République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

UNIVERSITÉ SAAD DAHLEB – BLIDA 1

Faculté de science

Département d'informatique

Mémoire de Projet de Fin d'Études Pour l'obtention du diplôme de Master en Informatique

Option : *Ingénierie des systèmes informatiques et Réseaux*

Theme

Architecture Encodeur-Encodeur pour la génération automatique de légendes textuelles des signaux audio

---

Présenté par :

- ✧ Azazi Khalida
- ✧ Ourchane Sara

Devant le jury composé de :

Dr.Kameche Abdallah Hicham	USDB	Promoteur
— Dr.Bacha Sihem	USDB	Président
Dr.Ferfara Sofien	USDB	Examinateur

Année Universitaire : 2023-2024

Remerciement

Tout d'abord, Louange à **Allah**, Seigneur de l'univers qui nous a armés de patience et de force tout au long de ces longues années et nous a bénis avec succès dans l'accomplissement de cet œuvre.

Nous offrons également nos sincères remerciements et gratitude à notre encadrant **Dr. Kameche Abdallah Hicham**, pour son aide tout au long de la période de travail, et pour tous les conseils, instructions et documents précieux qui ont grandement contribué à la mise en œuvre de ce travail.

Nous remercions également tous **les enseignants de l'Université SAAD DAHLEB** qui ont travaillé sur notre formation dès le premier jour, en particulier les enseignants du département d'informatique.

Nous les remercions tous, sans exception. Et nous n'oublions pas de remercier tous ceux qui ont contribué, directement ou indirectement, à la réalisation de ce travail.

Nous exprimons également notre gratitude absolue à nos **chers parents**, pour tout leur sacrifice, amour, tendresse, soutien, et nos **frères** et **sœurs** et nos **chères amis** pour tout ce qu'ils nous ont donné comme soutien financier et psychologique tout au long des années scolaires, et aussi à tous les amis dont les noms n'ont pas été mentionnés.

Merci à tous.

Dédicace

Je profite de l'occasion offerte par ce mémoire pour exprimer mes dédicaces :

À mes chers parents, Moussa et Aicha, qui n'ont jamais cessé de m'encourager. Ainsi pour leur soutien inconditionnel et leur amour indéfectible...

À mes frères Tarek, Anis et Nadim, et à mes sœurs Sabrina, Hanane et Maissa. Que Dieu vous protège et vous offre le paradis.

À mon ami et binôme Sara, ainsi qu'à tous mes amis...

À mes professeurs, qui m'ont enseigné durant mes études.

AZAZI Khalida

Dédicace

Je profite de l'occasion offerte par ce mémoire pour exprimer mes dédicaces :

À mes parents, Fadila et Mohamed, sans qui je n'aurais jamais pu atteindre mes buts. Merci pour votre amour, votre soutien inconditionnel et vos encouragements constants.

À mon frère, Allaa Eddine, pour sa présence et son soutien. À mes chers amis, Wiam, Khalida, Amina, Ihcene et Hadil pour leur amitié, leur soutien et les moments précieux partagés ensemble.

À mes professeurs, pour leur enseignement et leur guidance tout au long de mes études.

Ourchane Sara

Résumé

L'Automated Audio Captioning (AAC) est une discipline émergente visant à générer automatiquement des descriptions textuelles précises à partir de contenus audio. Le Joint Embedding est une technique utilisée pour projeter des données de différentes modalités (par exemple, audio et texte) dans un même espace vectoriel partagé. L'objectif est de représenter les deux types de données de manière à ce qu'ils soient directement comparables et que leurs relations sémantiques soient préservées dans cet espace commun.

Dans ce travail , On propose une approche de Joint Embedding, qui projette les représentations audio et textuelles dans un espace vectoriel commun. Pour encoder les descriptions textuelles,on utilise SBERT, tandis que des architectures spécialisées comme le CNN14 Encoder sont utilisées pour extraire des caractéristiques audio pertinentes à partir des spectrogrammes Mel.

On utilise le dataset Clotho, qui propose une variété de clips audio accompagnés de descriptions textuelles, pour développer et évaluer des modèles AAC.

Le modèles AAC obtenu peut générer automatiquement des légendes audio de haute qualité à partir du dataset Clotho, ouvrant la voie à de nombreuses applications pratiques dans l'accessibilité et la recherche multimodale.

Mots clés : Réseaux de neurones, traitement de signaux audios, génération automatique de légende pour audio, traitement de langage naturel, projection multimodale.. . .

Abstract

Automated Audio Captioning (AAC) is an emerging discipline aimed at automatically generating accurate textual descriptions from audio content. Joint Embedding is a technique used to project data from different modalities (e.g. audio and text) into the same shared vector space. The aim is to represent the two types of data in such a way that they are directly comparable and their semantic relationships are preserved in this common space.

In this work, we propose a Joint Embedding approach, which projects audio and text representations into a common vector space. SBERT is used to encode text descriptions, while specialized architectures such as CNN14 Encoder are used to extract relevant audio features from Mel spectrograms.

The Clotho dataset, which offers a variety of audio clips files accompanied by textual descriptions, is used to develop and evaluate AAC models.

The resulting AAC models can automatically generate high-quality audio captions from the Clotho dataset, paving the way for many practical applications in accessibility and multimodal research.

Key words : Neural networks, audio signal processing, automatic legend generation for audio, natural language processing, multimodal projection.. . .

ملخص

التعليق الصوتي الآلي AAC هو تخصص ناشئ يهدف إلى توليد أوصاف نصية دقيقة تلقائيًا من المحتوى الصوتي، والتضمين المشترك هو تقنية تُستخدم لإسقاط البيانات من طرائق مختلفة في فضاء متجه واحد مشترك، والهدف هو تمثيل نوعي البيانات بطريقة تجعلها قابلة للمقارنة المباشرة والحفاظ على علاقاتها الدلالية في هذا الفضاء المشترك.

في هذا العمل، نقترح نهج التضمين المشترك، الذي يعرض تمثيلات الصوت والنص في فضاء متجه مشترك. يُستخدم SBERT لترميز الأوصاف النصية، بينما تُستخدم البنى المتخصصة مثل برنامج التشفير CNN14 لاستخراج الميزات الصوتية ذات الصلة من المخططات الطيفية لمل.

يتم استخدام مجموعة بيانات Clotho ، التي تقدم مجموعة متنوعة من المقاطع الصوتية مصحوبة بأوصاف نصية، لتطوير وتقييم نماذج AAC .

يمكن لنماذج AAC الناتجة توليد تسميات توضيحية صوتية عالية الجودة تلقائيًا من مجموعة بيانات Clotho ، مما يمهد الطريق للعديد من التطبيقات العملية في مجال إمكانية الوصول والأبحاث متعددة الوسائط.

الكلمات المفتاحية : الشبكات العصبية، ومعالجة الإشارات الصوتية، وتوليد الأسطورة التلقائية للصوت، ومعالجة اللغة الطبيعية، والإسقاط متعدد الوسائط.

Table des matières

1 Notions Préliminaires **6**

Introduction 6

I L'apprentissage profond **7**

1.1 Introduction 8

1.2 l'apprentissage profond 8

1.3 Les réseaux de neurones 8

1.3.1 Les principales composantes du réseau de neurones 9

1.4 Réseaux de neurones convolutifs (CNN) 12

1.4.1 Les couches de réseaux de neurones convolutionnels 13

1.5 L'auto-encodeur 14

1.5.1 Le fonctionnement d'auto-encoder 14

1.6 Le Transformateur 16

1.6.1 Les Types de Transformateur 17

1.7 Le prétraitement de données pour le Deep Learning 18

II Le traitement d'audio **19**

1.1 Introduction 20

1.2 Le Son 20

1.2.1 L'onde sonore 20

1.2.2 Les Paramètres du Son 21

1.3 Conversion analogique-numérique (ADC) 24

1.4 Fonctionnalités d'audio 26
1.4.1 Fonctionnalités du domaine temporel 26
1.4.2 Fonctionnalités du domaine fréquentiel 27
1.5 La Visualisation du son 29
1.5.1 La forme d'onde d'un signale audio 29
1.5.2 Le Spectrogramme 30

III Traitement de texte 33
1.1 Introduction 34
1.2 Le traitement du langage naturel (NLP) 34
1.2.1 Les techniques de traitement de texte 35
1.2.2 Les techniques de vectorisation dans le NLP 35
1.2.3 Approches Transformateur pour le Traitement du Langage Naturel 37

IV Travaux connexes 38
1.1 Introduction : 39
1.2 Présentation des documents 39
1.2.1 Document 1 [1] 39
1.2.2 Document 2 [2] 40
1.2.3 Document 3 [3] 40
1.3 Analyse comparative de trois études connexes 42
1.4 Conclusion 42

2 Approche proposée 44
2.1 Introduction 44
2.2 L'architecture de la solution 45
2.3 Prétraitement de données 45
2.3.1 Prétraitement des données audio 46
2.4 Prétraitement des données textuelles 46
2.4.1 Intégration de données 46
2.4.2 Calcule et stockage de statistiques des données prétraitées 47
2.5 Extraction des caractéristiques 48

2.5.1 Extraction des caractéristiques audio 48
2.5.2 Extraction des caractéristiques de texte 53
2.6 L'architecture encodeur-encodeur 54
2.7 Décodage et génération de légendes textuelles 55
2.7.1 BART 55
2.7.2 Architecture et paramètres du modèle encodeur-décodeur : 56
2.7.3 Génération de légendes textuelles : 58
2.8 La fonction de perte : 59
2.9 Conclusion 60

3 Résultat pratiques **62**
3.1 Introduction 62
3.2 Les outils utilisés pour le développement 62
3.3 Description de DataSet 65
3.4 Métriques d'évaluation 66
3.4.1 BLEU (Bilingual Evaluation Understudy) : 66
3.4.2 ROUGE (Recall-Oriented Understudy for Gisting Evaluation) . . . 66
3.4.3 CIDEr (Consensus-based Image Description Evaluation) 67
3.4.4 SPICE (Semantic Propositional Image Caption Evaluation) 67
3.4.5 SPIDEr (SPICE + CIDEr) 67
3.4.6 SPIDEr_FL (SPICE, CIDEr et FL) 67
3.5 Paramètres des expériences 67
3.6 Résultat et évaluation 68
3.7 Analyse de résultats 76
3.8 Résultats de Test 77
3.9 Conclusion 80

Conclusion générale **81**

Travaux futurs **83**

Bibliographie **85**

Table des figures

1.1 Représentation mathématique/informatique d'un neurone biologique[4]. . . 9
1.2 Architecture de réseaux de neurones 10
1.3 Fonction d'activation[5]. 11
1.4 Exemple de fonctions d'activation[6]. 12
1.5 Architecture générale d'un CNN[7]. 13
1.6 Architecture d'un Auto-encodeur[8]. 15
1.7 Architecture d'encodeur et décodeur[9] 16
1.8 Architecteur de Transformateur[10]. 17
1.9 Ondes sonores 21
1.10 Système de traitement d'audio[11]. 25
1.11 (i) signal analogique (ii) signal échantillonné (iii) puis quantifié[12]. 25
1.12 Exemple de formes d'ondes d'un signale audio. 30
1.13 Exemple d'un spectrogramme 31
1.14 Mel Spectogramme[13]. 32
1.15 La formule mathématique de TF-IDF 36

2.1 L'architecture générale du système 45
2.2 Exemple des phrase S-BERT 54
2.3 L'architecture encodeur-encodeur de système d'AAC 55

3.1 Les valeurs de fonction de perte pendant l'entrainement de modèle 69
3.2 l'évolution des valeurs de taux d'apprentissage pendant l'entrainement . . . 70
3.3 Les valeurs de fonction de perte pendant la validation de modèle 71
3.4 Comparaison entre las valeurs de perte d'Entraînement et de Validation . . 73

3.5 Page d'acceuil de l'application 78
3.6 Page d'acceuil de l'application(start) 79

Liste des tableaux

1.1 Les paramètres du Son 24
1.2 Comparaison entre les travaux connexes 42

2.1 Description de contenu des dictionnaires générées par le prétraitement des données audio 46
2.2 Description du contenue du DataFrame générée pendant l'intégration de données textuelles 47
2.3 Description de contenu des dictionnaires de données 47
2.4 Les valeurs des paramètres utilisés pour l'extraction du Mel-Spectrogramme 48

3.1 Les outils utilisés 65
3.2 Les métriques utilisées 66
3.3 Hyperparamètres d'entraînement 68
3.4 Les scores obtenus pour les métriques d'évaluation 75
3.5 Comparaison des résultats des métriques des travaux connexes avec les résultats que nous avons obtenus 76
3.6 Les légendes générées par notre modèle pour différents fichiers audio 79

Liste des Acronymes et Abréviation

AI	Intelligence Artificielle
ML	Apprentissage Automatique
DL	Apprentissage Profond
API	Interface de Programmation Applicative
ADC	Convertisseur Analogique-Numérique
AAC	Légende Audio Automatisée
CNN	Réseau de Neurones Convolutif
RNN	Réseau de Neurones Récurrent
ReLU	Unité Linéaire Rectifiée
FT	Transformée de Fourier
DFT	Transformée de Fourier Discrète
FFT	Transformée de Fourier Rapide
STFT	Transformée de Fourier à Court Terme
SC	Centroïde Spectral
MFCC	Coefficients Cépstraux de Fréquence Mel
NLP	Traitement du Langage Naturel
RMS	Moyenne Quadratique
AE	Enveloppe d'Amplitude
BER	Taux d'Energie de Courbure
ZCR	Taux de Fréquence de Traversée de Zéro
ViT	Transformateurs de Vision
TF-IDF	Fréquence de Terme-Inverse Fréquence de Document
BERT	Représentations de Codage Bidirectionnel à partir de Transformateurs
sBERT	BERT de Phrase
GPT	Transformateur Pré-entraîné Génératif
FC	Full Connected

Introduction Générale

1. Motivations

Le développement d'une technologie de génération automatisée de légendes audio peut transformer la manière dont nous interagissons avec le contenu sonore et textuel. Cette technologie, appelée Automated Audio Captioning (AAC), permet de convertir des enregistrements audio en texte en décrivant de manière générale le contenu audio à l'aide d'un texte libre, facilitant ainsi l'accès à l'information pour les personnes malentendantes ou sourdes et améliorant l'inclusivité.

En automatisant la transcription des contenus audio, on peut non seulement gagner un temps précieux mais aussi réduire les coûts associés aux transcriptions manuelles, ce qui est particulièrement bénéfique pour les entreprises et les chercheurs.

De plus, la capacité de transformer des fichiers audio en texte améliore l'indexation et la recherche de contenus dans divers domaines, tels que les médias, le journalisme, et la recherche académique. Cette technologie offre également des avantages dans les environnements de sécurité et de surveillance, où elle peut faciliter la détection de comportements suspects ou la documentation précise d'événements en temps réel.

Dans les plateformes de médias sociaux et de streaming, les légendes audio augmentent l'engagement des utilisateurs en rendant les contenus plus accessibles et compréhensibles. Le développement de cette technologie, qui repose sur des avancées en intelligence artificielle et en traitement du langage naturel, promet d'avoir un impact significatif sur de nombreux aspects de notre société, y compris l'éducation, la formation, et l'interaction homme-machine, en rendant l'information et le divertissement plus accessibles et inclusifs.

2. Problèmes

Bien que l'AAC soit un domaine de recherche actif, il reste de nombreux défis à relever. Les signaux audio peuvent contenir des éléments ambigus et subjectifs qui peuvent être difficiles à interpréter avec précision. Et les différences perceptuelles entre les individus et les nuances contextuelles des signaux audio.

De plus, l'alignement temporel précis entre les segments audio et les mots correspondants dans les légendes demeure un défi majeur. Une synchronisation précise entre les segments audio et les mots correspondants dans les légendes est cruciale pour générer des descriptions cohérentes et contextuellement pertinentes.

3. Contributions

La contribution de ce projet de fin d'études réside dans la conception d'une architecture encoder-encoder innovante, spécifiquement adaptée à la génération automatique de légendes textuelles pour des signaux audio.

En réponse à la problématique soulevée, notre architecture vise à surmonter les défis liés à l'interprétation subjective et aux ambiguïtés présentes dans les signaux audio.

Premièrement, nous proposons un mécanisme d'encodage audio qui capture efficacement les caractéristiques pertinentes du signal, en tenant compte de sa nature complexe et variée. Cette étape d'encodage est cruciale pour garantir une représentation adéquate du contenu audio, permettant ainsi une légende textuelle précise.

Deuxièmement, notre approche d'encodage textuel est conçue pour capturer les nuances sémantiques et contextuelles nécessaires à la génération de légendes pertinentes. En utilisant un modèle de langage entraîné sur des corpus diversifiés.

Le cœur de notre contribution réside dans la fusion de ces deux encodages, audio et textuel, dans un espace de représentation commun, appelé le joint embedding. Cette fusion permet une intégration synergique des informations audio et textuelles, facilitant ainsi la génération de légendes textuelles cohérentes et fidèles au contenu audio d'origine.

En outre, nous abordons la question de l'alignement temporel entre les segments audio et les mots correspondants dans les légendes générées. Nous proposons une méthode de synchronisation fine, basée sur des techniques de traitement du signal et de traitement du langage naturel, afin de garantir la cohérence temporelle entre les descriptions textuelles et les segments audio correspondants.

En résumé, notre contribution apporte une solution novatrice à la tâche complexe de la génération automatique de légendes textuelles pour les signaux audio. En combinant des approches d'encodage audio et textuel avancées, ainsi qu'une méthodologie robuste d'alignement temporel, notre architecture encoder-encoder se positionne comme une avancée significative dans le domaine de l'Automated Audio Captioning (AAC).

4. Organisation de mémoire

Ce mémoire est divisé en trois chapitres. Le premier chapitre est consacré à une recherche bibliographique comportant cinq parties donnent un aperçu des principes fondamentaux d'apprentissage automatique, d'apprentissage profond, du traitement du son, du traitement du langage naturel et des travaux connexes. Le deuxième chapitre décrit notre approche proposée, et l'architectures utilisées. Le dernier chapitre présente une description des outils utilisée dans notre projet. Ainsi il présente les résultats de nos travaux, analyse les résultats et discute des travaux futurs.

Chapitre 01 :

Notions Préliminaires

Chapitre 1

Notions Préliminaires

Dans ce premier chapitre, nous posons les bases nécessaires à la compréhension des principaux concepts de deep learning, du traitement audio et du traitement du texte, ainsi que des travaux connexes dans la détection et la classification des événements acoustiques.

Nous commençons par,le deep learning, mettant l'accent sur les réseaux de neurones, les CNN, les auto-encodeurs, le transformateur, le prétraitement de données dans DL.

Ensuite, nous abordons le traitement audio en examinant les fondements du son, la conversion ADC, et les caractéristiques temporelles et fréquentielles des signaux, ainsi que des outils essentiels comme le spectrogramme et la waveform. Dans la troisième partie, nous discutons du traitement du texte, couvrant les méthodes de NLP, le traitement de texte, et la vectorisation, mettant en avant les approches basées sur les Transformateurs.

Enfin, nous examinons les travaux connexes dans le domaine de la détection et de la classification des événements acoustiques, situant notre recherche dans le contexte de la littérature existante. À travers cette exploration succincte, nous posons les bases nécessaires pour une analyse approfondie de notre sujet de recherche.

Première partie

L'apprentissage profond

1.1 Introduction

Le Deep Learning est une méthode d'apprentissage automatique basée sur des réseaux de neurones profonds. Le DL révolutionne notre capacité à résoudre des problèmes complexes de reconnaissance de motifs et de traitement des données. Dans cette partie, nous abordons la notion de DL et de réseaux de neurones, les Réseaux de neurones convolutifs, les auto-encodeurs, le transformateur et le prétraitement des données spécifique au DL et ses applications dans le domaine de l'audio. En résumé, nous examinons comment le DL permet d'extraire des modèles complexes des données brutes.

1.2 l'apprentissage profond

Le deep learning, ou apprentissage profond, est un sous-ensemble d'apprentissage automatique, Machine Learning, qui prend les données en entrée et prend des décisions intuitives et intelligentes à l'aide d'un réseau de neurones artificiels, RNA, superposés et contenant des couches cachées [14]. En exploitant la structure hiérarchique de ces réseaux de neurones, le deep learning peut extraire des caractéristiques de haut niveau à partir de données brutes, ce qui le rend particulièrement puissant pour la résolution de problèmes complexes dans divers domaines.

1.3 Les réseaux de neurones

Un réseau de neurones est une collection d'algorithmes inspirés par le fonctionnement du cerveau humain (voir figure 1.1). L'objectif de cette technologie est de simuler l'activité la transmission d'informations à travers différentes couches de connexions neuronales [15].

Un réseau de neurones est composé de différentes couches, et chaque couche est constituée de neurones, qui sont les éléments fondamentaux. Chaque neurone accomplit des opérations simples qui enrichissent la complexité et la capacité de traitement globale du réseau.

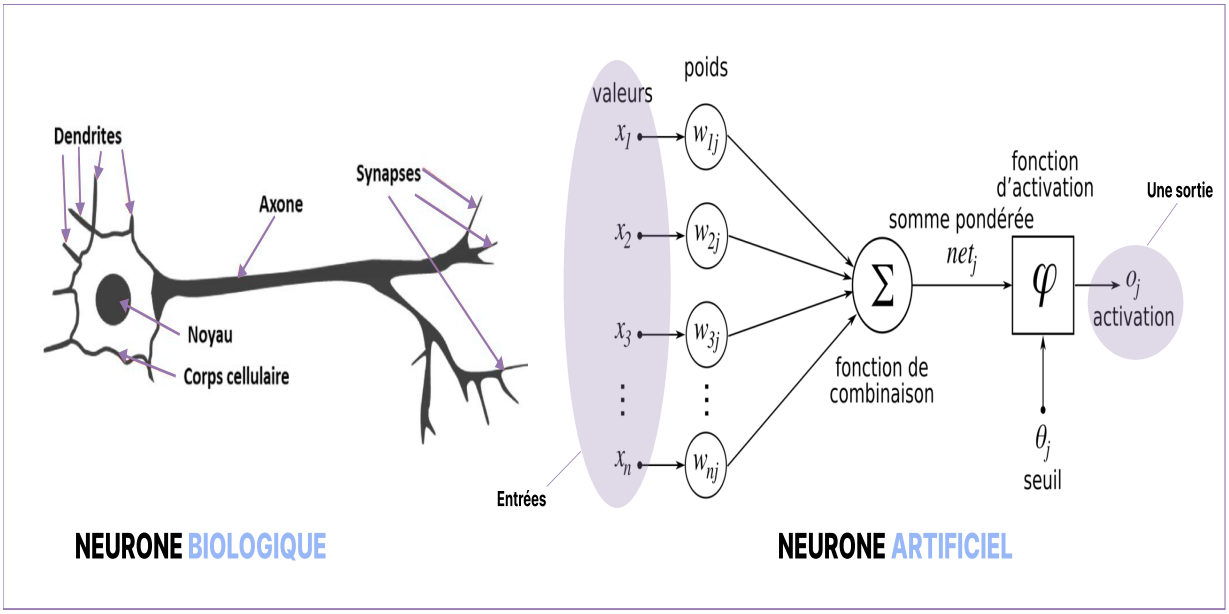


FIGURE 1.1 – Représentation mathématique/informatique d'un neurone biologique[4].

