République Algérienne Démocratique et Populaire

Université A. MIRA de Béjaia

Faculté des Sciences Exactes
Département de Recherche Opérationnelle

Mémoire présenté pour l'obtention du diplôme de master

Spécialité : Sciences de Données et Aide à la Décision

Reconnaissance vocale des lettres de la langue Amazighe (Kabyle)

Présenté par :
TETAH Ikram
BOURNINE Ikram

Sous la direction de : Mr. A Zaidi et Dr. L Asli

Défendu le 03/07/2024, devant le jury composé de :

B.BRAHMI	M.C.A	Président de jury	UAMB - Bejaia.
M.M'HAMDI	M.C.A	Examinateur	UAMB - Bejaia.
L.HAMZA	M.C.A	Examinatrice	UAMB - Bejaia.

Année Universitaire 2023 – 2024

Remerciments

En premier lieu, nous remercions ALLAH, le tout puissant, de nous avoir appris ce que nous ignorons, de nous avoir donné la santé et tout dont nous avons besoin pour l'accomplissement de ce mémoire.

Nous exprimons notre gratitude envers monsieur Zaidi Ali pour être à l'origine de ce thème de recherche et pour ses conseils précieux, ses encouragements constants et son soutien continu.

Nous remercions Monsieur Asli Larbi pour ses conseils avisés, sa disponibilité, ses encouragements et son précieux accompagnement tout au long de ce travail.

Ce travail a été mené au sein du centre de recherche en langue et culture amazighes. Nous souhaitons exprimer notre respect et notre gratitude envers tous les membres du personnel du centre.

Nous remercions sincèrement toutes les personnes qui ont contribué à notre collecte de données. Leur aide a été essentielle à la réalisation de ce projet.

Nous remercions également tous les enseignants qui ont assuré notre formation. Leur dévouement et leurs connaissances ont été essentiels à notre parcours. Grâce à eux, nous avons pu acquérir les compétences nécessaires pour mener à bien ce projet.

On tient à remercier les membres du jury pour avoir accepté d'évaluer notre travail.

On remercie tout particulièrement nos parents, ainsi que nos frères et sœurs, pour leur soutien inconditionnel tout au long de ces longues années d'études.

À toutes les personnes qui ont contribué de près ou de loin à l'élaboration de ce travail, nous disons, MERCI.

Table des matières

Table des figures

List of Algorithms

Liste des tableaux

Liste d'abriviations

API : Application Programming Interface
ASR : Automatic Speech Recognition
AUC : Area Under the Curve
BRNN : Bidirectional Recurrent Neural Network
CPU : Central Processing Unit
CNN : Convolutional Neural Networks
db : Décibel
DCT : Transformée en Cosinus Discrète
DTW : Dynamic Time Warping
FFT : Fast Fourier Transform
FN : Faux Négatif
FP : Faux Positif
FSF : Free Software Foundation
GPU : Graphics Processing Unit
IA : Intelligence Artificielle
IDE : Integrated Development Environment
JSON : JavaScript Object Notation
LSTM : Long Short-Term Memory
MFCC : Mel-Frequency Cepstral Coefficients
MGM : Modèles Gaussiens Mixtes
ML : Machine Learning
MMC : Modèle de Markov caché
MSE : Mean Squared Error
RAP : Reconnaissance automatique de la parole
RAM : Random Access Memory
RNA : Réseau de neurones artificiel
ROC : Receiver Operating Characteristic
RNN : Recurrent Neural Network
SNR : Signal-to-Noise Ratio
TALN : Traitement automatique du langage naturel
TFCT : Transformée de Fourier à Court Terme
TFICT : Transformée de Fourier Inverse à Court Terme
VN : Vrai Négatif
VP : Vrai Positif
WAV : Waveform Audio File Format

Introduction générale

Si l'homme est capable de comprendre un message vocal provenant de n'importe quel locuteur, même dans des environnements bruyants et avec diverses élocutions, syntaxes et vocabulaires. La machine peut-elle accomplir la même tâche ? Existe-t-il une solution capable de surmonter globalement ces défis ? La reconnaissance vocale est la solution technologique à cet enjeu, permettant aux systèmes informatiques d'interpréter le langage parlé aussi efficacement que les êtres humains, quel que soit le contexte.

Notre étude s'intègre dans le cadre du développement d'un modèle de reconnaissance vocale des lettres de la langue kabyle avec pour objectif d'interpréter et de transcrire de manière précise les phonèmes et les caractères spécifiques à cette langue. Le kabyle est une variante de la langue amazighe parlée principalement en Algérie, connue pour sa richesse linguistique et ses particularités phonétiques. Cette langue utilise un alphabet berbère adapté qui inclut des caractères spécifiques pour représenter des sons qui peuvent différer de ceux des autres langues.

La reconnaissance vocale passe par un processus complexe qui implique plusieurs étapes essentielles, telles que la collecte d'enregistrements vocaux représentant des phonèmes, des mots, des verbes et des phrases en kabyle. Ces enregistrements sont réalisés par des locuteurs de différentes régions, âges, sexes et statut social pour couvrir la variété linguistique et accentuelle de la langue. Une fois collectées, les données vocales doivent être prétraitées avant de pouvoir être efficacement utilisées pour la reconnaissance vocale.

L'une des étapes clés de ce prétraitement est l'extraction des caractéristiques acoustiques [51] qui permettent de transformer directement les enregistrements bruts en caractéristiques plus faciles à manipuler et à analyser, en capturant les propriétés spectrales essentielles du signal vocal.

La reconnaissance vocale des lettres a fait l'objet d'études approfondies ces dernières années, avec des chercheurs utilisant différents algorithmes tels que les machines à vecteurs de support (SVM), la comparaison dynamique, les modèles cachés de Markov (MMC) [55], les réseaux neuronaux artificiels (RNA) et les réseaux de neurones convolutionnels. Les résultats ont été variés et satisfaisants, mais ces méthodes partagent certains inconvénients communs. Elles nécessitent généralement un grand nombre de données étiquetées pour un entraînement efficace et des résultats précis. De plus, toutes ces méthodes sont sensibles aux conditions d'enregistrement vocal, telles que le bruit de fond, la qualité de l'enregistrement, les variations dans la prononciation, ainsi que la complexité du signal vocal.

Dans ce travail nous centralisons notre objectif à réaliser un système de reconnaissance vocale des lettres de la langue kabyle en utilisant un modèle de réseau de neurones récurrents bidirectionnel LSTM, en raison de sa capacité à modéliser les séquences temporelles complexes et à capturer les variations phonétiques subtiles propres à cette langue. Le LSTM seul permet au modèle de comprendre les dépendances à long terme dans les séquences vocales [64], tandis que la bidirectionnalité lui permet d'utiliser à la fois le contexte précédent et suivant pour améliorer la précision de la reconnaissance des lettres kabyles [14].

