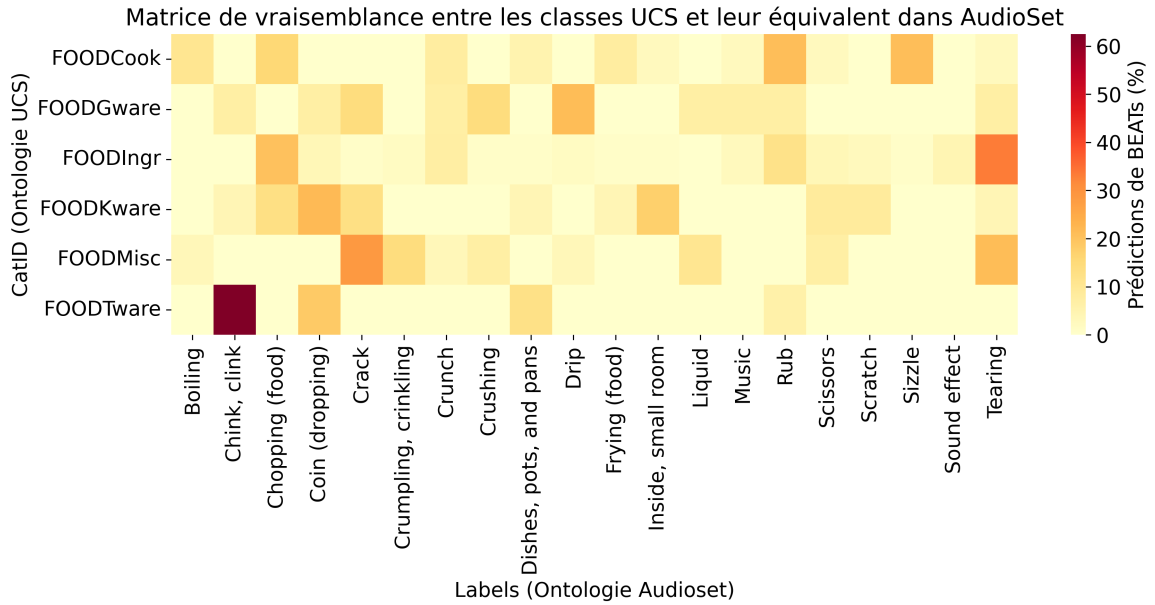


Figure 20 – Matrice de corrélation entre les sous-catégories de *FOOD&DRINK* et leur association par BEATs dans l'ontologie AudioSet.

L'analyse de la figure 20 fait principalement ressortir une valeur significative. En effet, on remarque une forte corrélation entre la catégorie UCS *tableware* et la classe d'onomatopées " *Chink, clink* " définie par " *Un son métallique, court, léger* "[1]. Ceci

1. A short light metallic sound.

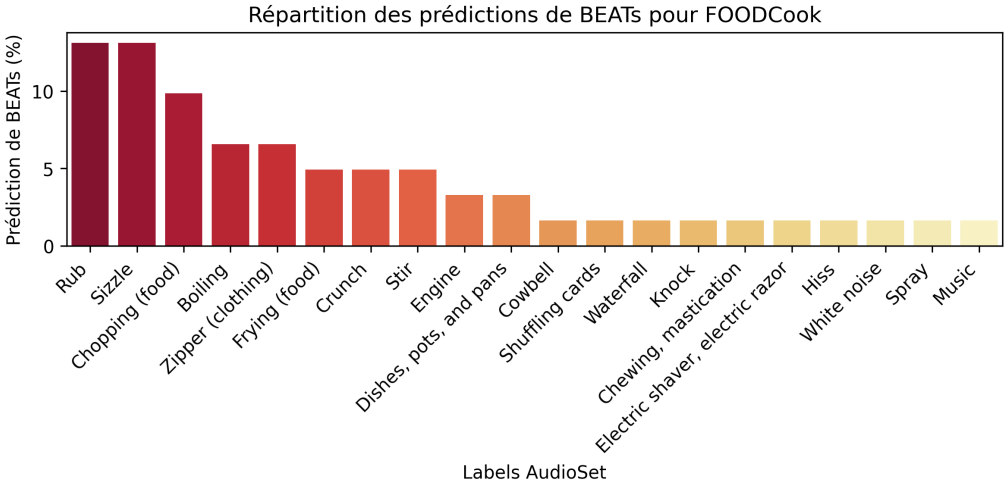
veut dire que lorsqu'on présente à BEATs des sons appartenant à la sous-catégorie *Tableware*, la prédiction sera, dans 60% des cas, la classe audioset *Chink, Clink*.

Mise à part cette valeur qui sort du lot, on a du mal à identifier des corrélations entre les classes UCS et celles d'Audioset. De ce fait, relier simplement les catégories qui semblent similaires entre elles est difficile.

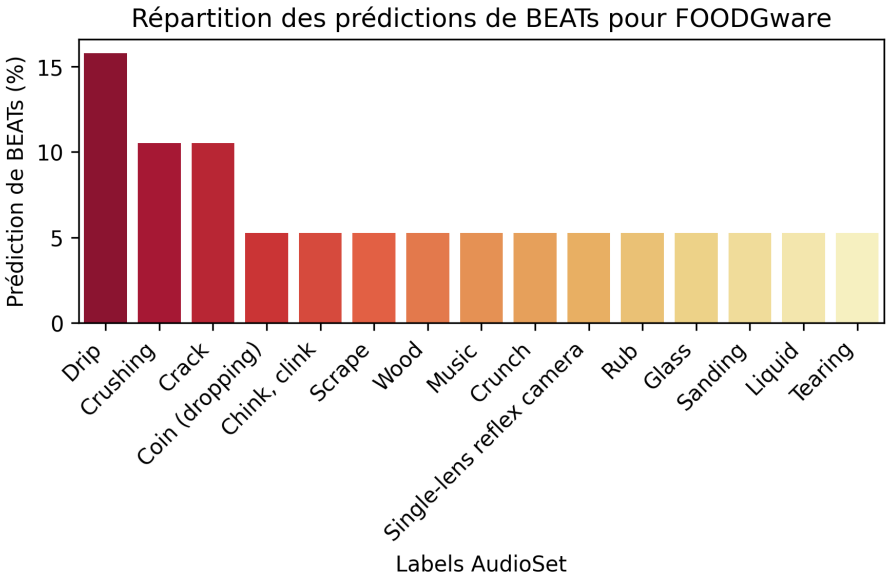
On représente en figures 21.1 et 21.2 pour chaque sous-catégorie UCS, la distribution des prédictions de BEATs dans l'ontologie Audioset. Après l'analyse des diagrammes, on remarque deux situations. La première correspond aux cas où l'algorithme n'est pas capable d'associer un *label pertinent* aux sons provenant d'une catégorie UCS particulière. C'est par exemple le cas de la catégorie *Kitchenware* (fig 21a) où la classe la plus corrélée avec le matériel de cuisine est la classe *Coin (dropping)* (pièces lâchées). De même, pour la catégorie *Ingredients* (fig 21c), on constate que la classe Audioset la plus proche selon BEATs est *Tearing* (pleurs). Dans ces cas, les labels associés par l'algorithme aux sous-catégories UCS n'ont rien à voir avec les sources réellement présentes dans le fichier. Ici, le problème concerne avant tout les performances de l'algorithme, car on vient de montrer grâce à ce simple test que BEATs avait du mal à traiter certaines données audio spécifiques. Malheureusement, nous ne pouvons pas y faire grand chose, BEATs est un des modèles les plus performants à l'heure actuelle et c'est une tâche trop lourde que de reconstruire un classifieur et de l'entraîner en partant de zéro.

La deuxième situation est celle où l'algorithme associe plusieurs *labels pertinents* à une sous-catégorie. Pour la figure 21a, les labels Audioset associés par BEATs sont *Rub*[2], *Sizzle*[3] et *Chopping (food)*[4]. On retrouve dans le top 10, des classes comme *Boiling* (Bouillir), *Frying (food)* (Frire) et *Stir* (remuer une substance fluide). On remarque ainsi que l'algorithme se révèle parfois très pertinent puisque l'ensemble des descripteurs ci-dessus sont des bons descripteurs du champ sonore de la cuisine. Néanmoins, cette situation n'est pas non plus satisfaisante dans la mesure où elle ne permet pas une association point par point entre les deux ontologies. Et, on ne peut pas ici blâmer la capacité de l'algorithme à classer comme il faut les sons qui lui sont soumis. En effet, hormis la catégorie *zipper (clothing)* qui dénote, un humain aurait été tout à fait capable de d'associer les sons de *Cooking* de manière similaire s'il n'avait sous la main que le dictionnaire de labels de l'ontologie Audioset. Dès lors, la situation n'est plus tant un défaut de performances de l'algorithme qu'un défaut dans la construction de l'ontologie Audioset (pour notre usage du moins).

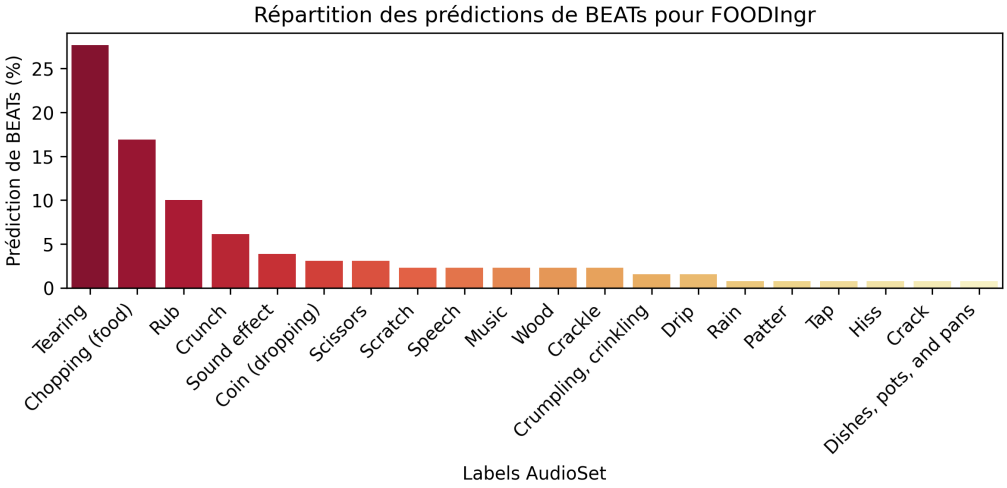
2. Frottement de 2 surfaces entre elles
3. Le bruit de nombreuses petites bulles qui éclatent, le plus souvent lorsque l'on fait revenir des aliments dans une poêle avec de la matière grasse
4. Le bruit de la découpe d'ingrédients en morceaux à l'aide d'un couteau ou d'un outil à lame similaire



(a) Répartition des prédictions de BEATs pour la sous-catégorie *Cooking*.

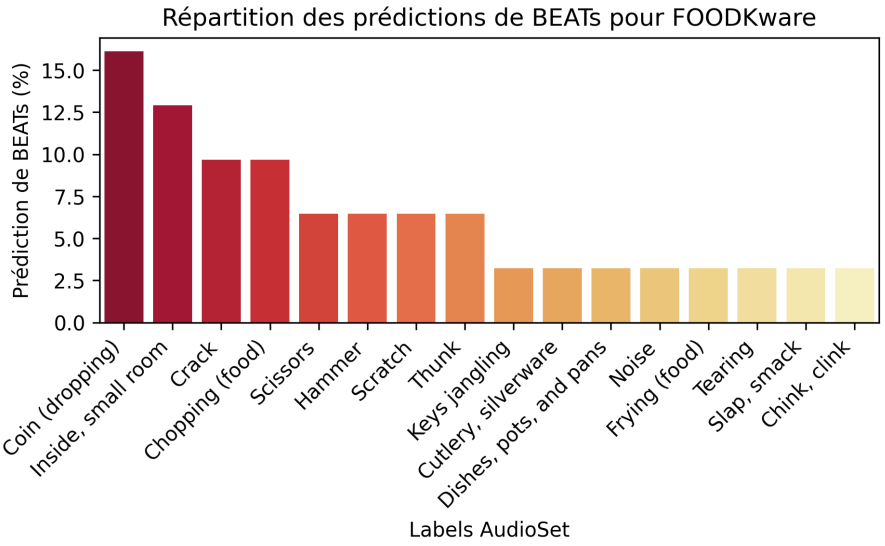


(b) Répartition des prédictions de BEATs pour la sous-catégorie *Glassware*.

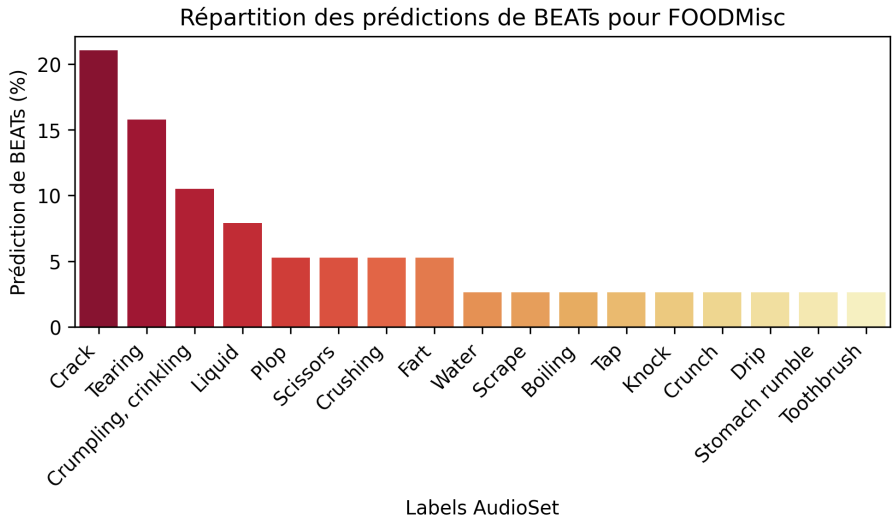


(c) Répartition des prédictions de BEATs pour la sous-catégorie *Ingredients*.

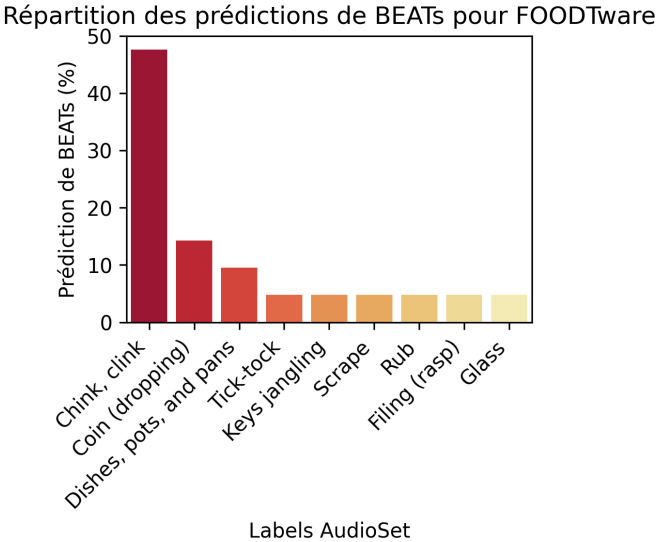
FIGURE 21 – Distribution des prédictions de BEATs pour les sous-catégories de *FOOD&DRINK* (1/2).



(a) Répartition des prédictions de BEATs pour la sous-catégorie *Kitchenware*.