1.3.1 Les principales composantes du réseau de neurones

Le réseau de neurones est composé des composants principaux suivants :

1.3.1.1 - Neurones : ensemble de fonctions

Dans un réseau de neurones, chaque neurone est une unité qui reçoit des informations, effectue des calculs simples et les transmet à d'autres unités. On distingue trois types de neurones dans un réseau artificiel :

- Les neurones d'entrée qui reçoivent les données du monde extérieur.
- Les neurones de traitement.
- Les neurones de sortie.

Ces neurones correspondent respectivement à la couche d'entrée (*input layer*), la couche cachée (*hidden layer*) et la couche de sortie (*output layer*) (voir figure 1.2). En pratique, un réseau de neurones peut comporter plusieurs couches à chaque niveau : entrée, traitement et sortie [16].

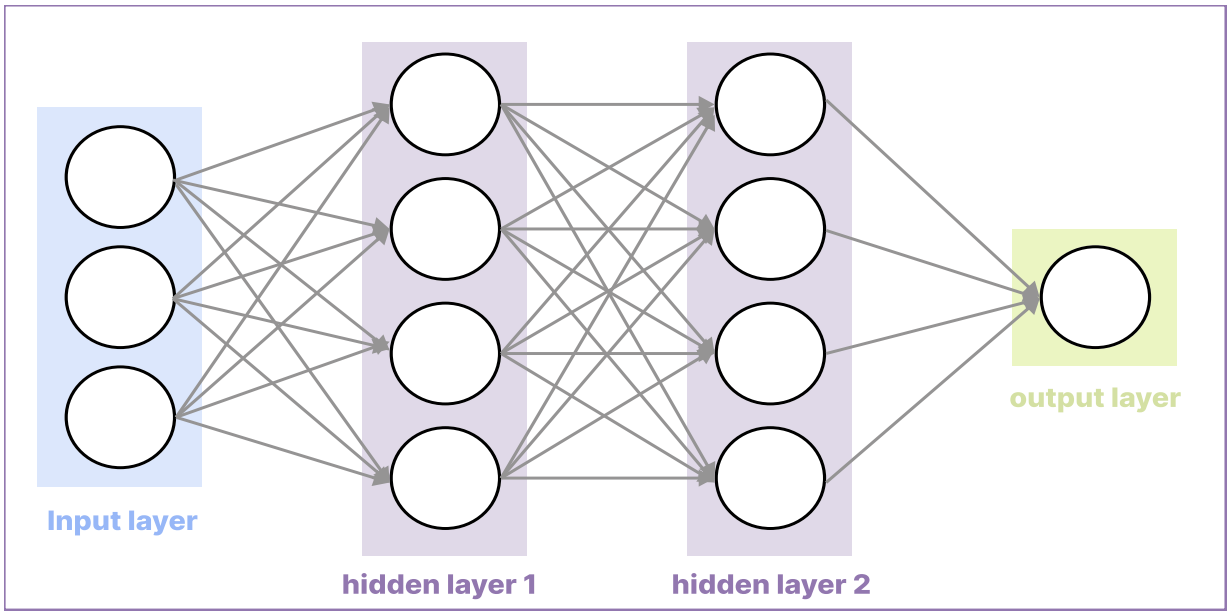


FIGURE 1.2 – Architecture de réseaux de neurones

1.3.1.2 - Les poids et les biais

Dans un réseau de neurones artificiels, les connexions entre les neurones, ne sont pas toutes équivalentes. Chaque connexion possède un poids spécifique, appelé weight en anglais, qui influence la transmission de l'information d'une unité à une autre. Les informations provenant d'une synapse avec un poids important seront ainsi dominantes à l'entrée du neurone suivant, comparativement à celles provenant de connexions à poids faible. Ces dernières ne sont cependant pas ignorées lors du traitement de l'information. Les poids sont mis à jour de façon continue par les réseaux de neurones artificiels pour améliorer la précision des sorties. Ce sont en réalité des valeurs d'auto-apprentissage.

Les biais sont des paramètres additionnels utilisés dans les réseaux de neurones pour ajuster les valeurs des entrées sur lesquelles les poids sont appliqués, avant d'obtenir les valeurs de sortie finales. Ils influencent le comportement de la fonction d'activation [16].

1.3.1.3 - La fonction d'activation

La fonction d'activation, exprimée mathématiquement, permet à chaque neurone de réguler ou de normaliser les données d'entrée qu'il reçoit avant de les transmettre ou non. Cette fonction devient active lorsque la valeur calculée dépasse le seuil requis fixé (voir figure 1.3). Dans le cas contraire, elle reste inactive et aucune donnée n'est transmise,

reproduisant ainsi le fonctionnement des neurones biologiques. Les fonctions d'activation jouent un rôle crucial dans le processus d'apprentissage et d'amélioration continue des neurones artificiels [16].

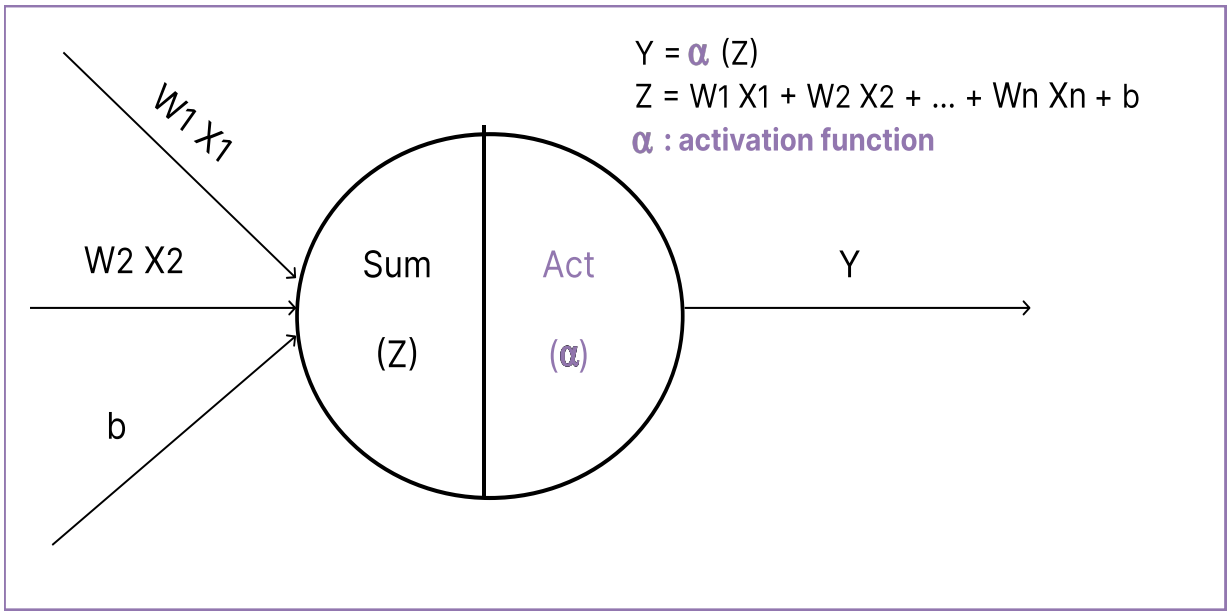


FIGURE 1.3 – Fonction d'activation[5].

Sachant que :

- X : C'est le vecteur des données d'entrée. Chaque composant de X représente une caractéristique de l'observation ou un signal entrant dans le neurone.
- W : C'est le vecteur des poids associés aux entrées. Chaque poids w_i indique l'importance de l'entrée x_i correspondante.
- b : C'est le biais. Il permet de décaler la fonction d'activation pour que le neurone puisse s'activer même si toutes les entrées sont nulles.

Plusieurs fonctions d'activation peuvent être présentes dans un réseau de neurones (voir figure 1.4) comme : La fonction sigmoïde qui réduit la sortie entre 0 et 1 en appliquant une fonction de journalisation, ce qui rend le problème de classification plus facile, L'unité linéaire rectifiée (ReLu) convertit tout ce qui est inférieur à zéro en zéro, souvent utilisée dans les couches cachées, La fonction Softmax, utilisée dans la couche de sortie, et La fonction de tangente hyperbolique $\tanh(x)$.

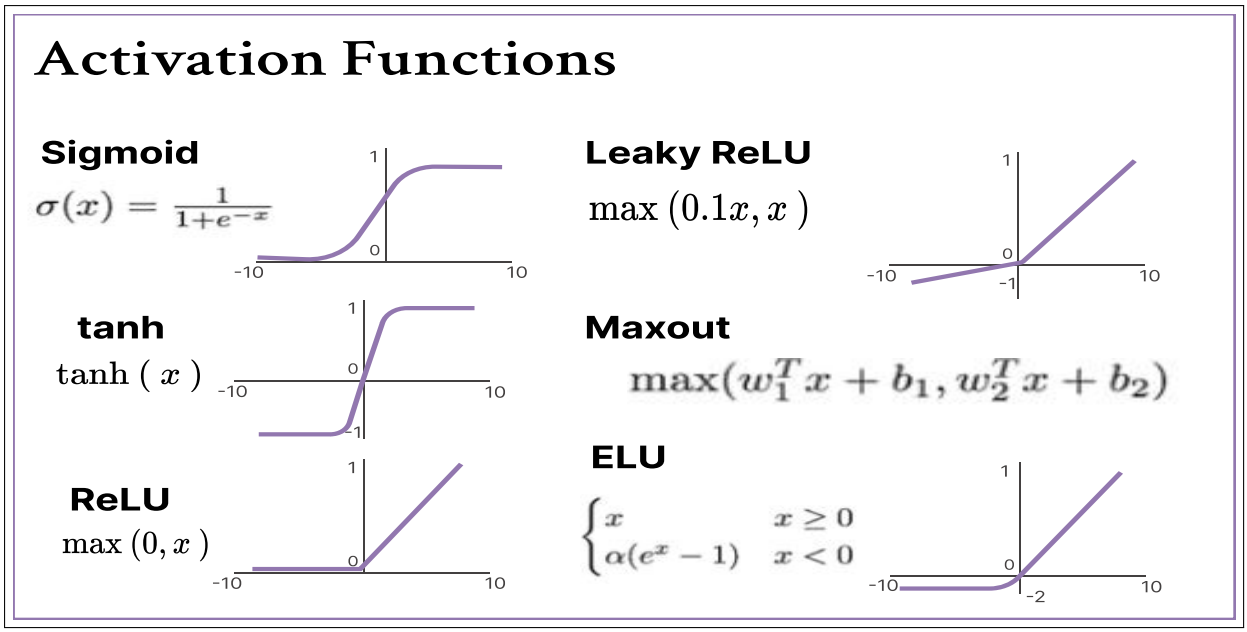


FIGURE 1.4 – Exemple de fonctions d'activation[6].

1.4 Réseaux de neurones convolutifs (CNN)

Un réseau de neurones convolutionnel (en anglais : Convolutional Neural Network), est une variante spécialisée des réseaux neuronaux artificiels. Il est constitué de multiples couches de convolution et est particulièrement efficace pour l'apprentissage automatique et les applications d'intelligence artificielle (IA), notamment dans la reconnaissance d'images et de la parole, le marketing ciblé, la vente, et bien d'autres domaines [17].

Les réseaux de neurones convolutifs sont à ce jour les modèles les plus performants pour classer des images. Souvent désignés par l'acronyme CNN, ils comportent deux parties bien distinctes. En entrée du CNN, une image est fournie sous la forme d'une matrice de pixels. Elle a 2 dimensions pour une image en niveaux de gris. La couleur est représentée par une troisième dimension, de profondeur 3 pour représenter les couleurs fondamentales [Rouge, Vert, Bleu] [18].

La première étape d'un CNN est sa partie convolutive, qui agit comme un extracteur de caractéristiques des images. L'image est soumise à une série de filtres, ou noyaux de convolution, générant ainsi de nouvelles images appelées cartes de convolutions. Certains filtres intermédiaires réduisent la résolution de l'image en utilisant une opération de maxi-

mum local. Ensuite, les cartes de convolutions sont aplaties et concaténées pour former un vecteur de caractéristiques, connu sous le nom de code CNN (voir figure 1.5).

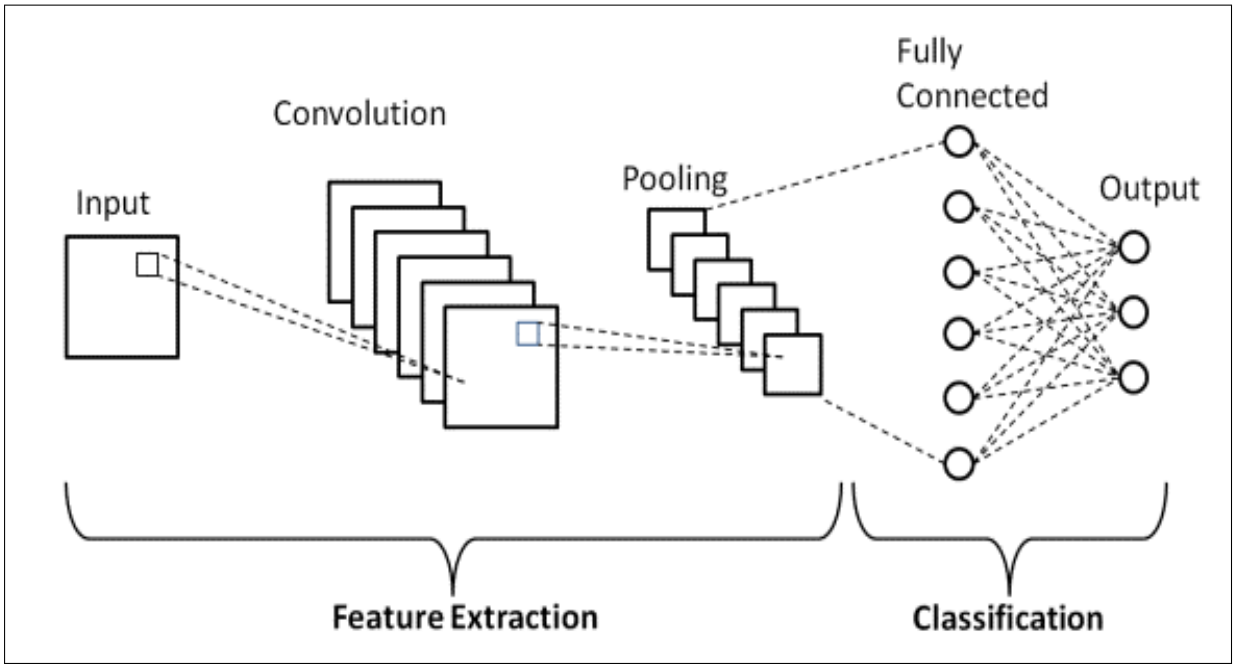


FIGURE 1.5 – Architecture générale d'un CNN[7].

Ce code CNN est ensuite utilisé comme entrée pour la deuxième partie du réseau, composée de couches entièrement connectées, également appelées perceptron multicouche. Cette partie a pour rôle de combiner les caractéristiques extraites du code CNN afin de classifier l'image. La sortie de ce processus est une dernière couche comportant un neurone par catégorie. Les valeurs numériques obtenues sont généralement normalisées entre 0 et 1, avec une somme totale de 1, pour produire une distribution de probabilité sur les différentes catégories [19].

1.4.1 Les couches de réseaux de neurones convolutionnels

✓ **Couche d'entrée (Input layer)** :

— Reçoit les données (souvent des images).

✓ **Couche de convolution (Convo layer : Convolution + ReLU)** :

— Applique des filtres pour extraire des caractéristiques, suivie d'une activation ReLU.

✓ **Couche de Pooling** :

éduit la dimension spatiale et rend la représentation plus robuste.

✓ **Couche entièrement connectée (Couche Fully connected)** :

— Combine les caractéristiques extraites.

✓ **Couche Softmax/logistique** :

— Transforme les scores en probabilités pour la classification multiclasse.

✓ **Couche de sortie (Output layer)** :

— Produit les prédictions finales (classification ou régression).

Chaque couche joue un rôle spécifique dans l'extraction et la transformation des caractéristiques des données d'entrée pour effectuer des tâches comme la classification, la détection d'objets ou la segmentation d'images.

1.5 L'auto-encodeur

En termes simples, un auto-encodeur est un type d'algorithme d'apprentissage non supervisé utilisé notamment dans le deep learning. Il est donc constitué d'un réseau neuronal profond permettant de construire une nouvelle représentation de données. Autrement dit, l'auto-encodeur permet de reconstruire une entrée à partir du code de données non étiquetées.

Par conséquent, le nombre de neurones sur la couche de sortie doit être identique à celui de la couche d'entrée. L'algorithme vise à ce que la sortie soit la plus proche de l'entrée.

Un auto-encodeur s'entraîne à extraire les parties les plus importantes d'une entrée afin de générer une sortie qui présente moins de descripteurs. En d'autres termes, le réseau ignore le bruit, généralement pour réduire la dimensionnalité de l'entrée [20].

1.5.1 Le fonctionnement d'auto-encoder

L'architecture d'un auto-encodeur se compose de deux ensembles de couches de neurones (voir figure 1.6). Le premier ensemble forme ce qui est appelé l'encodeur qui traite les données d'entrées pour construire de nouvelles représentations (code). Le deuxième ensemble, dit décodeur, tente de reconstruire les données à partir de ce code.

De ce fait, la performance d'un auto-encodeur se mesure par les différences entre les données d'entrées et les données de sortie. Par ailleurs, pour entraîner l'algorithme, ses paramètres sont modifiés de manière à réduire l'erreur de reconstruction [20].

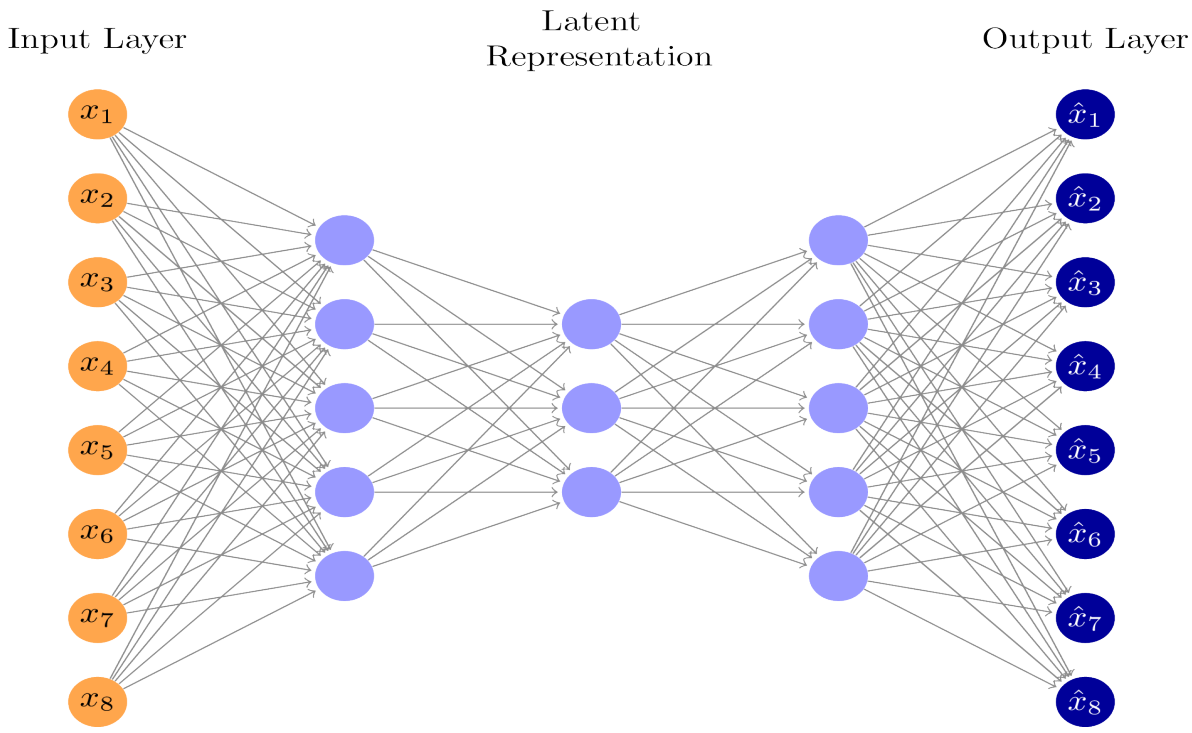


FIGURE 1.6 – Architecture d'un Auto-encodeur[8].

1.5.1.1 - L'encodeur

Dans le domaine des réseaux de neurones et des auto-encodeurs, un encodeur est un composant fondamental qui transformer les données d'entrée en une représentation compacte de dimension inférieure (Voir figure 1.7).

"L'encodeur est responsable de mapper les données d'entrée à une représentation cachée, souvent appelée espace latent, qui capture les caractéristiques essentielles des données d'origine. Cette transformation vise à réduire la dimensionnalité tout en préservant les informations pertinentes, facilitant ainsi des tâches telles que la compression de données et l'extraction de caractéristiques." (Goodfellow, Bengio et Courville, 2016) [21].