Ce mémoire commence par une introduction générale. Par la suite, il est réparti en cinq chapitres :

— **Le premier chapitre :** nous présentons brièvement les concepts de base liés à l'intelligence artificielle, au machine learning, au deep learning, au traitement automatique du langage naturel et à la reconnaissance automatique de la parole. Ce chapitre aborde également le phénomène de la production de la parole ainsi que les différentes caractéristiques d'un signal vocal.

— **Le deuxième chapitre :** traite de la langue Amazighe en mettant l'accent sur la variante kabyle, y compris sa répartition géographique, son système d'écriture en alphabet latin et ses phonèmes spécifiques.

— **Le troisième chapitre :** décrit les étapes essentielles de la création d'un corpus vocal, allant de la collecte des données à leur prétraitement préliminaire. Nous aborderons également la création détaillée du dataset vocal, y compris l'organisation des fichiers audio et leurs transcriptions.

— **Le quatrième chapitre :** couvre les étapes du prétraitement du signal audio, l'extraction des caractéristiques acoustiques, ainsi que les techniques de reconnaissance vocale.

— **Le cinquième chapitre :** est dédié à la description et à l'implémentation de notre modèle de reconnaissance vocale des lettres en kabyle, ainsi qu'à la présentation des résultats obtenus.

Le mémoire se termine par une conclusion générale et quelques perspectives.

1

Généralités sur la reconnaissance automatique de la parole & problématique

Introduction

Le présent chapitre a pour objectif de présenter les notions élémentaires liées à l'intelligence artificielle, au machine learning, au deep learning, au traitement automatique du langage et à la reconnaissance automatique de la parole. Ces concepts sont essentiels pour comprendre le fonctionnement et les applications de la reconnaissance vocale, ainsi que ses implications dans le contexte de la langue kabyle. De plus, nous aborderons également la problématique posée dans le cadre de notre projet.

Sommaire

1.1 Concepts de base de l'intelligence artificielle

Définition 1. ***Intelligence Artificielle*** *:*

L'intelligence artificielle (IA) est un terme qui a été inventé pour la première fois par ***John McCarthy*** *en 1956, il a défini le sujet comme "La science et l'ingénierie de la fabrication de machines intelligentes, en particulier des programmes informatiques intelligents".*

L'IA est une science technique qui étudie et développe des théories, des méthodes, des technologies et des applications pour permettre aux machines de fonctionner avec une pensée semblable à celle des êtres humains, rendant ainsi les machines capables de prendre des décisions intelligentes et d'interagir de manière plus sophistiquée avec leur environnement. Ainsi, l'intelligence artificielle repose sur un fondement théorique solide établi par Alan Turing le père de l'informatique moderne [39].

FIGURE 1.1 – Intelligence Artificielle.

Définition 2. ***Turing Test*** *:*

Pour évaluer si une machine peut être considérée comme intelligente ou non, ***Alan Turing*** *le célèbre mathématicien, a proposé une approche empirique connue sous le nom de "The Imitation Game".*

Dans le cadre du jeu de l'imitation, un juge (C) interagit par des échanges de messages écrits avec deux interlocuteurs anonymes, A et B, sans savoir qui est l'être humain et qui est la machine. Après plusieurs échanges, le juge doit déterminer l'identité de chaque interlocuteur.

Si la machine parvient à tromper le juge en se faisant passer pour un être humain, elle est considérée comme intelligente selon Turing [28] [60].

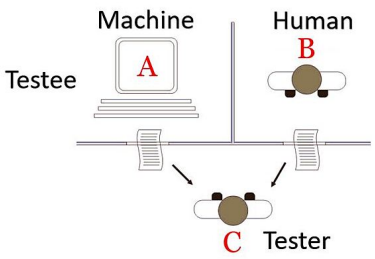


FIGURE 1.2 – Turing Test.

1.1.1 Domaines d'application de l'intelligence artificielle

L'intelligence artificielle révolutionne différents aspects de notre vie. Ci-dessous, nous citons quelques domaines où son impact est le plus significatif :

— Sécurité.
— Robots intelligents.
— Voitures autonomes.
— Maisons intelligentes.
— Traduction automatique.
— Achat en ligne.
— Casques de réalité virtuelle.
— Système de Positionnement Global.

FIGURE 1.3 – Domaines d'application de l'IA.

1.2 Machine Learning

Définition 3. ***Machine Learning***

Machine learning (ML) est un sous domaine de l'intelligence artificielle. Il explore la construction et l'étude d'algorithmes capables d'apprendre à partir de données et de faire des prédictions.

Il est étroitement lié au domaine des statistiques [15], c'est-à-dire ils utilisent des techniques statistiques pour permettre à un système d'apprendre des modèles et des relations à partir de données, qui permettent aux machines d'effectuer des tâches qui ne seraient autrement possibles que pour les humains, telles que la catégorisation d'images, l'analyse de données ou la prédiction des fluctuations de prix.

1.2.1 Techniques du Machine Learning

Apprentissage supervisé

L'apprentissage supervisé est une méthode d'apprentissage automatique où un algorithme apprend à partir de données étiquetées, constituées de paires d'entrées et de sorties attendues.

Les tâches les plus courantes dans ce cadre sont la "classification", qui consiste à séparer les données en catégories distinctes, et la "régression", qui vise à ajuster les données à une courbe ou à une fonction.

Par exemple, prédire l'étiquette de classe ou le sentiment associé à un morceau de texte tel qu'un tweet ou une critique de produit représente un cas concret de classification de texte, relevant de l'apprentissage supervisé [23].

Apprentissage non supervisé

L'apprentissage non supervisé, aussi appelé clustering, consiste à entraîner un modèle sur un ensemble de données non étiquetées. Étant donné que les exemples ne sont pas étiquetés, il n'y a pas de signal d'erreur ou de récompense pour évaluer une solution potentielle. Cette méthode peut être utilisée soit comme un objectif en soi, soit comme une étape de prétraitement pour un algorithme supervisé [15].

Par exemple, dans le cas d'un ensemble d'images de chiffres manuscrits, une méthode d'apprentissage non supervisé peut identifier 10 groupes de données. Ces groupes peuvent correspondre aux 10 chiffres distincts de 0 à 9. Cependant, étant donné que les données d'entraînement ne sont pas étiquetées, le modèle résultant ne peut pas fournir de signification sémantique aux groupes identifiés [31].

Apprentissage semi-supervisé

L'apprentissage semi-supervisé exploite à la fois des données étiquetées et non étiquetées. Cette combinaison permet de générer un modèle adapté à la classification des données [31].

En général, les données étiquetées sont rares tandis que les données non étiquetées sont abondantes. L'objectif de la classification semi-supervisée est de développer un modèle capable de prédire les classes des futures données de test de manière plus précise que celui obtenu en utilisant uniquement les données étiquetées [23].

Apprentissage par renforcement

Reinforcement learning est un type d'algorithme d'apprentissage automatique qui permet aux agents logiciels et aux machines d'évaluer automatiquement le comportement optimal dans un contexte ou un environnement particulier pour améliorer leur efficacité. Il s'agit d'une approche axée sur l'environnement, où l'apprentissage est basé sur des récompenses ou des pénalités.

