Projet : reconnaissance de bruits de chantier

Jacques Dié, Maël Fayolle
Etudiants au Master 1 Ingénierie des Systèmes Intelligents à Sorbonne Université
Email : jacques.die@etu.sorbonne-universite.fr
mael.fayolle@etu.sorbonne-universite.fr

***Abstract*— Ce papier rend compte de la méthode de reconnaissance vocale par codage du signal en Mel-Frequency Cepstral Coefficients (MFCC) puis de son utilisation sur des bruits de chantier dans un réseau de neurones artificiels profonds.**

I. INTRODUCTION

Depuis déjà quelques années, les avancées en intelligence artificielle ont mené à des changements importants dans la façon dont nous interagissons avec la société. L'une de ces avancées est le domaine du Deep Learning.

Que ce soit dans nos recommandations d'achat sur Amazon, dans la recherche sur Google ou même dans des domaines plus poussés comme les marchés financiers, le Deep Learning est un véritable atout pour peu que l'on sache l'utiliser.

Basées sur une reproduction de réseaux de neurones qui singent celles que nous avons dans notre cerveau, cet ensemble de méthodes permet d'apprendre des données sans supervision humaine tout en gardant les erreurs d'apprentissage et de prendre des décisions en conséquence.

Nous allons ici nous intéresser, dans le cadre de notre projet à Sorbonne Universités, à l'utilisation du Deep Learning avec TensorFlow et Keras pour reconnaître des bruits de chantier et classer les différents outillages que nous avons capté.

Pour cela nous allons regarder dans un premier temps la transformation du signal audio en MFCC. Ce codage permet de décrire la forme du spectre du signal entrant à l'aide d'un nombre réduits de coefficients. Une fois le signal modifié, nous appliquerons une architecture de Deep Learning pour prédire les résultats et les classifier efficacement.

II. CODAGE MEL-FREQUENCY CEPSTRAL COEFFICIENTS

A. Captation de signal audio

Avant de rentrer dans l'explication du deep learning, nous devons évoquer le pré-traitement des données que nous allons envoyer. En effet, l'une des phases les plus importantes du Machine Learning est de s'assurer que les données d'entrée soient adaptées au problème posé. Ici, nous allons étudier comment convertir une voix en codage MFCC avant de généraliser pour différents types de sons.

A l'arborescence du processus, nous créons des sons quasi-périodiques en poussant l'air de nos poumons pour faire vibrer nos cordes vocales. L'air va ensuite dans le pharynx et les cavités nasales et orales avant de s'échapper par la bouche et le nez.

Ensuite, le changement de pression de l'air produit par cette articulation est converti en voltage par un microphone et échantillonné avec un convertisseur « analog to digital ».
La sortie de cet enregistrement est un array uni dimensionnel de nombres représentant la discrétisation des échantillons provenant de la conversion digitale.

Cet array possède trois caractéristiques propres :

- Le taux d'échantillonnage, la fréquence à laquelle le signal analogique est capturé (en Hertz),

- Le nombre de canaux qui correspond au nombre de microphones utilisées pour capturer le son,

- La précision (bit depth) est le nombre de bits utilisées par échantillons, correspondant à la résolution de l'information.

Dans le cadre de la détection automatique de langage, la plupart des signaux sont alors traités pour filtrer le bruit ou réduire le dimensionnement de l'array concerné et récupérer des caractéristiques propres pouvant être manipulés.

Ces dernières proviennent d'une forme d'analyse spectrale qui convertit le signal audio à un jeu de caractéristique précis qui met en exergue les signaux récupérés par une oreille humaine.
Ces méthodes reposent sur le calcul d'une short time Fourier Transform (STFT) sur le signal avec l'aide d'une fast fourier transform, de banque de filtres ou une combinaison des deux.

Mel Frequency cepstral coefficient (MFCC) sont les caractéristiques les plus communément utilisées pour la détection automatique de langage d'une part parce que ce jeu de caractéristiques repose sur des filtres qui mimiquent

l'oreille humaine (et est donc approprié à la captation de paroles) et d'autres parts car possédant une faible dimensionnalité.