(b) Répartition des prédictions de BEATs pour la sous-catégorie *Miscellaneous*.



(c) Répartition des prédictions de BEATs pour la sous-catégorie *Tableware*.

FIGURE 21 – Distribution des prédictions de BEATs pour les sous-catégories de *FOOD&DRINK* (2/2).

Ceci s'explique en partie par le fait que la sous-catégorie *Cooking* est une classe regroupant les éléments sonores relatifs au *champ sémantique* de la cuisine et non pas regroupant les sources par similarités acoustiques. *Cooking* est une classe avec une *forte variabilité morphologique intra-classe* ce qui explique la répartition des labels Audioset : il y en a beaucoup avec une probabilité faible mais tout de même significative (entre 5% et 20%).

Par ailleurs, nous constatons qu'il y a un *fort recouvrement* entre les classes UCS si l'on se place dans l'ontologie Audioset. En effet, de nombreux labels se retrouvent dans le top 5 des corrélations avec les classes UCS. C'est, par exemple, le cas du label *Rub* qui se retrouve représentant des catégories *cooking* (10%), *ingredients* (17%) et *kitchenware* (9%), de même que *coin (dropping)* qui est corrélé avec *glassware* (5%), *kitchenware* (16%) et tableware (14%). Nous montrons donc, à travers ce test, l'incompatibilité profonde qui existe entre ces deux ontologies. Aussi, une simple liaison point par point entre les classes des deux ontologies est impossible. Il y a de fait, un besoin de réenseigner à l'algorithme à penser "différemment" et cela ne peut se faire qu'au travers d'un nouvel apprentissage, ce qui justifie l'étude menée dans le chapitre 2.4 et dont les résultats sont exposés dans la section suivante.

3.1.2 Utilisation de BEATs après ajustement.

Nous présentons dans cette partie, les résultats de l'algorithme BEATs ajusté sur une classification UCS.

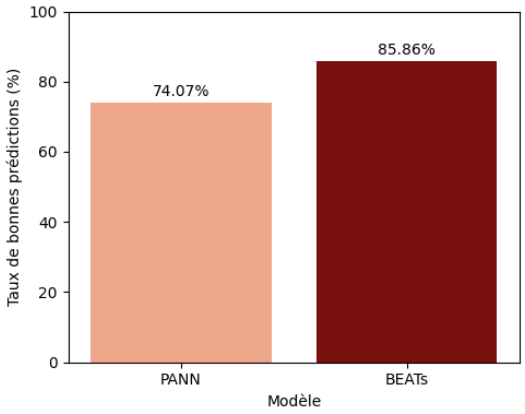
A Comparaison entre deux modèles ajustés sur le jeu de données construit.

Le choix de l'algorithme pré-entraîné BEATs a été justifié au préalable dans la partie 2.2. Toutefois, afin de confirmer ce choix, nous avons appliqué la même méthode que celle détaillée dans le chapitre 2 avec le classifieur PaNNs. PaNNs (Kong *et al.*, 2020) est un modèle basé sur des réseaux de neurones à convolutions qui utilisent l'attention ; son fonctionnement diffère largement de celui de BEATs, qui lui, est basé sur une architecture transformeur. Les performances des deux modèles ont été comparées [5], ceux-ci ont tous deux été évalués en utilisant le même jeu d'évaluation en appliquant une mesure d'exactitude, les résultats sont exposés sur la figure 22.

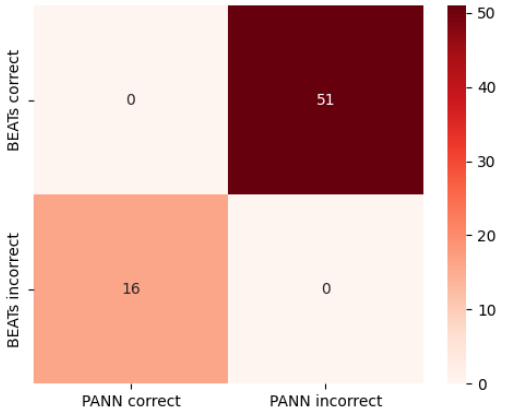
Le diagramme en barre 22a montre que sur le jeu d'évaluation, BEATs effectue environ 10% de bonnes précisions de plus que PaNNs. À côté, la figure 22b montre la matrice de contingence entre PaNNs et BEATs, révélant le nombre d'occurrences

5. Il est à noter que les jeux d'apprentissage et d'évaluation ne contenaient pas l'entièreté des données à notre disposition mais uniquement une partie de celles-ci, le but ici étant simplement de comparer deux algorithmes, des petits jeux de données suffisaient.

où l'un des deux modèles a fait une prédiction correcte pendant que l'autre a fait une prédiction incorrecte. Le tableau mentionne 16 occurrences où PaNNs à prédit correctement et BEATs non, et 51 occurrences ou BEATs a prédit correctement et PaNNs non.



(a) Pourcentage de prédictions correctes sur le jeu d'évaluation pour les deux modèles ajustés.



(b) Matrice de contingence entre BEATs et PaNNs.

FIGURE 22 – Étude des performances relatives des deux modèles ajustés BEATs et PaNNs.

Le test de McNemar (MCNEMAR, 1947) permet de comparer deux modèles évalués dans les mêmes conditions. Soit A le nombre d'éléments identifiés correctement par BEATs et incorrectement par PaNNs et B le nombre d'éléments identifiés correctement par PaNNs et incorrectement par BEATs, alors la statistique de test de Mac Nemar est défini par la relation :

$$K = \frac{(A-B)^2}{A+B}$$

Ce test rend aussi compte d'une valeur p en supposant que la distribution de probabilité de l'hypothèse nulle suit une loi du χ^2 à 1 degrés de liberté. La *p-value* est définie comme étant la probabilité qu'un modèle statistique vérifiant l'hypothèse nulle donne un résultat au moins aussi extrême que celui observé.
Concrètement :

- Si $p < 0,01$: Très forte chance d'écarter l'hypothèse nulle.
- Si $p < 0,05$: Forte chance d'écarter l'hypothèse nulle
- Si $0,05 < p < 0,1$: Faible chance d'écarter l'hypothèse nulle
- Si $p > 0,1$: L'hypothèse nulle ne peut pas être écartée.

Dans notre cas, l'hypothèse nulle est la suivante : PaNNs et BEATs présentent les mêmes performances de classification sur le jeu d'évaluation constitué.

L'étude statistique des résultats obtenus conduit à une statistique de test $\mathbf{K = 17,2537}$ et une valeur $\mathbf{p = 3,27.10^{-5}}$. La valeur p est largement inférieure à 0.01, ce qui nous permet de rejeter l'hypothèse nulle dans plus de 99% des cas. La différence de performance entre les deux modèles est significative : **l'architecture BEATs se révèle bien être la plus appropriée pour la classification de sons de nourritures et breuvages (*FOOD&DRINK*).** Dans la suite de ce travail nous exposerons donc les résultats de ce modèle.

B Performances de l'algorithme.

Lors de l'apprentissage nous avons justifié la nécessité de tester plusieurs hyperparamètres (cf. 2.4.2). La figure 23 présente les résultats pour l'exactitude globale du modèle BEATs entraînés avec différents hyperparamètres sur un jeu d'évaluation réduit [6]. La taille des lots (*batch size*) est désignée par les lettres BS ; le pas d'apprentissage (*learning rate*) est abrégé par les lettres LR. L'exactitude globale d'un modèle correspond au nombre de sons correctement classés, divisé par le nombre total de sons dans le jeu d'évaluation.

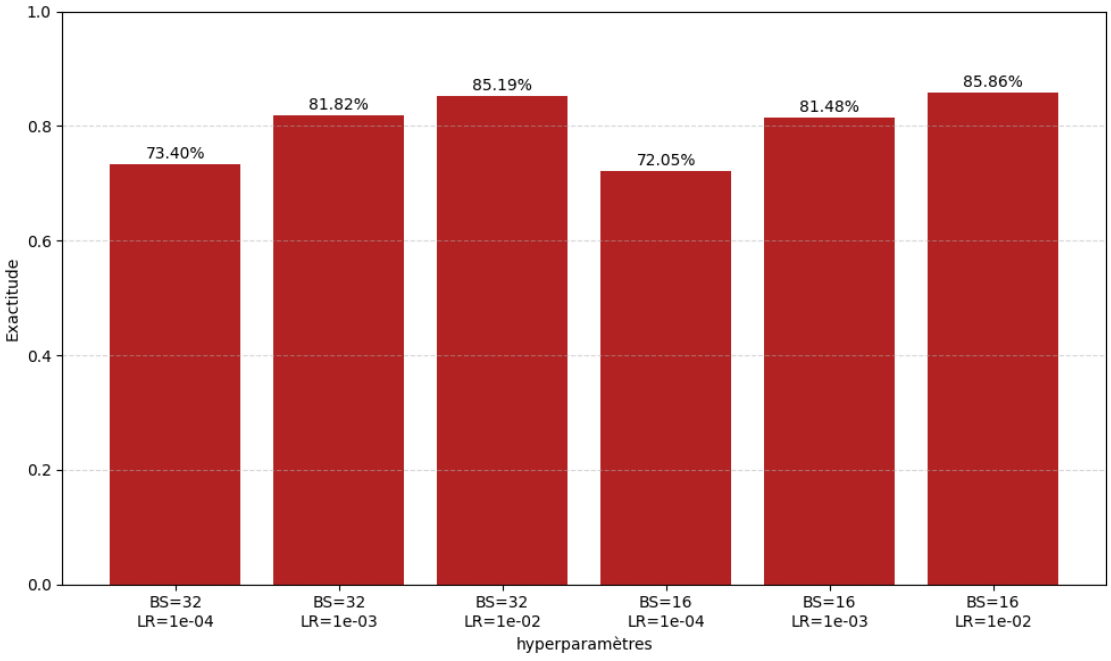


FIGURE 23 – Exactitude globale en fonction des hyperparamètres.

La figure 23 montre que l'exactitude du modèle dépend majoritairement du pas d'apprentissage, et que l'influence de la taille des lots n'est que peu significative.

6. Ici encore, l'entraînement et l'évaluation des têtes de classification ont été effectués sur des jeux de données réduits comprenant une portion de la totalité des ressources disponibles. Réduire artificiellement le nombre de données permet un gain de temps considérable lors de la phase d'apprentissage, tout en laissant clairement apparaître des résultats exploitables.

Le modèle exhibant la meilleure performance toutes classes confondues est celui entraîné avec une taille de lot de 16 fichiers et un pas d'apprentissage de 0.01, réalisant un score de **85,86%** sur un jeu d'évaluation de taille réduite. Maintenant que les hyperparamètres optimaux ont été identifiés, nous pouvons maintenant entraîner notre tête de classification sur l'entièreté des données en notre possession. Les résultats présentés dans la suite seront ceux de cette tête de classification.

La *matrice de corrélation* ou *matrice de confusion* renseigne sur la capacité d'un algorithme à prédire correctement une classe. La figure 24 fait apparaître la *classe réelle* (vérité terrain) sur l'axe vertical, et la *classe prédite* par l'algorithme sur l'axe horizontal. On remarque que la classe *Misc* est la plus correctement identifiée. La sous-catégorie *Misc* est victime de sur-représentation, c'est-à-dire que l'algorithme a tendance à trop souvent attribuer cette classe quitte à se tromper. La colonne *FOODMisc* a une somme supérieure à 1 (1.68) et les catégories réelles *Kitchenware* et *Cooking* se retrouvent donc parfois injustement attribuées à la catégorie *Misc* par l'algorithme. Nous sommes tentés d'expliquer cette sur-représentation de la manière suivante : la classe *Misc* regroupe tous les sons qui n'appartiennent pas explicitement aux 5 autres sous-catégories, elle présente donc des éléments très hétérogènes ce qui explique la corrélation entre les classes *Cooking* et *Kitchenware* avec la classe *Misc*.

Au contraire, la classe *Kitchenware* est la plus difficilement identifiée avec seulement 35% de prédictions correctes. Il est intéressant de se poser les questions d'un tel échec, en particulier, le peu d'exemples de cette sous catégorie dans le jeu d'entraînement comme le montre le tableau 2.3 laisse imaginer qu'avec plus de données il n'y aurait pas ce problème.

Par ailleurs, l'algorithme semble faire une corrélation entre *Glassware* et *Tableware* : si un son de *Tableware* est présenté au classifieur, celui-ci a une probabilité de se tromper en le classant dans la catégorie *Glassware* égale à 26%. Cette anomalie ne peut cette fois pas s'expliquer par un manque de données puisque, toujours en accord avec le tableau 2.3, la catégorie *Tableware* est une des plus représentées. Néanmoins on peut rapprocher les natures des actions performées dans les audios des deux catégories Glassware et Tableware, les deux étant semblables et consistant majoritairement en des impacts, des poses et des saisies, ce qui peut amener l'algorithme à les confondre.

Les données de notre jeu d'entraînement sont des fichiers audios d'une durée de 4 secondes, ce choix a été justifié à la fin du 2.1. Il est donc cohérent de tester l'algorithme sur des fichiers eux aussi d'une durée de 4 secondes, cependant, il est aussi important de rappeler que notre recherche se place dans le cadre d'une pratique professionnelle concrète. En réalité, les fichiers de sonothèques ne font pas tous exactement 4 secondes, et ces derniers durent généralement entre quelques secondes

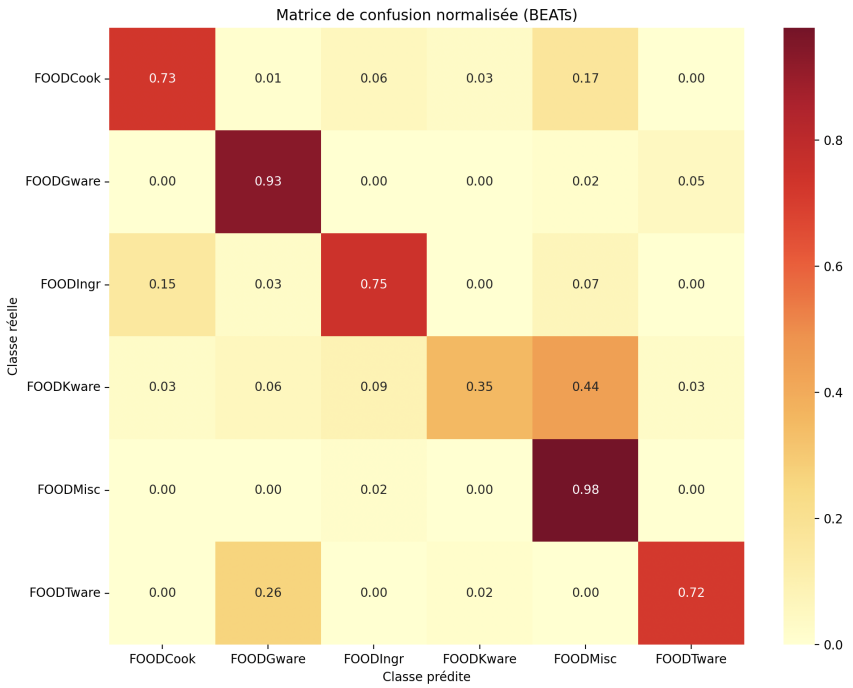


FIGURE 24 – Matrice de confusion normalisée par ligne (BEATs).

et quelques minutes. Aussi nous avons souhaité évaluer l'algorithme dans une situation réelle. Dans la figure 25, l'exactitude du modèle après ajustement a été mesurée sur le jeu d'évaluation, ce dernier étant constitué des fichiers tantôt entiers, tantôt découpés en échantillons de 4 secondes.