Le but ultime est d'utiliser les informations obtenues de l'interaction avec l'environnement pour prendre des mesures qui maximisent les récompenses ou minimisent les risques.

C'est un outil puissant pour entraîner des modèles d'IA qui peuvent aider à accroître l'automatisation ou à optimiser l'efficacité opérationnelle de systèmes sophistiqués tels que la robotique, les tâches de conduite autonome, la fabrication et la logistique de la chaîne d'approvisionnement [23].

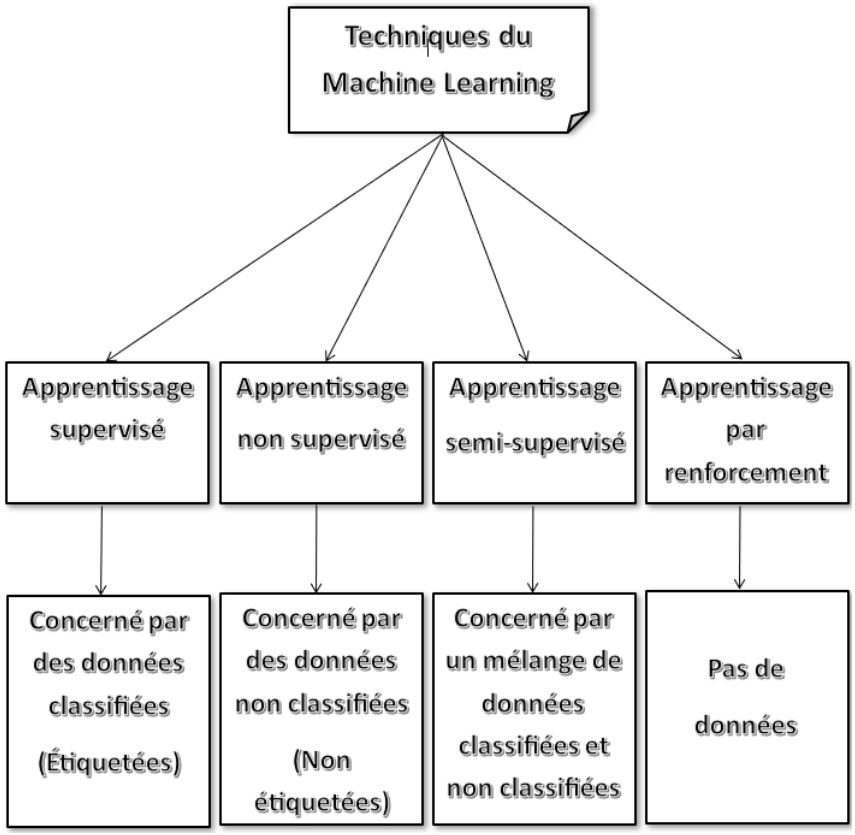


FIGURE 1.4 – Techniques du machine learning [43].

1.3 Deep Learning

Définition 4. ***Deep Learning*** *:*

Le deep learning, ou apprentissage profond, représente une branche de l'intelligence artificielle qui évolue à partir du machine learning (apprentissage automatique). Contrairement à une programmation classique où la machine suit des instructions préétablies, le deep learning permet à la machine d'apprendre de manière autonome.

Il repose sur l'utilisation de réseaux de neurones artificiels, inspirés du fonctionnement du cerveau humain. Ces réseaux sont structurés en plusieurs couches de neurones, pouvant aller de quelques dizaines à plusieurs centaines. Chaque couche reçoit et traite les informations provenant de la couche précédente.

Par exemple, le système peut apprendre à reconnaître des lettres avant de comprendre des mots dans un texte, ou à détecter la présence d'un visage sur une photo avant d'identifier la personne représentée [27].

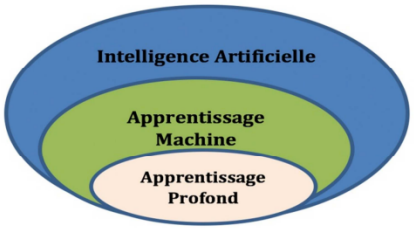


FIGURE 1.5 – Deep learning [27].

1.3.1 Réseaux de neurones

Un réseau neuronal est constitué de multiples neurones qui sont des unités de calcul simples connectés les uns aux autres, ces neurones sont organisés en couches. En associant les neurones, le réseau peut simuler une fonction, c'est-à-dire approximer une relation entre plusieurs paramètres, en calculant des valeurs de sortie à partir de valeurs d'entrée.

La capacité d'un réseau neuronal à mémoriser des paires de valeurs d'entrées et de valeurs de sortie augmente avec le nombre de neurones présents dans le réseau. Le réseau peut apprendre cette fonction en lui présentant des exemples, c'est-à-dire des jeux de données constitués de valeurs d'entrées et de valeurs de sortie correspondantes. Il peut également continuer à apprendre en temps réel de nouveaux exemples, pour autant que ces exemples soient appropriés.

En agissant de cette manière, le réseau neuronal peut simuler une forme d'intelligence humaine, en répondant à des situations avec des actions appropriées, à travers le mécanisme d'action-réaction [27].

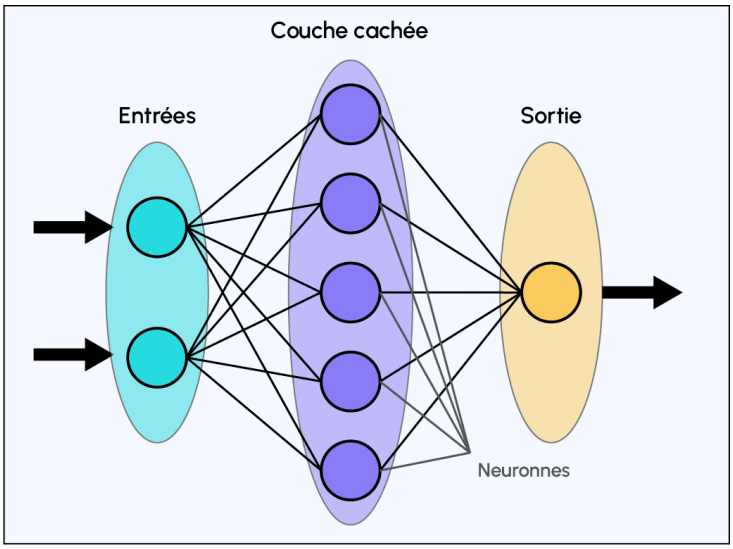


FIGURE 1.6 – Forme d'un réseau de neurones.