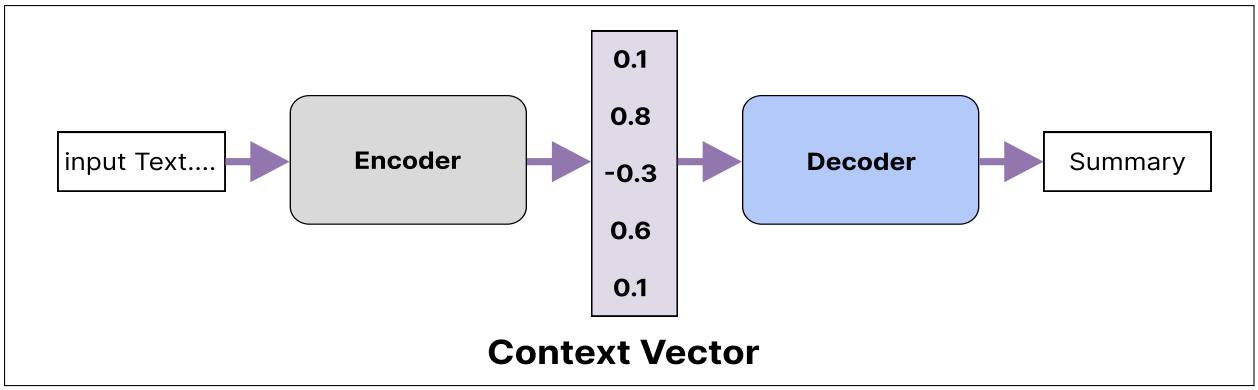


FIGURE 1.7 – Architecture d'encodeur et décodeur[9]

.

1.5.1.2 - Le décodeur

À l'inverse de l'encodeur, le décodeur décompresse l'espace latent pour reconstituer les données. Son défi consiste à utiliser les caractéristiques contenues dans le vecteur condensé pour essayer de reconstruire le plus fidèlement possible le jeu de données [20].

1.6 Le Transformateur

Le Transformateur est une architecture de réseau de neurones introduite par Vaswani et al. En 2017, principalement utilisée pour des tâches de traitement du langage naturel (NLP). Il est basé sur des mécanismes d'attention pour modéliser les dépendances entre les éléments d'une séquence d'entrée et d'une séquence de sortie sans utiliser de structures récurrentes ou convolutionnelles traditionnelles. Le Transformateur permet à la machine d'apprendre des séquences de manière automatique, sans avoir été programmée spécifiquement à cet effet.

"Le Transformateur, une architecture de modèle qui évite la récurrence et repose entièrement sur un mécanisme d'attention pour établir des dépendances globales entre l'entrée et la sortie. Il est particulièrement bien adapté à l'entraînement sur des ensembles de données très volumineux et à la mise en œuvre sur des accélérateurs matériels spécialisés." (Vaswani et al., 2017)[22].

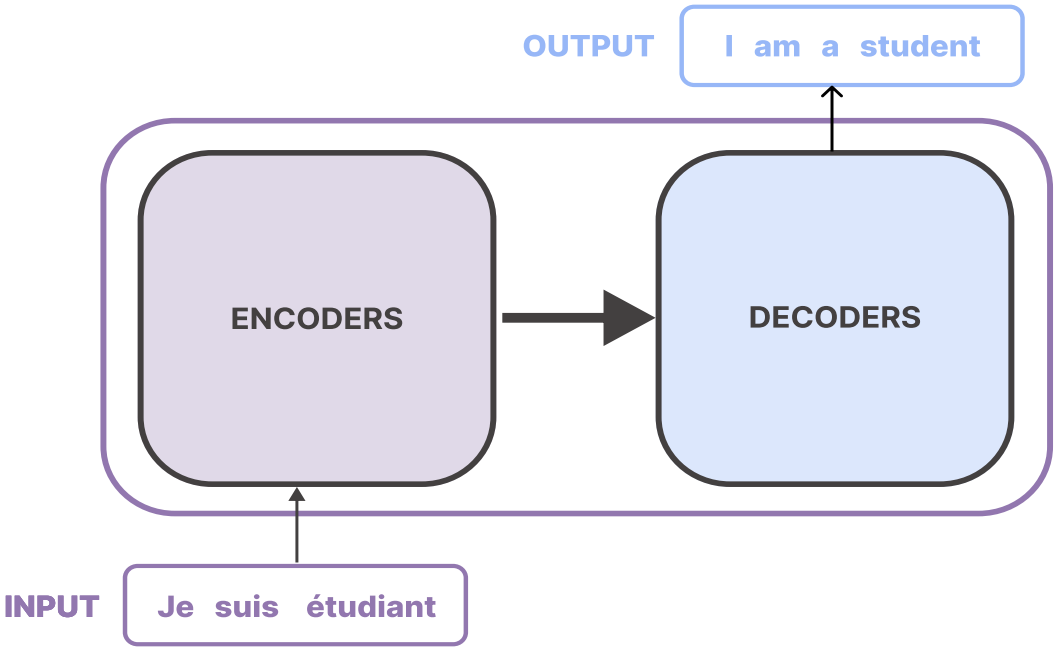


FIGURE 1.8 – Architecteur de Transformateur[10].

1.6.1 Les Types de Transformateur

Il existe différents types de transformateur, notamment :

- **Transformateurs d'encodeur :** Ces modèles se composent uniquement d'un encodeur, qui transforme les données d'entrée en une représentation dans un espace latent. Ils sont principalement utilisés dans des domaines comme la modélisation du langage pour prédire le mot suivant dans une séquence.
- **Transformateurs à décodeur uniquement :** Ces modèles possèdent uniquement un décodeur, responsable de générer des données de sortie en se basant sur une entrée et une représentation apprise. Ils sont fréquemment employés dans des tâches de traduction, visant à produire une phrase dans une langue cible à partir d'une phrase source.
- **Transformateurs d'encodeur-décodeur :** Ces modèles intègrent à la fois un encodeur et un décodeur. Ils sont adaptés aux tâches de séquence à séquence comme la traduction automatique, la synthèse de texte, et les agents conversationnels.
- **Transformateurs de vision (ViT) :** Il s'agit d'une adaptation de l'architecture des transformateurs spécifiquement conçue pour la vision par ordinateur. Les ViT sont utilisés dans des applications telles que la classification d'images, la détection

d'objets et la segmentation d'images [23].

1.7 Le prétraitement de données pour le Deep Learning

Le prétraitement des données est une première étape courante du workflow de Deep Learning pour préparer les données brutes dans un format acceptable par le réseau. Par exemple, nous pouvons redimensionner un fichier audio en entrée pour correspondre à la taille de la couche d'entrée pour les données audio. Nous pouvons également prétraiter des données pour améliorer les caractéristiques souhaitées ou réduire les artefacts qui peuvent biaiser le réseau. Par exemple, nous pouvons normaliser le volume ou supprimer le bruit des données audio d'entrée [15].

Deuxième partie

Le traitement d'audio

1.1 Introduction

Avec l'avènement des médias numériques, l'audio est devenu essentiel dans notre quotidien, allant de la musique aux communications en ligne. La compréhension et le traitement de l'audio sont cruciaux pour améliorer la qualité sonore et extraire des informations utiles. Parmi les avancées notables, l'Automated Audio Captioning (AAC) facilite la création de sous-titres automatiques pour l'accessibilité.

Pour bien comprendre l'audio, il faut connaître les bases du son et la conversion analogique-numérique (ADC). Les caractéristiques d'audio, tant temporelles que fréquentielles, offrent des perspectives pour l'analyse des signaux. Enfin, des outils comme le spectogramme et la waveform permettent une visualisation claire du son.

Dans cette partie, nous aborderons ces aspects du traitement de l'audio, en explorant les techniques et notions clés.

1.2 Le Son

Le son est une manifestation physique résultant de vibrations d'un objet, entraînant l'oscillation et la collision des molécules d'air environnantes. Ces interactions engendrent des variations de pression dans l'air, formant des ondes sonores qui se propagent à travers l'espace. Les ondes sonores sont des ondes mécaniques, nécessitant un milieu matériel comme l'air pour se propager. Graphiquement, les formes d'onde représentent ces variations de pression au fil du temps, offrant des informations sur la fréquence, l'intensité et la durée du son.

Ainsi, le son est une manifestation de l'énergie mécanique transférée à travers un milieu matériel par le biais de vibrations et d'ondes sonores.

1.2.1 L'onde sonore

La manière dont le son se propage dans le milieu acoustique. L'onde est la propagation d'une perturbation dans un milieu matériel. Elle transporte de l'énergie, mais pas de matière.

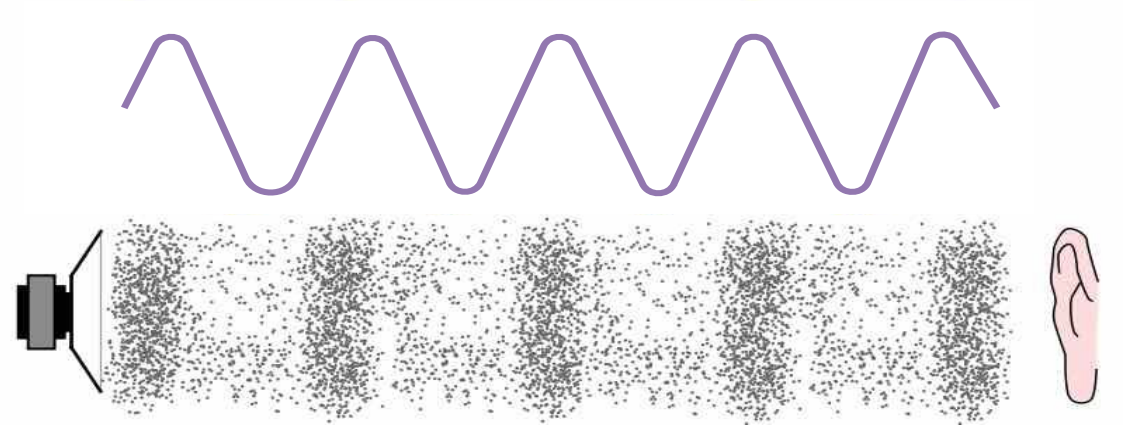
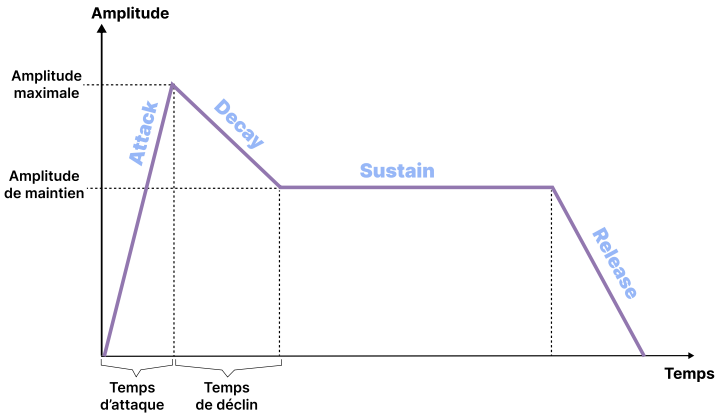


FIGURE 1.9 – Ondes sonores

1.2.2 Les Paramètres du Son

Type de paramètres	**Paramètres**	**Description**
Paramètres temporels	La Durée	Elle dépend du temps pendant lequel le milieu est perturbé ou c'est l'intervalle de temps pendant lequel le son est perçu. On la mesure généralement en secondes (s).
	L'attaque	C'est le temps de montée, la partie initiale transitoire du son avant d'atteindre son régime permanent. Une attaque rapide donne une impression de sécheresse.
	La décroissance ou déclin	C'est le temps que met le son pour décroître après l'extinction de la source ou c'est la manière dont un son diminue en intensité après son pic initial. Une décroissance lente donne une impression de réverbération.

<table>
<tr><td></td><td>Le temps d'enveloppe</td><td>Ou enveloppe ADSR, décrit la façon dont le volume d'un son évolue au fil du temps. Il se divise en quatre phases :
Attaque : C'est le temps que prend le son pour atteindre son amplitude maximale après le début.
Décroissance : C'est quand l'amplitude commence à diminuer après le maintien.
Maintien : C'est la partie où l'amplitude reste relativement constante.
Relâchement : Le son diminue progressivement jusqu'à retourner au silence. 

Ces phases, contrôlées par des paramètres de temps, influencent le caractère et la texture du son.</td></tr>
<tr><td rowspan="2">Paramètres fréquentiels</td><td>Fréquence fondamentale (f0)</td><td>C'est la fréquence la plus basse d'un son périodique. Elle détermine la hauteur perçue du son.</td></tr>
<tr><td>Spectre/Contenu fréquentiel</td><td>C'est la décomposition du son en ses différentes composantes fréquentielles et leurs amplitudes relatives.</td></tr>
</table>

	Bande passante	C'est la plage de fréquences sur laquelle le son a une amplitude significative ou c'est la plage de fréquences sur laquelle un dispositif ou un système peut fonctionner efficacement. Elle détermine la richesse spectrale du son.
Paramètres d'amplitude	Amplitude	C'est la valeur maximale de la variation de pression (ou de déplacement) par rapport à la pression (ou position) moyenne.
	Niveau sonore (intensité)	C'est la quantité d'énergie acoustique transportée par unité de surface et de temps. Le niveau sonore perçu est lié à la sensation de « volume » du son. Elle s'exprime en décibels (dB).
	Dynamique	C'est la différence entre les niveaux d'amplitude les plus bas et les plus élevés dans un son. Elle présente l'étendue des variations d'amplitude dans un signal sonore.
Paramètres perceptuels	Hauteur (Pitch)	C'est la perception de la fréquence d'un son. Elle détermine si un son est grave ou aigu (Une fréquence plus élevée est perçue comme un son plus aigu, tandis qu'une fréquence plus basse est perçue comme un son plus grave).
	Loudness (Intensité sonore)	C'est la perception de l'intensité ou de l'amplitude d'un son. Elle dépend de l'amplitude, mais aussi d'autres facteurs tels que la fréquence et est généralement mesurée en décibels (dB).
	Timbre	C'est la qualité tonale ou sonore qui permet de distinguer différents instruments ou voix, même lorsqu'ils jouent la même note à la même hauteur et avec la même intensité. Le timbre est déterminé par la forme d'onde du son et la distribution de ses harmoniques.

Paramètres spatiaux	Localisation spatiale	C'est la capacité à localiser la source d'un son dans l'espace, que ce soit en termes de direction (gauche/droite) ou de distance.
	Direction	C'est la perception de la provenance du son dans l'espace horizontal (azimut) et vertical (élévation).
	Distance	C'est la perception de l'éloignement du son. Elle dépend de l'atténuation de l'intensité et des indices monauraux comme la réverbération.
Paramètres dynamiques	Modulation d'amplitude	C'est la variation cyclique de l'amplitude du son, comme un tremblement. Elle peut créer des effets expressifs comme le vibrato ou le trémolo.
	Dynamique Enveloppe dynamique	C'est la courbe d'évolution de l'amplitude du son au cours du temps. Elle définit le profil dynamique caractéristique d'un fils.
	Transitoires	Ce sont les variations rapides d'amplitude au début et à la fin du son.

TABLE 1.1 – Les paramètres du Son

1.3 Conversion analogique-numérique (ADC)

Le monde physique est par nature analogique. Il est perçu via des signaux analogiques, le traitement numérique des données prend le pas sur les approches purement analogiques. Le recours au numérique permet en effet un stockage aisé de l'information, une excellente reproductibilité des traitements, la possibilité de développer relativement aisément des fonctionnalités complexes, une réduction des coûts de production, etc.

L'interface nécessaire entre le monde analogique et un traitement numérique donné est réalisé par des convertisseurs analogique – numérique (CAN, ou ADC pour Analog to Digital Converter en anglais) et numérique – analogique
Un convertisseur analogique – numérique (ADC) est un dispositif électronique permettant la conversion d'un signal analogique en un signal numérique (Voir figure 1.10).

- **Signal analogique :** signal continu en temps et en amplitude.
- **Signal numérique :** signal échantillonné et quantifié, discret en temps et en amplitude.

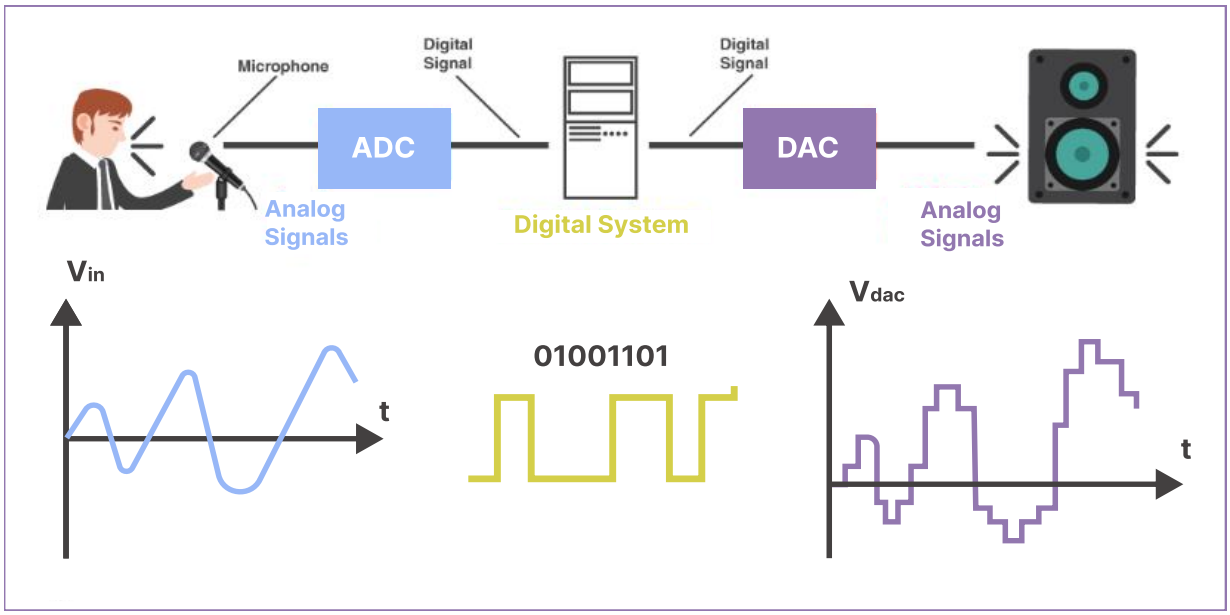


FIGURE 1.10 – Système de traitement d'audio[11].

Conceptuellement, la conversion analogique – numérique peut être divisée en trois étapes : l'échantillonnage temporel, la quantification et le codage. (Voir figure 1.11)

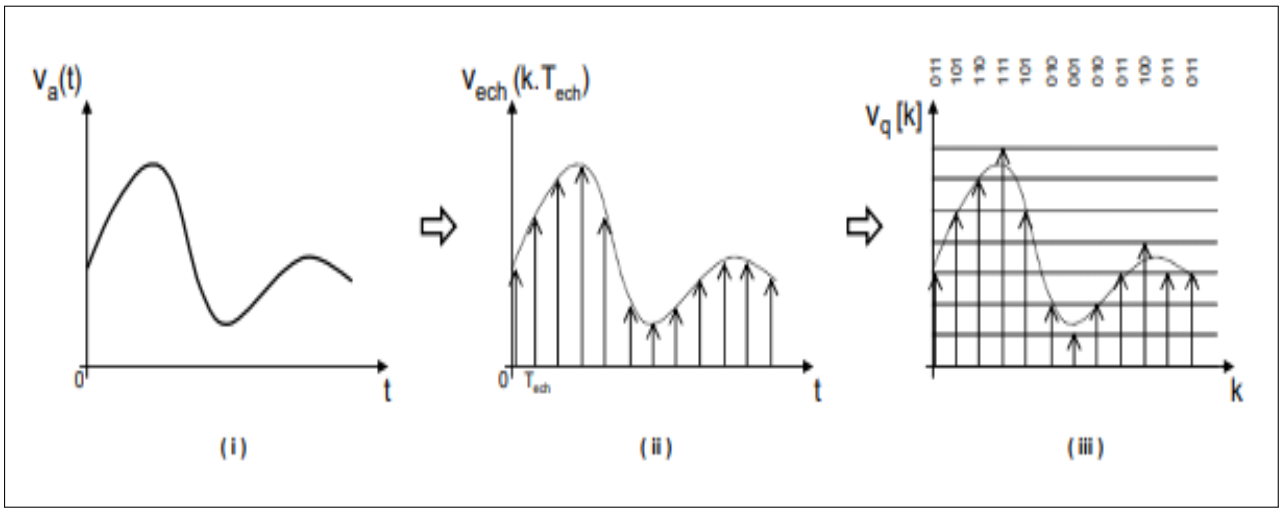


FIGURE 1.11 – (i) signal analogique (ii) signal échantillonné (iii) puis quantifié[12].

1. **L'échantillonnage temporel** : consiste à prendre des instantanés périodiques du signal audio à une certaine fréquence, appelée fréquence d'échantillonnage.

2. **La quantification** : La quantification convertit chaque échantillon analogique en une valeur numérique, en attribuant une valeur numérique discrète à chaque niveau de l'échantillon.
3. **Le Codage** : Le codage convertit les valeurs numériques quantifiées en un format binaire (0s et 1s) pour être traitées, stockées ou transmises numériquement.

1.4 Fonctionnalités d'audio

On distingue deux fonctionnalités principales, fonctionnalités du domaine temporel et fonctionnalités du domaine fréquentiel. Les fonctionnalités du domaine temporel se concentrent sur l'évolution du signal dans le temps, tandis que les fonctionnalités du domaine fréquentiel se concentrent sur les composantes de fréquence du signal. Chacune de ces catégories de fonctionnalités a ses propres avantages et peut être utilisée pour différents types de tâches d'analyse ou de traitement du signal audio.

1.4.1 Fonctionnalités du domaine temporel

Les fonctionnalités du domaine temporel sont basées sur l'évolution du signal sonore dans le temps. Elles capturent des informations sur la variation du signal à différents moments.
Exemples de fonctionnalités du domaine temporel :

- **Énergie du signal** : L'énergie du signal audio représente la quantité d'énergie présente dans le signal sur une période définie. Une énergie élevée correspond à un son puissant, tandis qu'une énergie faible indique un son plus doux. L'énergie est un indicateur clé qui reflète le volume sonore et l'amplitude du signal. Elle est employée dans de nombreux algorithmes de traitement audio, tels que la détection de transitoires et la compression.
- **ZCR (Zero Crossing Rate)** : Le ZCR désigne le taux auquel un signal audio numérique traverse la ligne zéro, passant d'une polarité positive à négative ou vice versa. Cette mesure met en évidence la présence de hautes fréquences dans le signal. Un ZCR élevé signifie un signal comportant beaucoup de hautes fréquences, alors qu'un ZCR bas suggère une prédominance de basses fréquences. Il est largement utilisé dans la reconnaissance vocale et le traitement musical.