L'étude de la figure 25 révèle que l'algorithme performe mieux sur des fichiers de durée identique à ceux sur lequel il a été entraîné en atteignant un score d'exactitude de **87,65%**. En revanche, les résultats baissent de plus de 10% atteignant **74.75%** d'exactitude lorsqu'on considère des fichiers entiers, c'est-à-dire dans une situation réelle.

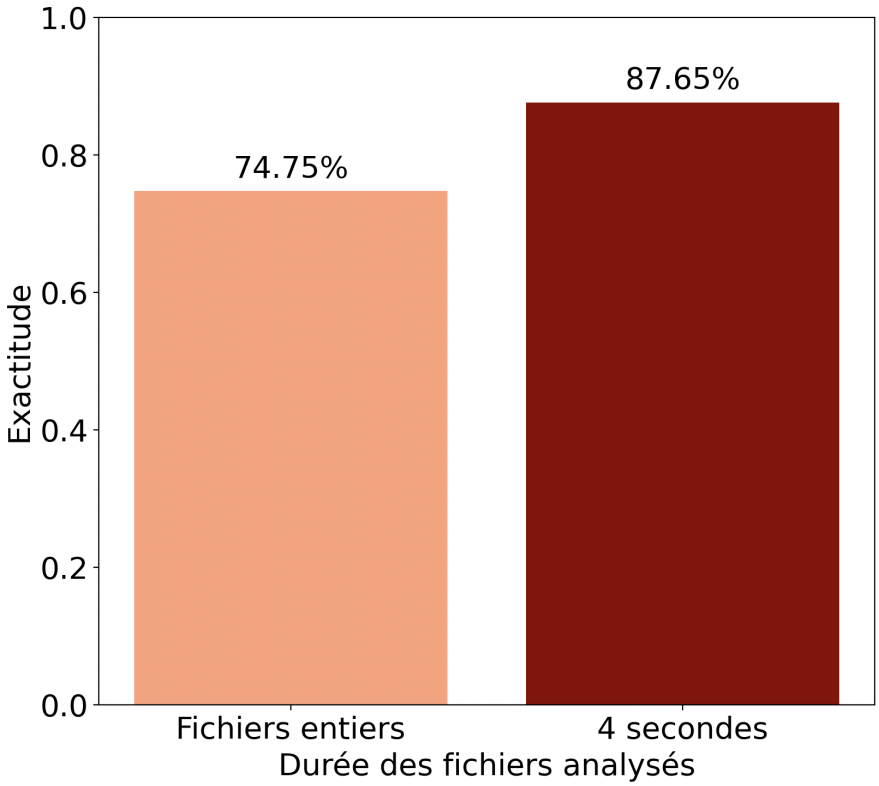


FIGURE 25 – Performances du modèle BEATs ajusté, comparaison entre des fichiers entiers et des fichiers d'une longueur de 4 secondes.

3.2 DISCUSSIONS.

3.2.1 Discussions sur les performances de l'algorithme.

Lors de cette étude, nous avons présupposé un certain nombre d'hypothèses sur la base de notre expérience personnelle et de témoignages recueillis de manière informelle. Une de ces hypothèses correspond au fait de considérer que les sons en sonothèques ne possèdent qu'une seule source principale. Bien que ce soit en majorité le cas pour le jeu de données que nous avons constitué, on trouve tout de même des exemples de sons labellisés *Cooking* qui sont plus des sons d'ambiance de cuisine de restaurant que des effets sonores analytiques. On peut légitimement se poser la question de comment labelliser un tel son et remettre ici en question l'UCS. Est-ce qu'il est possible d'inventer une catégorisation plus pertinente ? Cette question n'a pas été traitée dans ce mémoire et les études sur ce sujet, comme celle de Jean-Michel DENIZARD (2017), ne trouvent pas encore d'application concrète dans le monde professionnel.

3.2.2 Extension du modèle à une ontologie plus détaillée.

Nous nous sommes restreints, dans le cadre de ce mémoire, au cas des sons de nourritures et breuvages. Nous considérons que notre algorithme obtient un score de classification satisfaisant sur les 6 classes qui lui ont été présentées. Cependant, une prochaine étape consisterait à appliquer la même méthode sur un plus grand nombre de classes. Il serait intéressant d'observer si la performance se maintient, ou s'il existe éventuellement un *seuil critique* du nombre de classes à partir duquel la précision chute drastiquement. L'idée d'un classifieur capable de répartir des sons dans la quasi-totalité des sous-catégories UCS, soit plus de 700 classes, avec un score acceptable ne paraît pas irréaliste, il faudra néanmoins prendre le temps de constituer un jeu de données couvrant l'ensemble des catégories. Si le modèle se retrouvait mis en défaut par un grand nombre de classes, on pourrait alors se poser la question d'entraîner le modèle à partir de zéro, c'est-à-dire en effectuant à nouveau un pré-apprentissage. Malheureusement, les puissances de calculs nécessaires à l'entraînement d'algorithmes effectuant des tâches complexes sont souvent démesurées. À titre d'exemple *llama3* le modèle de langage développé par Facebook a été entraîné pendant près de 7,7 millions d'heures GPU, évidemment le traitement étant fait en parallèle sur plusieurs cartes graphiques dotées de plusieurs cœurs de traitement. Cela représente tout de même 3 mois en continu sur une centaine de cartes graphiques possédant chacune 32 cœurs de traitement (AI@Meta, 2024).

La constitution d'un large jeu de données en UCS permettraient un premier apprentissage des modèles sur des fichiers de taille identiques et normées (par exemple 4 secondes) permettant à l'algorithme d'identifier un certains nombre de motifs dans le spectre. Il faudrait ensuite pratiquer un apprentissage par transfert en utilisant des fichiers réels de sonothèque de longueurs variables afin de maximiser les performances de l'algorithme dans le cas d'une situation réelle de montage son.

3.2.3 Expériences avec des humains.

Par ailleurs, nous parlons de score satisfaisant, mais on est en droit de se demander qu'est ce qu'il est bon de considérer comme "acceptable". Nous proposons dans ce paragraphe, deux expériences qu'il serait intéressant de mener sur des populations humaines, qui pourraient servir cette étude. La première expérience consisterait à demander à plusieurs monteurs sons de constituer une scène sonore sur une vidéo muette, à partir de sons de sonothèques annotés en UCS. Les uniques critères de recherche dans la sonothèque seraient la catégorie et la sous-catégorie UCS (les fichiers seraient anonymisés et toutes les autres métadonnées écrasées), les monteurs seraient alors invités à constituer 4 scènes différentes avec des sonothèques plus ou moins bien organisées, c'est-à-dire avec 60%, 70%, 80% et 100% de sons bien

rangés. En définitive, les monteurs devraient remplir un questionnaire reposant sur des échelles de Likert qui permettraient d'établir un *seuil de rangement critique* en dessous duquel il devient vraiment difficile de travailler correctement. Chaque expérience de montage pourrait être chronométrée afin d'avoir une mesure objective de la difficulté à créer la scène sonore, en plus du ressenti subjectif du monteur. Une liste de questions qui nous semblent pertinentes pourraient être :

- Quel âge avez-vous ?
- Depuis combien de temps exercez-vous ce métier ?
- Sur une échelle de 1 à 5, quelle était la difficulté à faire le montage son de cette scène ?
- Sur une échelle de 1 à 5, quelle aurait été la difficulté à faire le montage son de la scène en travaillant avec une sonothèque optimale ?
- Quels sont les caractéristiques de la sonothèque qui vous ont le plus mis en défaut ?
  - Le fait que certains sons de correspondent pas à ce qui est écrit dans leur catégorie UCS (de 1 à 5) ?
  - L'absence de noms de fichiers clairs (de 1 à 5) ?
  - Autre chose ? Précisez

Une deuxième expérience consisterait à demander à un groupe d'humains, pas nécessairement spécialistes, de classer une dizaine de sons de quelques secondes dans des catégories UCS. On pourrait ainsi en déduire un score de précision d'une population humaine sur un jeu d'évaluation qui serait mis en regard des performances d'un algorithme sur le même jeu d'évaluation. En effet, espérer 100% de classifications correctes sur le groupe humain, paraît tout de même irréaliste, une grande partie des sons étant polysémiques, ils pourraient appartenir à plusieurs classes, ce qui conduirait un sujet à mal les classifier.

Le Travail de sonothécaire

Au travers de l'entretien mené auprès de Rodrigo Sacic, sonothécaire chez HAL, nous nous sommes vite rendus compte de la complexité d'organiser une sonothèque. Si une part de notre problématique de recherche à été étudiée, il reste néanmoins beaucoup à faire si l'on veut étudier l'ergonomie d'une sonothèque et le *workflow* généralisé du monteur son. La norme UCS est certes une bonne initiative, mais elle n'est pas une solution complète à la description multifactorielle des sons, n'intégrant, par exemple, pas ou peu de critères relatifs à la typologie Schaefferienne des sons. De plus, il faut noter que choisir la catégorie et la sous-catégorie n'est au final pas

ce qui prend le plus de temps au sonothécaire, et des champs comme les mots-clés (*Keywords*) ou la Description sont en fait assez longs et bien plus complexes à remplir. Nous pourrions alors nous intéresser à la *détection* de sources au sein d'un fichier. En se basant sur une ontologie comme Audioset, le thésaurus de *Sons de France*, ou une autre, à inventer, contenant un grand nombre de sources explicites. Pour chaque son, on pourrait chercher à lui associer un certain nombre de mots-clés (dans le champ de Soundminer *Keywords*) qui auraient été détectés dans le fichier, en plus de la classification en sous-catégories UCS.

Cependant, les champs comme *Keywords* et Description nécessitent une compréhension profonde de la polysémie des sons, relatant la vaste étendue de ses intégrations possibles dans le montage d'un produit audiovisuel. L'indexation est un *savoir spécialisé*, un humain normal (entendons par normal un individu pas particulièrement sensibilisé au montage son) aussi intelligent soit-il serait sans doute incapable d'effectuer cette indexation de manière optimale. À l'instar du neurochirurgien, le sonothécaire est un spécialiste. À l'heure actuelle, nous pensons qu'il y a peu de chances qu'un automate soit apte à effectuer cette tâche d'indexation efficacement, et ceci repose principalement sur l'incompatibilité profonde entre le caractère polysémique des sons et la manière dont sont construites les ontologies pour l'apprentissage automatique.

3.2.4 Le problème de la polysémie pour l'apprentissage automatique.

Comme le souligne Georges Lakoff (1986), la simple catégorisation binaire (appartient/n'appartient pas) ne saurait s'étendre aux *concepts abstraits* comme la linguistique (ou pour ce qui nous concerne, les sons). Le sonore est un domaine ou la pluralité des symboles véhiculés par le même objet est conséquente, aussi, le caractère immatériel, intangible du son, invite fortement à la polysémie. Si cette caractéristique permet à l'artisan du son de façonner sa "matière" afin d'influer la perception de celle-ci, la polysémie n'en demeure pas moins un obstacle au regard de la catégorisation automatique. Les classifieurs algorithmiques se basent sur une *ontologie* pré-définie, comme détaillé dans le C, chaque ontologie se doit d'exprimer *sans ambiguïté* les relations entre les différents concepts, or **il ne peut y avoir de lecture polysémique sans ambiguïté**. Il demeurera donc impossible de classer les sons selon plusieurs hiérarchies en parallèle tant que les ontologies ne changeront pas. Bien sûr la *polysémie* d'un son peut se diriger vers une perception *monosémique* en fonction du contexte auquel il appartient, c'est le cas du *moulin à café de Raymond Murray Schafer* A.1.2, qui une fois situé dans son contexte, ne véhicule presque plus que l'information causale qui lui est attachée. En ce sens, le perfection-

nement du *procédé d'attention* nécessaire à l'analyse du contexte, dans les modèles de classifications actuelles, semble constituer une piste à explorer pour cerner la pluralité des symboles véhiculés par chaque objet.

3.2.5 Les I.A. génératives.

Que peut-on dire des intelligences artificielles qui génèrent du son de manière synchrone à l'image ? La recherche en matière d'I.A. générative s'est largement développée au cours des dernières années et le milieu de l'audiovisuel a lui aussi été fortement affecté. En effet, il n'est plus rare de voir des images fixes ou animées générées à partir d'un *prompt*[7] dans des contextes professionnels. De même, des modèles comme *Fugatto* chez NVIDIA sont capables de créer la bande son d'une vidéo muette de manière synchrone, ou de générer des effets sonores à partir d'un *prompt* textuel. En dépit de résultats parfois jugés insuffisants par les utilisateurs, ces outils restent tout de même l'objet de nombreux débats. Dans un contexte où les intelligences artificielles de plus en plus performantes, remettent en question la notion d'art, la question éthique du remplacement des techniciens de l'audiovisuel se pose également.

7. cf. Glossaire D.1

Conclusion.

Ce mémoire a pris pour point de départ une problématique concrète du monde professionnel audiovisuel : le temps conséquent consacré par les monteurs sons à l'*indexation manuelle* des fichiers dans les *sonothèques*. Cette tâche, bien que fondamentale pour garantir la fluidité du travail de création sonore, demeure fastidieuse et peu valorisée. L'essor de l'*apprentissage automatique* et des systèmes de *classification* des sons nous a poussés à interroger la pertinence d'un tel outil pour alléger cette charge de travail. En particulier, nous avons évalué l'utilisation du modèle *BEATs* pour classer les sons selon l'ontologie *UCS*, aujourd'hui massivement adoptée dans les milieux professionnels de la *postproduction audiovisuelle*.

Le cas d'usage étudié s'inscrit dans une approche que nous qualifierons de *classification causale* majoritairement acceptée. En effet, comme le soulignent plusieurs travaux, les sons à faible ambiguïté sont souvent catégorisés selon leur *source causale*, ce qui correspond à l'usage opérationnel majoritaire des monteurs son. Si cette logique ne couvre pas l'ensemble des situations — notamment les sons transformés ou fortement polysémiques — elle constitue néanmoins un standard robuste auquel le modèle *BEATs* a été confronté. Dans ce cadre bien défini, les performances obtenues s'avèrent satisfaisantes : avec un score de précision supérieur à 85 % sur le jeu d'évaluation, le modèle montre une capacité claire à identifier des sons dans des sous-catégories UCS, malgré les limites intrinsèques au jeu de données utilisé.

Les résultats sont d'autant plus encourageants que les sous-catégories testées, issues du domaine *FOOD&DRINK*, présentaient une variabilité interclasse importante, ainsi qu'une séparation de niveau sémantique élevée. Certaines catégories, comme *Cooking* ou *Ingredients*, intègrent une diversité morphologique notable, tant du point de vue *acoustique* que *symbolique*. Ce constat est crucial : si un classifieur est capable de traiter efficacement ces catégories complexes, on peut légitimement espérer qu'il soit capable de *généraliser* à un plus grand nombre de classes dans l'architecture complète de l'*UCS*.