1.4 Traitement automatique du langage naturel

Dans l'étude du langage, quelques définitions clés sont importantes à comprendre :

Définition 5. ***Langue*** *:*

La langue, selon ***Ferdinand de Saussure****, est définie comme un produit social résultant de la faculté de langage, ainsi qu'un ensemble de conventions établies par la société pour permettre l'utilisation de cette faculté par les individus. En d'autres termes, la langue représente l'ensemble des signes linguistiques utilisés par un individu pour communiquer et s'exprimer avec les autres [48] .*

Définition 6. ***Phonétique***

Plusieurs définitions ont été données à cette branche. ***Jean Pierre Cuq*** *déclare que la phonétique est : « la discipline qui étudie la composante sonore d'une langue dans sa réalisation concrète, des points de vue acoustique, physiologique (articulatoire) et perceptif (auditifs) ». Alors la phonétique s'intéresse à l'étude des sons de la parole [26].*

Les branches de la phonétique sont :

- **La phonétique articulatoire** : c'est une activité qui s'occupe des actions des cordes vocales de la bouche.
- **La phonétique acoustique** : cette phonétique concerne l'action de transférer un message sonore pour que l'auditeur le reçoive.
- **La phonétique auditive** : c'est la branche qui examine comment l'appareil auditif reçoit un message.

Définition 7. ***Phonème***

Le phonème est la plus petite unité distinctive et significative de la parole, capable d'être identifiée et isolée au sein d'une chaîne parlée [26].

Définition 8. ***Mot*** *:*

Un mot est une unité linguistique constituée d'une ou plusieurs lettres qui, lorsqu'elle est prononcée ou écrite, a une signification ou une fonction grammaticale dans une langue donnée.

Définition 9. ***Phrase*** *:*

D'un point de vue strictement linguistique, en mettant de côté toute considération logique ou psychologique, ***Antoine Meillet*** *définit la phrase comme un ensemble d'articulations liées entre elles par des rapports grammaticaux et qui, ne dépendant grammaticalement d'aucun autre ensemble, se suffisent à elles-mêmes [38].*

Définition 10. ***Traitement automatique du langage naturel*** *:*

Le traitement automatique du langage naturel (TALN) est un domaine à la frontière de la linguistique et l'informatique, il a pour objectif de développer des logiciels capable de traiter de façon automatique des données linguistiques exprimées dans une langue naturelle donnée et pour une application bien définie. Cet objectif passe nécessairement par l'explicitation des règles de la langue, leur représentation dans un formalisme calculable, puis leur implémentation à l'aide de programmes informatiques [17].

1.4.1 Fonctionnement du traitement automatique du langage naturel

Les approches du traitement automatique du langage naturel, ont évolué de méthodes basées sur des règles classiques vers des techniques modernes basées sur le deep learning. Alors que les approches antérieures du TALN utilisaient des algorithmes basés sur des règles pour identifier des mots et des phrases dans le texte, le deep learning repose sur des modèles complexes qui apprennent à comprendre le langage naturel à partir de vastes ensembles de données étiquetées.

Cependant, l'obtention de ces données étiquetées en quantité suffisante constitue l'un des principaux défis actuels du TALN [36].

Trois outils largement utilisés dans ce domaine :

— **NLTK** : est un module Python open source qui fournit des ensembles de données et des didacticiels pour le TALN.
— **Gensim** : est une bibliothèque Python spécialisée dans la modélisation de sujets et l'indexation de documents.
— **Intel TALN Architect** : offre des outils avancés pour les topologies et les techniques d'apprentissage en profondeur dans le cadre du TALN.

1.4.2 Domaines d'application du TALN

Le traitement automatique du langage naturel trouve des applications dans divers domaines tels que la santé, la finance, le commerce et la recherche scientifique pour améliorer les performances et l'efficacité dans divers aspects de la vie quotidienne et des entreprises. Le tableau ci-dessus illustre quelques-unes :

Domaines	Définitions
Classification de texte	La classification de texte est une tâche fondamentale en traitement automatique du langage naturel qui consiste à catégoriser les données textuelles en étiquettes prédéfinies ou en catégories en fonction de leur contenu et de leur contexte.
Texte prédictif	Le TALN est utilisé dans les applications de texte prédictif sur les smartphones, où le système suggère des mots en fonction de la saisie de l'utilisateur, apprenant des motifs de texte pour fournir des prédictions de mots précises au fil du temps.
Analyse des sentiments	Il permet l'analyse des sentiments, permettant aux entreprises d'évaluer les réactions des clients, de surveiller les mentions sur les médias sociaux et de comprendre le sentiment des clients à l'égard des produits ou services, ce qui aide à prendre des décisions éclairées.
SpeechToText/TextToSpeech	Transformer les commandes vocales en texte écrit et vice versa.
Résumé automatique	Le TALN facilite le résumé automatique de documents texte, en extrayant les informations clés pour fournir des résumés concis, ce qui aide à parcourir efficacement de vastes quantités de données.
Filtrage des e-mails	Il est utilisé pour catégoriser, trier et filtrer les e-mails, distinguant entre les messages importants et le spam, améliorant ainsi la fiabilité et la sécurité de la communication par e-mail.
Assistants intelligents	Il alimente des assistants intelligents comme Siri, Alexa et Cortana, permettant aux utilisateurs d'interagir avec des appareils en utilisant un langage naturel, améliorant l'expérience utilisateur et permettant une communication efficace avec les machines.

TABLE 1.1 – Domaines d'application du TALN.

1.5 Reconnaissance automatique de la parole

Les premiers systèmes de reconnaissance vocale datent des années 1950, avec l'IBM Shoebox capable de reconnaître 16 mots. Dans les années 1970, le système Harpy de l'université Carnegie Mellon comprenait environ 1000 mots. Aujourd'hui, des systèmes modernes comme Siri, Alexa et Google Assistant utilisent l'apprentissage profond et les réseaux de neurones pour offrir une reconnaissance vocale très précise et comprendre des commandes complexes.

1.5.1 Mécanisme de production de la parole

La production de la parole est un processus complexe qui implique une interaction entre les systèmes neurologiques et physiologiques. Initialement, cela démarre par une activité neurologique, suivie de l'activation des organes phonatoires par le cerveau. Le fonctionnement de ces organes est principalement de nature physiologique [63].

Le fonctionnement de l'appareil phonatoire humain repose sur l'interaction entre trois entités : les poumons, le larynx et le conduit vocale.

Le processus de production de parole peut être résumé en trois étapes essentielles :

— La création d'un flux d'air qui servira de base à la production sonore.
— La source sonore est produite soit par la vibration régulière des cordes vocales, soit par un blocage partiel du conduit vocal, ce qui crée un bruit.
— Les cavités au-dessus des cordes vocales sont ajustées pour produire le son voulu.