- **RMS (Root Mean Square)** : Le RMS est une mesure statistique qui représente la valeur efficace du signal audio sur une période spécifique. Il se calcule en prenant la racine carrée de la moyenne des carrés des valeurs d'échantillons du signal. Cette valeur RMS donne une indication de la puissance ou de l'énergie moyenne du signal et est directement liée à la perception du volume sonore. Un signal ayant un RMS plus élevé sera perçu comme plus fort.

1.4.2 Fonctionnalités du domaine fréquentiel

Les fonctionnalités du domaine fréquentiel représentent les caractéristiques du signal en termes de composantes de fréquence. Elles sont généralement dérivées de la transformation de Fourier ou d'autres techniques similaires qui décomposent le signal en ses composantes fréquentielles.

Exemples de fonctionnalités du domaine fréquentiel :

- **La transformée de Fourier (TF)** : La transformée de Fourier est un outil mathématique qui joue un rôle central dans le traitement du domaine fréquentiel. Il nous permet de convertir un signal du domaine temporel vers le domaine fréquentiel, révélant les composantes fréquentielles sous-jacentes qui composent le signal. En décomposant un signal en composantes fréquentielles, nous pouvons analyser et manipuler des fréquences ou des plages de fréquences spécifiques, offrant ainsi une compréhension plus approfondie des caractéristiques du signal. Elle est définie comme suit :

$$X(f) = \int_{-\infty}^{\infty} x(t) e^{-j2\pi ft}\, dt$$

Pour un signal temporel $x(t)$, nous pouvons obtenir sa version dans le domaine fréquentiel $X(f)$ en utilisant cette formule . Nous représenterons la version temporelle d'une fonction par $x(t)$ ou $y(t)$, et la version fréquentielle correspondante par $X(f)$ et $Y(f)$. Notez qu'on utilise la variable "t" pour le temps, et "f" pour la fréquence. Le "j" est simplement l'unité imaginaire pure.

Pour revenir au domaine temporel à partir de la fréquence, c'est presque la même chose.

$$x(t) = \frac{1}{2\pi} \int X(f) e^{j2\pi ft} df$$

De nombreux manuels utilisent plutôt ω à la place de $2\pi f$. ω est la fréquence angulaire en radians par seconde, alors que f est en Hz.

$$\omega = 2\pi f$$

- **Transformée de Fourier à Court Terme (TFCT)** : La STFT pour ("Short-Time Fourier Transform" en anglais) est une technique utilisée en traitement du signal et en analyse spectrale pour étudier les variations de fréquence d'un signal au fil du temps. Elle est particulièrement utile pour analyser des signaux qui ne sont pas stationnaires, c'est-à-dire des signaux dont les propriétés statistiques changent avec le temps.

  Mathématiquement, la STFT d'un signal $x(t)$ est donnée par la formule :

$$\text{STFT}x(t)(t,\omega) = \int_{-\infty}^{\infty} x(\tau) \cdot w(t-\tau) \cdot e^{-i\omega\tau} d\tau$$

- **La Transformée de Fourier Discrète (DFT)** : est une technique mathématique utilisée en traitement du signal et en analyse spectrale pour convertir un signal temporel en son équivalent fréquentiel. Elle est la version discrète de la Transformée de Fourier continue (FT), adaptée pour les signaux numériques.

  Mathématiquement, la DFT d'un signal discret $x[n]$ de longueur N est définie comme suit :

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi}{N}kn}$$

  Pour calculer la DFT, on utilise généralement l'algorithme de la transformée rapide de Fourier (FFT) pour réduire le temps de calcul.

- **Le rapport d'énergie de bande (BER)** : Le BER est une mesure utilisée pour évaluer la distribution de l'énergie dans différentes bandes de fréquences d'un signal. Il s'agit généralement du rapport de l'énergie dans une bande de fréquence spécifique par rapport à l'énergie totale du signal. Cela peut être utilisé pour caractériser les propriétés spectrales d'un signal. Le BER pour une bande de fréquence k est calculé comme suit :

$$\text{BER}k = \frac{E_k}{\sum_{i=1}^{N} E_i}$$

- **Le Spectral Centroid (SC)** : Le Spectral Centroid est une mesure qui représente le centre de gravité spectral d'un signal. Mathématiquement, il s'agit de la moyenne

pondérée des fréquences contenues dans le signal, où chaque fréquence est pondérée par son amplitude ou son énergie. Un Spectral Centroid élevé indique que la majorité de l'énergie du signal est concentrée à des fréquences élevées, tandis qu'un Spectral Centroid bas indique une concentration d'énergie à des fréquences plus basses.
Le Spectral Centroid SC est calculé en utilisant la formule suivante :

$$SC = \frac{\sum_{f=0}^{f_{max}} f \cdot S(f)}{\sum_{f=0}^{f_{max}} S(f)}$$

- **Mel-Frequency Cepstral Coefficients (MFCC)** : Le système auditif humain est supposé traiter les signaux vocaux de manière non linéaire. Il est bien connu que les composantes de basse fréquence du signal vocal contiennent plus d'informations. C'est pourquoi un filtre non linéaire à échelle de Mel a été conçu pour mettre l'accent sur les composantes de basse fréquence plutôt que sur les composantes de haute fréquence. Le cepstrum de fréquence Mel est une représentation du spectre de puissance à court terme d'une trame de parole à l'aide d'une transformation cosinus linéaire du logarithme du spectre de puissance sur une échelle de fréquence Mel non linéaire. La conversion de la fréquence normale f en fréquence mél m est donnée par l'équation :

$$m = 2595 \log_{10} \left(\frac{f}{700} + 1 \right)$$

1.5 La Visualisation du son

La visualisation du son est un aspect essentiel pour analyser et comprendre les caractéristiques d'un signal audio. Deux des représentations les plus couramment utilisées pour visualiser le son sont et la forme d'onde et le spectrogramme.

1.5.1 La forme d'onde d'un signale audio

La forme d'onde(waveForm) est une représentation du son permettant de voir les variations d'intensité du signal réparti dans le temps, de gauche à droite. Cette visualisation permet de distinguer l'enveloppe dynamique du fichier sonore (sa forme générale) et de s'y repérer. Plus le son est fort, plus l'amplitude de l'onde est grande [24].

La forme d'onde est généralement obtenue en enregistrant les échantillons du signal à des intervalles réguliers. Chaque échantillon représente la valeur du signal à un instant précis (voir figure 1.12). En traçant ces échantillons dans un graphique, on obtient la forme d'onde du signal.

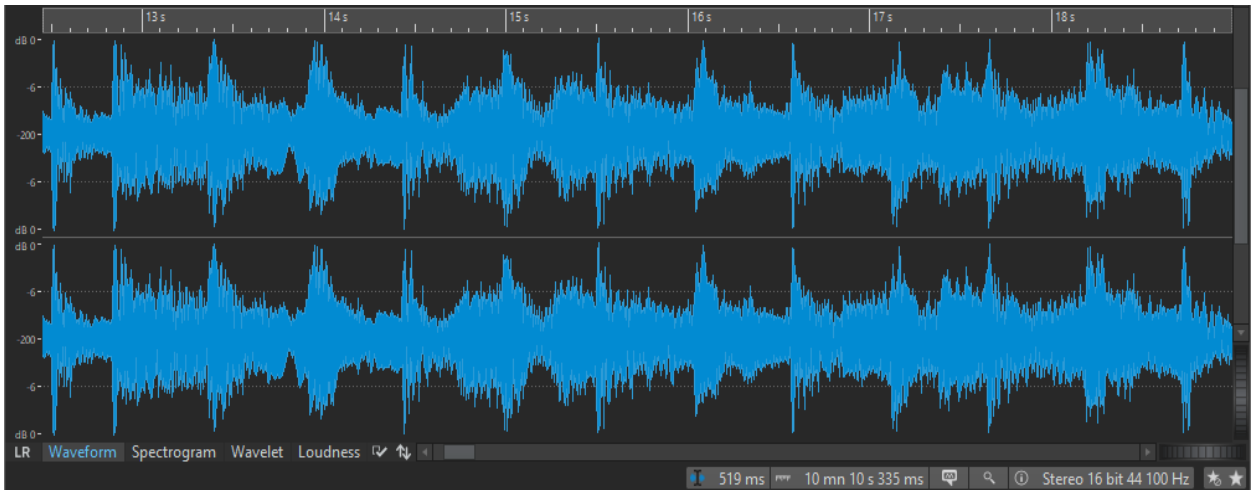


FIGURE 1.12 – Exemple de formes d'ondes d'un signale audio.

1.5.2 Le Spectrogramme

Le spectrogramme est une représentation graphique du contenu en fréquence d'un signal audio au fil du temps. Il permet de visualiser simultanément le temps, la fréquence et l'amplitude sur un même graphique.

Le spectrogramme utilise une gamme de couleurs pour indiquer l'amplitude de chaque fréquence, où les couleurs plus claires représentent une énergie de signal plus élevée (voir figure 1.13). Il est créé en analysant de courts segments du signal audio, généralement quelques millisecondes, et en calculant la transformée de Fourier discrète de chaque segment pour obtenir son spectre de fréquences. Chaque bande verticale dans le spectrogramme représente le spectre du signal à un moment précis dans le temps, illustrant comment l'énergie du signal est répartie dans les différentes fréquences présentes à ce moment-là.

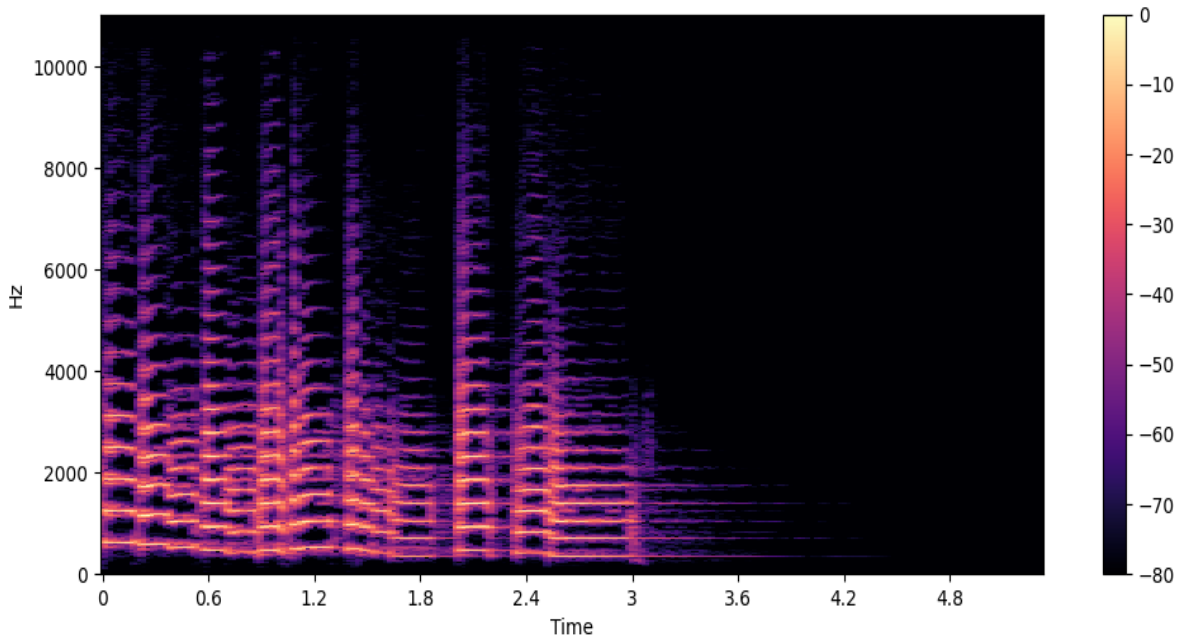


FIGURE 1.13 – Exemple d'un spectrogramme

1.5.2.1 - Mel-Scale

L'échelle Mel est une échelle perceptuelle des hauteurs tonales, fondée sur la sensibilité auditive humaine aux différentes fréquences . Elle est largement utilisée dans les applications de traitement de la parole et du son pour modéliser plus fidèlement le système auditif humain.

Cette échelle représente les hauteurs de manière à ce que les auditeurs perçoivent des intervalles égaux comme étant à des distances égales les unes des autres. Le point de référence est défini en attribuant une hauteur de 1000 mels à un son de 1000 Hz, 40 dB au-dessus du seuil auditif de l'auditeur. En dessous de 500 Hz, les échelles Mel et Hertz sont pratiquement identiques, tandis qu'au-dessus, les intervalles perçus comme égaux en hauteur deviennent de plus en plus grands [25].

1.5.2.2 - Spectogramme Mel

Un spectrogramme Mel est un spectrogramme dont les fréquences sont converties en échelle Mel. Il est similaire à un spectrogramme en ce sens qu'il montre le contenu en fréquence d'un signal audio au fil du temps (voir figure 1.14), mais sur un axe de fréquence

différent. L'échelle mel est une échelle perceptuelle qui se rapproche de la réponse en fréquence non linéaire de l'oreille humaine. Pour créer un spectrogramme mel, le STFT est utilisé comme auparavant, divisant l'audio en segments courts pour obtenir une séquence de spectres de fréquence. De plus, chaque spectre est envoyé à travers un ensemble de filtres, appelé mel filterbank, pour transformer les fréquences à l'échelle mel [26].

L'utilisation de l'échelle Mel permet de capturée les caractéristiques essentielles de l'audio tout en filtrant les détails superflus, ce qui permet de créer des modèles plus efficaces et moins gourmands en calcul.

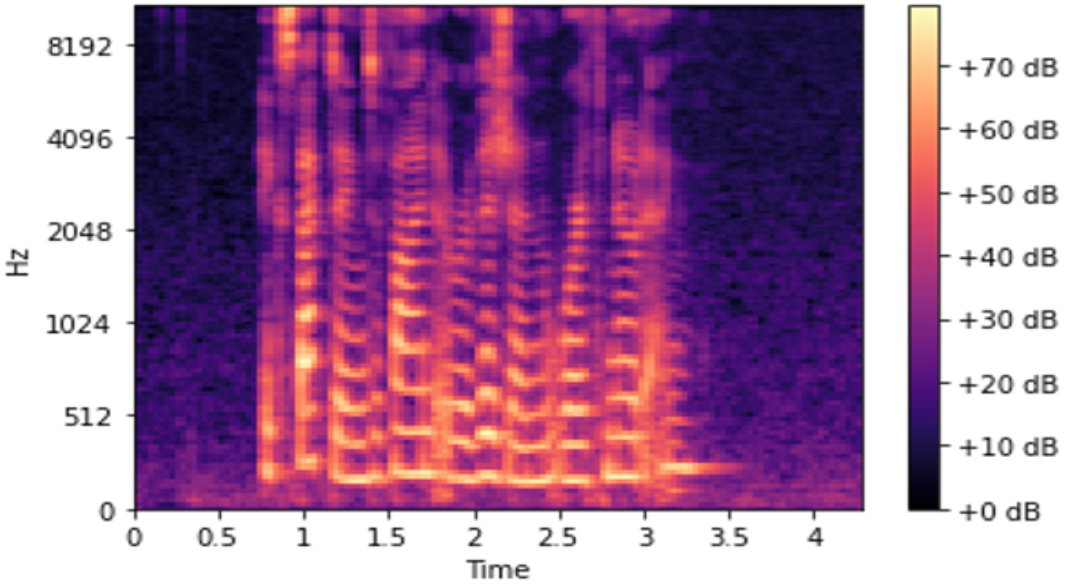


FIGURE 1.14 – Mel Spectogramme[13].

Troisième partie

Traitement de texte

1.1 Introduction

Le traitement du langage naturel (NLP) est un domaine de l'IA visant à permettre aux machines d'analyser et générer des langues humaines. Un aspect clé est le traitement de texte, qui consiste à appliquer diverses techniques pour extraire des informations utiles à partir de données textuelles brutes. Parmi ces techniques, on trouve la vectorisation qui représente le texte sous forme numérique pour les modèles d'apprentissage automatique.

Cette partie couvrira en détail les différentes méthodes de NLP, de traitement de texte et de vectorisation, en mettant l'accent sur les approches Transformateurs.

1.2 Le traitement du langage naturel (NLP)

Le NLP est un domaine interdisciplinaire de l'informatique qui englobe plusieurs domaines d'étude. Il s'appuie sur la linguistique, l'intelligence artificielle, l'apprentissage automatique, les mathématiques, la robotique, et bien d'autres disciplines pour ses applications variées [27]. Cette technologie a pour but de permettre aux machines de lire, d'analyser, de comprendre et d'interpréter le langage humain afin de lui donner du sens[28].

En utilisant la vectorisation de texte, les outils NLP transforment le texte en une forme compréhensible par une machine. Ensuite, les algorithmes d'apprentissage automatique utilisent des données d'entraînement et des sorties attendues pour apprendre à établir des associations entre une entrée donnée et sa sortie correspondante.

Après cela, des méthodes d'analyse statistique sont employées pour déterminer quelles caractéristiques du texte sont les plus pertinentes. Ces caractéristiques sont ensuite utilisées pour faire des prédictions sur de nouvelles données.

En fin de compte, plus on alimente ces algorithmes de NLP avec des données, plus les modèles d'analyse de texte seront précis.

1.2.1 Les techniques de traitement de texte

Les techniques de traitement de texte en traitement automatique du langage naturel (NLP) sont variées et continuent d'évoluer avec le temps. Voici quelques-unes des techniques les plus utilisées :

- **Nettoyage :** Variable selon la source des données, cette phase consiste à réaliser des tâches telles que la suppression d'urls, d'emoji, etc.
- **La tokenisation :** est le processus de décomposition d'un texte en éléments plus petits, appelés "tokens". Ces tokens peuvent être des mots individuels, des phrases, des symboles de ponctuation ou même des caractères.
- **Lemmatisation :** il s'agit d'une tâche similaire à la racinisation, mais avec une analyse plus approfondie du vocabulaire et de la structure des mots. La lemmatisation vise à éliminer seulement les terminaisons flexionnelles pour isoler la forme canonique du mot, appelée lemme. [29].
- **Stemming :** Il est défini comme le processus qui produit des variantes d'un mot racine/de base. En d'autres termes, il réduit un mot de base à son mot souche. Nous utilisons le stemming pour raccourcir la recherche et normaliser les phrases pour une meilleure compréhension [30].
- D'autres opérations peuvent inclure la suppression des chiffres, de la ponctuation, des symboles et des mots vides (stopwords), ainsi que la mise en minuscule du texte.

1.2.2 Les techniques de vectorisation dans le NLP

La vectorisation consiste à transformer des données textuelles en une forme numérique. Cette étape de transformation est essentielle pour convertir un format non structuré de texte en une représentation structurée, rendant l'analyse et la manipulation des données plus simples et plus efficaces pour les algorithmes d'apprentissage automatique.
Parmi les techniques de vectorisation les plus utiles :

- **TF-IDF :** Abréviation de Term Fequency-Inverse Document Frequency, est une mesure numérique utilisée pour évaluer l'importance d'un mot dans un document en fonction de sa fréquence d'apparition dans ce document et dans une collection donnée de documents (voir figure 1.15). L'intuition de cette mesure est la suivante :

Si un mot apparaît fréquemment dans un document, il doit être important et nous devons lui attribuer un score élevé. En revanche, si un mot apparaît dans un trop grand nombre d'autres documents, il ne s'agit probablement pas d'un identifiant unique, et nous devrions donc lui attribuer un score moins élevé [31].

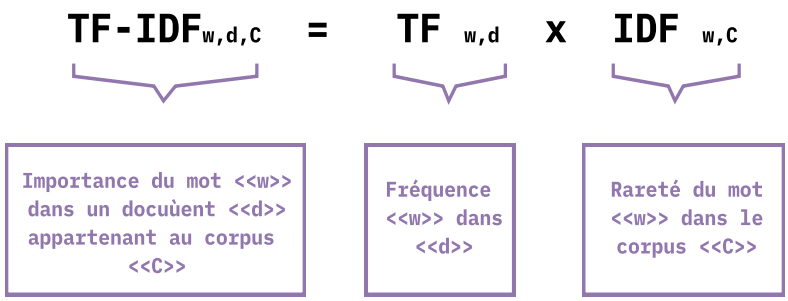


FIGURE 1.15 – La formule mathématique de TF-IDF

- **n-grammes :** consiste à représenter un texte sous forme de vecteur en considérant les séquences de n mots consécutifs présentes dans celui-ci. Chaque séquence de n mots unique devient une dimension du vecteur, dont la valeur correspond au nombre d'occurrences de cette séquence dans le texte.
- **BOW (Bag-of-Words) :** Le sac de mots (BoW) est une stratégie de traitement du langage naturel permettant de convertir un document textuel en nombres utilisables par un programme informatique. Cette méthode consiste à convertir un texte en un vecteur basé sur la fréquence des mots dans le texte, sans tenir compte de l'ordre ou du contexte des mots [32].
- **One-Hot Encoding :** L'encodage à une chaleur est une technique courante dans l'apprentissage automatique, en particulier lorsqu'il s'agit de variables catégorielles. Elle consiste à représenter chaque catégorie sous la forme d'un vecteur binaire. Dans ce processus, un vecteur binaire est créé pour chaque catégorie unique, tous les éléments étant mis à zéro sauf celui correspondant à la catégorie d'une observation donnée, qui est mis à un. [33].
- **Word Embedding :** est une méthode de représentation des mots et des textes. Il s'agit d'un vecteur numérique qui symbolise un mot dans un espace vectoriel de dimension réduite. Cette technique permet aux mots ayant des sens proches d'être

représentés par des vecteurs qui se ressemblent.

- **Word2Vec :** est une technique utilisée pour générer des embeddings de mots, c'est-à-dire des représentations numériques de mots sous forme de vecteurs dans un espace de dimension réduite distribuées reflétant les caractéristiques des mots. Ces caractéristiques englobent le contexte dans lequel les mots spécifiques apparaissent.

- **GloVe (Global Vectors for Word Representation) :** est un algorithme d'apprentissage non supervisé, il capture à la fois les statistiques globales et les statistiques locales pour générer les embeddings. GloVe vise à capturer le sens sémantique des mots en le représentant sous forme de vecteurs denses dans un espace vectoriel à haute dimension.