B. Etapes pour obtenir un codage MFCC

Il y a sept étapes pour obtenir des caractéristiques de la MFCC :

1) Pre-emphasis

Application d'un filtre au signal d'entrée qui abaisse les amplitudes des basses fréquences et augmentent les hautes (dans la production vocale, l'énergie des hautes fréquences a tendance à être plus faible). On peut voir cela comme une normalisation de données.

2) Framing

Le signal acoustique étant en constante évolution, sa modélisation est effectuée à partir de petits segments échantillonnés du signal d'entrée vu comme stationnaire.
Le framing consiste à séparer les échantillons du signal audio et à les mettre dans des segments à taille fixe, des frames.

Les frames vont s'apparenter à la limite de la représentation phonétique des sons. Sachant qu'un phonème (élément sonore du langage parlé, considéré comme une unité distinctive) peuvent prendre entre 5 à 100 ms, il faudra adapter la longueur de la frame en conséquence.

En général, on prendra un segment correspondant à 20 à 30 ms d'informations pour le considérer comme quasi-stationnaires et cela toutes les 10 à 15 ms pour balayer efficacement le signal.

3) Windowing

Les échantillons contenus dans les frames doivent être normalisés pour réduire les potentielles disparités qui peuvent apparaître aux bords de ces dernières. On peut utiliser de nombreuses fenêtres, les plus utilisées pour le speech recognition restent la fenêtre de Hann ou celle de Hamming.

4) Fast Fourier Transform

Pour passer du domaine temporel au domaine fréquentiel sur toutes nos frames, on utilise une short time fourier transform (application d'une discret fourier transform sur chaque frame).

On peut utiliser à ce niveau un spectrogramme pour afficher notre transformation de la FFT sur chaque frame et observer les différences entre elles (Le spectrogramme représentant dans un seul diagramme à deux dimensions le temps, la fréquence et la puissance sonore du signal étudié).

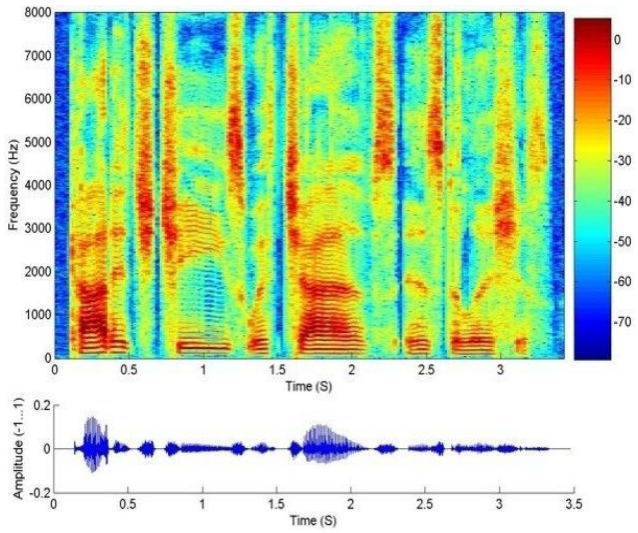

Figure 1 : exemple de spectrogramme ou la puissance est associée à une couleur ordonnée par les fréquences et le temps,

5) Bancs de filtres Mel

L'échelle de Mel est une échelle psychoacoustique de hauteurs des sons, au sens de leur repérage entre grave et aigu, dont l'unité est le mel.

Les bancs de filtres Mel sont des filtres qui s'apparentent à l'oreille humaine, linéaires aux fréquences basses et logarithmiques pour les hautes fréquences.

On applique tout cet ensemble de filtres sur les frames et on obtient la somme pondérée des fréquences spectrales qui correspond à chaque filtre. Ces valeurs cartographient les fréquences d'entrée dans l'échelle de Mel.