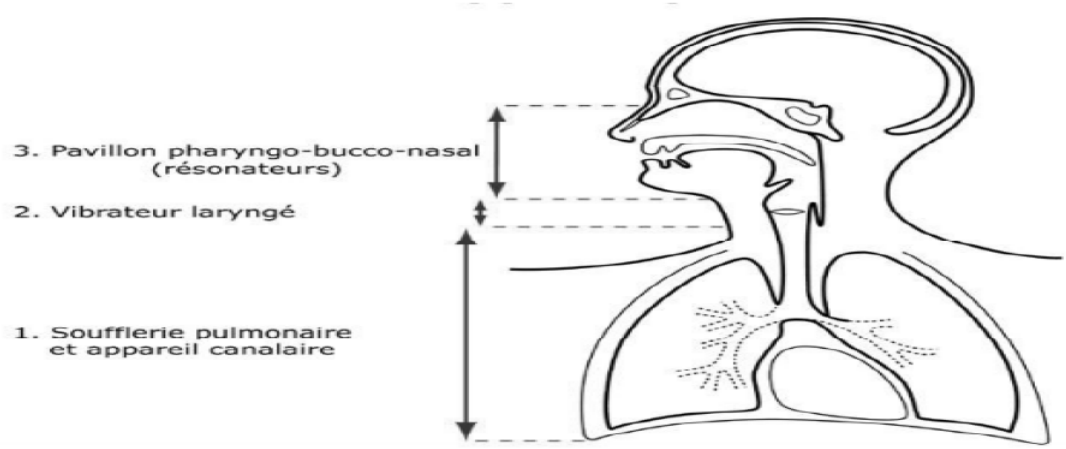


FIGURE 1.7 – Schéma de l'appareil phonatoire [51].

Définition 11. ***La parole*** *:*

La parole est un signal continu, d'énergie finie, non stationnaire. Sa structure est complexe et variable dans le temps.

L'information portée par le signal de parole peut être analysée par plusieurs façons. On en distingue généralement plusieurs niveaux de description non exclusifs : acoustique, phonétique, phonologique, morphologique, syntaxique, sémantique et pragmatique [63].

— **Acoustique :** elle se concentre sur la description et l'analyse des signaux sonores produits par l'appareil vocal humain lors de la parole.

— **Phonétique :** comme cité précédemment, la phonétique étudie les sons de la parole humaine, leur production, leur transmission et leur perception .

— **Phonologique :** la phonologie étudie comment les sons sont utilisés dans les langues pour distinguer les mots et créer du sens.

— **Morphologique :** la morphologie étudie la structure des mots et les règles de leur formation.

— **Syntaxique :** elle étudie la syntaxe d'une langue et ses règles grammaticales.

— **Sémantique :** c'est la branche de la linguistique qui étudie le sens des mots, des phrases et des textes.

— **Pragmatique :** la pragmatique étudie comment le contexte et l'intention des locuteurs influencent le sens et l'interprétation des énoncés.

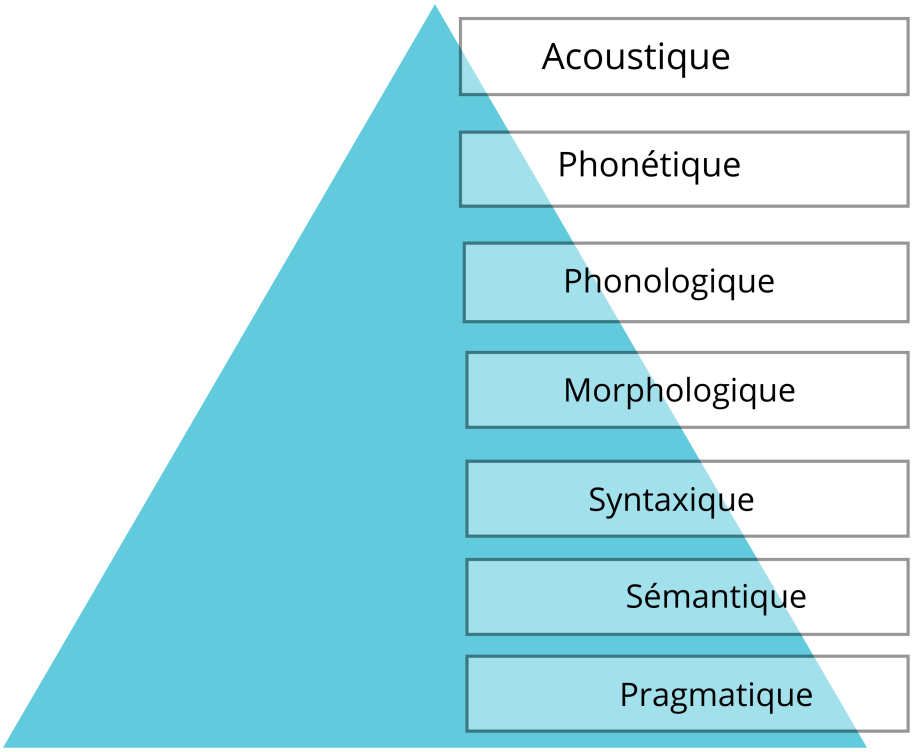


FIGURE 1.8 – Niveaux d'analyse du signal de parole [51].

1.5.2 Paramètres acoustiques du signal de parole

Le signal parole est généralement caractérisé par trois paramètres [51] :

La fréquence fondamentale

Elle représente la fréquence du cycle d'ouverture/fermeture des cordes vocales. Elle varie :

— De 80Hz à 200Hz pour une voix masculine.

— De 150Hz à 450Hz pour une voix féminine.

— De 200Hz à 600Hz pour une voix d'enfant.

L'énergie

L'amplitude du signal de la parole varie au cours du temps selon le type de son, son énergie dans une trame est donnée par :

$$E = \sum_{n=0}^{N-1} S^2(n) \tag{1.1}$$

Avec :

S : signal de parole.

N : taille de la trame.

n : chaque échantillon de signal à l'intérieur de la trame.

E : énergie du signal.

Le spectre

Le spectre représente l'intensité de la voix selon la fréquence, elle est généralement obtenue par une analyse de fourier à court terme.

La quasi-stationnarité du signal vocal signifie que ses caractéristiques restent assez constantes sur de courtes périodes, ce qui permet d'utiliser des fenêtres de temps d'environ 20 à 30 millisecondes, appelées trames, pour analyser et modéliser le son. Ces fenêtres se chevauchent pour assurer une analyse temporelle continue.

La transformée de Fourier à court terme (TFCT) d'un signal échantillonné est par définition la transformée du signal pondéré.

$$\hat{S}(k) = \hat{S}\left(f = \frac{k}{N}\right) = \sum_{n=0}^{N-1} S(n) \cdot W(n) \cdot \exp\left(-\frac{2J\pi nk}{N}\right), \quad 0 \leq k \leq N \tag{1.2}$$

— N : la taille du trame.

— $\hat{S}(k)$: spectre complexe.

— $S(n)$: segment analysé.

— $W(n)$: fenêtre de temps.

Le spectre de puissance (appelé aussi densité spectrale de puissance de la transformée de Fourier) est donné par :

$$|\hat{S}(k)|^2, \quad 0 \leq k \leq \frac{N}{2} \tag{1.3}$$

Définition 12. ***Reconnaissance automatique de la parole :***

La reconnaissance automatique de la parole (RAP), ou automatic speech recognition (ASR) en anglais, est la tâche dédiée à l'extraction automatique de paramètres d'un flux de parole composé des signaux acoustiques.