1.2.3 Approches Transformateur pour le Traitement du Langage Naturel

- **BERT (Bidirectional Encoder Representations from Transformers)** : est un framework de Machine Learning open source dédié au traitement du langage naturel (NLP). Il vise à améliorer la compréhension des ordinateurs face au langage ambigu en utilisant le contexte du texte. Entraîné sur d'énormes volumes de données textuelles non annotées, BERT utilise deux approches : prédire les mots manquants et anticiper la phrase suivante, afin de produire des embeddings contextuels.
- **SBERT (Sentence-BERT)** : est une modification du réseau BERT pré-entraîné qui utilise des structures de réseaux siamois et triplets pour dériver des embeddings de phrases sémantiquement significatifs qui peuvent être comparés à l'aide de la cosinusimilarité [34].
- **GPT (Generative Pre-trained Transformer)** : Les Transformateurs génératifs pré-entraînés sont des modèles de langage développés, comme GPT-3 développé par OpenAI en 2018, sont des outils pré-entraînés qui simplifient la création d'applications d'apprentissage automatique. Ils rendent également plus accessible le développement d'applications à ceux qui n'ont pas une grande expérience en programmation [35].

Quatrième partie

Travaux connexes

1.1 Introduction :

L'identification et la classification des scènes et événements acoustiques jouent un rôle crucial dans l'évolution de la reconnaissance audio. Les recherches récentes se concentrent sur le développement de systèmes automatisés capables de créer des légendes audio précises. Les travaux antérieurs ont mis en évidence l'importance de concevoir des systèmes fiables pour interpréter automatiquement le contenu sonore. Malgré les progrès, des défis persistent, notamment en ce qui concerne l'amélioration de la précision et de la robustesse des systèmes dans des contextes variés. Cette section passe en revue trois travaux connexes dans ce domaine.

1.2 Présentation des documents

1.2.1 Document 1 [1]

Ce rapport pour la détection et la classification des scènes et événements acoustiques intitulé " Label-Refined Sequential Training with Noisy Data for Automated Audio Captioning " présente une méthode d'architecture encodeur-décodeur où les développeurs ont recueilli divers ensembles de données, notamment Clotho, AudioCaps, WavText5K et BBC Sound Effects. Chaque ensemble de données à ses propres caractéristiques, certains contenant des échantillons audio bien prétraités et des sous-titres en texte clair, tandis que d'autres ont des étiquettes de texte bruitées et des distributions de données différentes. L'étude souligne l'importance de l'architecture encodeur-décodeur dans ce contexte, où les données audio sont traitées par un encodeur audio afin d'extraire des vecteurs de caractéristiques. Ces vecteurs de caractéristiques sont ensuite décodés par un décodeur de texte pour générer des descriptions textuelles du contenu audio. L'encodeur audio du modèle est basé sur des PANN (réseaux neuronaux audio pré-entraînés), avec l'utilisation spécifique du modèle CNN14 pré-entraîné réputé pour son efficacité dans des tâches telles que la tâche de marquage AudioSet.

D'autre part, le composant de décodage de texte utilise BART (Bidirectional and Auto-Regressive Transformateurs), un modèle de langage dérivé de BERT, connu pour sa compétence dans des tâches telles que le résumé de texte.

Ce cadre sophistiqué d'encodage et de décodage joue un rôle essentiel dans le sous-titrage audio automatisé, en permettant la traduction des caractéristiques audio en descriptions textuelles cohérentes. La sélection et l'intégration minutieuses de ces composants soulignent la capacité du modèle à traiter et à interpréter efficacement les données audio pour la génération de sous-titres, ce qui montre l'importance de l'encodeur audio et du décodeur de texte dans l'amélioration des performances globales du système.

1.2.2 Document 2 [2]

Ce rapport pour la détection et la classification des scènes et événements acoustiques intitulé "PEACS : PREFIX ENCODING FOR AUDITORY CAPTION SYNTHESIS" présente un système de génération automatique de légendes audio. Cette approche utilise une architecture encodeur-décodeur avec un encodeur basé sur un modèle HTS-AT pré-entraîné et un décodeur basé sur GPT2. Un réseau de mappage léger est utilisé pour traduire les représentations audio en un préfixe, guidant ainsi le décodeur. Le modèle est pré-entraîné sur divers ensembles de données de légendage audio et affiné sur Clotho.

L'objectif du rapport est de décrire cette méthode de génération de légendes audio en utilisant des modèles pré-entraînés et une architecture encodeur-décodeur. Le modèle utilise CLAP (Contrastive Language-Audio Pre-Training) avec un backbone HTS-AT pour l'encodeur et GPT2small pour le décodeur. Les clips audio sont transformés en mono canal à 48 kHz et en log mel-spectrograms sur des segments de 10 secondes. SpecAugment est appliqué pour augmenter la robustesse du modèle, et les légendes sont prétraitées et tokenisées avec Byte-Pair-Encoding (BPE).

Les ensembles de données utilisés incluent Clotho, AudioCaps, MACS, SoundDescs et WavText5k. La fonction de perte est basée sur l'entropie croisée entre les légendes prédites et véritables, et la stratégie de décodage utilise le beam search avec une taille de faisceau de 8, avec des pénalités pour répétition et prévention des n-grammes.

1.2.3 Document 3 [3]

Ce rapport pour la détection et la classification des scènes et événements acoustiques présente deux sous-tâches : le légendage automatique de l'audio (AAC) et la recherche

basée sur le langage.

Pour le légendage automatique de l'audio, le système génère des descriptions textuelles du contenu audio en utilisant le modèle PANN avec l'extracteur de caractéristiques CNN-14 et l'encodeur et décodeur BART, évalué par des métriques telles que SPIDEr, METEOR, CIDEr et SPICE. Le dataset utilisé est Clotho v2.

Pour la Tâche, un modèle séquence-à-séquence avec un encodeur et un décodeur BART est utilisé. Les métriques utilisées pour le légendage incluent BLEU, ROUGE, METEOR, CIDEr, SPICE et SPIDEr. La fonction de perte utilisée est l'entropie croisée. Les hyperparamètres incluent l'extraction de caractéristiques avec un spectrogramme log-mel, l'utilisation du modèle CNN14, des couches de décodeur transformeur, et une optimisation par AdamW, avec une durée d'entraînement d'environ 2 heures sur un GPU GTX 1080ti.

En conclusion, ce rapport présente une approche détaillée pour le légendage automatique de l'audio et la recherche basée sur le langage, en utilisant des modèles avancés et des métriques de performance rigoureuses, avec le dataset Clotho v2 fournissant une base solide pour l'entraînement et l'évaluation des modèles .

1.3 Analyse comparative de trois études connexes

Documents	modélisation audio	modélisation de texte	audio feature	Word embeddings	Dataset	Fonction de perte
Document 1	CNN	Transformateur	PANNs	BART	Clotho AudioCaps WavText5K SoundDescs	Crossentropy
Document 2	Transformateur	GPT2	CLAP log-mel spectrograms	/	Clotho AudioCaps MACS SoundDescs WavText5k	crossentropy
Document 3	PANNs avec CNN-14	BART	Log-mel energies	BART	Clotho v2	Crossentropy

Table 1.2 – Comparaison entre les travaux connexes

1.4 Conclusion

Ce chapitre introductif a posé les bases essentielles pour comprendre les principaux domaines abordés dans ce mémoire : l'apprentissage automatique, le deep learning, le traitement audio, le traitement du texte, et les travaux connexes dans la détection et la classification des événements acoustiques.

Nous avons exploré les concepts clés de chaque domaine, fournissant un cadre conceptuel solide pour la suite de notre étude.

Chapitre 02 :

Approche proposée

Chapitre 2

Approche proposée

2.1 Introduction

La génération automatique de légendes audio (AAC) comprend plusieurs étapes cruciales, chacune contribuant de manière significative à la qualité et à la précision des légendes générées.

L'architecture proposée repose sur un pipeline séquence à séquence qui intègre un encodeur pré-entraîné et un décodeur, assurant une transformation efficace des signaux audio en descriptions textuelles. Le processus débute par un prétraitement des données audio, incluant la transformation en Log-Mel Spectrogramme pour extraire les caractéristiques essentielles du signal sonore. Ces caractéristiques sont ensuite encodées en représentations audio-embeddings à l'aide d'un encodeur audio. Parallèlement, les transcriptions textuelles associées sont converties en embeddings sémantiques denses grâce à un encodeur de texte.

Les embeddings audio et textuels sont alignés dans un espace commun, où leur similarité sémantique est maximisée pour faciliter l'appariement correct des paires audio-texte. La combinaison de ces représentations permet d'entraîner un modèle capable de générer des légendes audio précises et cohérentes. Ce chapitre détaillera les différentes étapes du prétraitement des données audio et textuelles, l'extraction des caractéristiques, ainsi que l'architecture encodeur-encodeur , le mécanisme de joint embeddings et l'architecture encodeur-décodeur. Nous démontrerons comment cette approche rigoureuse et novatrice permet de capturer efficacement les relations complexes entre les modalités audio et textuelles.

2.2 L'architecture de la solution

Le système de génération de légendes audio proposé repose sur une architecture de séquence à séquence, intégrant un encodeur pré-entraîné et un décodeur. Cette architecture est implémentée à l'aide du framework PyTorch HuggingFace.

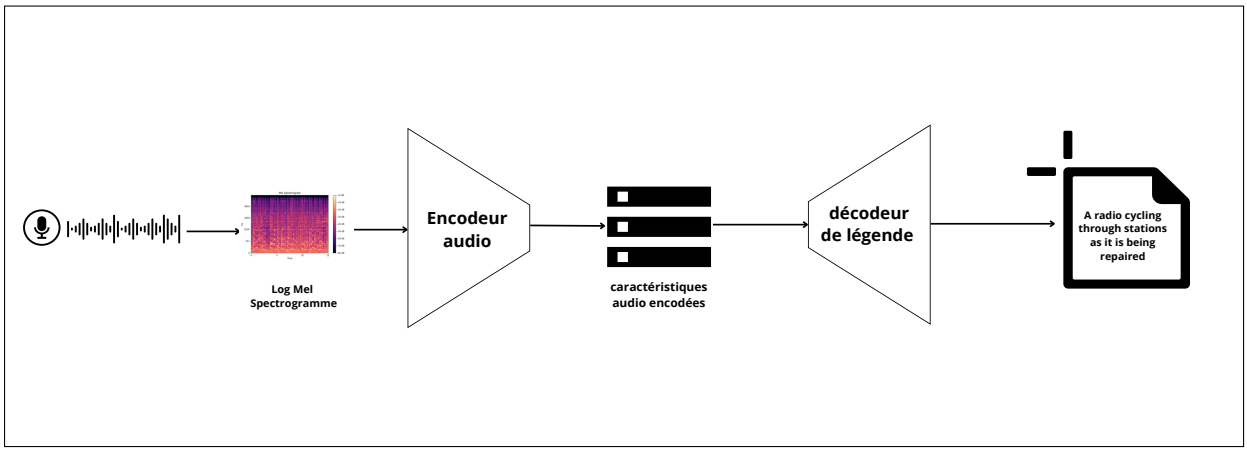


FIGURE 2.1 – L'architecture générale du système

L'architecture présentée illustre le pipeline de système d'AAC basé sur le modèle encodeur-décodeur. À titre d'illustration, le processus débute avec un signal audio prétraité, suivi d'une transformation en Log-Mel Spectrogramme afin de procéder au traitement et à l'extraction des caractéristiques fondamentales du signal, en vue d'une analyse ultérieure. Les caractéristiques audio ainsi extraites servent alors d'entrée à l'encodeur audio, où elles sont converties en représentations audio-embeddings. Ces embeddings sont ensuite décodés par un écodeur de texte, générant ainsi une légende textuelle du signal audio d'origine.

Une exploration plus approfondie de ces composants ainsi que des mécanismes utilisés dans cette architecture sera réalisée dans les sections ultérieures.

2.3 Prétraitement de données

Le prétraitement de données consiste à nettoyer, tansformer et préparer les données brutes (audio et textuelles) pour les rendre utilisables par les algorithmes d'apprentissage automatique. Ce processus améliore la qualité des données et optimise les performances

des modèles prédictifs.

2.3.1 Prétraitement des données audio

Le prétraitement permet à la création de deux dictionnaires : «audio_fid2fname» et «audio_durations» qui sont utilisés pour stocker respectivement les associations entre les identifiants de fichier et les noms de fichiers, et les durées des fichiers audio valides dans la dataset.

Clé	Description
audio_fid2fname	Pour chaque échantillon audio valide, un identifiant unique généré de manière aléatoire et encodé en base64 ainsi que le nom du fichier WAV sont stockés dans ce dictionnaire.
audio_durations	Les durées des échantillons audio valides sont stockées dans ce dictionnaire.

TABLE 2.1 – Description de contenu des dictionnaires générées par le prétraitement des données audio

Ces dictionnaires sont sérialisés ensuite au format binaire et enregistrés dans un fichier `audio_info.pkl` au format pickle.

2.4 Prétraitement des données textuelles

2.4.1 Intégration de données

Dans cette étape les données textuelles (les noms des fichiers audio, le texte brut et le texte nettoyé de leurs descriptions) sont parcourues pour construire un DataFrame contenant les informations suivantes :

Nom de fichier audio	Le nom de fichier .WAV
Texte brut	La légende textuelle de l'audio
Texte nettoyé	La légende textuelle de l'audio en miniscule
L'identifiant unique de l'audio	« file name id » déjà généré dans la phase de prétraitement audio
L'identifiant unique de la légende audio	« Token id » un identifiant unique généré de manière aléatoire et encodé en base64
Tokeniser	Le texte nettoyé est tokenisé en utilisant des espaces comme séparateur.

TABLE 2.2 – Description du contenue du DataFrame générée pendant l'intégration de données textuelles

2.4.2 Calcule et stockage de statistiques des données prétraitées

Trois dictionnaires de données «word_bags», «split_infos», «vocabulary» ont été créés pour stocker différentes informations sur notre ensemble de données. Le contenu de chaque dictionnaire est illustré dans le tableau suivant :

Dictionnaires	Contenu
word_bags	Contient des sacs de mots pour chaque fraction de données, où chaque sac de mots est une liste de tous les mots présents dans les légendes associées à cette fraction.
split_infos	Stocke des informations statistiques sur chaque fraction de données, telles que le nombre de clips audio, le nombre total de légendes et le nombre total de mots
Vocabulary	Contient tous les mots uniques présents dans les légendes de l'ensemble de données (le vocabulaire complet de toutes les données)

TABLE 2.3 – Description de contenu des dictionnaires de données

Les données sur le vocabulaire, les sacs de mots et les informations relatives aux fractions de données sont enregistrées dans un fichier binaire «vocab_info.pkl». Ce fichier est utilisé pour stocker les résultats de l'analyse et les pré-traitements réalisés sur l'ensemble de données. Il permet une récupération rapide des informations nécessaires pour les étapes ultérieures de l'expérience.

2.5 Extraction des caractéristiques

2.5.1 Extraction des caractéristiques audio

L'extraction des caractéristiques audio est un processus crucial dans le domaine du traitement du signal audio. Il vise à identifier et extraire les informations clés contenues dans l'audio, permettant ainsi une analyse approfondie du contenu sonore. Voici les étapes qui nous fournir pour l'extraction des caractéristiques :

2.5.1.1 - Extraction du Mel Spectrogramme

Nous avons employé le Log-Mel Spectrogramme pour le traitement des signaux audio. Cette approche transforme les formes d'onde audio en une représentation spectrale, capturant ainsi les caractéristiques essentielles du signal pour l'analyse ultérieure.

Le tableau suivant illustre les valeurs des paramètres utilisés pour l'extraction du Mel-Spectrogramme.

Paramètres	**Valeur attribuées**
Durée de chaque fenêtre d'analyse	40 ms
Chevauchement entre les fenêtres	20 ms
Nombre de bandes de Mel à générer	64

TABLE 2.4 – Les valeurs des paramètres utilisés pour l'extraction du Mel-Spectrogramme

2.5.1.1.2 - Transformation de Fourier pour chaque fenêtre temporelle

Pour chaque fenêtre temporelle, une transformation de Fourier est appliquée. elle permet de convertir le signal audio, qui est représenté dans le domaine temporel (amplitude en

fonction du temps), en son équivalent dans le domaine fréquentiel (amplitude en fonction de la fréquence), La longueur de la FFT est généralement choisie comme la plus petite puissance de 2 qui est supérieure ou égale à la longueur de la fenêtre, ce qui permet d'optimiser les calculs FFT pour une meilleure efficacité. Cela permet d'identifier les différentes composantes fréquentielles présentes dans chaque fenêtre temporelle du signal audio.

2.5.1.1.3 - Regroupement des fréquences en bandes de Mel

Les fréquences résultantes de la transformation de Fourier sont ensuite regroupées en bandes de Mel. L'échelle de Mel est une échelle de fréquences conçue pour refléter la perception humaine des sons. Contrairement à une échelle linéaire, où chaque unité représente une augmentation constante en Hz, l'échelle de Mel tient compte de la manière dont les fréquences sont perçues par l'oreille humaine. Ainsi, les bandes de Mel sont plus larges à basse fréquence et plus étroites à haute fréquence, pour mieux correspondre à la perception humaine des sons.

2.5.1.1.4 - Construction du spectrogramme de Mel

Un spectrogramme de Mel est construit en représentant graphiquement l'intensité de chaque bande de Mel pour chaque fenêtre temporelle. Typiquement, les intensités sont représentées par des couleurs ou des nuances de gris, où des couleurs plus vives ou des valeurs plus élevées indique une amplitude plus élevée.

2.5.1.1.5 - Transformation en échelle logarithmique

Les valeurs de l'intensité de chaque bande de Mel dans le spectrogramme de Mel sont converties en échelle logarithmique. Cette transformation est réalisée en prenant le logarithme des valeurs de l'intensité. Le but de cette transformation est de mieux représenter les différences dans les niveaux d'énergie du signal audio.

En prenant le logarithme des valeurs, les contrastes entre les différentes fréquences sont amplifiés, ce qui peut faciliter l'analyse des caractéristiques importantes du signal.

Le Mel Spectrogramme représente une avancée majeure dans l'analyse des signaux audio, offrant une représentation précise et discriminante des caractéristiques acoustiques.

Son utilisation s'étend à de nombreux domaines pas uniquement dans le domaine de sous titrage automatique. En permettant une analyse fine des fréquences et des modulations temporelles, le Mel Spectrogramme constitue un outil essentiel pour la compréhension et l'exploitation des signaux audio dans diverses applications.

2.4.1.2 - Encodage d'audio (CNN14)

Nous avons choisi d'utiliser une architecture CNN profonde appelée CNN14Encoder pour encoder des spectrogrammes audio en représentations vectorielles de dimension fixe.

Le CNN14 Encoder se compose de six blocs convolutifs successifs, chacun comprenant deux couches convolutives suivies d'une normalisation par lots (BatchNorm), d'une fonction d'activation ReLU, d'un avg-pooling et d'une couche de dropout pour la régularisation. Ces blocs sont ensuite suivis de deux couches fully-connected avec une fonction d'activation ReLU et une couche de dropout supplémentaire. La dernière couche fully-connected projette la représentation interne du CNN dans un espace de dimension fixe, en l'occurrence 300, pour obtenir une représentation vectorielle compacte des spectrogrammes audio.

Pour accélérer la convergence du modèle et réduire les besoins en données d'entraînement, on a utilisé la technique de transfert d'apprentissage. Cela implique d'initialiser les poids du CNN14Encoder avec ceux d'un modèle CNN14 pré-entraîné sur une tâche similaire en audio. Le processus comprend une étape de mappage des noms de paramètres entre les deux modèles, suivie du chargement et de l'initialisation des poids correspondants.

Voici l'architecture de l'encodeur CNN14 :

Conv2D Block 1 : (batch_size, 1, time_steps, Mel_bands)
Conv2D_1(1, 64, 3x3, stride=1, padding=1, bias=False)
BatchNorm2d_1(64)
ReLU_1
Conv2D_2(64, 64, 3x3, stride=1, padding=1, bias=False)
BatchNorm2d_2(64)
ReLU_2
AvgPool2d_1 (kernel_size=2)
Dropout_1 (p=0.2)
Output_1 : $(\textbf{batch\_size}, 64, \frac{\textbf{time\_steps}}{2}, \frac{\textbf{Mel\_bands}}{2})$

Conv2D Block 2 : (batch_size, 64, time_steps/2, Mel_bands/2)
Conv2D(64, 128, 3x3, stride=1, padding=1, bias=False)
BatchNorm2d_4(128)
ReLU_3
Conv2D(128, 128, 3x3, stride=1, padding=1, bias=False)
BatchNorm2d_4(128)
ReLU_4
AvgPool2d_2(kernel_size=2)
Dropout_2(p=0.2)
Output_2 : $(\text{batch\_size}, 128, \frac{\text{time\_steps}}{4}, \frac{\text{Mel\_bands}}{4})$

Conv2D Block 3 : (batch_size, 128, time_steps/4, Mel_bands/4)
Conv2D_5(128, 256, 3x3, stride=1, padding=1, bias=False)
BatchNorm2d_5(256)
ReLU_5
Conv2D_6(256, 256, 3x3, stride=1, padding=1, bias=False)
BatchNorm2d_6(256)
ReLU_6
AvgPool2d_3(kernel_size=2)
Dropout_3(p=0.2)
Output_3 : $(\text{batch\_size}, 256, \frac{\text{time\_steps}}{8}, \frac{\text{Mel\_bands}}{8})$

↓

Conv2D Block 4 : (batch_size, 256, time_steps/8, Mel_bands/8)
Conv2D_7(256, 512, 3x3, stride=1, padding=1, bias=False)
BatchNorm2d_5(512)
ReLU_7
Conv2D_8(512, 512, 3x3, stride=1, padding=1, bias=False)
BatchNorm2d_6(512)
ReLU_8
AvgPool2d_4(kernel_size=2)
Dropout_4(p=0.2)
Output_4 : $(\text{batch\_size}, 512, \frac{\text{time\_steps}}{16}, \frac{\text{Mel\_bands}}{16})$

↓

Conv2D Block 5 : (batch_size, 512, time_steps/16, Mel_bands/16)
Conv2D_9(512, 1024, 3x3, stride=1, padding=1, bias=False)
BatchNorm2d_9(1024)
ReLU_9
Conv2D_10(1024, 1024, 3x3, stride=1, padding=1, bias=False)
BatchNorm2d_10(1024)
ReLU_10
AvgPool2d_5(kernel_size=2)
Dropout_5(p=0.2)
Output_5 : $(\text{batch\_size}, 1024, \frac{\text{time\_steps}}{32}, \frac{\text{Mel\_bands}}{32})$

Conv2D Block 6 : (batch_size, 1024, time_steps/32, Mel_bands/32)
Conv2D_11(1024, 2048, 3x3, stride=1, padding=1, bias=False)
BatchNorm2d_11(2048)
ReLU_11
Conv2D_12(2048, 2048, 3x3, stride=1, padding=1, bias=False)
BatchNorm2d_12(2048)
ReLU_12
AvgPool2d_6(kernel_size=2)
Dropout_6(p=0.2)
Output_6 : $(\text{batch\_size}, 2048, \frac{\text{time\_steps}}{64}, \frac{\text{Mel\_bands}}{64})$

2.5.2 Extraction des caractéristiques de texte

2.5.2.1 - Encodage de texte (S-BERT)

L'encodeur de texte est chargé de convertir les transcriptions textuelles (texte brut) associées aux segments audio en représentations vectorielles denses et sémantiquement riches.