Cependant, il convient de nuancer ce résultat. L'*indexation sonore* ne se réduit pas à l'attribution d'une catégorie et d'une sous-catégorie *UCS*. Comme l'ont rappelé les professionnels interrogés durant des discussions informelles, le remplissage

de champs tels que les *mots-clés*, les *descriptions* ou les *attributs contextuels* représente une part substantielle du travail du *sonothécaire*. Ces éléments nécessitent une compréhension fine de la *polysémie* sonore, de son potentiel narratif, mais aussi de sa fonction dans une séquence audiovisuelle. Une automatisation complète paraît donc prématurée ; l'objectif réaliste serait plutôt un outil d'aide à l'indexation, permettant de guider le monteur ou le sonothécaire sans le remplacer.

Plusieurs pistes de recherche s'ouvrent désormais. D'abord, la constitution d'un *jeu de données de référence* entièrement labellisé selon l'UCS apparaît comme une priorité. Les données existent : de nombreuses *sonothèques professionnelles* sont déjà annotées selon cette norme, il suffirait alors de centraliser ces ressources pour constituer un jeu de données collectées utilisable ensuite pour l'entraînement d'algorithmes spécialisés et adaptés aux besoins de l'industrie audiovisuelle. Cela permettrait aussi d'augmenter le nombre de classes étudiées, de diversifier les contextes sonores, et de vérifier la robustesse du modèle développé à grande échelle.

Par ailleurs, deux expérimentations basées sur une population humaine seraient particulièrement éclairantes. La première consisterait à faire classer manuellement un sous-ensemble de sons par plusieurs auditeurs, et à comparer leurs performances à celles de *BEATs*. Cette expérience permettrait de déterminer ce qu'on peut attendre raisonnablement d'un algorithme, dans un domaine où l'ambiguïté est parfois irréductible. La seconde viserait à évaluer l'impact de la qualité d'indexation sur le travail de montage lui-même : en soumettant à des monteurs des bases de données dont l'indexation est plus ou moins fiable, on pourrait chercher à déterminer un *seuil de précision critique* au-delà duquel le gain de temps et de qualité devient significatif.

Enfin, plusieurs pistes techniques peuvent être explorées. L'une consisterait à *remplacer la projection linéaire* finale par un modèle plus complexe, capable de mieux prendre en compte les interactions entre classes ou les corrélations entre *caractéristiques acoustiques* et *sémantiques*. Une autre, plus ambitieuse, consisterait à envisager un *ré-entraînement complet de BEATs*, en utilisant directement des *ontologies audiovisuelles* (comme l'UCS) au lieu d'Audioset, qui demeure inadaptée à ces usages professionnels spécifiques.

En définitive, ce travail s'inscrit dans une démarche exploratoire et pragmatique. Il démontre qu'une liaison est possible entre la recherche académique en *classification des sons environnementaux* et les usages professionnels des *monteurs son*. Si l'automatisation complète de l'indexation n'est pas encore à portée immédiate, une *assistance intelligente et contextuelle* apparaît déjà comme une perspective réaliste et utile. C'est dans cette direction que les recherches futures pourront s'orienter.

Bibliographie

Abayomi-Alli, O. O., Damaševičius, R., Qazi, A., Adedoyin-Olowe, M., & Misra, S. (2022). Data Augmentation and Deep Learning Methods in Sound Classification: A Systematic Review. *Electronics*, *11*(22), 3795 (cf. p. 50).

Adjiman, R. (2015, février). *Sémiotique des sons et cognition située*. (Cf. p. 16, 17).

AI@Meta. (2024). Llama 3 Model Card (cf. p. 79).

Bahdanau, D., Cho, K., & Bengio, Y. (2016). Neural Machine Translation by Jointly Learning to Align and Translate. (Cf. p. 61).

Ballas, J. (1993). Common Factors in the Identification of an Assortment of Brief Everyday Sounds. *Journal of experimental psychology. Human perception and performance*, *19*, 250-67 (cf. p. 57).

Bansal, A., & Garg, N. K. (2022). Environmental Sound Classification: A descriptive review of the literature. *Intelligent Systems with Applications*, *16*, 200115 (cf. p. 27, 28).

Bello, J. P., Mydlarz, C., & Salamon, J. (2018). Sound analysis in smart cities. *Computational analysis of sound scenes and events*, 373-397 (cf. p. 28).

Bellos, A. (2014). 'Seven' triumphs in poll to discover world's favourite number. *The Guardian* (cf. p. 14).

Bhat, A. S., Amith, V., Prasad, N. S., & Mohan, D. M. (2014). An Efficient Classification Algorithm for Music Mood Detection in Western and Hindi Music Using Audio Feature Extraction. *2014 Fifth International Conference on Signal and Image Processing*, 359-364 (cf. p. 97, 98).

Cano, E., FitzGerald, D., Liutkus, A., Plumbley, M. D., & Stoter, F.-R. (2019). Musical Source Separation: An Introduction. *IEEE Signal Processing Magazine*, *36*(1), 31-40 (cf. p. 99).

Cayton, L. (2005). Algorithms for manifold learning (cf. p. 105, 106).

Chachada, S., & Kuo, C.-C. J. (2013). Environmental sound recognition: A survey. *APSIPA Transactions on Signal and Information Processing*, *3*, 1-9 (cf. p. 50).

Chandrakala, S., & Jayalakshmi, S. L. (2019). Environmental Audio Scene and Sound Event Recognition for Autonomous Surveillance: A Survey and

Comparative Studies [Place: New York, NY, USA Publisher: Association for Computing Machinery]. *ACM Comput. Surv.*, *52*(3) (cf. p. 28).

Chaudhuri, R., & Fiete, I. (2016). Computational principles of memory. *Nature Neuroscience*, *19*(3), 394-403 (cf. p. 106).

Chen, L., Gunduz, S., & Ozsu, M. T. (2006). Mixed Type Audio Classification with Support Vector Machine. *2006 IEEE International Conference on Multimedia and Expo*, 781-784 (cf. p. 28).

Chen, S., Wu, Y., Wang, C., Liu, S., Tompkins, D., Chen, Z., Che, W., Yu, X., & Wei, F. (2023). BEATs: Audio Pre-Training with Acoustic Tokenizers [ISSN: 2640-3498]. *Proceedings of the 40th International Conference on Machine Learning*, 5178-5193 (cf. p. 39, 44, 46, 47, 54, 59).

Chion, M. (1995). *Guide des objects sonores: Pierre Schaeffer et la recherche musicale* (Nouv. éd). (Cf. p. 91, 95).

Chion, M. (2010). *Le son: traité d'acoulogie* (2e éd. revue et corrigée). A. Colin. (Cf. p. 24, 94).

Chion, M. (2021). *L'audio-vision - 5e éd : Son et image au cinéma.* (Armand Collin). (Cf. p. 91).

Chong, D., Wang, H., Zhou, P., & Zeng, Q. (2022). Masked Spectrogram Prediction For Self-Supervised Audio Pre-Training [_eprint: 2204.12768]. (Cf. p. 44).

Couvreur, C., Fontaine, V., Gaunard, P., & Mubikangiey, C. G. (1998). Automatic Classification of Environmental Noise Events by Hidden Markov Models. *Applied Acoustics*, *54*(3), 187-206 (cf. p. 28).

Dai, W., Dai, C., Qu, S., Li, J., & Das, S. (2016, octobre). Very Deep Convolutional Neural Networks for Raw Waveforms [arXiv:1610.00087 [cs]]. (Cf. p. 111).

Delplancq, J. (2009). *La modélisation des connaissances dans le montage son au cinéma* [Mémoire de master]. ENS Louis-Lumiere. (Cf. p. 15-17).

Denizard, J.-M. (2017). L'émergence des significations chez le monteur son, au cours de la recherche et de la sélection des sons : une approche communicationnelle et cognitive (cf. p. 12, 16, 17, 24, 78).

Dong, X., Yin, B., Cong, Y., Du, Z., & Huang, X. (2020). Environment Sound Event Classification With a Two-Stream Convolutional Neural Network. *IEEE Access*, *PP*, 1-1 (cf. p. 111).

f. 1703, A. (s. d.). Max Pooling. (Cf. p. 107).

Gemmeke, J. F., Ellis, D. P. W., Freedman, D., Jansen, A., Lawrence, W., Moore, R. C., Plakal, M., & Ritter, M. (2017). Audio Set: An ontology and human-labeled dataset for audio events. *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 776-780 (cf. p. 35, 37, 53, 59).

Gong, Y., Chung, Y.-A., & Glass, J. (2021). AST: Audio Spectrogram Transformer. *Interspeech 2021*, 571-575 (cf. p. 59).

Gruber, T. R. (1993). A translation approach to portable ontology specifications. *Knowledge Acquisition*, *5*(2), 199-220 (cf. p. 14).

Gygi, B., Kidd, G., & Watson, C. (2004). Spectral-temporal factors in the identification of environmental sounds. *The Journal of the Acoustical Society of America*, *115*, 1252-65 (cf. p. 49).

Harb, H. (2001). *Classification d'un signal sonore en vue d'une indexation par le contenu des documents multimédias* [thèse de doct.]. (Cf. p. 53).

Huang, P.-Y., Xu, H., Li, J., Baevski, A., Auli, M., Galuba, W., Metze, F., & Feichtenhofer, C. (2023). Masked Autoencoders that Listen [_eprint: 2207.06405]. (Cf. p. 47, 48).

Kong, Q., Cao, Y., Iqbal, T., Wang, Y., Wang, W., & Plumbley, M. D. (2020). PANNs: Large-Scale Pretrained Audio Neural Networks for Audio Pattern Recognition [_eprint: 1912.10211]. (Cf. p. 52, 59, 73).

Koutini, K., Schlüter, J., Eghbal-zadeh, H., & Widmer, G. (2022). Efficient Training of Audio Transformers with Patchout. *Interspeech 2022, 23rd Annual Conference of the International Speech Communication Association, Incheon, Korea, 18-22 September 2022*, 2753-2757 (cf. p. 59).

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. *Advances in Neural Information Processing Systems*, *25* (cf. p. 104).

Lakoff, G. (1986). *Women, fire, and dangerous things: what categories reveal about the mind* (paperback ed., [Nachdr.]). The Univ. of Chicago Press. (Cf. p. 11, 81).

Larras, B. (2015, décembre). *CMOS analog implementation of clique-based neural networks* [thèse de doct.]. (Cf. p. 101).

Laurin, J. (2018). *Enregistrement et traitement de métadonnées dynamiques dans le flux de travail audionumérique* [Mémoire de master]. ENS Louis-Lumiere. (Cf. p. 20).

Mallat, S., & Geoffrey, P. (2020). Modèles multi-échelles et réseaux de neurones convolutifs. (Cf. p. 111).

McNemar, Q. (1947). Note on the Sampling Error of the Difference Between Correlated Proportions or Percentages. *Psychometrika*, *12*(2), 153-157 (cf. p. 74).

Mesaros, A., Heittola, T., & Ellis, D. (2018). Datasets and Evaluation. In T. Virtanen, M. D. Plumbley & D. Ellis (éd.), *Computational Analysis of Sound Scenes and Events* (p. 147-179). Springer International Publishing. (Cf. p. 54).

Moles, A. (1959). Classification d'une sonothèque. *E. Chiron, 1953-1976, La revue du son*(73) (cf. p. 18, 19).

Mushtaq, Z., Su, S.-F., & Tran, Q.-V. (2021). Spectral images based environmental sound classification using CNN with meaningful data augmentation. *Applied Acoustics*, *172*, 107581 (cf. p. 27).

Noack-Wilhelm, G. (2025). *Sound Morph Diffusion : Un outil de morphing vocal basé sur la synthèse par réseau de neurones* [Mémoire de master]. ENS Louis-Lumiere. (Cf. p. 109).

Pacquin, K. (2020). Metadata Style Guide (cf. p. 23).

Pan, S. J., & Yang, Q. (2010). A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, *22*(10), 1345-1359 (cf. p. 30, 62).

Pascal, J. (2023). Neural Networks: Unleashing the Power of Latent Space Compression. *Medium* (cf. p. 109).

Pashine, S., Dixit, R., & Kushwah, R. (2020). Handwritten Digit Recognition using Machine and Deep Learning Algorithms [arXiv:2106.12614 [cs]]. *International Journal of Computer Applications*, *176*(42), 27-33 (cf. p. 103).

Patterson, K., Nestor, P. J., & Rogers, T. T. (2007). Where do you know what you know? The representation of semantic knowledge in the human brain. *Nature Reviews Neuroscience*, *8*(12), 976-987 (cf. p. 44).

Piczak, K. J. (2015). ESC: Dataset for Environmental Sound Classification [Place: Brisbane, Australia]. *Proceedings of the 23rd Annual ACM Conference on Multimedia*, 1015-1018 (cf. p. 34, 57, 59).

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, *323*, 533-536 (cf. p. 118).

Saada, M. (2017). *Écoute et language* [thèse de doct., ENS Louis-Lumiere]. (Cf. p. 17).

Salamon, J., Jacoby, C., & Bello, J. (2014). A Dataset and Taxonomy for Urban Sound Research. *Proceedings - 22nd ACM International Conference on Multimedia* (cf. p. 28, 34, 57, 65, 66).

Sawhney, N., & Maes, P. (1997). Situational awareness from environmental sounds. *Project Rep. for Pattie Maes*, 1-7 (cf. p. 28).

Schaeffer, P. (1966). Traité des objets musicaux. In F. Wörner & M. Wald-Fuhrmann (éd.), *Lexikon Schriften über Musik: Band 2: Musikästhetik in Europa und Nordamerika* (p. 735-744). Bärenreiter-Verlag. (Cf. p. 17, 91, 93).

Schafer, R. M., & Gleize Sylvette. (1979). *Le paysage sonore / R. Murray Schafer ; traduit de l'anglais par Sylvette Gleize*. J.-C. Lattès. (Cf. p. 95, 96).

Serror, T. (2018). *Un Moteur De Recherche Sémantique Pour Sonothèques* [Mémoire de master]. ENS Louis-Lumiere. (Cf. p. 12).

Stowell, D., & Plumbley, M. D. (2013). An open dataset for research on audio field recording archives: freefield1010. (Cf. p. 34).

Thayer, R. E. (1989). *The Biopsychology of Mood and Arousal*. Oxford University Press USA. (Cf. p. 97).

Turing, A. M. (1950). I.—Computing Machinery And Intelligence. *Mind*, *59*(236), 433-460 (cf. p. 28).

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł. u., & Polosukhin, I. (2017). Attention is All you Need. *Advances in Neural Information Processing Systems*, *30* (cf. p. 40, 41).

Viardot, J. (2023). *La séparation de sources pour le remixage en son immersif de masters monophoniques des années 1950-60* [Mémoire de master]. (Cf. p. 100).

Virtanen, T., Plumbley, M. D., & Ellis, D. (éd.). (2018). *Computational Analysis of Sound Scenes and Events*. Springer International Publishing. (Cf. p. 30, 37, 49, 50, 54).

Whiteley, N., Gray, A., & Rubin-Delanchy, P. (2025, mars). Statistical exploration of the Manifold Hypothesis [arXiv:2208.11665 [stat]]. (Cf. p. 106).

Williams, A. H., Kim, T. H., Wang, F., Vyas, S., Ryu, S. I., Shenoy, K. V., Schnitzer, M., Kolda, T. G., & Ganguli, S. (2018). Unsupervised Discovery of Demixed, Low-Dimensional Neural Dynamics across Multiple Timescales through Tensor Component Analysis. *Neuron*, *98*(6), 1099-1115.e8 (cf. p. 106).