Le signal brut étant impossible à traiter, les informations pertinentes sont d'abord échantillonnées en trames ou vecteurs afin d'extraire les paramètres acoustiques.

L'approche principale utilisée dans ces systèmes est basée sur l'approche probabiliste avec des techniques d'apprentissage automatique. Toutefois, les approches neuronales ont démontré des résultats intéressants ces dernières années dans diverses tâches liées à la RAP [6].

1.5.3 Fonctionnement d'un système de reconnaissance de parole

La reconnaissance automatique de la parole est une technique informatique qui permet d'analyser la voix humaine captée au moyen d'un microphone pour la transcrire sous la forme d'un texte exploitable par la machine [51].

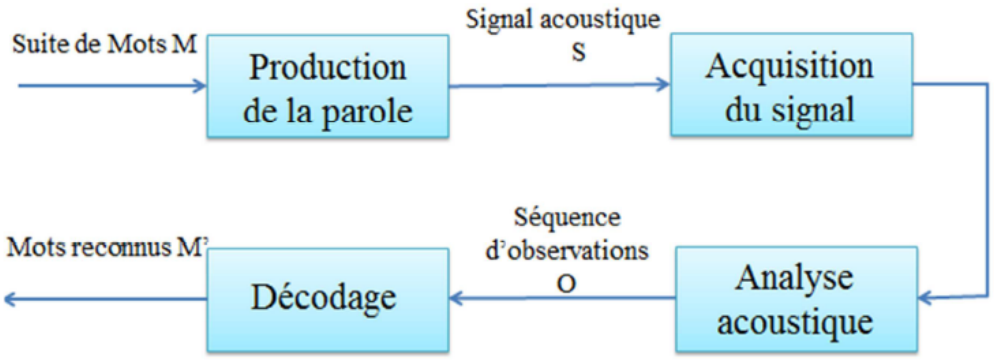


FIGURE 1.9 – Principe de fonctionnement [51].

La suite de mots prononcés M est convertie en un signal acoustique S par l'appareil phonatoire. Ensuite le signal acoustique est transformé en une séquence de vecteurs ou d'observations O, ces vecteurs représentent les caractéristiques acoustiques du signal à différents moments dans le temps.

Finalement, le module de décodage consiste à associer à la séquence d'observations O une séquence de mots reconnus M', en adaptant une stratégie de comparaison bien définie, qui peut impliquer l'utilisation de modèles linguistiques, de modèles acoustiques et d'algorithmes de reconnaissance de motifs pour trouver la meilleure correspondance entre les observations et les mots possibles.

1.5.4 Composition d'un système de reconnaissance automatique de la parole

Un programme typique de reconnaissance automatique de la parole est composé des étapes suivantes :

1. Prétraitement du signal audio qui consiste à normaliser, reduire le bruit des signaux audio et ainsi de suite.
2. Extraction des paramètres caractéristiques (MFCC, LPC, PLP, ...)
3. Choix d'une méthode de reconnaissance (CNN, RNN, HMM, ...) pour la reconnaissance des caractéristiques extraites.
4. Evaluation de la reconnaissance.

1.5.5 Mesures de performance d'un système de reconnaissance automatique de la parole

Il existe plusieurs valeurs mesurant les performances d'un système de reconnaissance automatique de la parole :

— **Taux de reconnaissance** : le nombre ou le pourcentage de mots parfaitement reconnus.

— **Taux de substitution** : le nombre ou le pourcentage de mots pour lequels le système fait erreur de reconnaissance.

— **Taux de rejet** : le nombre ou le pourcentage de mots que le sysème n'a pas compris.

— **Taux d'omission** : le pourcentage de mots non détectés.

— **Taux d'insertion** : le nombre ou le pourcentage de réponses inopinées.

1.6 Présentation du sujet

Notre projet vise à créer un corpus vocal dédié à la langue kabyle que nous utiliserons pour concevoir un système de reconnaissance vocale des lettres dans cette langue qui est souvent oubliée dans le domaine technologique. Notre projet sera le point de départ vers une transcription complète de toute communication vocale en kabyle.

1.7 Contexte théorique

La création d'un corpus vocal dédié à la langue kabyle et la conception d'un modèle de reconnaissance vocale efficace exigent une expertise approfondie en linguistique, traitement du signal, deep learning, et adaptation aux spécificités phonétiques et grammaticales du kabyle. Ce processus complexe vise à garantir que le modèle final soit robuste et précis pour identifier distinctement les lettres spécifiques de cette langue.

1.8 Problématique

La reconnaissance automatique des phonèmes en langue kabyle est confrontée à plusieurs défis majeurs en raison de ses caractéristiques intrinsèques et des particularités de la langue.

— La langue n'est pas aussi largement connue ou parlée que certaines autres langues notamment celles avec une diffusion plus mondiale.

— le manque de données linguistiques disponibles sur la langue kabyle.

— **Ambiguïté phonémique :** la langue kabyle présente des ambiguïtés au niveau phonémique, ce qui rend difficile la reconnaissance et l'interprétation automatiques des sons.

— **Ambiguïté syntaxique :** les structures syntaxiques complexes de la langue kabyle peuvent conduire à plusieurs interprétations possibles d'une même séquence de phonèmes, ajoutant ainsi une complexité à la reconnaissance automatique des phonèmes.

— **Coarticulation :** comme dans toute langue, la coarticulation est présente en kabyle, où chaque phonème est influencé par ceux qui l'entourent, rendant ainsi la reconnaissance précise des phonèmes plus difficile.

— **Variabilité intra-locuteur :** la prononciation des phonèmes en kabyle peut varier considérablement même chez un seul locuteur, en raison de différents modes d'élocution tels que la voix chantée, criée, murmurée, ou affectée par des conditions telles qu'un rhume, une irritation de la gorge, le stress ou le bégaiement.

— **Variabilité inter-locuteurs :** En plus de la variabilité intra-locuteur, la variabilité entre locuteurs, comprenant des différences de timbres vocaux, de voix masculines, féminines et d'enfants [33], ajoute une complexité supplémentaire à la reconnaissance automatique de la parole en Kabyle.

Pour remédier à ces problèmes, notre objectif est de développer un modèle de reconnaissance vocale capable de comprendre tous les dialectes kabyles, en se concentrant spécifiquement sur les phonèmes. Pour atteindre une performance optimale, nous incluons des voix provenant de divers âges, genres et régions. En créant un corpus linguistique varié, nous assurons une reconnaissance précise et efficace.

Conclusion

Ce chapitre a exploré les bases de l'intelligence artificielle, de l'apprentissage automatique, de l'apprentissage profond, du traitement automatique du langage naturel et de la reconnaissance automatique de la parole.

Il a abordé le processus humain de production de la parole, les paramètres acoustiques de la parole et l'importance de bien comprendre ces éléments pour analyser efficacement les caractéristiques du signal vocal et identifier la source d'excitation. Pour faciliter cette compréhension, une brève description de l'appareil phonatoire a été incluse, permettant ainsi de mieux appréhender les phénomènes liés à la production de la parole et leurs connexions.