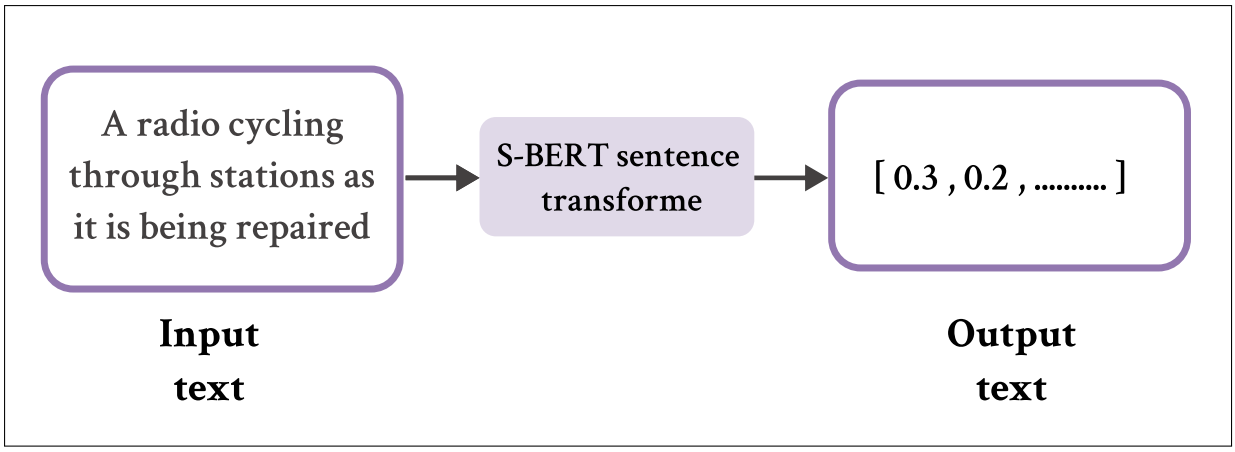


FIGURE 2.2 – Exemple des phrase S-BERT

L'encodeur de texte utilise S-BERT (Sentence-BERT) pour convertir les transcriptions textuelles en représentations vectorielles denses et sémantiquement riches. S-BERT, une variante de BERT, génère des embeddings de phrases qui capturent le sens, la structure et le contexte. (voir figure 2.2).

Le modèle 'all-mpnet-base-v2' de S-BERT est utilisé pour produire des vecteurs d'embeddings de 768 dimensions pour chaque transcription. Ces vecteurs sont ensuite réduits à 300 dimensions via une couche de projection pour s'aligner avec les embeddings audio.

Les embeddings textuels sont appariés avec les embeddings audio (extraits par CNN14) pour former des paires audio-texte utilisées dans l'entraînement du modèle. Cette approche permet de capturer la sémantique profonde des transcriptions, facilitant la génération de sous-titres précis pour les segments audio(voir figure 2.3).

2.6 L'architecture encodeur-encodeur

L'architecture encodeur-encodeur pour le système d'Automated Audio Captioning (AAC) est conçue pour traiter des paires audio-texte en entrée, afin de générer des légendes textuelles descriptives pour les segments audio. Comme illustré dans la figure 2.3 ci-dessous, cette architecture schématisée commence par un lot (batch) de N paires d'audio-texte. Chaque paire est ensuite traitée par deux encodeurs distincts : un encodeur audio basé sur CNN14 pour les audio et un encodeur de texte basé sur SBERT pour les données textuelles.

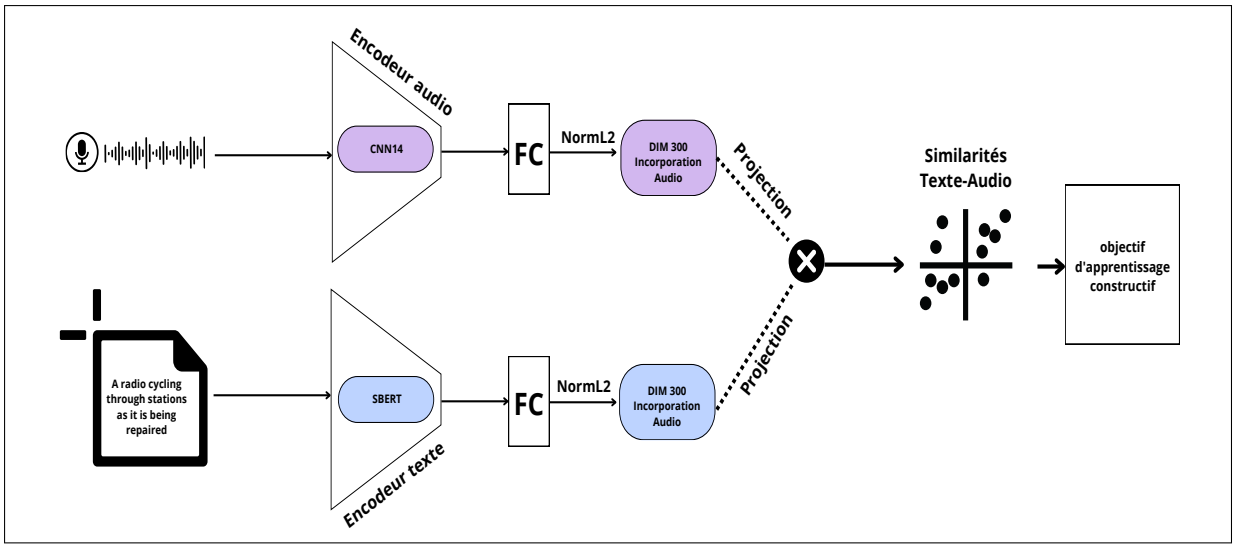


FIGURE 2.3 – L'architecture encodeur-encodeur de système d'AAC

L'encodeur audio (CNN14) et l'encodeur de texte (SBERT) transforment respectivement les spectrogrammes audio et les transcriptions textuelles en représentations vectorielles. Ces représentations passent ensuite par des couches entièrement connectées (FC) pour affiner les représentations. Une normalisation L2 est appliquée aux sorties des couches FC pour rendre les vecteurs invariants à l'échelle. Les représentations audio et texte, désormais dans un espace commun de 300 dimensions, sont utilisées pour calculer des similarités. Un objectif d'apprentissage contrastif est ensuite appliqué pour encourager le rapprochement des embeddings des paires audio-texte correspondantes et l'éloignement des paires non correspondantes.

Cette architecture encodeur-encodeur permet de capturer efficacement les relations complexes entre les signaux audio et leurs descriptions textuelles, facilitant la génération de légendes audio précises et sémantiquement riches.

2.7 Décodage et génération de légendes textuelles

2.7.1 BART

BART (Bidirectional and Auto-Regressive Transformers) est un modèle de séquence à séquence préentraîné, proposé par Lewis et al. (2019). Il combine les avantages des modèles

auto-régressifs classiques avec ceux des modèles encodeur-décodeur bidirectionnels. L'architecture de BART comprend deux composants principaux :

- **Encodeur bidirectionnel :** Ce composant, similaire à celui du modèle BERT, permet une attention bidirectionnelle sur la séquence d'entrée, capturant ainsi efficacement les dépendances contextuelles dans les deux sens de la séquence.
- **Décodeur auto-régressif :** Ce décodeur génère les jetons de sortie un par un, de manière séquentielle, en fonction des jetons précédents et de la représentation encodée de l'entrée.

BART est préentraîné sur une tâche de reconstruction de texte, où il doit reconstituer une séquence complète à partir d'une version corrompue. Cette tâche de débruitage et de génération de texte permet au modèle d'apprendre des représentations riches et transférables à diverses tâches de génération de texte.

Une fois préentraîné, BART peut être affiné sur des données spécifiques à une tâche cible, comme la génération de légendes. Lors de cette phase, l'encodeur encode les données d'entrée tandis que le décodeur génère la sortie souhaitée jeton par jeton.

Pour la tâche d'AAC, seule la capacité de génération de texte du décodeur auto-régressif de BART est requise. Son préentraînement sur une tâche de reconstruction de texte lui confère une robustesse et une capacité de généralisation accrues, ce qui en fait un choix pertinent pour cette tâche.

2.7.2 Architecture et paramètres du modèle encodeur-décodeur :

Composant	Paramètre	Valeur
Prétraitement de données pour le décodeur BART	Caractéristiques Audio	Remplissage avec des vecteurs nuls
	Légendes Tokenisées	Ajout d'un token de début de phrase `<s>` Ajout d'un token de fin de phrase `</s>` Remplissage avec un token spécial `<pad>`
L'encodeur CNN14	Embeddings de sortie	2048 dimensions
Couche affine	Fonction d'activation	Aucune
	Transformation	2048 dimensions en 768 dimensions
Décodeur transformateur (BART)	Self-attention multi-têtes	12 têtes, suivi d'une connexion résiduelle qui ajoute l'entrée de la couche à la sortie de la self-attention, puis normalisation de couche.
	Cross-attention multi-têtes	12 têtes, suivi d'une connexion résiduelle qui ajoute l'entrée de la couche à la sortie de la cross-attention, puis normalisation de couche.
	Couches Affines	2Deux couches affines de dimensions 3072 et 768, avec connexions résiduelles après chaque couche, suivies de la normalisation de couche.

2.7.3 Génération de légendes textuelles :

Après l'entraînement de l'encodeur CNN14 pour l'extraction de représentations audio discriminantes, ce dernier est gelé lors de l'entraînement du modèle de génération de légendes audio. Les couches finales de moyenne temporelle et les couches denses sont omises, permettant d'obtenir des séquences d'embeddings audio de dimension 2048. Ces embeddings subissent ensuite une transformation frame par frame via une couche affine, réduisant leur dimension à 768, avant d'être transmis au décodeur.

Le décodeur, quant à lui, exploite les sorties de l'encodeur pour générer les légendes de manière autorégressive. Dans ce processus, les mots précédemment générés sont tokenisés et transformés en embeddings pour servir d'entrées au décodeur. Le tokenizer utilisé dans le modèle de référence est préentraîné avec un processus d'encodage par paires de bytes (byte-pair-encoding), où chaque token correspond à un sous-mot du vocabulaire du modèle plutôt qu'à un mot entier en anglais. Ce tokenizer dispose d'un vocabulaire de 50265 tokens. Chaque token de la séquence précédente est alors associé à un vecteur de caractéristiques via une table d'embeddings, avant d'être fourni au décodeur.

Au sein du décodeur, chaque couche applique une auto-attention sur les tokens précédemment générés, ainsi qu'une attention croisée (cross-attention) sur l'ensemble de la séquence d'embeddings audio en sortie de l'encodeur. La dimension des embeddings de chaque couche du décodeur est fixée à 768. Enfin, une tête de classification composée d'une couche linéaire avec activation softmax génère une distribution de probabilités sur l'ensemble du vocabulaire de tokens, permettant de prédire le prochain token à générer.

Lors de l'évaluation du modèle, la génération des légendes est effectuée par recherche avec faisceau (beam search), bien qu'un décodage glouton soit également fourni dans le code de référence à des fins de comparaison.

Cette architecture encodeur-décodeur, avec un encodeur CNN figé pour l'extraction de représentations audio et un décodeur auto-régressif entraîné pour la génération de légendes, permet d'exploiter efficacement les informations complémentaires présentes dans les modalités audio et textuelle pour la tâche de génération de légendes audio.

2.8 La fonction de perte :

Dans la tâche d'automated audio captioning, l'objectif principal est d'entraîner un modèle à générer des légendes textuelles pertinentes et descriptives pour des segments audio donnés. Cette tâche de génération de séquence peut être formulée comme un problème de modélisation de la distribution de probabilité conditionnelle P(y—x), où x représente l'entrée audio et y la légende textuelle cible.

Une métrique d'entraînement communément utilisée pour ce type de tâche est la cross-entropy (entropie croisée). La cross-entropy mesure la divergence entre la distribution de probabilité prédite par le modèle et la distribution de probabilité réelle des données. En minimisant cette métrique, le modèle est encouragé à attribuer des probabilités élevées aux légendes correctes et des probabilités faibles aux légendes incorrectes.

Soit N le nombre d'exemples d'entraînement, $\mathbf{x}_i$ l'entrée audio du i-ème exemple, et $y_i^1, y_i^2, ..., y_i^t$ la légende textuelle cible correspondante de longueur T. La cross-entropy peut être formulée comme suit :

$$L = -\frac{1}{N} \sum_i \sum_t \log P(y_i^t | \mathbf{x}_i, y_i^1, y_i^2, ..., y_i^{(t-1)}) \tag{2.1}$$

Où $P(y_i^t | \mathbf{x}_i, y_i^1, y_i^2, ..., y_i^{(t-1)})$ représente la probabilité prédite par le modèle pour le t-ème token de la légende, étant donné l'entrée audio $\mathbf{x}_i$ et les tokens précédents de la légende.

Cette formulation reflète la nature auto-régressive du processus de génération de légende, où chaque token est prédit en fonction de l'entrée audio et des tokens précédemment générés. La cross-entropy encourage ainsi le modèle à prédire des probabilités élevées pour les tokens corrects de la légende cible, tout en pénalisant les prédictions incorrectes.

Lors de l'entraînement, le modèle ajuste ses paramètres de manière à minimiser cette perte de cross-entropy sur l'ensemble des exemples d'entraînement. Cela permet d'apprendre une représentation interne des données audio et textuelles, ainsi que les relations complexes entre ces modalités, afin de générer des légendes pertinentes pour de nouveaux segments audio.

2.9 Conclusion

En conclusion, notre chapitre offre une perspective novatrice et complète sur la manière de relever ce défi complexe. En intégrant de manière rigoureuse l'architecture de la solution, le prétraitement des données, l'extraction des caractéristiques audio et textuelles, ainsi que l'architecture encodeur-encodeur et l'architecture encodeur-décodeur nous avons posé les fondations d'un système robuste et performant. En exploitant les relations entre les modalités audio et textuelle dans un espace de "joint embedding", nous sommes en mesure de produire des légendes textuelles pertinentes et informatives. Cette approche représente une avancée significative dans le domaine de la génération automatique de légendes, ouvrant la voie à de nouvelles applications et possibilités dans divers domaines.

Chapitre 03 :

Résultat pratiques

Chapitre 3

Résultat pratiques

3.1 Introduction

Faisant suite au chapitre précédent qui exposait les fondements théoriques de notre approche, nous décrivons ici les outils et technologies exploités pour la mise en œuvre concrète de notre solution AAC. Ce chapitre présente en détail les différents aspects pratiques liés au développement et à l'évaluation de notre système de génération automatique de légendes textuelles à partir de signaux audio (AAC).
Nous commençons par passer en revue les différents outils utilisés pour le développement. Ensuite, nous poursuivons avec une description du dataset qui a servi de base à nos expériences, ainsi que de la procédure suivie pour l'analyse et l'évaluation des résultats obtenus. Enfin, nous illustrons les résultats obtenus dans la phase de test.

3.2 Les outils utilisés pour le développement

L'environnement de développement		
PyCharm		est un environnement de développement intégré spécialement conçu pour le langage de programmation Python. Il propose une variété de fonctionnalités visant à simplifier la programmation et à la rendre plus efficace, telles que la coloration syntaxique, l'auto-complétion de code, les outils de débogage, l'intégration du contrôle de version, et le support des frameworks web.

Le Langage de programmation		
Python		Python est un langage de programmation interprété, interactif et orienté objet. Il fournit des structures de données de haut niveau telles que les listes et les tableaux associatifs, le typage dynamique et la liaison dynamique, les modules, les classes, les exceptions, la gestion automatique de la mémoire, etc. Il possède une syntaxe remarquablement simple et élégante, tout en étant un langage de programmation puissant et généralisé. Il a été conçu en 1990 par Guido van Rossum. Comme beaucoup d'autres langages de script, il est gratuit, même à des fins commerciales, et peut être exécuté sur pratiquement n'importe quel ordinateur moderne [36] .
Les Services en ligne		
Google Collab		Google Colaboratory est un outil gratuit permettant de prototyper des modèles de Machine Learning avec des GPU et TPU, directement dans un navigateur web. Il s'agit d'un service de notebooks Jupyter hébergé, facile à utiliser sans configuration requise. Les machines virtuelles de Colab offrent des ressources informatiques puissantes mais sont limitées par des quotas et des interruptions en cas d'inactivité. Pour plus de ressources et une utilisation prolongée, des versions payantes comme Colab Pro et Colab Pro+ sont disponibles.
Kaggle		est une plateforme en ligne populaire prisée pour la science des données, offrant notamment une variété de ressources telles que des ensembles de données, des compétitions, des cours et des notebooks collaboratifs. Kaggle propose des notebooks Jupyter intégrés pour écrire, exécuter et partager du code Python directement sur la plateforme et des GPU gratuits. Il est capable de traiter des calculs intensifs.

Google Drive		est un service de stockage sur cloud, nous l'utilisons pour le stockage et l'organisation des données du projet et pour faciliter la collaboration à distance. Sauvegarder et synchroniser les fichiers et nos résultats expérimentaux. En bref, Google Drive offre une solution de stockage de données, de partage et de collaboration qui est essentielle pour la gestion efficace de notre projet.
Overleaf		Overleaf est une plateforme en ligne collaborative spécialisée dans la création, l'édition et la publication de documents LaTeX. Principalement dédiée aux travaux académiques et scientifiques, elle permet une collaboration en temps réel et offre un aperçu instantané du document final.
Les bibliothèques		
Librosa		Nous permettons de Pouvoir lire, écrire et manipuler des fichiers audios.
PYyaml		pour travailler avec des fichiers YAML, nous l'utilisons pour la configuration ou le stockage de données structurées.
Pytorch		est utilisée pour la création de modèles d'apprentissage profond, ce qui est crucial pour la génération automatique de légendes textuelles.
Numpy		est une bibliothèque fondamentale pour le calcul numérique en Python, nous l'utilisons pour effectuer des opérations mathématiques sur les données.
Transformers		est une bibliothèque populaire pour le traitement du langage naturel basé sur des modèles de Transformateur, ce qui est précieux pour la génération de texte.
Matplotlib		est une bibliothèque de visualisation en Python, utile pour créer des graphiques et des représentations visuelles des données.
Pandas		est une bibliothèque efficace pour la manipulation et l'analyse de données structurées, nous l'utilisons pour lire les métadonnées et gérer les données de notre projet.

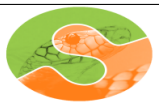
Scikit-image		est une bibliothèque dédiée au traitement d'images en Python, il est utilisé pour manipuler des images liées à nos signaux audio.
Mutagen		utile pour la manipulation de métadonnées audio, ce qui va être pertinent pour extraire des informations des signaux audio.
Pickle		est un module Python pour la sérialisation et la désérialisation d'objets Python, utile pour sauvegarder des modèles ou des données.
H5py		pour travailler avec des fichiers HDF5, nous l'utilisons pour stocker de grandes quantités de données.
Tqdm		pour afficher des barres de progression lors de l'exécution de boucles, ce qui va être pratique pour suivre l'avancement de nos processus.
sBert		permet de générer des embeddings de phrases, de texte de pointe.

TABLE 3.1 – Les outils utilisés

3.3 Description de DataSet

Clotho est un dataset pour l'audio captioning comprenant :

- **4981 échantillons audio de 15-30 secondes**
- **24 905 captions de 8-20 mots**
- **Audio varié :** ambiances naturelles, bruits de machines, sons d'animaux, bruits urbains
- **Divisions :** Développement, Validation, Évaluation
- **Caractéristiques :**

  - Audio provenant de Freesound

  - Captions par Amazon Mechanical Turk et annotateurs anglophones

La version 2.1 corrige environ 150 fichiers corrompus et remplace certains caractères illégaux[37].

Le dataset est organisé pour faciliter l'entraînement et l'évaluation des modèles d'Audio Automatic Captioning (AAC).

3.4 Métriques d'évaluation

Nous présentons un aperçu des principales mesures d'évaluation utilisées dans l'AAC, notamment BLEU, ROUGE_L, CIDEr, SPICE, SPIDEr et SPIDEr_FL en soulignant leur importance et les aspects uniques qu'elles apportent au processus d'évaluation.

Voici une brève description mentionnée dans ce tableau pour ces métriques utilisée :

Métrique	Plage de score	Description
BLEU	[0, 1]	Mesure la précision des n-grammes
ROUGE_L	[0, 1]	Évalue le rappel des n-grammes
CIDEr	[0, 10]	Utilise TF-IDF pour pondérer les n-grammes
SPICE	[0, 1]	Compare les graphes de scène sémantique
SPIDEr	[0, 5.5]	Moyenne des scores SPICE et CIDEr
SPIDEr_FL	[0, 5.5]	Combinaison pondérée des scores SPIDEr et de la qualité linguistique FL

TABLE 3.2 – Les métriques utilisées

Voici le résumé avec les références conservées :

3.4.1 BLEU (Bilingual Evaluation Understudy) :

BLEU, introduit en 2002 [38], est une métrique d'évaluation pour la traduction automatique adaptée à l'AAC [39]. Elle mesure la correspondance entre les n-grammes des légendes générées et des références humaines, avec un score de 0 à 1.

3.4.2 ROUGE (Recall-Oriented Understudy for Gisting Evaluation)

ROUGE évalue le chevauchement entre la prédiction du modèle et la référence humaine. Il comprend plusieurs variantes (ROUGE-N, ROUGE-L) basées sur différents types de

chevauchements [40].