Xu, Y., Huang, Q., Wang, W., Foster, P., Sigtia, S., Jackson, P. J. B., & Plumbley, M. D. (2017). Unsupervised Feature Learning Based on Deep Models for Environmental Audio Tagging [Publisher: Institute of Electrical and Electronics Engineers (IEEE)]. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, *25*(6), 1230-1241 (cf. p. 52).

Zhou, Z.-H. (2012, juin). *Ensemble Methods: Foundations and Algorithms* (T. 14) [Journal Abbreviation: Ensemble Methods: Foundations and Algorithms Publication Title: Ensemble Methods: Foundations and Algorithms]. (Cf. p. 103).

Chapitre A

Annexes A

A.1 Les quatre écoutes, un conditionnement de la classification.

Rappelons dans un premier temps qu'il n'existe pas *une* classification, mais bien *des* classifications. En effet, certaines classifications peuvent être objectives et basées sur des critères physiques, d'autres subjectives, c'est le cas des classifications qui font appel aux émotions ou à la symbolique. Dans cette annexe nous parcourrerons les quatre types d'écoutes théorisées, et illustrerons comment ces écoutes engendrent des hiérarchies sonores différentes.

La taxonomie [1] est l'étude des règles qui régissent une classification. Pour ce qui est de classer les sons, le nombre de classifications possibles est très grand, infini peut-être. Encore faut-il pouvoir juger de leur pertinence, pour cela nous avons besoin d'outils d'analyse.

Cette taxonomie dépend nécessairement de la manière dont le son est écouté, Pierre Schaeffer (1966) puis Michel Chion (1995) distinguent trois écoutes actives.

Schaeffer ajoute à ces trois écoutes actives, une écoute passive, *"Ouïr"*, Michel Chion (2021), précise une sous-partie du procédé d'ouïr qu'il nomme *l'écoute affective* pour qualifier une écoute émotionnelle et subjective.

Dans la suite de ce mémoire, nous utiliserons la terminologie de Chion car elle nous paraît plus claire.

Les classifications de sons ne peuvent exister qu'au sein d'un paradigme prédéfini, c'est-à-dire un ensemble d'axiomes permettant une classification logique et justifiée. Le type d'écoute dans lequel on se place est de ce fait nécessairement antérieur au choix d'une classification. Les quatre écoutes développées précédemment,

1. cf. Glossaire D.1

Schaeffer	Chion	Définition
Entendre	**Écoute réduite**	Considérer le son tel qu'il est, de manière objective, description basée sur des critères physiques/acoustiques.
Écouter	**Écoute causale**	Identifier l'objet qui est la cause de ce son.
Comprendre	**Écoute sémantique ou Écoute codale**	Interpréter le son comme porteur de sens, le message est décodé (e.g. la parole, un message en morse, etc.).

TABLEAU A.1 – Équivalence des types d'écoutes selon Schaeffer et Chion.

nous permettent de construire quatre familles taxinomiques, nous les détaillons dans les paragraphes suivants.

On définira le terme d'Ontologie[2] de la manière suivante : *un ensemble de concepts et de catégories dans un domaine, qui montre leurs propriétés et les relations qu'ils entretiennent entre eux.*

Bien qu'en français le terme ontologie renvoie essentiellement à une doctrine philosophique, il est présent dans la littérature scientifique comme équivalent du terme anglais "ontology", nous l'utiliserons en ce sens.

A.1.1 L'Écoute réduite : une classification basée sur les caractéristiques physiques des signaux.

Un des premiers systèmes de classification des sons auquel on est tenté de penser est le solfège. Les différentes notes utilisées en musique, correspondent aux noms donnés aux sons en fonction de leur fréquence fondamentale. La musique occidentale décompose les fréquences en octaves et les octaves en 11 demi-tons permettent la définition des 12 notes. De même, la musique occidentale est dotée d'un formidable outil à classifier les durées, la notion de rythme. Sur une partition, les différents formes des notes caractérisent la durée de ces dernières tandis que leur hauteur sur la portée renvoie à la notion de fréquence.

Seulement, même si ce système de classification s'est avéré très efficace pour la tâche qu'on lui a confié (lire et écrire la musique) on en voit assez vite les limites. D'abord, il ne permet pas de qualifier tous les sons ; les sons qui n'ont pas de hauteur tonale, par exemple, ne peuvent pas être représentés par une note comme Si ou Mi. De plus, il est peu précis ; s'il est capable de discriminer deux sons de hauteur différente, il est en revanche incapable de différencier deux sons de même hauteur

2. Traduction de l'anglais "Ontology" : a set of concepts and categories in a subject area or domain that shows their properties and the relations between them. [Oxford Language]

mais ayant un timbre [3] différent. Il est impossible de différencier, avec ce système, un trompettiste jouant un son cuivré (riche en harmoniques aiguës) d'un autre qui préférera jouer la même partition avec un son plus feutré (moins riche en harmoniques aiguës).

Pour cette raison, les partitions ont commencées à être annotées de mots désignant des intentions de jeu ou des adjectifs, le plus souvent en italien, cf. 26.

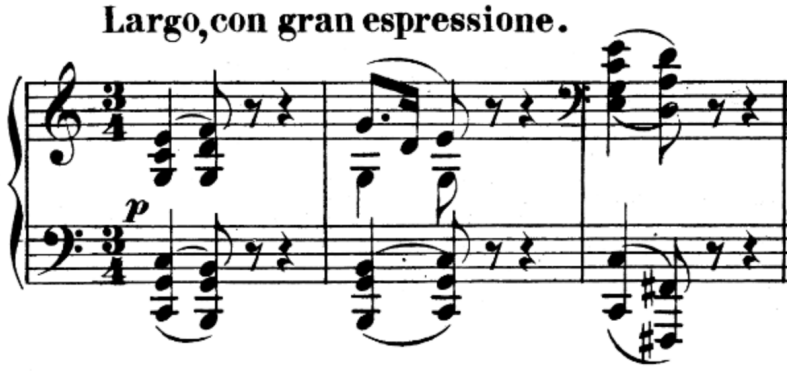


FIGURE 26 – Indication de jeu *"Lent, avec expression"* à destination de l'interprète, *L. W. Beethoven* "Sonate pour piano n°4", *deuxième mouvement.*

La musique classique est libre d'interprétation, et c'est d'ailleurs le rôle du chef d'orchestre de proposer et de transmettre sa vision à ses musiciens. Cependant, dans la deuxième moitié du XXème siècle, avec l'arrivée du numérique, la musique contemporaine et les nouveaux moyens de création digitalisés, il devient nécessaire de créer une classification précise objective et complète des sons musicaux. C'est la tâche à laquelle s'attelle Pierre SCHAEFFER (1966) dans le chapitre *Morphologie et Typologie des objets sonores* de son *Traité des objets musicaux* paru en 1966.

Avant tout développement, il semble nécessaire d'introduire quelques définitions.
Morphologie : La Morphologie est l'étude de la forme d'un son, l'idée consiste à trouver un ensemble de caractéristiques descriptives, un glossaire permettant à deux individus de se comprendre lorsqu'ils échangent au sujet d'un son.
Typologie : La typologie est l'étude des types, cette discipline vise à définir un certain nombre de cases dans le but de permettre une classification.

> *" La morphologie tend a une qualification du sonore, tandis que la typologie répond à une nécessité d'identification des objets sonores ".*
> - Pierre SCHAEFFER (1966) -

On comprend vite qu'il est impossible d'établir une typologie sans l'existence de

3. La notion de timbre est complexe mais nous pouvons dans un premier temps la résumer à la répartition des composantes fréquentielles, définissant le spectre de la note considérée.

critères permettant la discrimination. En d'autres termes, le choix d'une morphologie influencera forcément la typologie qui en découle.

La parabole du grenier :

Pierre Schaeffer compare les sons à un grenier plein d'objets mal rangés. Pour s'attaquer à cette tâche de mise en ordre, on est d'abord tenté de les ranger selon des critères physiques ; *" le simple et le mesurable se présentent aussitôt "*. Ainsi ranger un tas de planches par taille croissante, ou des fioles en fonction de leur capacité fait sens. Mais l'auteur nous interroge alors sur l'impossibilité de mettre en relation ces planches, ces vieux bocaux et l'oiseau empaillé qui traîne aussi là. Mesurer la longueur ou le volume de l'oiseau pour le ranger dans l'une ou l'autre des deux catégories ne paraît pas idéal ; *" On voit que la physique ne m'est d'aucun secours "*. Dans sa typologie, l'auteur propose de classifier les sons par rapport à leur emploi. Il n'oublie pas de nous rappeler que toute typologie est forcément orientée, la sienne sera celle d'une typologie adaptée aux sons musicaux ; *" La recherche d'un typologie absolue est illusoire "*.

On entend par caractéristique physique du signal les données qui peuvent être mesurées de manière objective. Des exemples de caractéristiques physiques des signaux sont le niveau sonore, la fréquence instantanée, le barycentre spectral, la durée et l'enveloppe temporelle, entre autres. Pierre Schaeffer définit **l'écoute réduite** que Michel Chion résume en ces termes :

> *" C'est l'écoute qui fait volontairement et artificiellement abstraction de la cause et du sens, pour s'intéresser au son considéré pour lui-même, dans ses qualités sensibles non seulement de hauteur et de rythme, mais aussi de grain, matière, forme, masse et volume. "*
>
> - Michel Chion (2010) -

Schaeffer identifie plusieurs caractéristiques servant à organiser les sons.

1. *Durée* : *temps perçu* d'un objet sonore, contrairement au temps chronométrique qui est absolu et quantifiable. Il considère 3 durées. Courtes (trop courtes) ; moyennes (idéales) ; étendues (trop longues).

2. Entretien : l'entretien d'un son qualifie la partie maintien (*sustain*) de son enveloppe ADSR [4] : Il introduit 3 possibilités.

   - *Impulsion :* L'entretien du son est nul, c'est le cas pour un son de coup de fouet.

4. Attack decay sustain release D.1

- *Entretien passif :* l'environnement prolonge et colore le son de manière déterminée après l'attaque, *e.g.* un piano ou une guitare dont la caisse de résonance entretient le son.
- *Entretien actif :* le son est entretenu par un appoint renouvelé en énergie, cela correspond au cas des cordes frottées ou des instruments à vent.

3. Facture : c'est une qualification de l'entretien, à comprendre la manière dont un son est entretenu, on parle de facture continue pour les instruments à vents et de facture itérative pour un *staccato* de violon par exemple. Un son impulsif comme un coup de cymbale bénéficie d'une facture nulle.

4. Variation : c'est la quantité de *"ce qui bouge en fonction du temps"*, de manière analogue à la durée, Pierre Schaeffer considère qu'il ne faut pas qu'un son varie trop, sinon il est perçu comme aléatoire donc comme un bruit, tout en étant un minimum variant pour ne pas être monotone.

Lorsqu'on écoute un son, la notion de contexte est importante, pour illustrer ce propos, prenons l'exemple d'un son quelconque tel qu'un grincement de chaise sur le sol. L'acoustique dans laquelle le son sera écouté va grandement jouer sur les caractéristiques physiques du son. Selon que le son est écouté dans une chambre ou dans une église, les caractéristiques comme la durée, le barycentre spectral vont varier.

A.1.2 L'Écoute affective : une classification basée sur le ressenti émotionnel.

Dans son analyse des travaux de Pierre Schaeffer, Chion (1995) souligne qu'il faut séparer *l'objet sonore*, c'est-à-dire la manière dont un son est perçu, du signal physique qui lui est associé. Le signal physique, ce sont les caractéristiques physiques objectives de ce signal, tandis que sa perception est subjective. Dans la suite de ce mémoire, nous utiliserons le terme "d'objet sonore" pour parler de la perception de celui-ci.

Nous prendrons ici appui sur les travaux de Raymond Murray Schafer et Gleize Sylvette (1979), l'auteur consacre un chapitre de son livre à la classification des sons. Il y développe l'idée d'une ontologie basée sur des critères esthétiques. Cette théorie vient de la nécessité des urbanistes d'améliorer le confort des citadins en proposant des paysages sonores agréables. Schafer s'interroge alors sur la possibilité de créer un système de mesure universel de réactions esthétiques aux sons. Réduite à sa plus simple expression, l'esthétique distingue le laid du beau, ce qui conduit à l'idée de classer les sons par ordre de préférence dans un environnement sonore. Toutefois il se heurte rapidement à la réalité.

> *“ Les sons affectent différemment chaque individu et provoquent parfois des réactions si diverses [...] que cette approche a été jugée trop subjective pour que les résultats soient significatifs ”*
>
> - R. M. Schafer et Gleize Sylvette (1979) -

Cette thèse est confirmée par les résultats du *Rapport international des préférences sonores*[5] dans lequel il est montré que la perception des sons dépend de trop de facteurs tels que le contexte culturel. Pour prendre un exemple, datant des années 70, les pays ayant une technologie *"avancée"*[6] comme l'Europe ou l'Amérique du Nord ne consideraient pas les bruits d'animaux comme agaçants, contrairement à la Jamaïque. De manière analogue, les européens ont tendance à classer les bruits de machines, circulations, transports dans les sons désagréables ; là où les jamaïcains les considèraient comme des sons neutres. Les américains ont eu le temps d'oublier le hululement de la chouette, ce son est sorti de leur habitude et leur paraît extraordinaire, ils le considèrent comme agréable. Ce que montre, en revanche, le *rapport international des préférences sonores*, c'est qu'un son qui est souvent entendu contre notre volonté, est plus susceptible d'être classé comme un son désagréable. De même, un son auquel nous sommes rarement exposés, sera plus facilement jugé agréable car constituant une expérience extraordinaire.

Le caractère agréable d'un son dépend aussi du contexte spatio-temporel dans lequel il est entendu. On peut se baser sur l'exemple du *Moulin à café de Schafer*. Quand ce dernier est écouté sur bande, le son est perçu comme effrayant ou menaçant, mais remis dans son contexte (8 h de matin dans un appartement), les réactions à son égard se tempèrent.

Cas des algorithmes de streaming musical

Avec la disparition progressive des supports physiques et l'essor du numérique, l'écoute musicale s'est largement dématérialisée, favorisant ainsi l'essor des plateformes de streaming. Ces dernières sont désormais le principal mode de consommation musicale. Elles intègrent toutes des algorithmes de classification permettant de trier les morceaux selon différents critères. La catégorisation par genre musical (rock, jazz, etc.) repose sur des paramètres objectifs tels que le tempo, les motifs rythmiques ou encore la formation instrumentale. Par ailleurs, certaines plateformes proposent également un tri basé sur l'humeur (*mood*) du morceau[7].

À l'instar de la classification par genres, l'identification des morceaux selon leur *mood* représente un enjeu majeur pour les plateformes telles que Deezer ou Spotify.

5. Annexe II de (Schafer & Gleize Sylvette, 1979)
6. Raymond Muray Schafer
7. Le *mood* d'un morceau, traduit en français par “humeur”, désigne l'émotion suscitée à l'écoute de la musique.

L'analyse des humeurs musicales s'appuie sur le modèle établi par le psychologue Robert Thayer, qui ordonne les *moods* selon deux axes : le premier associé au niveau de stress (variant de l'absence de stress à un état anxieux) et le deuxième à l'énergie (allant du calme à l'excitation) (Thayer, 1989).