Le chapitre suivant sera consacré spécifiquement à la langue kabyle, explorant ses particularités phonétiques, grammaticales et linguistiques. Cette exploration permettra de mieux comprendre les défis et les opportunités liés à l'application des technologies de reconnaissance vocale à cette langue.

2
Langue Kabyle

Introduction

La langue Amazighe, est un riche héritage linguistique d'Afrique du Nord. Elle se déploie en une mosaïque de dialectes variés, parmi lesquels la langue kabyle occupe une place importante.

Ce chapitre explore la diversité linguistique Amazighe, en mettant un accent particulier sur la langue kabyle et ses multiples facettes. Nous examinerons la répartition géographique des dialectes amazighs et kabyles, en soulignant les régions spécifiques où ces langues sont parlées. De plus, nous plongerons dans le système d'écriture du kabyle, notamment l'alphabet latin, et nous détaillerons les phonèmes particuliers qui caractérisent cette langue.

Sommaire

2.1 La langue amazighe à travers le temps

La langue amazighe, également connue sous le nom de tamazight, est une langue polynomique avec de nombreuses variantes parlées dans différentes régions d'afrique du nord et du sahara. Bien que chaque dialecte ait ses propres caractéristiques, ils partagent tous une origine commune.

Historiquement, l'amazighe a été écrite dans divers systèmes d'écriture, notamment l'alphabet libyco-berbère et l'alphabet arabe, et aujourd'hui, l'alphabet latin est également utilisé pour certains dialectes [46].

La colonisation européenne à partir du XIXe siècle, en particulier par la France et l'Espagne, a eu des impacts significatifs sur la langue amazighe et les politiques linguistiques dans la région. Les colonisateurs ont souvent promu les langues européennes au détriment de l'amazighe, entraînant un déclin de son utilisation. Cependant, malgré ces tentatives visant à éradiquer la langue, elle a persisté et même été revitalisée grâce aux mouvements nationalistes et aux initiatives d'associations culturelles.

Des progrès significatifs ont été réalisés en termes de reconnaissance officielle de la langue amazighe. Par exemple, le Maroc lui a accordé le statut de langue officielle en 2001, et en Algérie, elle a été reconnue comme langue nationale en 2002, devenant officielle en 2016.

Cependant, la diversité des dialectes, tels que le tamazight, le tachelhit et le tamahaq, présente des défis pour la standardisation et la promotion de cette langue. Le processus de standardisation de l'amazighe est en évolution, visant à établir une langue unifiée pour faciliter son enseignement et son utilisation dans divers secteurs tels que l'éducation et les médias.

Jusqu'à ce jour, la langue amazighe n'a pas cessé d'évoluer malgré les circonstances. Son développement continu témoigne de sa vitalité et de sa capacité à s'adapter aux changements socio-politiques et linguistiques. Cette évolution reflète également les efforts des locuteurs, des chercheurs et des institutions pour promouvoir et préserver cette riche tradition linguistique et culturelle [46].

Définition 13. ***Langue amazighe***

L'Algérie est un pays où l'on trouve deux principales variétés de langues nationales : les variétés de Tamazight et celles de l'arabe.

Le terme "berbère" trouve son origine dans le grec "barbaroi", repris par les romains sous la forme "barbarus", puis emprunté par les arabes sous "barbar". Enfin, les français l'ont adopté sous la forme "berbère". À l'origine, ce terme désignait principalement les "gens dont on ne comprend pas la langue", c'est-à-dire les étrangers. Les berbères, quant à eux, se désignent eux-mêmes par le terme "Imazighen" au pluriel, et par "Amazigh" au singulier.

Le mot "tamazight" désigne leur langue (berbère), bien que l'on trouve également l'expression "langue amazighe". Quant au terme "Tamazgha", il fait référence au territoire auquel ils appartiennent, c'est-à-dire la berbérie. Enfin, "Amazigh" signifie "homme noble" ou "homme libre".

La langue berbère regroupe une diversité d'idiomes répartis de manière discontinue à travers toute l'afrique du nord, s'étendant de l'est de l'égypte jusqu'au littoral marocain en passant par la libye, la tunisie, le niger, l'algérie, le mali et la mauritanie.

Une caractéristique majeure du berbère est sa fragmentation en un grand nombre de dialectes distincts. Ainsi, le berbère ne constitue pas une langue uniforme [1].