3.4.3 CIDEr (Consensus-based Image Description Evaluation)

CIDEr mesure le consensus entre la description candidate et les références, en utilisant une pondération TF-IDF sur les n-grammes [41].

3.4.4 SPICE (Semantic Propositional Image Caption Evaluation)

SPICE évalue la similarité sémantique en comparant les graphes de scène des phrases candidates et de référence, se concentrant sur le sens plutôt que sur la correspondance lexicale [41].

3.4.5 SPIDEr (SPICE + CIDEr)

SPIDEr combine SPICE et CIDEr pour équilibrer la fidélité sémantique et la fluidité syntaxique des légendes générées [42].

3.4.6 SPIDEr_FL (SPICE, CIDEr et FL)

SPIDEr_FL intègre SPIDEr avec des mesures de qualité linguistique (FL) pour une évaluation plus complète des légendes audio générées automatiquement.

3.5 Paramètres des expériences

Pour entraîner efficacement un modèle de génération automatique de légendes audio, plusieurs paramètres doivent être soigneusement réglés. Voici dans le tableau suivant les principaux paramètres utilisés dans nos expériences, ainsi que leurs valeurs respectives (tableau 3.3) :

Nous avons choisi une taille de batch de 4 exemples, combinée avec une accumulation de gradients sur 2 étapes, pour optimiser l'utilisation de la mémoire. L'optimiseur AdamW a été utilisé pour ses avantages en termes de régularisation des poids et d'ajustement dynamique des taux d'apprentissage. Le taux d'apprentissage initial de 10^{-5} permet des

Paramètre	Valeur
Taille de batch	4 exemples
Accumulation de gradients	2 étapes
Nombre d'époques	20
Optimiseur	AdamW
Taux d'apprentissage	10^{-5}

TABLE 3.3 – Hyperparamètres d'entraînement

mises à jour stables et progressives des poids du modèle. L'entraînement a été effectué sur 20 époques.

Ces paramètres ont été choisis pour optimiser la performance et la stabilité de l'entraînement du modèle.

3.6 Résultat et évaluation

Afin d'évaluer l'efficacité et la convergence de notre modèle de génération automatique de légendes audio, nous avons suivi de près les valeurs de la fonction de perte (cross-entropy) tout au long de l'entraînement.

La fonction de perte mesure la différence entre les légendes générées par le modèle et les légendes réelles, fournissant ainsi une indication de la performance du modèle. Une diminution progressive des valeurs de la fonction de perte au fil des époques indique que le modèle apprend à mieux capturer les relations entre les entrées audio et les légendes correspondantes.

Dans la courbe suivante (voire figure 3.1), nous présentons l'évolution des valeurs de la fonction de perte pendant les 20 époques d'entraînement. Initialement, la fonction de perte est élevée (plus de 7) au cours de début de la première époque, ce qui est typique au début de l'entraînement. Cependant, dès la fin de cette première époque, nous observons une diminution rapide et significative de la perte, atteignant des valeurs proches de 0,5 juste avant la deuxième époque. Cette réduction rapide indique que le modèle commence à apprendre efficacement dès les premières itérations.

Par la suite, la fonction de perte continue de diminuer, montrant une tendance générale vers la convergence.

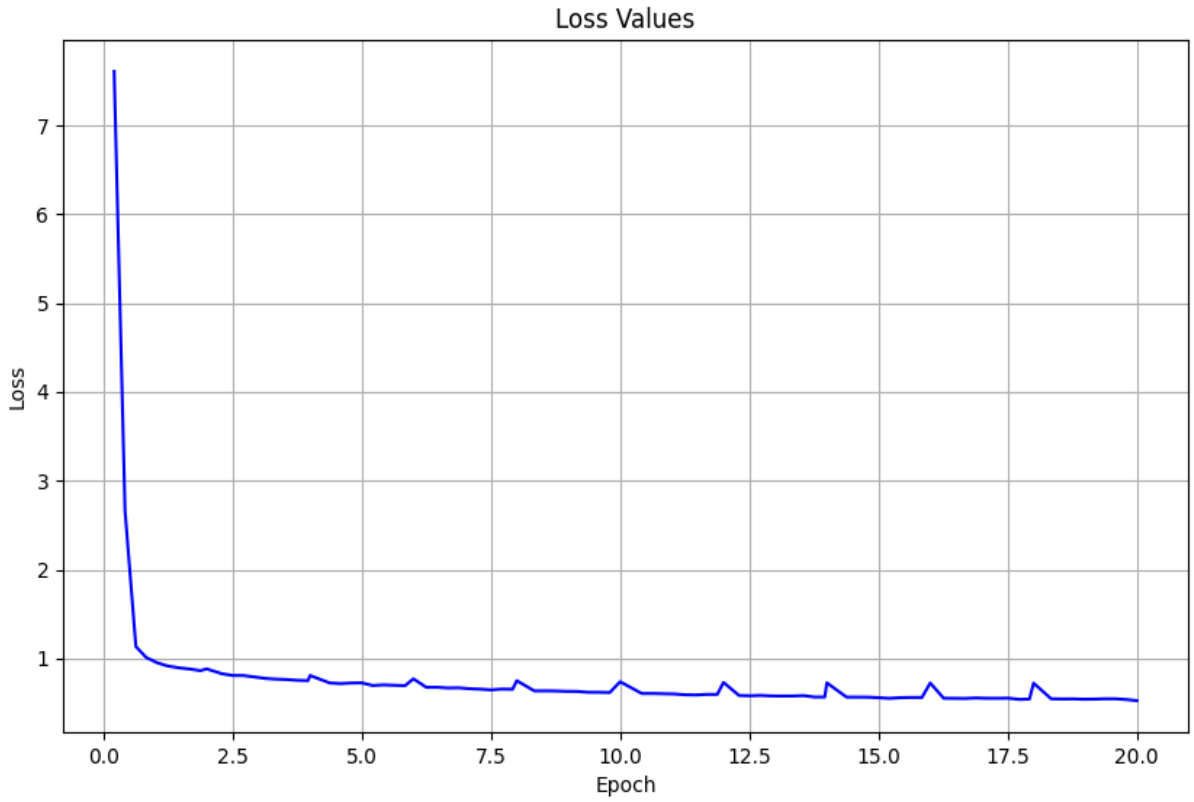


FIGURE 3.1 – Les valeurs de fonction de perte pendant l'entrainement de modèle

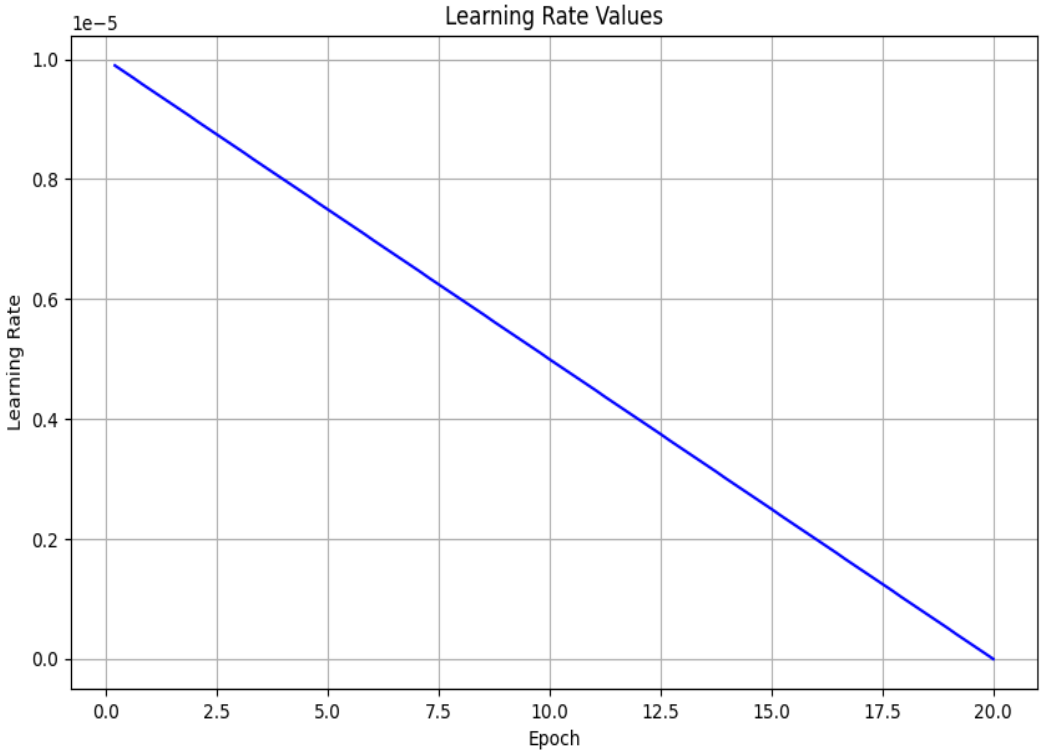


FIGURE 3.2 – l'évolution des valeurs de taux d'apprentissage pendant l'entrainement

La courbe ci-dessus illustre l'évolution du taux d'apprentissage en fonction des époques d'entraînement pour notre modèle.

Analyse de la Courbe

Diminution Linéaire : Le taux d'apprentissage décroît de manière linéaire de 1e-5 à 0 au cours des 20 époques. Cette approche progressive est couramment adoptée pour ajuster les poids du modèle de manière stable, en minimisant les variations potentielles.

Interprétation

Initialisation Élevée : En démarrant avec un taux d'apprentissage relativement élevé (1e-5), le modèle peut effectuer des mises à jour significatives des poids dès le début de l'entraînement. Cela facilite une descente rapide du gradient, permettant au modèle d'éviter les minima locaux peu profonds.

Diminution Progressive : La réduction continue du taux d'apprentissage au fil des époques permet d'affiner progressivement les poids du modèle. Vers la fin de l'entraînement, un taux d'apprentissage plus faible stabilise les ajustements des poids, favorisant ainsi une

convergence précise vers un minimum global ou local profond.

Implications pour l'Optimisation

Stabilité de Convergence : L'approche linéaire de diminution du taux d'apprentissage contribue à une convergence stable du modèle. Cette méthode permet d'ajuster précisément les paramètres du modèle, minimisant ainsi les risques de fluctuations indésirables.

Équilibre entre Exploration et Exploitation : En début d'entraînement, un taux d'apprentissage plus élevé favorise l'exploration de l'espace de recherche, tandis qu'une diminution progressive favorise l'exploitation des connaissances acquises. Cet équilibre est crucial pour optimiser les performances du modèle sans compromettre sa généralisation.

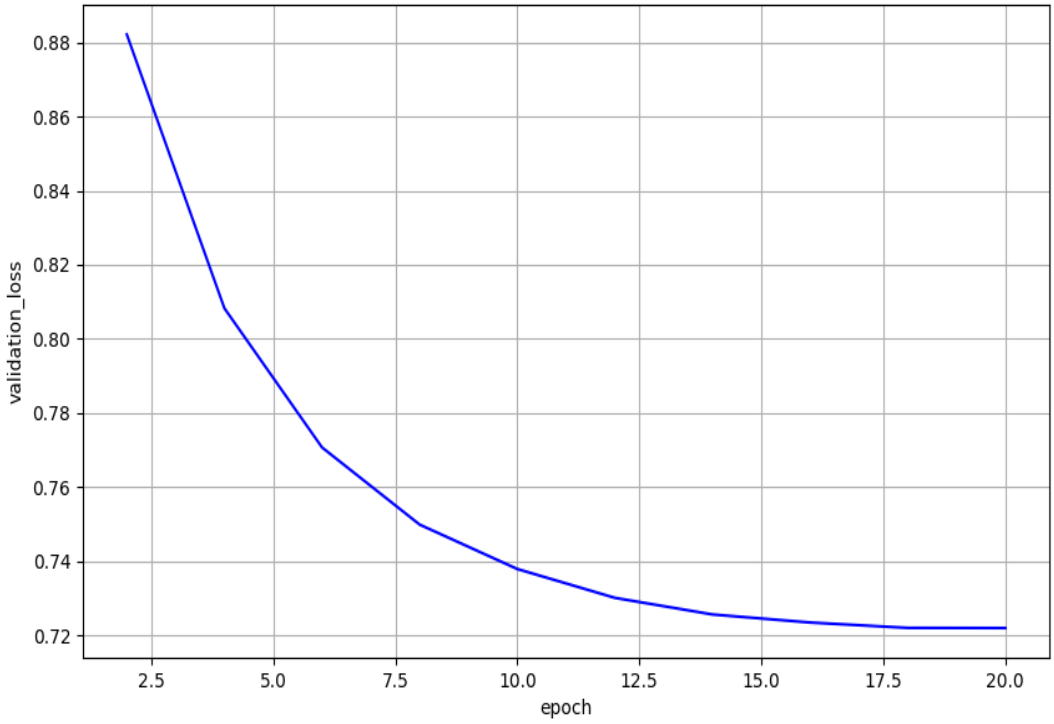


FIGURE 3.3 – Les valeurs de fonction de perte pendant la validation de modèle

La courbe présentée ci-dessus illustre l'évolution de la perte de validation au cours des époques d'entraînement de notre modèle.

Analyse de la Courbe

Diminution Continue : La perte de validation diminue de manière régulière à mesure que le nombre d'époques augmente, passant d'environ 0.88 à 0.72 au cours des 20

époques. Cette diminution progressive indique que le modèle assimile efficacement les caractéristiques des données d'entraînement, améliorant ainsi ses performances sur l'ensemble de validation.

Stabilisation : Aux alentours de la 17ème époque, la courbe commence à se stabiliser, ce qui suggère que le modèle approche un point où les gains en performance deviennent moins significatifs. Cette stabilisation est un indicateur positif de la convergence du modèle, évitant ainsi le risque de surapprentissage.

Interprétation

Bonne Convergence : La diminution régulière de la perte de validation sans fluctuations abruptes indique une convergence satisfaisante du modèle. Cette capacité à généraliser efficacement sur les données de validation est essentielle pour garantir des performances robustes sur des données inédites.

Absence d'Overfitting : L'absence de remontée significative de la perte de validation suggère que le modèle ne souffre pas d'overfitting. Habituellement, une augmentation de la perte de validation indiquerait une mémorisation excessive des données d'entraînement plutôt qu'un apprentissage de motifs généralisables.

Implications pour l'Optimisation

Nombre d'Époques : Étant donné que la courbe se stabilise vers la 17 ème époque, limiter l'entraînement à environ 20 époques pourrait être optimal pour prévenir le surapprentissage tout en économisant des ressources de calcul.

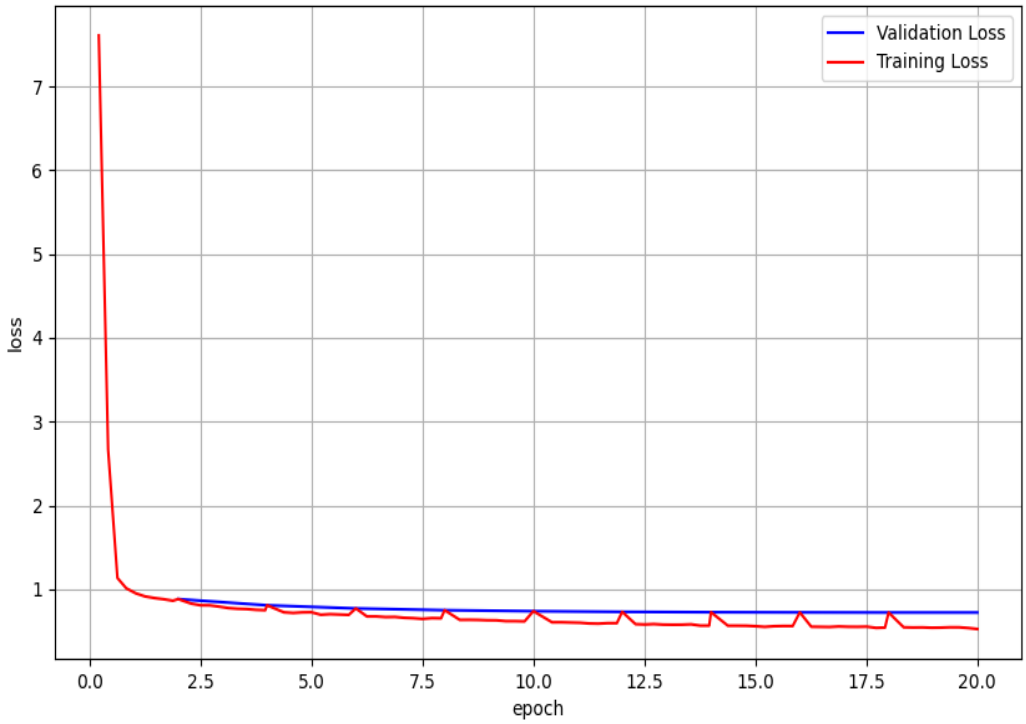


FIGURE 3.4 – Comparaison entre las valeurs de perte d'Entraînement et de Validation

La courbe présentée ci-dessus illustre l'évolution simultanée de la perte d'entraînement (en rouge) et de la perte de validation (en bleu) en fonction des époques pour notre modèle.

Analyse des Courbes

Diminution Initiale Rapide : La perte d'entraînement diminue de manière significative dès les premières époques, passant de plus de 7 à environ 1. Cette rapide réduction indique une adaptation rapide du modèle aux données d'entraînement.

Stabilisation : Après cette phase initiale, tant la perte d'entraînement que la perte de validation montrent une tendance à la stabilisation. Elles convergent toutes deux vers des valeurs proches de 1 au fil des époques, indiquant que le modèle a atteint un point où les gains supplémentaires sont limités.

Oscillations Minimes : Bien que quelques petites oscillations soient observées dans la perte d'entraînement, elles restent modérées et n'altèrent pas la trajectoire globale des courbes.

Interprétation

Convergence Rapide : La diminution rapide de la perte d'entraînement initiale reflète une bonne initialisation des poids et une adaptation efficace du modèle aux données. Cette phase initiale est cruciale pour établir des bases solides dès le début de l'apprentissage.

Bonne Généralisation : La proximité entre les courbes de perte d'entraînement et de perte de validation indique que le modèle généralise bien aux nouvelles données. Cette cohérence témoigne de l'absence de surapprentissage, où le modèle maintient des performances similaires sur les données de validation.

Stabilité : La stabilité observée dans les courbes reflète une convergence robuste du modèle, capable de maintenir ses apprentissages sur les données d'entraînement et de validation.

Ces analyses des courbes de perte d'entraînement et de validation fournissent des insights cruciaux pour optimiser le processus d'apprentissage du modèle, assurant à la fois stabilité et généralisation robuste des performances.

L'évaluation des performances de notre modèle est essentielle pour mesurer sa précision et sa capacité à produire des résultats pertinents. Pour cette évaluation, nous utilisons plusieurs métriques standardisées, chacune offrant un aperçu unique de la qualité des légendes générées. Les scores obtenus pour ces métriques sont présentés ci-dessous (tableau 3.4), fournissant une vue d'ensemble complète de la performance de notre modèle sur différentes dimensions d'évaluation.

Métrique	**Score**
BLEU_1	0.585
BLEU_2	0.388
BLEU_3	0.260
BLEU_4	0.166
ROUGE_L	0.387
SPICE	0.119
CIDEr	0.420
SPIDEr	0.270
SPIDEr_FL	0.261

TABLE 3.4 – Les scores obtenus pour les métriques d'évaluation

Les résultats montrent une performance variée selon les différentes métriques :

Scores BLEU : La diminution progressive des scores de BLEU1 à BLEU4 montre que le modèle est capable de capturer des mots individuels et des paires de mots avec une précision raisonnable, mais a plus de difficulté avec des séquences plus longues et complexes. Les scores BLEU3 et BLEU4 plus bas indiquent que la génération de phrases plus longues et cohérentes reste un défi.

ROUGE-L : Le score ROUGE-L de 0.387 indique que le modèle capture assez bien les structures de phrases longues, bien qu'il y ait encore des marges d'amélioration.

SPICE : Avec un score de 0.119, SPICE révèle que le modèle a encore des progrès à faire en matière de compréhension et de génération de relations sémantiques complexes.

CIDEr : Un score de 0.420 en CIDEr suggère que les légendes générées sont modérément similaires aux légendes de référence en termes de contenu, mais il y a toujours de la place pour des améliorations.

SPIDEr et SPIDEr-FL : Les scores de 0.270 et 0.261 respectivement montrent que le modèle offre une performance équilibrée en termes de précision et de qualité sémantique, mais il y a des aspects de fluence qui nécessitent une attention particulière.

3.7 Analyse de résultats

Pour une évaluation approfondie de nos résultats, nous avons réalisé une comparaison entre les performances de notre modèle et celles des travaux connexes précédents (décrit dans le chapitre 1). Cette analyse comparative vise à mettre en lumière les forces et les faiblesses de notre approche par rapport aux approches antérieures.

Modèle / Métrique	CNN+ Transformateurs	Transformateur+ GPT2	(PANNS +CNN14) +BART	Notre modèle
ROUGE_L	0.365	0.239	0.386	0.166
CIDEr	0.395	0.365	0.419	0.420
SPICE	0.117	0.118	0.121	0.119
SPIDEr	0.256	0.241	0.270	0.270
SPIDEr_FL	0.253	0.239	/	0.261

TABLE 3.5 – Comparaison des résultats des métriques des travaux connexes avec les résultats que nous avons obtenus

Cette comparaison offre des insights précieux sur la performance de notre modèle par rapport aux approches précédentes, à travers diverses métriques d'évaluation. Pour mieux comprendre ces résultats, nous allons procéder à une analyse comparative détaillée des performances de notre modèle par apport au autres modèle sur ces différentes métriques :

ROUGE-L : Notre modèle a obtenu un score de 0.166, ce qui est inférieur aux autres modèles. Cela suggère que, bien que notre modèle soit performant sur d'autres aspects, il pourrait bénéficier d'améliorations pour mieux capturer la structure globale des phrases. Les modèles 1 et 3, en particulier, montrent une meilleure performance avec des scores de 0.365 et 0.386 respectivement.

CIDEr : Notre modèle a obtenu un score de 0.420, le plus élevé parmi tous les modèles comparés. Cela indique que les légendes générées par notre modèle sont très similaires aux légendes de référence, en termes de contenu et de précision des n-grammes.

SPICE : Le score SPICE de notre modèle est 0.119, ce qui est comparable aux autres modèles (Modèle 1 : 0.117, Modèle 2 : 0.118, Modèle 3 : 0.121). Cela montre que notre modèle est compétitif en termes de capture des relations sémantiques dans les légendes générées.