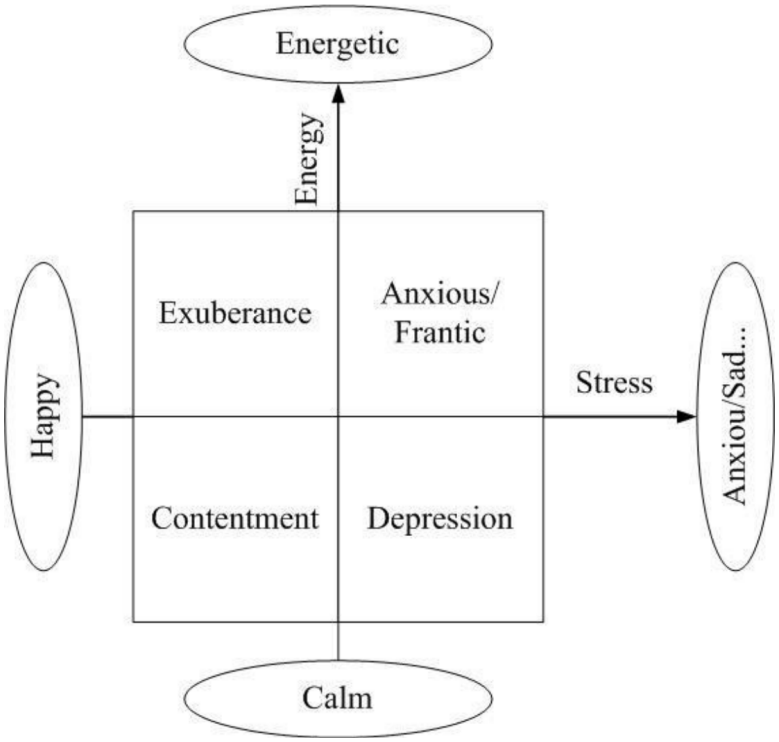


Figure 27 – Modèle de Thayer pour la classification des musiques en fonction des humeurs (Extrait de Bhat *et al.* (2014)).

L'étude de Bhat *et al.* (2014) propose une méthode de classification des émotions musicales à partir de caractéristiques acoustiques. Elle identifie quatre grandes catégories de paramètres, dites *caractéristiques supra-segmentaires*, qui sont l'intensité, la couleur, la hauteur et le rythme. Chacune de ces catégories repose sur des descripteurs plus spécifiques appelés *caractéristiques segmentaires*.

L'étude établit une correspondance entre ces critères acoustiques et les *moods* perçus par l'auditeur (cf. fig 28).

Ces données permettent ainsi d'établir un lien entre des *moods* perçus (des informations de haut niveau, porteuses de sens pour l'être humain mais difficilement mesurables directement) et des caractéristiques acoustiques du signal (qui relèvent d'un niveau bas et sont directement exploitables par les algorithmes de classification).

A.1.3 Écoute sémantique ; Classification basée sur le sens.

La parole étant un signal sonore porteur de sens, de nombreuses recherches, s'attellent depuis 1980 au décodage automatique du signal parlé. Ce champ de re-

Mood	Intensity	Timbre	Pitch	Rhythm
Happy	Medium	Medium	Very High	Very High
Exuberent	High	Medium	High	High
Energetic	Very High	Medium	Medium	High
Frantic	High	Very High	Low	Very High
Sad	Medium	Very Low	Very Low	Low
Deppression	Low	Low	Low	Low
Calm	Very Low	Very Low	Medium	Very Low
Contentment	Low	Low	High	Low

FIGURE 28 – Tableau représentant la corrélation entre les caractéristiques acoustiques des signaux et l'humeur perçue de ce son (BHAT *et al.*, 2014)

cherche est appelé reconnaissance automatique de la parole, en anglais *Automatic Speech Recognition* (ASR).

Les algorithmes discours vers texte, (speech to text)

Quand on s'intéresse au langage il est important de bien séparer le *lexique*[8] et la *sémantique*[9]. Si le lexique regroupe les mots en tant qu'assemblage de phonèmes - entendus dans une écoute réduite - ceux-ci, (les mots), n'acquièrent un sens qu'après décodage, c'est-à-dire au travers d'un dictionnaire sémantique forgé par l'expérience.

Les algorithmes *speech to text* ont d'abord été développés pour les traducteurs en temps réel. Le principe consiste à transcrire un discours en un texte de manière instantanée. Notons qu'il ne s'agit pour l'instant que d'une considération lexicale, d'un ensemble de mots parlés vers un ensemble de mots écrits. Cependant, une bonne traduction n'est pas une traduction mot-à-mot, évidemment le sens des mots dans un phrase doit au maximum être conservé. Il est important de considérer la polysémie de certains mots, *e.g.* pêcher (le poisson) ou pêcher (aller à l'encontre des valeurs chrétiennes).

En fait l'écoute sémantique ne trouve pas réellement d'application concrète dans l'apprentissage automatique. On préférera de manière systématique transcrire la voix en texte et c'est sur celui-ci que sera faite l'étude sémantique. La raison étant qu'il existe beaucoup plus de ressources textuelles que de ressources audio sur internet pour entraîner un algorithme, de même le texte est une manière très compressée de stocker de l'information sémantique, le support textuel prend beaucoup moins de place que le support audio (même s'il est compressé). Il n'est pas rare de constater qu'un livre de trois cent pages puisse prendre le même espace sur un disque dur qu'un fichier mp3 de quelques minutes. En fait le texte est même la forme de donnée la plus compressée possible compréhensible par un humain.

8. cf. Glossaire D.1
9. cf. Glossaire D.1

A.1.4 L'Écoute causale : une classification par sources acoustiques.

L'écoute causale est de loin notre mode d'écoute privilégié, il rappelle à la fonction primale de l'ouïe, signaler un danger et donc identifier la cause de ce danger potentiel. De manière générale, on désigne par écoute causale le procédé - conscient ou non - visant à identifier l'objet source produisant un son.

La séparation de sources instrumentales en musique

Les problématiques liées au signal musical sont nombreuses, les disciplines peuvent être séparées en deux catégories majeures : la séparation de sources, qui vise à extraire d'une mixture sonore un ou plusieurs *stems*[10] musicaux, et la classification. Cette dernière peut se faire à différentes échelles, il peut venir à l'idée de classer les morceaux par tempo, par genre musical ou par *mood* comme développé précédemment.

L'objectif de la MSS *(music source separation)* est d'extraire des sources instrumentales appelées *target sources* d'une mixture composée d'un ensemble de sources. Il s'agit donc ici non pas d'un problème de *tagging*[11] mais plutôt de *clusterization*[12]. Dans ce contexte la machine n'a pas nécessairement besoin de savoir à quel instrument correspond la source, mais peut se contenter de la séparer aussi bien que possible du reste du signal.

De manière évidente, on considère la mixture comme la somme des sources qui la composent.

$$x(t) = \sum_{i=0}^{N} s_i(t)$$

Les sources musicales se divisent en 3 types (Cano *et al.*, 2019). Il y a d'abord les sources harmoniques dont les composantes fréquentielles ont une fréquence égale à un multiple entier de la fréquence fondamentale. Ensuite, il y a les sources percussives, qui possèdent beaucoup d'énergie sur l'ensemble du spectre mais qui ne durent pas dans le temps. Les percussions sont caractérisées par des sons non entretenus et peu tonaux. Enfin, il y a la voix, qui pourrait être considérée comme une source harmonique pour les voyelles, mais qui, pour les plosives, se rapproche plus de la percussion. Et, pour les fricatives, c'est-à-dire des phonèmes comme le /sh/ (/ʃ/), elle s'apparente a un son entretenu mais non tonal, comparable à un bruit blanc.

Même si la problématique de la séparation de sources n'est pas à proprement parler un problème de classification, il est pertinent une fois la séparation effectuée

10. cf. Glossaire D.1
11. cf. glossaire D.1
12. Idem

de classer les instruments individuels en fonction de leur timbre. On parle alors d'un problème de reconnaissance de timbre. Nous renvoyons ici à des travaux comme celui de Jean Viardot (2023) qui fait une synthèse sur le sujet.

Chapitre B

Annexes B

B.1 Architectures des réseaux de neurones.

A Les modèles type perceptron.

Le neurone artificiel.

Les neurones artificiels sont basés sur le fonctionnement des neurones biologiques. Dans le cerveau des êtres vivants, les signaux électriques transitent via les *dendrites* qui relient les différents neurones entre eux. Au cours de notre vie, ces liaisons se renforcent lorsqu'elles sont beaucoup utilisées tandis que celles qui ne sont pas entretenues finissent par disparaître. Lorsqu'un neurone présente une activité électrique et transmet un signal à ses voisins, on dit qu'il est actif. Un neurone artificiel (ou unité logique) fonctionne de la même manière : c'est une fonction mathématique possédant plusieurs entrées et une sortie. Concrètement un neurone effectue une combinaison linéaire de ses entrées, composée par une *fonction d'activation* non linéaire. Cette fonction d'activation permet l'apprentissage de phénomènes non linéaires, qui, en réalité, représentent la vaste majorité des phénomènes rencontrés dans la nature. Chaque entrée est pondérée par un poids, on note $\mathbf{E} = (E_1, E_2, \ldots, E_n)$ le vecteur des entrées et $\mathbf{w} = (w_1, w_2, \ldots, w_n)$ celui des poids de pondération des synapses. On note S_j la sortie du neurone j.

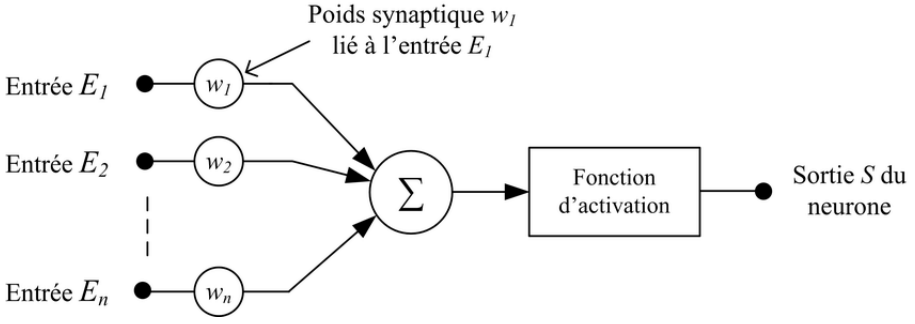
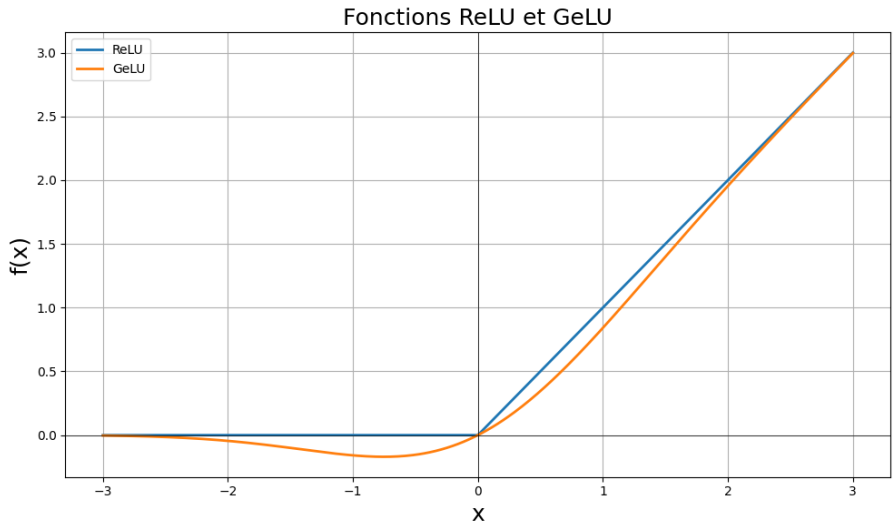


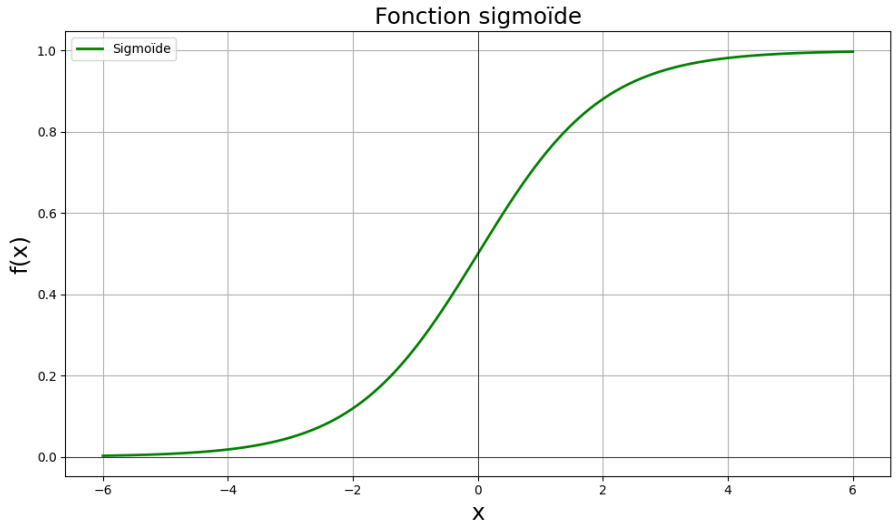
FIGURE 29 – Un neurone artificiel (LARRAS, 2015)).

$$S_j(\mathbf{E}, \mathbf{w}) = \varphi\Big(\sum_{i=1}^{n} w_i E_i\Big) \tag{B.1}$$

Avec φ la fonction d'activation. Le plus couramment, il s'agit de fonctions sigmoïdes[1], ReLU, ou GeLU (version dérivable de ReLU), cf. figure 30.



(a) Fonctions ReLU et GeLU sur l'intervalle $[-3, 3]$.



(b) Fonction sigmoïde sur l'intervalle $[-6, 6]$.

FIGURE 30 – Les principales fonctions d'activation.

La fonction ReLU est nulle pour tout $x < 0$ et est définie par $y = x$ pour $x \geq 0$. Si la sortie du neurone est négative, sa contribution est ramenée à 0, si elle est positive alors le neurone est actif.

Dans les faits, en plus des poids associés à leurs entrées, les neurones possèdent un biais, une variable indépendante des neurones précédents. Ce biais représente l'idée

1. la fonction sigmoïde vérifie $\sigma = \frac{1}{1+e^{-x}}$

que toutes les unités logiques n'ont pas la même contribution, certaines s'activent "plus facilement" que d'autres. Le biais b_j du neurone j peut-être vu comme la valeur numérique du seuil qu'il est nécessaire de franchir pour que le neurone s'active. En reprenant l'expression B.1, il vient naturellement :

$$S_j(\mathbf{E}, \mathbf{w}) = \varphi\Big(\underbrace{\sum_{i=1}^{n} w_i E_i + b_j}_{\text{activation si } \geq 0} \Big) \tag{B.2}$$

Ce neurone artificiel très fréquemment utilisé est le modèle McCulloch-Pitts (*M-P model)* (Zhou, 2012).