A noter qu'il existe d'autres échelles « perceptives » comme celle de Bark, les filtres ERB, Gamma tone…

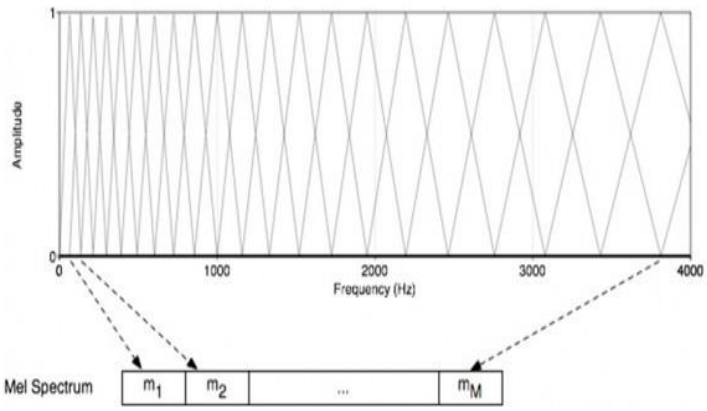

Figure 2 : bancs de filtres mel, les plages de fréquence en entrée sont converties dans l'échelle de Mel

6) Discret cosinus transform

Les valeurs obtenues avec les bancs de filtres passent ensuite par une transformation cosinus discrète pour revenir dans un domaine pseudo-temporel.

La transformation cosinus discrète agit comme une ifft mais seulement sur l'ensemble des réels (puisque le spectre d'amplitude que l'on cherche à obtenir est réel et symétrique). Cette transformation a aussi pour effet de compresser les data en input qu'on nommera vecteurs acoustiques dans un jeu de coefficients cosinus qui correspondent aux oscillations dans la fonction aussi appelé la MFCC.

7) Delta Energy et delta spectrum

Etant donné que notre signal change en fonction des frames, le delta energy et delta spectrum va donner des informations sur la valeur des transitions entre les frames :
Delta energy sur les différences entre les coefficients des frames consécutives et delta spectrum sur les différences entre les delta energy. Ces informations seront ensuite manipulées par notre architecture de Deep Learning.

Nous nous retrouvons au final avec des coefficients sepsctraux calculés appliquée au spectre de puissance du signal d'origine, espacées logarithmiquement selon l'échelle de Mel.
Mais comment utiliser ces données ? La réponse étant : en faisant recours à la biologie.

III. DEEP LEARNING

A. Neurones au sens biologique

De manière générale, le cerveau humain est composé de 86 à 100 milliards de cellules appelées neurones dont le rôle est d'acheminer et de traiter des messages dans notre organisme.

Chaque neurone possède alors des rôles prédéfinis s'apparentant par exemple à des mouvements précis quand d'autres s'occupent de notre digestion.

Un neurone est composé d'un corps cellulaire, d'un noyau, de plusieurs ramifications (dendrites) qui sont des points d'entrée de l'information circulant dans le cerveau, d'un chemin de sortie de l'information (axone), des synapses connectant les neurones entre eux et d'une gaine de myéline protégeant l'axone.

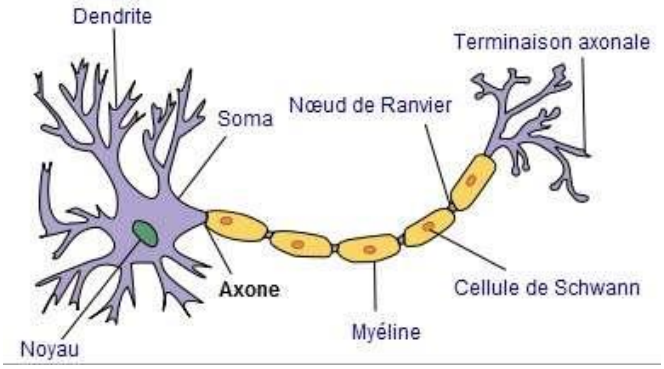

Figure 3 : schéma d'un neurone organique

La communication entre neurones s'opère par l'échange de messages sous forme de variation de tension électrique. Un neurone peut recevoir plusieurs messages de la part d'autres neurones auxquels il est lié.

Il va alors réaliser la somme des influx nerveux reçus puis, si cette somme dépasse un certain seuil, il « s'active » et transmet à son tour un message via son axone aux neurones connectés à celui-ci et ainsi de suite jusqu'à complétion du chemin et donc de l'action entreprise.