SPIDEr : Avec un score de 0.270, notre modèle est à égalité avec le Modèle 3, et légèrement supérieur au Modèle 1 (0.256) et au Modèle 2 (0.241). Cette métrique combinée (CIDEr et SPICE) suggère que notre modèle offre un bon équilibre entre précision et qualité sémantique.

SPIDEr-FL : Notre modèle a obtenu un score de 0.261, le plus élevé des modèles comparés qui ont cette métrique disponible. Cela indique que notre modèle génère des légendes qui sont non seulement précises et sémantiquement riches, mais aussi fluides et lisibles.

Points Forts et Limites : Les résultats montrent que notre modèle excelle particulièrement sur les métriques CIDEr et SPIDEr-FL, ce qui suggère une forte capacité à générer des légendes précises et lisibles. Cependant, les scores plus faibles en ROUGE-L indiquent des défis dans la capture de la structure globale des phrases.

Précision et Contenu : Les scores élevés de CIDEr et SPIDEr indiquent que notre modèle génère des légendes qui sont très similaires aux légendes de référence.

Lisibilité : Le score SPIDEr-FL le plus élevé parmi les modèles comparés souligne la qualité fluide et naturelle des légendes générées par notre modèle.

Structure Globale : Le score ROUGE-L inférieur suggère que notre modèle pourrait améliorer la capture de la structure globale et de la cohérence des phrases.

3.8 Résultats de Test

Pour illustrer les résultats obtenus par notre système AAC et mettre en évidence les différences significatives, nous avons développé une application de bureau. Cette application permet de mener des expériences plus rapidement avec nos fichiers de tests audio. Les deux figures ci-dessous montrent l'interface que nous avons conçue pour présenter nos

expériences.

Cette application comporte deux champs principaux :

- **Entrée (audio) :** Permet de choisir n'importe quel fichier audio_test.wav.
- **Sortie (légende) :** Affiche la légende prédite par notre modèle (voir figure 3.6)

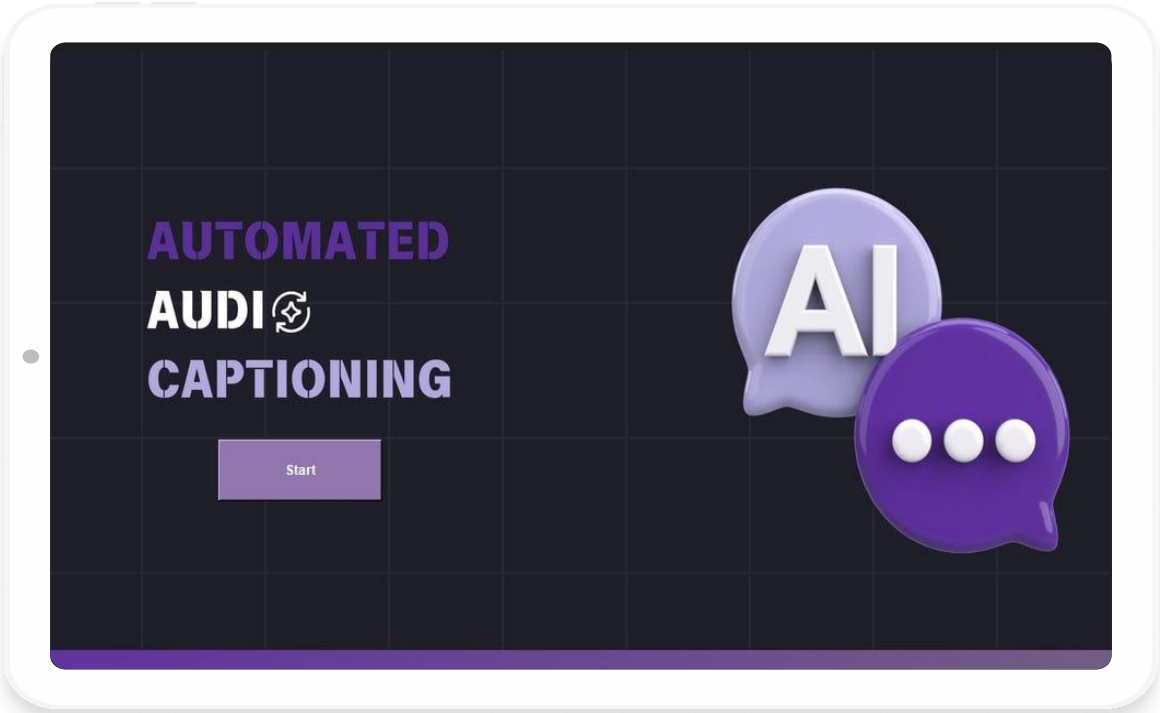


FIGURE 3.5 – Page d'acceuil de l'application

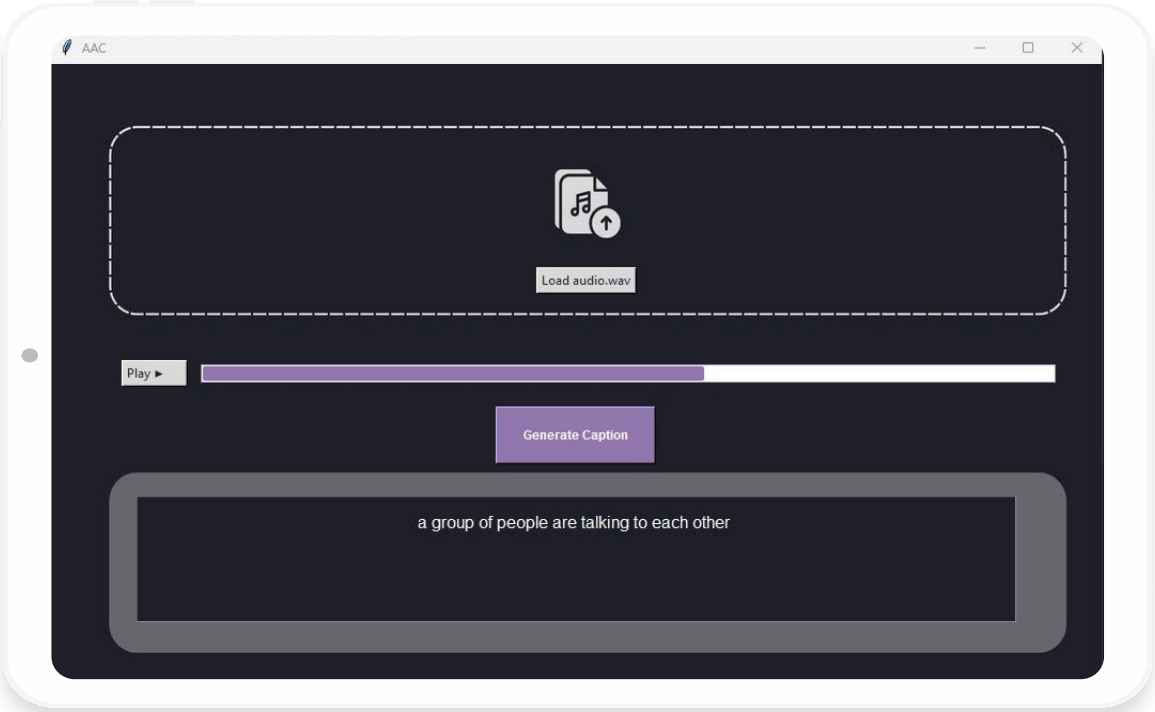


FIGURE 3.6 – Page d'acceuil de l'application(start)

Les résultats ci-dessous montrent les légendes générées par notre modèle pour différents fichiers audio. Ces résultats illustrent la capacité de notre modèle à décrire avec précision une variété de scénarios audio.

Fichier Audio	**Légende Générée**
test_0076.wav	a group of people are talking to each other
test_0107.wav	a bird chirps while water drips in the background
test_0131.wav	a person is knocking on a wooden floor
test_0148.wav	a washing machine is running at a constant rate
test_0153.wav	someone is flipping through the pages of a book

TABLE 3.6 – Les légendes générées par notre modèle pour différents fichiers audio

Ce tableau 3.6 présente les légendes générées par notre modèle pour plusieurs fichiers audio de test (issues de Clotho). Chaque légende est une description concise de l'événement audio présent dans chaque fichier. Par exemple, pour le fichier « test_0076.wav », le modèle a généré la légende « a group of people are talking to each other », ce qui indique que le modèle peut identifier et décrire avec précision des interactions humaines dans un environnement sonore.

Les résultats montrent que notre modèle est capable de fournir des descriptions claires et spécifiques pour divers types de sons, tels que le chant des oiseaux, les bruits de l'eau, le bruit des pas sur un sol en bois, le fonctionnement constant d'une machine à laver, et le feuilletage des pages d'un livre. Ces descriptions démontrent l'efficacité de notre modèle à capter les nuances des événements audio et à les traduire en légendes textuelles précises.

Cependant, bien que les résultats soient prometteurs, il est important de noter que des améliorations peuvent encore être apportées pour augmenter la précision et la richesse des descriptions générées. Des ensembles de données plus grands et plus diversifiés pourraient améliorer encore les performances de notre modèle.

3.9 Conclusion

Dans ce chapitre, nous avons abordé les aspects pratiques de la mise en œuvre de notre solution de génération automatique de légendes textuelles à partir de signaux audio. Après avoir établi les fondements théoriques et l'approche proposée dans les chapitres précédent, nous avons décrit en détail les outils et technologies utilisés pour développer notre système AAC. Nous avons commencé par passer en revue les différents outils de développement, puis nous avons présenté le jeu de données utilisé comme base pour nos expériences. Ensuite, nous avons exposé la procédure suivie pour l'analyse et l'évaluation des résultats obtenus, mettant en lumière les différentes étapes nécessaires à la mise en œuvre de nos méthodes. Enfin, nous avons illustré les résultats obtenus lors de la phase de test et notre interface utilisateur proposée, démontrant ainsi l'efficacité de notre approche.

Conclusion générale

Ce mémoire présente une approche novatrice d'architecture encodeur-encodeur pour relever le défi complexe de la génération automatique de légendes textuelles à partir de signaux audio. Notre étude vise à surmonter les principaux défis rencontrés, notamment l'ambiguïté et la subjectivité des signaux audio ainsi que l'alignement temporel précis. Nous avons réalisé notre expérience en utilisant l'ensemble de données Clotho.

Au cœur de notre contribution se trouve un mécanisme d'encodage audio robuste capable de capturer efficacement les caractéristiques complexes du signal. Couplé à un encodeur textuel conçu pour saisir les nuances sémantiques, notre architecture fusionne ces représentations audio et textuelles dans un espace commun de "joint embedding". Cette fusion synergique permet la génération de légendes cohérentes et fidèles au contenu audio source.

De plus, nous proposons une méthode d'alignement temporel fine, basée sur des techniques de traitement du signal et du langage naturel, assurant une synchronisation précise entre les segments audio et les mots des légendes.

L'architecture proposée repose sur un pipeline séquence-à-séquence intégrant un encodeur audio pré-entraîné CNN14, un encodeur textuel SBERT, et un décodeur BART pour la génération de légendes. Les résultats obtenus démontrent les performances prometteuses de notre approche.

En combinant des approches d'encodage audio et textuel de pointe avec un mécanisme robuste d'alignement temporel, ce mémoire positionne notre architecture encodeur-encodeur

comme une avancée majeure dans le domaine émergent de la génération automatique de légendes audio. Nous avons détaillé les différentes étapes de notre approche, de la recherche bibliographique initiale à la mise en œuvre pratique et à l'évaluation des résultats, en soulignant l'importance des outils et technologies utilisés.

Travaux futurs

Suite aux résultats prometteurs de notre approche actuelle, plusieurs axes de recherche peuvent être explorés pour optimiser davantage les performances et l'efficacité de la génération automatique de légendes audio. Voici quelques pistes d'amélioration :

1. Nouvelles Architectures de Réseaux : Expérimenter avec des décodeurs basés sur des réseaux de neurones convolutifs (CNN) pour évaluer leur impact sur la génération de légendes.

2. Intégration dans des Plateformes Grand Public : Déployer le système sur des plateformes populaires telles que des sites web, des applications mobiles ou des services de streaming. Cela permettrait à un public plus large d'accéder facilement à cette technologie, facilitant son adoption pratique.

3. Génération de Légendes en Temps Réel : Développer des algorithmes optimisés pour la génération de légendes en temps réel. Cela nécessiterait une réduction significative de la latence.

4. Capacité de Généralisation : Évaluer le système sur des ensembles de données audio provenant de divers domaines (musique, podcasts, vidéos éducatives, etc.) pour tester sa capacité de généralisation. Analyser les biais lexicaux et structurels pour améliorer l'équité et l'exactitude des légendes générées.

5. Personnalisation et Adaptation au Contexte : Intégrer des mécanismes de personnalisation permettant d'adapter les légendes aux préférences et aux besoins spécifiques des

utilisateurset développer des capacités de génération de légendes en plusieurs langues et intégrer des modules de traduction automatique pour les légendes générées. Cela permettrait de rendre le contenu audio accessible à des utilisateurs parlant différentes langues. Par exemple, ajuster le niveau de détail ou le style de langage utilisé.

En conclusion, cette étude ouvre de nouvelles perspectives passionnantes et inclusives dans le domaine de l'AAC, avec des applications potentielles dans divers domaines tels que l'éducation, le divertissement et la sécurité. Nous espérons que notre travail contribuera à faire progresser la recherche dans ce domaine prometteur et à rendre les contenus audio plus accessibles pour tous.

Bibliographie

[1] J. Sim, E. Kim, and K. Lee. Label-refined sequential training with noisy data for automated audio captioning, 2023.

[2] T. Schaumlöffel, M. G. Vilas, and G. Roig. Peacs : Prefix encoding for auditory caption synthesis, 2023.

[3] G. Karanth, N. Rao, S. Subramanian, and A. Shah. Dcase 2023 task 6 automated audio captioning and language-based retrieval, 2023.

[4] Deeply Learning. Fonctionnement du neurone artificiel. Repéré sur https ://deeplylearning.fr/cours-theoriques-deep-learning/fonctionnement-du-neurone-artificiel/.

[5] Analytics Vidhya. Activation functions and their derivatives : A quick complete guide, April 2021. Repéré sur https ://www.analyticsvidhya.com/blog/2021/04/activation-functions-and-their-derivatives-a-quick-complete-guide/.

[6] Université Batna 2. Deep learning (chapitre 2), 2020-2021. Cours pour les Master 2 SDS. Page 6.

[7] Convolutional neural network (cnn) architecture [image]. Repéré sur ResearchGate : https ://www.researchgate.net/figure/Convolutional-Neural-Network-CNN-Architecture-37$_fig3_3$47920378.

[8] TikZ. Autoencoder. Repéré sur https ://tikz.net/autoencoder/.

[9] A. Nama. Exploring the power of encoder-decoder models : Pros, cons, and applications,

2023. Repéré sur https ://medium.com/@anishnama20/exploring-the-power-of-encoder-decoder-models-pros-cons-and-applications-8bfbe2e66e76.

[10] J. Alammar. The illustrated transformateur. Repéré sur https ://jalammar.github.io/illustrated-Transformateur/.

[11] Électronique Mixte. Cours 1 : Conversions analogique-numérique et numérique-analogique. Repéré sur https ://www.electronique-mixte.fr/formation-pdf/cours-pdf-cours-divers-en-electronique/cours-1-conversions-analogique-numerique-et-numerique-analogique/.

[12] P. Dutertre. Conceptions avancées des circuits intégrés analogiques. convertisseurs a/n et n/a, 2009. Page 2.

[13] R. Weynand. Audio deep learning made simple (part 2) : Why mel spectrograms perform better, May 28 2020. Repéré sur https ://towardsdatascience.com/audio-deep-learning-made-simple-part-2-why-mel-spectrograms-perform-better-aad889a93505.

[14] M. Guedri. La reconnaissance des chiffres manuscrits isolés en utilisant l'apprentissage profond, 2019. Page 21.

[15] DataScientest. Deep neural network. Repéré sur https ://datascientest.com/deep-neural-network.

[16] Jedha. Comment se compose un réseau de neurones ? Repéré sur https ://www.jedha.co/formation-ia/reseau-neurones-deep-learningcomment-se-compose-un-reseau-de-neurones.

[17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

[18] M. Z. Mokri. Classification des images avec les réseaux de neurones convolutionnels, July 3 2017. Master en Informatique. Page 62.

[19] A. Medjoudja and M. Chibane. La reconnaissance de la langue des signes algérienne : La mise en place d'un système de traduction signes/mots en temps réel, June 24 2024. Master

Professionnel en Informatique, Spécialité : Administration et Sécurité des Réseaux. Pages 37-38.

[20] Intelligence Artificielle. Auto-encodeur : Guide complet. Repéré sur https ://intelligenceartificielle.com/auto-encodeur-guide-complet/.

[21] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.

[22] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008, 2017.

[23] B. S. Siddhartha. An interpretation of lemmatization and stemming in natural language processing. *Journal of University of Shanghai for Science and Technology*, 2021.

[24] INAGRM. La forme d'onde. Repéré sur https ://sites.inagrm.com/parcours/co/La$_{f}orme_{do}nde.html$.

[25] Sonic Studio. Mel frequency cepstral coefficient (mfcc). Repéré sur https ://www.sfu.ca/sonic-studio-webdav/handbook/Mel.html.

[26] Hugging Face. Audio data. Repéré sur https ://huggingface.co/learn/audio-course/fr/chapter1/audio$_{d}ata$.

[27] A. B. Louam. Deep learning basé sur les méthodes de réduction pour la reconnaissance de visage, July 6 2019.

[28] S. D. Saidj. Techniques de nlp pour la détection des fausses nouvelles, 2022. Mémoire de Master. Pages 17-18.

[29] DataScientest. Introduction au nlp (natural language processing). Repéré sur https ://datascientest.com/introduction-au-nlp-natural-language-processing.

[30] S. G. Koolagudi, D. Rastogia, and K. S. Rao. Identification of language using mel-frequency cepstral coefficients (mfcc). In *International Conference on Modelling, Optimisation and Computing (ICMOC)*, 2012.

[31] Ethen. Tf-idf (term frequency-inverse document frequency). Repéré sur https ://ethen8181.github.io/machine-learning/clustering$_{o}ld/tf_{i}df/tf_{i}df.html$.

[32] Aysel Aydin. 4 bag of words model in nlp. Repéré sur https ://ayselaydin.medium.com/4-bag-of-words-model-in-nlp-434cb38cdd1b.

[33] J. Samuels. One-hot encoding and two-hot encoding : An introduction. Repéré sur ResearchGate.

[34] N. Reimers and I. Gurevych. Sentence-bert : Sentence embeddings using siamese bert-networks, 2019.

[35] Devoteam France. Lstm, transformateurs, gpt, bert : guide des principales techniques en nlp. Repéré sur https ://france.devoteam.com/paroles-dexperts/lstm-Transformateurs-gpt-bert-guide-des-principales-techniques-en-nlp/.

[36] M. F. Sanner. Python : A programming language for software integration and development. 10550 North Torrey Pines Road, La Jolla, CA-92037.

[37] F. Font, G. Roma, and X. Serra. Freesound technical demo. In *Proceedings of the 21st ACM International Conference on Multimedia*, pages 411–412, New York, NY, USA, 2013. Association for Computing Machinery. Event-place : Barcelona, Spain.

[38] K. Wołk and K. Marasek. Enhanced bilingual evaluation understudy, June 2 2014.

[39] J. Rownlee. How to calculate bleu score for text in python. Repéré sur https ://machinelearningmastery.com/calculate-bleu-score-for-text-python/.

[40] JetBrains Academy. Natural language processing : Step 29669. Repéré sur https ://hyperskill.org/learn/step/29669.

[41] M. Kilickaya, A. Erdem, N. Ikizler-Cinbis, and E. Erdem. Re-evaluating automatic metrics for image captioning.

[42] S. Liu, Z. Zhu, N. Ye, S. Guadarrama, and K. Murphy. Improved image captioning via policy gradient optimization of spider. arXiv :1612.00370v4.

[43] C. Azencott. Introduction au machine learning.

[44] Pure Storage. What is machine learning pipeline ? Repéré sur https ://www.purestorage.com/fr/knowledge/what-is-machine-learning-pipeline.html.

[45] IBM. Machine learning pipeline. Repéré sur https ://www.ibm.com/topics/machine-learning-pipeline.

[46] datafranca.org. Réseau de neurones profond. Repéré sur https ://datafranca.org/wiki/R

[47] S. Janny, S. Nathan, and W. Shu-Quartier. Introduction à l'apprentissage automatique, May 24 2022. Pages 6-7.

[48] X. Xu, Z. Xie, M. Wu, and K. Yu. Beyond the status quo : A contemporary survey of advances and challenges in audio captioning, November 16 2023.

[49] Papers with Code. Audio captioning. Repéré sur https ://paperswithcode.com/task/audio-captioning.

[50] X. Mei, X. Liu, and M. D. Plumbley. Automated audio captioning : an overview of recent progress and new challenges. *J Audio Speech Music Proc*, 26, 2022.

[51] Hugging Face. Spectrogramme. Repéré sur https ://huggingface.co/learn/audio-course/fr/chapter1/audio$_d$*ataspectrogramme*.

[52] N. Berri. Utilisation des techniques de deep learning pour l'extraction des concepts à partir des documents textuels, 2019. Mémoire de Master académique en Informatique. Page 22.

[53] Pure Storage. What is machine learning pipeline ? Repéré sur https ://www.purestorage.com/fr/knowledge/what-is-machine-learning-pipeline.html.

[54] M. Chibi. Détection des piétons par apprentissage profond, 2022/2023. Mémoire de Fin d'études Master, Filière : Informatique, Option : Système Informatique. Page 35.

[55] N. Berri. Utilisation des techniques de deep learning pour l'extraction des concepts à partir des documents textuels, 2019. Mémoire de Master académique en Informatique. Page 22.

[56] K. Drossos, S. Lipping, and T. Virtanen. Clotho : an audio captioning dataset. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Barcelona, Spain, 2020.

[57] Enjoy Algorithms. Supervised, unsupervised, and semi-supervised learning. Repéré sur https ://www.enjoyalgorithms.com/blogs/supervised-unsupervised-and-semisupervised-learning.

[58] Arize AI. Meteor score. Repéré sur https ://arize.com/glossary/meteor-score/.