Un réseau de neurones est organisé en plusieurs couches, d'abord une couche d'entrée qui reçoit les données. Ensuite, viennent une ou plusieurs couches cachées ; on parle de réseau de neurones profond (*deep neural network* - DNN) lorsqu'il y a plus d'une couche cachée. Enfin, vient la couche de sortie, cette dernière contient autant de neurones que de sorties possibles. Pour résoudre un problème binaire, un neurone suffit, s'il s'active la réponse est "oui". Pour les tâches de classification, on a en sortie un neurone par classe. Par exemple, pour la reconnaissance de chiffres écrits à la main, le réseau possède en sortie dix neurones, pour les dix chiffres arabes. (Pashine *et al.*, 2020)

Fonction softmax.

Dans un problème de classification, afin de pouvoir comparer les différentes valeurs des neurones de la couche de sortie, il est pertinent de les normaliser. Une manière standard d'effectuer cette normalisation consiste à calculer la probabilité d'appartenance d'un élément à chaque classe.

La fonction SoftMax est la généralisation à plusieurs variables des fonctions logistiques (dont fait partie la fonction sigmoïde). Son intérêt est de **convertir un vecteur de K nombres réels en une distribution de probabilité sur K choix.** En d'autres termes, cette fonction sert à transformer les amplitudes individuelles, en probabilités.

Considérons un vecteur $\mathbf{z} = (z_1, z_2, \ldots, z_k)$, alors le vecteur de sortie de la fonction SoftMax est défini par $\sigma = (\sigma_1, \sigma_2, \ldots \sigma_k)$, où chacune des composantes σ_j est décrite par la formule suivante :

$$\sigma(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$$

On normalise par rapport à la somme des exponentielles des composantes de $\mathbf{z}$. De

ce fait, la somme des σ_j donne 1.

Réseaux de neurones denses.

Les réseaux de neurones denses (*Dense Neural Network - DNN*) sont des réseaux dont chacune des couches est très connectée à la couche suivante. En fait, chacun des neurones de la couche n est connecté à l'ensemble des neurones de la couche $n+1$, évidemment, par récurrence cela vaut aussi pour les couches suivantes, jusqu'à la sortie du réseau. On dit parfois que ces couches sont entièrement connectées entre elles *(fully connected layers)*.

Apprentissage profond (Deep learning).

Les algorithmes de *deep learning* reposent sur une multitude de niveaux plutôt que sur une seule couche essayant de trouver des corrélations entre les entrées et les prédictions, on rencontre le terme de *multilayer perceptron*. Si l'approche *deep learning* est connue depuis longtemps, elle n'était jusqu'autour de 2010 que peu applicable dans un contexte réel. En effet, la puissance de calcul nécessaire au fonctionnement de plusieurs couches de neurones était bien trop élevée pour les machines. En 2012 les travaux admirables d'Alex Krizhevsky *et al.* (2012) sur la reconnaissance d'image avec un réseau profond à convolution font l'effet d'une bombe.

B Réseaux de neurones à convolution.

Les réseaux de neurones à convolution sont eux aussi organisés en couches successives. Ce type d'architecture est particulièrement utilisée dans le traitement d'images, notamment la reconnaissance de formes. En traitement du signal sonore, ce type de réseau est fréquemment utilisé aussi. En fait, la représentation la plus utilisée pour les signaux sonores en apprentissage automatique est leur représentation temps-fréquence, c'est-à-dire une image à deux dimensions. Des techniques de reconnaissances de formes sont ensuite appliquées sur cette image afin d'extraire des informations sur l'audio.

Caractéristiques bas niveau, caractéristiques haut niveau.

Tout objet peut-être décrit par des caractéristiques qui lui sont propres. On appelle *caractéristiques bas niveau* les descriptions à basse échelle. Dans le cas de l'apprentissage automatique, les données sont numérisées, les caractéristiques bas niveau décrivent de manière objective et très précise la donnée. Il s'agit d'une description de l'ordre du bit de donnée ou de l'octet. Pour un fichier au format *WAV*, une caractéristique bas niveau serait l'amplitude instantanée de chaque échantillon. Pour une image il s'agirait de la valeur numérique codant la proportion de chacune

des composantes rouge, verte et bleu de chaque pixel. Ces descriptions bas-niveau sont des données lourdes, non compressées, ce qui leur vaut l'appellation *"raw datas"* dans la littérature scientifique.

Les caractéristiques bas niveau s'opposent aux *caractéristiques haut niveau*, ces dernières sont des caractéristiques plus globales de l'objet, à moyenne ou grande échelle. Pour le même fichier au format *WAV*, on ne considère plus les amplitudes de chaque échantillons, mais son évolution temporelle globale, par exemple via une description ADSR [2]. Pour l'image, les caractéristiques haut niveau sont des formes, ou des textures.

Ces données sont nécessairement compressées et ordonnées, elles sont plus proches de la manière dont un humain analyse son environnement. En effet, notre perception est macroscopique, il évident que nous percevons l'objet "chaise" plus que comme une succession d'atomes. La perception de l'objet "chaise" est multidimensionnelle, sa forme, sa taille, le matériau duquel elle est constituée, etc, sont autant de dimensions sur lesquelles on peut placer cet objet. Cependant, ces seules descriptions liées aux caractéristiques physiques ne suffisent pas à décrire l'objet "chaise" pleinement, l'humain attache de l'importance à la *sémantique*. *"À quoi sert une chaise ? - À s'asseoir dessus"*. Aussi, la *fonction* d'un objet est un autre exemple de descripteur haut-niveau. Il nécessite une analyse, un travail mental basé sur la perception sensorielle que nous avons de cet objet, tout en faisant intervenir la mémoire, au travers des expériences vécues.

Hypothèse de la variété.

Les réseaux de neurones à convolution répondent à un problème concret, celui de la *très grande dimension*. Les données multimédias sont denses, pour une image carrée en niveau de gris, de 28 pixels de côtés, cela représente $28 \times 28 = 784$ éléments. Deux secondes de son échantillonnée à 48 kHz correspondent à 96000 éléments. Or, les ensembles de caractéristiques trop complexes ralentissent l'apprentissage de l'algorithme et rendent difficile la recherche d'optima globaux (Cayton, 2005).

> *" Chaque exemple d'un dataset est constitué de milliers de caractéristiques, mais il peut être décrit comme une fonction de quelques paramètres sous-jacents seulement. En d'autres termes, les exemples sont en fait des points sur une sous-variété à faible dimension qui est contenue dans un espace à haute dimension. "* [3]
>
> - Lawrence Cayton (2005) -

L'hypothèse de la variété, en anglais *manifold hypothesis* est un principe largement

2. Attack decay sustain release, cf. Glossaire, D.1
3. Traduction libre

accepté de l'apprentissage automatique qui affirme que les données de grande dimension peuvent être compressées sur une sous-variété de basse dimension, intégrée dans un espace de haute dimension (Whiteley *et al.*, 2025), c'est-à-dire passer d'une *description bas niveau* à une *description haut niveau*.

La perception et la mémoire humaine sont d'ailleurs basées sur ce même principe : quand quelqu'un regarde une image, il ne voit pas les pixels qui la constituent de manière individuelle, il perçoit plutôt des groupes de pixels, des formes, des contours, etc. Ainsi, dans notre cerveau, un objet peut-être représenté dans un sous-espace de dimension moindre que celui des caractéristiques de l'objet (Chaudhuri & Fiete, 2016) (Williams *et al.*, 2018). C'est en fait un problème de compression, la question étant de savoir quelles sont les caractéristiques pertinentes pour représenter un objet. Dans le formalisme mathématique, une perception $\mathbf{P}$ est donc un vecteur dans un espace $\mathcal{E}$ de dimension d. Le but du jeu consiste à réduire la dimension de l'entrée en caractéristiques globales qui conservent l'essence de l'objet dans une représentation plus légère. Dans le cas d'une ACP, cela consiste à mettre à 0 toutes les dimensions qui présentent des faibles variances. On obtient ainsi un vecteur $\hat{\mathbf{P}}$ de dimension d dont la plupart des coordonnées ont été mises à 0. On peut donc ensuite représenter ce vecteur dans un sous-espace $\mathcal{M}$ de dimension $m < d$ qui ne possède que les dimensions des composantes non nulles de $\hat{\mathbf{P}}$. Une compression de données à été effectuée permettant de restreindre la perception $\mathbf{P}$ à ses composantes principales $\hat{\mathbf{P}}$ (Cayton, 2005).

En d'autres termes, le principe consiste à simuler une analyse de caractéristiques globales, c'est-à-dire qu'on va parcourir l'image avec l'idée d'analyser des blocs de pixels dans le but d'identifier des objets. Ensuite, chaque groupe d'objets pourra signifier un méta-objet, (par exemple, des pattes + des grands yeux + des moustaches + des oreilles = chat).

Noyau convolutif

Un noyau *(kernel)* ou filtre convolutif désigne une matrice de transformation, et, il en existe plusieurs pouvant servir à différentes choses : détection de contours, augmentation du contraste, floutage et d'autres. La représentation temps-fréquence d'un son est typiquement une image en niveau de gris, c'est-à-dire une matrice en 2 dimensions, les axes x et y représentent la position du pixel dans l'image, la valeur de chaque case correspond à la luminance du pixel (0 vaut pour une absence de luminosité et 255 vaut pour une luminosité maximale). [4]

Les couches du *convolutional neural network* CNN sont constituées de ces noyaux de convolution. Chaque neurone est une matrice visant à extraire certaines des

4. Pour une image en couleur il faut ajouter une troisième dimension à la matrice, généralement on a une profondeur de 3 correspondant aux 3 canaux R,V,B.

caractéristiques de l'image à moyenne et grande échelle. Les valeurs de chaque noyau de convolution sont mises à jour au cours de la phase d'apprentissage du modèle. Ce genre de réseau est capable d'extraire des informations de textures, de formes, ainsi que la notion d'objets comme celle de chat ou de visage.

Les couches de pooling.

Les couches de convolution sont suivies de couches appelées *pooling*. Bien que "mise en commun" ou "couches d'union" soient des traductions possibles, ces termes ne sont pas employés dans la littérature. Ces couches de *pooling* contiennent des filtres qui effectuent une opération de maximum *(MaxPooling)* ou de moyenne *(average pooling)* sur les données qu'elles reçoivent. Un exemple concret est illustré en figure 31. Ces couches ont vocation à réduire la taille des matrices entrantes, ce qui

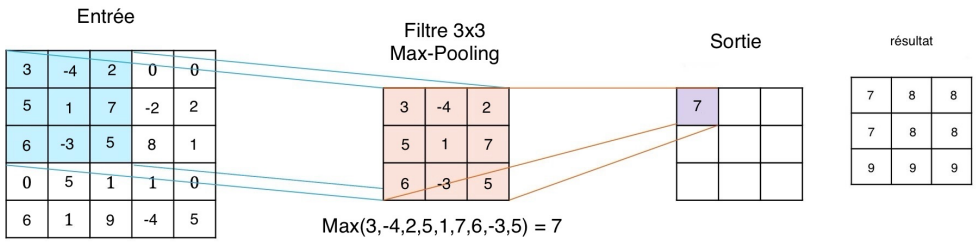


FIGURE 31 – Application d'un filtre *MaxPooling* sur une matrice 5×5 Image de f. 1703 (s. d.)

limite les opérations à effectuer pour les couches suivantes. Pour le cas de la illustré on passe d'une matrice de 25 éléments à une matrice 9 éléments. De plus, les couches de pooling apportent de la robustesse en réduisant l'impact de petites déformations sur l'image d'entrée. Enfin, en réduisant le nombre de paramètres libres, elles protègent l'algorithme contre la sur-interprétation et le rendent moins sensible au bruit contenu dans les caractéristiques bas-niveau des fichiers.

Le son n'est pas une image

Historiquement, on a d'abord traité les problèmes audio en se basant sur des représentations graphiques pour les donner à des réseaux de neurones convolutionnels. Mais en fait une représentation temps-fréquence n'est pas une image naturelle ! En effet dans les images naturelles, les 2 axes représentent la même grandeur : une longueur. On ne peut évidemment pas en dire autant d'un représentation temps-fréquence. Pour les images naturelles, un objet dans une image est cantonné à une zone de celle-ci, les pixels sont semblables de proche en proche car il y a une cohérence de grandeur physique dans les 2 axes.

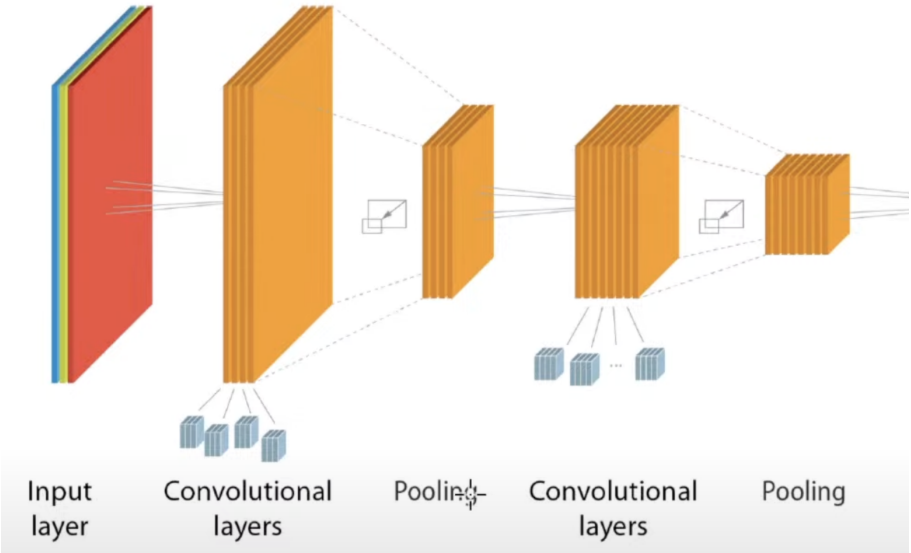


FIGURE 32 – Architecture d'un réseau de réseau de neurone à convolution.

Dans le cas d'une représentation temps-fréquence, deux pixels proches qui semblent appartenir au même objet graphique peuvent appartenir à deux sources différentes, par exemple une flûte et un violon jouant la même note posséderont les mêmes harmoniques mais un timbre différent. Par ailleurs, il y a une *invariance de la source acoustique* selon l'axe du temps. Le son issu d'une cymbale frappée à deux instants différents provient de la même source. En revanche ce n'est pas du tout le cas sur l'axe des fréquences. Une cymbale "pitchée" vers le grave ne ressemble plus à une cymbale, si bien qu'on aura du mal à considérer les deux sons comme provenant de la même source.

Pour adapter les CNN à l'analyse de représentations temps-fréquence, on a développé des filtres s'étendant sur toutes les fréquences et réalisant une convolution 1-D sur l'axe temporel uniquement, nous détaillerons cela plus tard.

Concept d'espace latent

La donnée voyage donc d'une couche vers les suivantes jusqu'à être "décomposée" en différentes *caractéristiques haut niveau*. Le réseau à convolution vise à représenter les entrées selon des concepts plus ou moins abstraits dans un *espace latent*, on appelle cette étape le plongement (*embedding*). La représentation de la figure 33 est une réduction à 2 dimensions d'un espace latent à 10 dimensions. La tâche du réseau de neurones décrit ici, est de discriminer et classifier des écritures manuscrites des dix chiffres arabes. On remarque que cet algorithme permet une bonne *clusterisation* des données dans l'espace des caractéristiques. De ce fait l'algorithme est capable de trouver l'hyperplan séparateur correspond à la frontière entre chaque classe de chiffres, c'est la généralisation à plus grande dimension de la figure 2.