B. Neurone artificiel : le perceptron

Ce mode de fonctionnement va être repris par un algorithme d'apprentissage supervisé : le perceptron.
Sous des traits de neurone formel suivant une règle d'apprentissage prédéfinie, le perceptron va déterminer automatiquement le poids synaptique des données d'entrée de manière à appliquer une classification au problème posé.

Mais là ou une autre méthode de classification s'arrêterait, la grande particularité des réseaux neuronaux est la notion de rétropropagation.
Ils peuvent apprendre de leurs erreurs et se corriger au fur et à mesure de ses epochs (cycles d'apprentissage de l'algorithme).

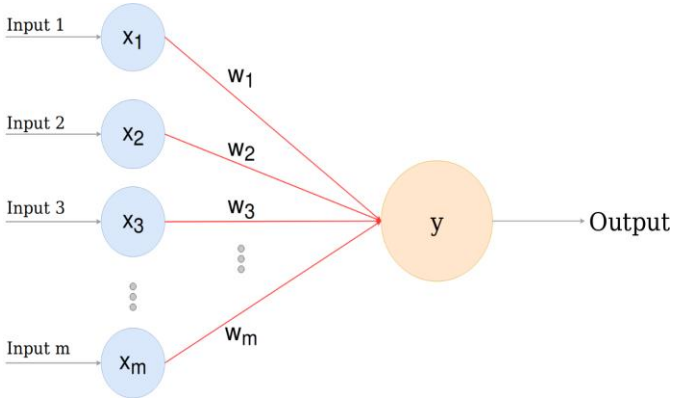

Figure 4: schéma d'un perceptron

Pour un problème linéairement séparable, un perceptron simple couche est suffisant mais dès lors que le problème est plus complexe, il faut déployer une architecture multicouche.

C. Problèmes à plusieurs classes : le perceptron multicouche

Dans le perceptron multicouche à rétropropagation, les neurones d'une couche sont reliés à la totalité des neurones des couches adjacentes. Ces liaisons sont soumises à un coefficient (poids) altérant l'effet de l'information sur le neurone de destination.

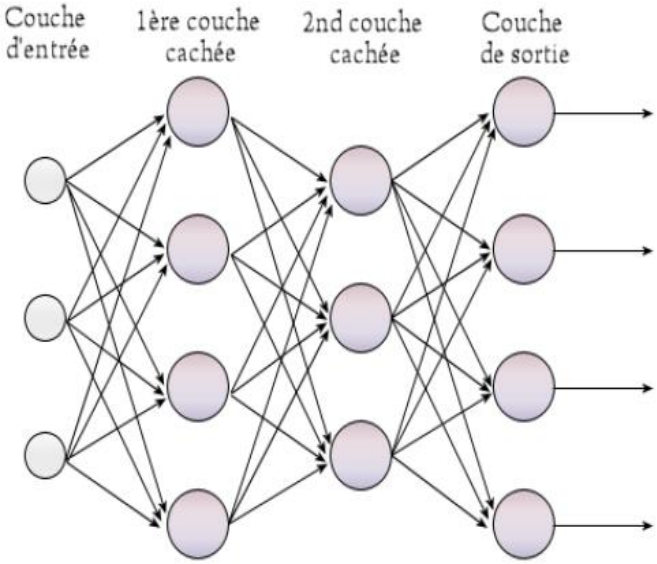

Figure 5 schéma d'un perceptron multicouche

La mise en place d'un Perceptron multicouche pour résoudre un problème passe donc par la détermination des meilleurs poids applicables à chacune des connexions inter-neuronales.

Ici, cette détermination s'effectue au travers d'un algorithme de rétropropagation.

1. Présentation d'un motif d'entraînement au réseau.
2. Comparaison de la sortie du réseau avec la sortie ciblée.
3. Calcul de l'erreur en sortie de chacun des neurones du réseau entre la cible et la prédiction du réseau
4. Calcul, pour chacun des neurones, de la valeur de sortie qui aurait été correcte.
5. Définition de l'augmentation ou de la diminution nécessaire pour obtenir cette valeur (erreur locale).
6. Ajustement du poids de chaque connexion vers l'erreur locale la plus faible déterminée par une descente stochastique de gradients.
7. Attribution d'un bias term à tous les neurones précédents
8. Recommencer à partir de l'étape 4, sur les neurones précédents en utilisant les poids mis à jour avec le bias pour réduire l'erreur.

Le Deep Learning repose sur cette architecture de perceptron multicouche.

D. Les méthodes de Deep Learning pour la reconnaissance des sons

Appliqué au traitement de signal audio, le Deep Learning est un outil puissant pour la reconnaissance de bruits.

En utilisant les coefficients sepsctraux (calculées pendant le codage MFCC) en entrée de notre architecture multicouche, nous pouvons arriver à classer ces données en fonction d'archétype prédéfinis qu'on aura entraîné auparavant sur des exemples déjà étiquetés.

En détails, l'architecture que nous déployons va être composé de couches de convolutions appliquées aux entrées pour déterminer si les caractéristiques que nous recherchons sont présentes en effectuant des produits de convolution sur les données reçues en fonction d'une profondeur, d'un pas et d'une marge.

Elles sont souvent combinées à un pooling pour limiter les opérations en sous échantillonnant leur entrée (pour rappel, plus il y a de données et plus il y aura de multiplications à faire à chaque couche ce qui risque de prendre un temps considérable pour une grande base de données).

Ensuite, une fois que les données ont été sous échantillonnées et qu'on a extrait les caractéristiques qui nous intéressent, elles passent dans les couches denses du réseau qui évalueront en fonction de leur fonction d'activation et des poids que les caractéristiques possèdent à quelle catégorie les données appartiennent. C'est à ce moment que l'algorithme de rétropropagation rentre en jeu.

Aussi, pour éviter le sur-apprentissage du modèle, nous utilisons des Dropout qui indiquent à un layer de désactiver un nombre de neurones précis d'une façon aléatoire et ainsi de garantir que les poids sur les connexions

Après avoir fait tourner la méthode selon un nombre d'epoch choisies, nous obtenons alors un taux de reconnaissance qui nous indiquera le pourcentage de précision que le réseau a obtenu. Si on le soumet à un jeu de données MFCC, il pourra prédire avec cette même précision à quelle classe il appartient.

C’est ainsi que nous pouvons reconnaître les sons que ce soit une voix ou des bruits environnants.

IV. RÉALISATION DU PROJET

A. Mise en forme des données

Le dataset est composé de 7 classes qui correspondent à un outil de travail. Le réseau de neurones est réalisé sur Kaggle, une plateforme web dotés de puissants outils pour manipuler les données utilisées pour le data science.

Pour simplifier le processus, nous avons créé deux fichiers contenant respectivement les transformées MFCC et le label de la classe associée que nous avons uploadés sur Kaggle. Nous avons décidé de conserver 13 cepstrum, avec une fenêtre de 1024 éléments pour la transformée de Fourier.

Ces coefficients sont calculés sur des trames de 10ms. De cette manière, la base de données est de taille conséquente (90103 éléments). De plus, ceci nous permettra de déterminer des changements de classe avec une plus grande précision temporelle (tout en sachant qu'une classe est très rarement représentée moins d'une seconde).

B. Architecture du réseau de neurones

Les données transformées en vecteurs acoustiques sont de taille 13x1, c'est à dire 1D. De ce fait, nous utilisons les fonctions de convolution et de maxpooling 1D de la bibliothèque Keras. Par ailleurs, les objets traités étant de petite dimension, nous n'avons pas besoin d'utiliser une très grande architecture. Voici celle que nous avons retenue après plusieurs essais :

	Architecture		
	type	*dimension*	*activation*
0	Input	*13x1*	-
1	Convolution 1D	*32, (5)*	*relu*
2	Convolution 1D	*64, (3)*	*relu*
3	MaxPooling 1D		
4	Convolution 1D	*128, (3)*	*relu*
5	Convolution 1D	*128, (3)*	*relu*
6	MaxPooling 1D		
7	Convolution 1D	*128, (3)*	*relu*
8	Convolution 1D	*128, (3)*	*relu*
9	Flatten		
10	Dense	*512*	*relu*
11	Dropout 0.3		
12	Dense	*64*	*relu*
13	Droput 0.3		
14	Dense	*7*	*softmax*