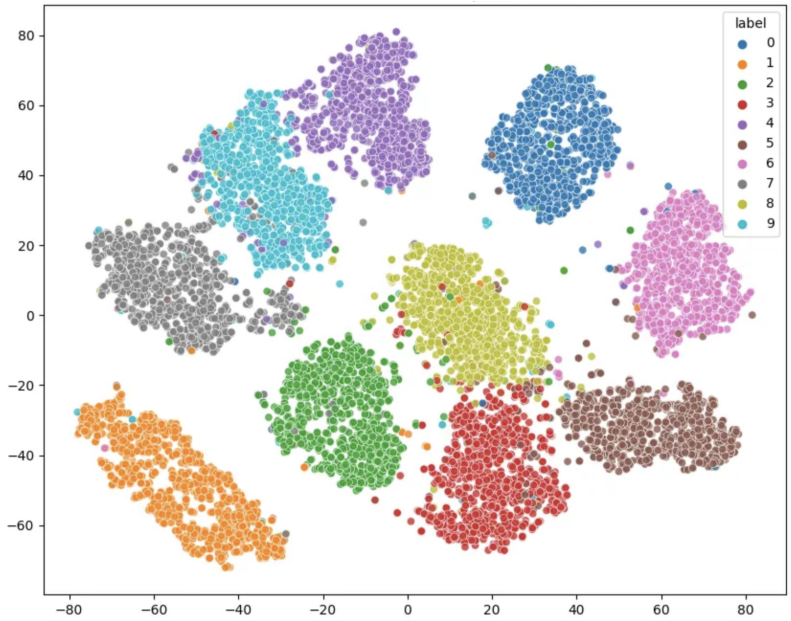


FIGURE 33 – Réduction à 2-D de l'espace latent d'un réseau de convolution visant à classifier les chiffres écrits à la main (PASCAL, 2023).

En conclusion, nous avons décrit ici les deux principales architectures de réseaux de neurones, ces blocs peuvent être assemblés pour créer des modèles plus grands et ajustables. Il en existe beaucoup nous détaillerons notamment un cas particulier de réseau *transformeur* dans la partie suivante 1.3.

Pour des informations complémentaires sur les différentes architectures de réseaux de neurones et les spécificités de celles-ci, le lecteur est invité à regarder le mémoire de Gaspard NOACK-WILHELM (2025) qui traite en détail de ces sujets.

Chapitre C

Annexes C

C.1 Soundminer et les métadonnées.

Soundminer est un gestionnaire de sonothèque qui permet (entre autres) de rentrer des informations non-audio dans des fichiers audios. Ces divers champs descripteurs permettent de définir un son de manière plurifactorielle et donc de manière précise. Ces champs sont nécessaires pour permettre au monteur son une recherche selon différents critères en fonction de la situation à laquelle il doit faire face. Le tableau 34 les recense.

Les classifieurs basés sur des données brutes (raw-datas).

Les *hand-designed features* ont montré leur intérêt en matière d'économie de calcul et de performances et sont toujours très utilisées. Néanmoins, avec la démocratisation des *GPU*, il devient maintenant possible d'ignorer cette phase d'extraction de caractéristiques pour laisser l'algorithme faire sa prédiction sur des données brutes (*raw-datas*). Si les inconvénients de cette méthode concernent avant tout la consommation d'énergie et la nécessité d'importantes ressources de calculs, les avantages n'en sont pas moins nombreux. En se basant uniquement sur les données brutes, le réseau de neurones apprend de manière plus *objective*. En effet, présenter la donnée sous une forme de spectrogramme ou une suite de vecteurs MFCC est un biais en soi. Il n'est pas garanti que la manière dont nous percevons le monde qui nous entoure soit la plus efficace pour classer les sons. De plus, ces représentations introduisent des approximations liées à la quantification et à la compression des données [1].

Depuis une dizaine d'années, on a montré que des performances équivalentes aux modèles basés sur des représentations temps-fréquences, pouvaient être obtenues en utilisant une représentation de type forme d'onde (*raw*) et des réseaux de neurones très profonds (34 couches) (Dai *et al.*, 2016).

Une autre idée développée par Xifeng Dong *et al.* (2020) est consiste à utiliser la représentation en temps-amplitude en parallèle d'une analyse basée sur un *log-mel spectrogram* [2] spectrogramme de Mel pondéré par une fonction logarithme sur l'intensité sonore, de sorte à s'approcher de la perception humaine qui est logarithmique selon les fréquences mais aussi selon le niveau sonore. Les résultats basés sur les deux flux sont ensuite combinés pour tirer le meilleur parti de chacun. Cette méthode permet d'exploiter au maximum les données en fréquentiel et en temporel.

1. Geoffrey Peter dans (Mallat & Geoffrey, 2020) https://youtu.be/UlyEAGhhHFo
2. Le *log-mel spectrogram* est un

FIGURE 34 – Liste des champs descripteurs interprétable par Soundminer

Chapitre D

Annexes D

D.1 Glossaire.

Abstraction (capacité d') :

L'abstraction est l'opération mentale, par laquelle les propriétés générales, universelles et nécessaires d'un objet sont distinguées de ses propriétés particulières et occasionnelles. Par cette opération, notre pensée prend une distance par rapport à l'expérience sensible et forme l'ensemble de nos idées qui seront consignées dans un *concept*.

Ajustement :

Technique dans le champ de l'apprentissage automatique permettant de spécialiser un modèle généraliste en vue d'une application précise. Cette phase se fait généralement à la fin de l'apprentissage sur des données annotées qui satisfont des conditions précises. On parle en anglais de phase de *fine-tuning*.

Apprentissage par transfert, eng. Transfert learning :

En apprentissage automatique, il désigne la capacité d'un algorithme à transférer ses connaissances d'une tâche source vers une tâche cible. Il peut être vu comme la capacité d'un modèle à appliquer ses connaissances, apprises sur une tâche antérieure, vers une nouvelle tâche, ces dernières partageant des similarités. Ce transfert se fait généralement par un entraînement supervisé appelé *phase d'ajustement* avec des données correspondant à la tâche cible. Attention cependant à ne pas confondre le *transfert de connaissances* et la *généralisation*, la généralisation étant simplement l'extension d'une même tâche à des données inconnues.

BOOM (Sound effect)

: Un son grave, court et dense, grands coups profonds dans le style de ceux que l'on retrouve dans les *trailers*.

Bruitage :

Sons enregistrés de manière synchrone à l'image par un professionnel, permettant d'ajouter de la matière sonore à une production audio-visuelle.

Buffer :

Zone de mémoire temporaire dans laquelle les données (en l'occurrence audios) sont stockées pendant qu'elles sont traitées ou transférées. Le terme de mémoire tampon est parfois utilisé en français.

Chutier (Audiovisuel) :

Un chutier désigne initialement un sac pour récupérer les chutes de pellicules. Cependant, dans une session de montage numérique, il s'agit d'un répertoire contenant des objets (clips vidéos ou sons).

Classification d'évènements sonores, eng. Sound event classification :

Champ de recherche relatif au traitement du signal audio visant à dire si un évènement sonore cible est présent ou non dans une scène sonore et si oui à dire où elle commence et où elle finit. Un algorithme qui répond à cette question effectue une tâche de *détection*.

Classification de scènes sonores, eng. Acoustic scene categorisation :

Vise à classifier l'entièreté d'une scène sonore dans une case (intérieur/extérieur, ville/campagne etc). On parle alors d'une tâche dite de classification car il n'y a qu'une seule sortie possible.

Classifieur (algorithme) :

On désigne par classifieur un algorithme, généralement entraîné par apprentissage supervisé, capable de répartir les objets qu'on lui présente en différentes classes appartenant à une ontologie pré-définie.

Couverture :

Le jeu de données doit contenir autant de catégories que nécessaires à la tâche de classification.

Design sonore :

ou conception sonore en français est l'art d'utiliser des éléments sonores afin d'obtenir un effet désiré.

Différentiabilité :

La différentiabilité généralise la notion de dérivabilité aux fonctions à plusieurs variables.

Drone (Sound effect) :

Caractérise un type d'effet sonore, de niveau constant, très peu dynamique, généralement tonal ou harmonique. On peut également qualifier ces sons de "bourdons" ou de "nappes".

Enveloppe ADSR :

L'enveloppe ADSR d'un son est séparée en 4 phases, l'attaque *(attack)* est la partie montante du son, plus l'attaque est courte, plus la transitoire est raide ; le déclin, (decay), c'est la partie du son qui vient juste après l'attaque durant laquelle le son décroit afin d'atteindre un niveau stable ; l'entretien *(sustain)* constitue la partie du son à volume stable ; enfin le relâchement *release*, caractérise la manière dont le son s'éteint.

Espace des caractéristiques/Espace latent,
eng. Feature space/Latent space :

Espace abstrait de dimension moindre que la dimension d'entrée d'un algorithme au sein duquel les données ont été compressées pour n'être représentées qu'en fonction de leur caractéristiques principales.

Étiquetage, eng. Tagging :

Tâche visant à attribuer des labels (généralement des noms de sources) aux différents évènements sonores constituants une scène sonore.

Extradiégétique :

Qui est extérieur à la diégèse, c'est-à-dire qui ne fait pas partie de l'action ou qui n'est pas lié aux événements dans une œuvre de fiction. Dans cette scène du film, la musique est un son extradiégétique car elle ne peut pas être entendue par les personnages ; c'est une musique d'ambiance pour le spectateur.

Exactitude :

En métrologie, l'exactitude d'une mesure correspond à l'étroitesse de l'accord entre une valeur mesurée et une valeur vraie. Elle mesure la proportion d'éléments bien classés qu'ils soient positifs ou négatifs parmi l'ensemble des observations.

Fonction coût/perte, eng. loss/cost function :

Lors de l'apprentissage automatique d'un algorithme, la fonction coût est la différence entre la vérité terrain et la prédiction de l'algorithme, minimiser cette fonction revient à maximiser le taux de bonnes précisions de l'algorithme.

Indexation :

L'indexation (d'un son dans une sonothèque) correspond à la tâche d'inscription des métadonnées afin qu'il puisse être retrouvé dans la banque de sons. Les champs comme le nom du fichier, la description, l'auteur, les catégories UCS et d'autres sont renseignées à ce moment là.

Inférence :

Mouvement de la pensée qui consiste à admettre une proposition en raison de son lien avec une proposition préalable tenue pour vraie. Dans le champ de l'apprentissage automatique, l'inférence statistique est l'ensemble des techniques permettant d'induire les caractéristiques d'un groupe général (la population) à partir de celles d'un groupe particulier (l'échantillon). On qualifie d'inférence la phase d'application (après l'apprentissage).

Jeu de données, eng. Dataset :

Ensemble d'échantillons visant à entraîner une machine. Le jeu de données est un ensemble d'exemples tenus pour vrais (vérité terrain), qui permettent à l'algorithme, par le principe d'inférence et de généralisation, d'étendre ces connaissances à des données qu'il ne connaît pas. Les données peuvent être annotées (apprentissage supervisé) ou non (apprentissage non supervisé).

Layering :

Fait de cumuler plusieurs couches de sons afin de former une bande sonore riche et cohérente.

Lexique :

Ensemble des mots (d'une langue).

Métadonnée :

Caractéristique formelle normalisée et structurée utilisée pour la description et le traitement des contenus des ressources numériques.

Ontologie :

Ensemble de concepts et de catégories dans une matière ou un domaine qui montre leurs propriétés et les relations qu'ils entretiennent entre eux.

Pas d'apprentissage, eng. Learning rate :

Cet hyperparamètre quantifie la "vitesse" à laquelle un modèle peut apprendre. Le but de l'apprentissage automatique est de miniser l'erreur entre la prédiction et la vérité terrain, ceci se fait via l'otpimisation de la fonction coût. Dans une descente de gradient sur la fonction coût, on se déplace à chaque itération d'une quantité lr vers un minimum local de la fonction, lr désigne le pas d'apprentissage. Plus il est grand, plus vite le minimum est atteint, au détriment de la précision, (avec un pas trop grand, il est impossible d'aller à l'endroit exact où la fonction coût est minimale).

Polysémie :

Se dit d'un objet dont la perception peut induire plusieurs significations. L'objet sonore est un bon exemple de polysémie, le même objet sonore pouvant être perçu différemment en fonction de son contexte (cf. moulin à café de Schaeffer A.1.2).

Post-production :

Ensemble du travail fait après la phase d'enregistrement des images et du son dans la création d'un produit audiovisuel. La post-production comprend par exemple les phases de montage, mixage, étalonnage ...

Prompt :

Les intelligences artificielles génératives utilisent souvent un texte dans lequel l'utilisateur détaille sa requête et l'objet qu'il attend en sortie de l'algorithme ; ce texte est appelé un prompt. Le prompt est donc l'entrée d'un modèle génératif, c'est ce texte qui va être décomposé en vecteurs et compressé dans l'espace latent de la machine, avant d'être transformé en image, son, texte répondant à une question, *etc.*

Regroupement, eng. Clusterization :

Tâche visant à regrouper des entrées partageant une ou plusieurs caractéristiques bas niveau communes.

Rétropropagation, eng. Backpropagation :

Le principe de rétropropagation vise à transmettre la valeur du gradient des couches finales du réseau vers les couches d'entrée. Les poids synaptiques sont ajustés d'autant qu'ils contribuent à une erreur. cf. Rumelhart *et al.* (1986) pour une explication détaillée.

Riser (Sound effect) :

Un son texturé d'entretien monotone dont la fréquence augmente au cours du temps, servant généralement à faire monter la pression.

Rumble (Sound effect) :

Un son grave continu et résonnant, comme un souffle dans le bas du spectre ou le tonnerre au loin.

Sémantique :

Qui concerne le sens, la signification.

Sonothécaire :

À l'instar du bibliothécaire qui range et ordonne les livres, le sonothécaire, lui, range et ordonne les sons.

Stem :

Regroupement de plusieurs sources instrumentales, généralement par timbres similaires, par fonction musicale, ou pour toute autre raison amenant à un regroupement pertinent.

Taille (d'un jeu de donnée) :

Pour chaque catégorie, il doit y avoir un "assez grand" nombre d'exemples.

Taille de lot, eng. Batch size :

Lors du procédé d'apprentissage d'un réseau de neurones, les paramètres du modèle sont mis à jour. Pour assurer un meilleur apprentissage, on met à jour les paramètres toutes les x itérations. Le nombre x désigne la taille du lot, c'est-à-dire le nombre d'exemples qui sont présentés à l'algorithme avant qu'il mette à jours ses poids et biais.

Taxinomie, taxonomie :

Étude des lois régissant une classification.

Tête d'attention (eng. Attention head) :

Une tête d'attention est un module de traitement au sein d'un algorithme transformeur. Ce module gère le *procédé d'attention* qui permet au transformer de privilégier certaines structures par rapport à d'autres dans le fichier d'entrée.

Variabilité :

Au sein d'une même catégorie d'un jeu de données, il doit y avoir des exemples avec différentes conditions d'émission, d'enregistrement, d'acoustique etc. Dans le but de représenter l'ensemble des caractéristiques propre à cette classe.

WHOOSH (Sound effect) :

Son généralement avec une texture de bruit blanc utilisé pour imiter un objet qui se rapproche à grande vitesse.

Workflow :

Séquence de processus industriels, administratifs ou autres par lesquels un travail passe de sa conception à l'achèvement.