Figure 6 : Architecture du réseau neuronal utilisé

Sur 50 epochs, des batch de taille 512 et l'optimiseur Adam, nous obtenons un taux de reconnaissance de 97.2%. En rappelant que les prédictions sont effectuées toutes les 10ms, ce résultat permet de déterminer avec précision à quelle classe appartient chaque instant d'une bande.

Vous pourrez retrouver les poids du modèle ainsi que les fichiers utilisés dans l'archive du projet.

V. CONCLUSION

Nous avons pu voir comment est-ce que l’on pouvait reconnaitre des signaux audios à travers la création de vecteurs acoustiques et de leur manipulation dans une architecture de réseaux neuronaux profonds.

Le Deep Learning a permis d’optimiser la reconnaissance audio cependant, il existe encore de nombreux défis. En effet, dans notre base de données, les signaux étaient facilement identifiables mais par exemple, dans un environnement bruité, capté par un microphone de qualité moindre, la reconnaissance devient alors bien plus compliquée étant donné l’aléatoire du jeu de données que l’on pourrait soumettre à notre architecture.

Plus spécifiquement, dans la reconnaissance vocale, des paroles avec un fort accent met à l’épreuve les différentes méthodes de reconnaissance et ont besoin d’une attention particulière. On pourrait aussi s’intéresser à un reconnaisseur automatique vocal qui reconnaîtrait plusieurs voix superposées…

Toutes ces problématiques ont besoin de nouvelles techniques pour le signal processing avec des algorithmes d’apprentissage qui seraient assez robustes pour d’une part intégrer la force des modèles d’architecture dynamiques tout en étant assez maniable pour traiter des cas particuliers mais aussi de s’intéresser à d’autre disciplines scientifiques que ce soit dans les sciences cognitives, la linguistique automatique, l’intelligence artificielle et le machine learning bio inspiré.

VI. References

A. *Figures*

Figure 1 : https://www.futura-sciences.com/sante/definitions/biologie-neurone-209/

Figure 2 : https://blog.groupe-sii.com/ral/

Figure 3 : https://towardsdatascience.com/the-perceptron-3af34c84838c

Figure 4 : https://towardsdatascience.com/the-perceptron-3af34c84838c

Figure 5 : https://fr.wikipedia.org/wiki/Perceptron_multicouche

B. *Bibliographie*

[1] Automatic Speech Recognition A Deep Learning Approach, Dong Yu ,Li Deng (2015)

[2] Abdel-Hamid, O., Deng, L., Yu, D.: In: Exploring Convolutional Neural Network Structures and Optimization Techniques for Speech Recognition, pp. 3366–3370 (2013)

[3] Deep Learning with Applications Using Python, Navin Kumar Manaswi (2018)

[4] Deep Learning for NLP and Speech Recognition, Uday Kamath, John Liu, James Whitaker (2019)

[5] Marc Parizeau, Réseaux de Neurones (Le perceptron multicouche et son algorithme de rétropropagation des erreurs), Université Laval, Laval, 2004, 272 p.

[6] Geoffroy Peeters, Master 2 ATIAM (2017-2018) Extraction automatique d'une structure musicale et génération d'un résumé audio http://recherche.ircam.fr/anasyn/peeters/pub/cours/20171031_Peeters_20172018_ATIAM_Cours_Structure.pdf

[7] Adam Geitgey, Machine Learning is Fun Part 6: How to do Speech Recognition with Deep Learning (2016) https://medium.com/@ageitgey/machine-learning-is-fun-part-6-how-to-do-speech-recognition-with-deep-learning-28293c162f7a

[8] Baelde M., Biernacki C. et Greff R. (2017), A mixture model-based real-time audiosources classification method,The 42nd IEEE International Conference on Acoustics,Speech and Signal Processing, ICASSP2017.