2.1.1 La géographie linguistique de la langue amazighe

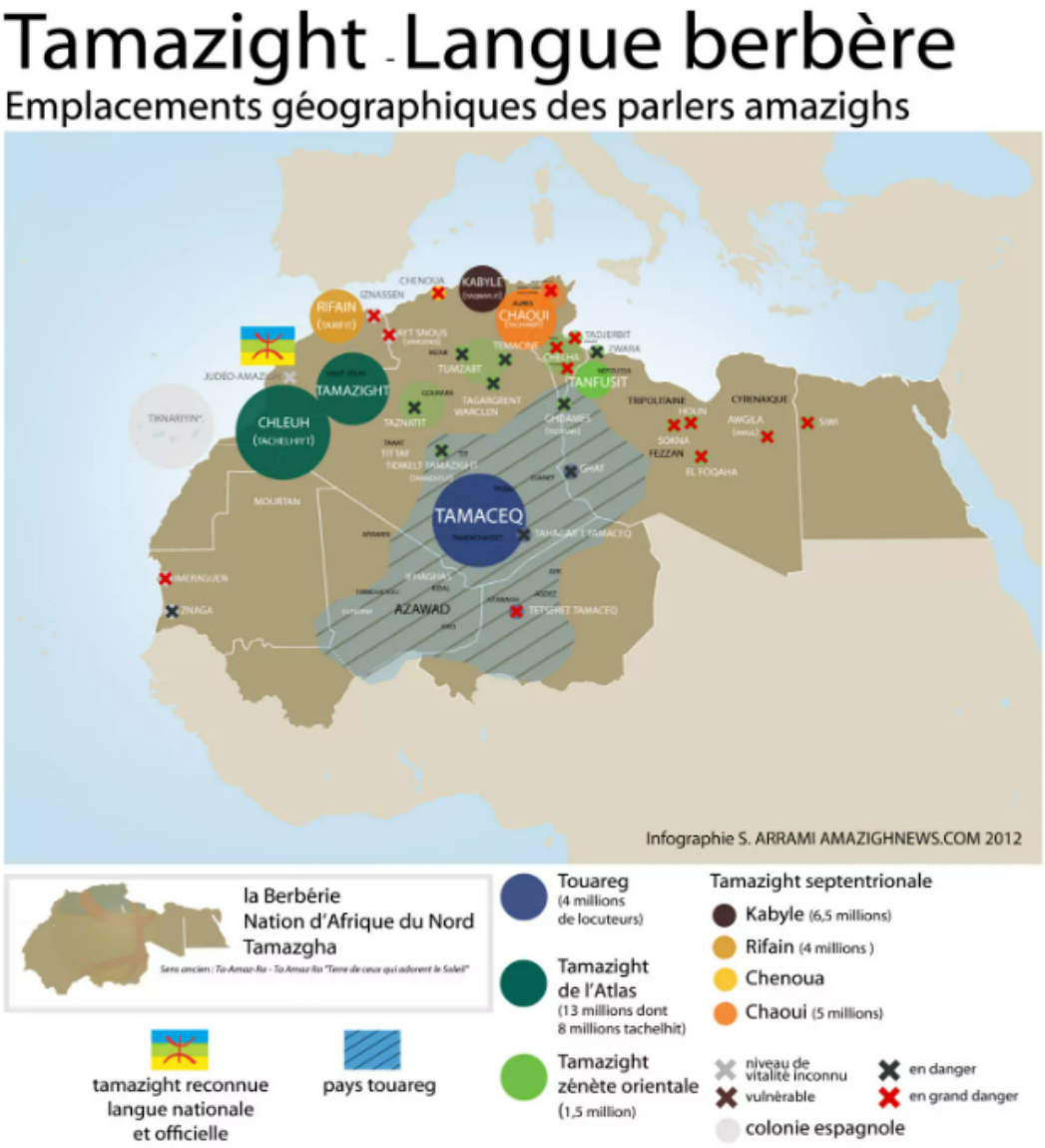


FIGURE 2.1 – Répartition Géographique des dialectes amazighs [56].

La carte illustre la présence de plusieurs variantes dialectales de la langue amazighe en afrique du nord, couvrant des régions telles que tamazgha centrale, la kabylie en algérie, le Hoggar kel tamachaq (espace touareg), Chenoua, Bilda amazighe, Tlemcen, Aurès, Mzab, le Grand Atlas, le Rif, l'Anti-Atlas Souss, le Tafilalet, la Mauritanie, la Libye et Siwa.

Ces régions témoignent de la diversité linguistique et culturelle de l'Afrique du Nord, avec différentes communautés berbères préservant leurs langues et leurs traditions dans ces zones géographiques distinctes.

2.1.2 Les dialectes de la langue amazighe

Il existe plusieurs formes de langues berbères, parmi lesquelles on trouve le chaoui, le soussi, le rifain, le kabyle, le chenoui, le mozabite, le tamasheq et le nafoussi, qui sont parmi les variétés les plus importantes de la langue amazighe.

Ces variétés sont parlées dans différentes régions, telles que l'Algérie (avec le kabyle, la chaouia, le mozabite), le Maroc (avec le rifain, l'atlas et le chleuh), et chez les Touaregs en Algérie, au Mali, au Niger, en Mauritanie et au Burkina Faso.

La survie de la culture berbère est due en partie au fait que chaque région était relativement isolée, ce qui a permis de préserver ses dialectes et ses traditions distinctes [1].

Pays	Appelation	Variétés linguistiques	Population
Algérie	amazighe	kabyle, chaouia, tamazight, hassaniyya, tumzabt, taznatit	15-20 millions.
Maroc	amazighe	tachelhit, tamazight, tarifit, ghomara	12-15 millions
Tunisie	amazighe	chaouia, nafusi, sened, ghadamès	100 000
Libye	tamache9	nafusi, tamaha9, ghadamès, sawknah, awjilah	220 000 (env)
Niger	amazighe	tamaja9 , tayart, touaregh	720 000
Mauritanie	zenaga	zenaga	200 (env)
Mali	tamaje9_kidal	tamaja9, tamashe9	440 000 (env)

TABLE 2.1 – Dialectes de la langue amazighe [1].

2.2 Variante kabyle

Définition 14. ***Variante kabyle***

Le kabyle, communément désigné sous le nom de taqbaylit, se démarque comme le principal dialecte berbère en Algérie, représentant probablement les deux tiers des locuteurs berbérophones du pays. En compagnie du touareg et du tachelhit, parlés dans le Sud-Ouest du Maroc, le kabyle

figure parmi les variantes régionales les plus étudiées et les plus familières [57].

Cette variante de la langue amazighe est pratiquée par les kabyles, une population berbère principalement localisée dans la région montagneuse de Kabylie, au nord de l'Algérie. L'usage de l'alphabet latin prévaut dans sa transcription. Comme c'est souvent le cas dans les langues, le kabyle présente des variations régionales et dialectales, avec des nuances linguistiques notables entre les différentes zones de la kabylie.

FIGURE 2.2 – Répartition géographique de la kabylie [65].

2.2.1 Système d'écriture kabyle

Pour transcrire kabyle, les berbères en général, les berbèrisants en particulier avaient fait recours à trois systèmes d'écriture [69] :

— **Tifinagh** : comme alphabet authentique attesté dans les inscriptions libyques depuis l'antiquité.

— **L'alphabet arabe** : suite à l'arrivée des arabes à la fin du 7ème siècle.

— **Le latin** : dès la fin du 18ème siècle.

L'alphabet Tifinagh

Le Tifinagh, tel que présenté par **M. Haddadou**, est un système d'écriture polyvalent, pouvant s'écrire de droite à gauche, de gauche à droite, ou du bas vers le haut. Il s'agit d'un système consonantique composé de barres, de cercles et de points.

Pour noter la voyelle "a" en finale, un point appelé "taghrit" est utilisé, tandis que les voyelles finales "i", "u" et "o" sont représentées par les mêmes signes que "y" et "w".

Il n'existe pas de séparation entre les mots ni de ponctuation. Le Tifinagh n'est pas homogène, car de nombreuses variations de caractères existent selon les régions, bien que ces variantes soient très similaires [69].

L'alphabet Latine

La plupart des kabyles emploient l'alphabet latin pour lire et écrire en langue kabyle.

L'agemmay amazigh latin contient 23 lettres latines standards et 11 lettres supplémentaires. Sa forme a été fixée par le linguiste **Mouloud Mammeri** dans les années 1960 [16].

— **Les voyelles** : traditionnellement, l'alphabet kabyle utilise quatre voyelles qui sont : « a », « e», « i » et « u », elles se lisent différemment selon qu'elles sont placées ou non à proximité des lettres dites empâtées .

— **Les consonnes** : le kabyle utilise en plus des consonnes latines, d'autres consonnes pour pouvoir répondre à sa diversité phonétique ,ces dernières font l'objet de notre étude. Les linguistes, jusqu'à nos jours utilisent 9 lettres non latines ou construites à base de lettres latines qui sont :

č, ḍ, ǧ, ḥ, ɣ, ṛ, ṣ, ṭ, ẓ, ɛ.

FIGURE 2.3 – Lettres spécifiques de la langue kabyle.

Particularités phonologiques du kabyle

La langue kabyle est réputée difficile à apprendre, en raison de ses sonorités très spécifiques. Certains sons kabyles n'existent pas en français. Vous retrouvez le th anglais, le r roulé, le h aspiré, le ch allemand, le r espagnol de Juan, le w anglais, des sons plus complexes emphatiques co-articulés, le g' prononcé avec une aspiration du palais entre le g, le h et le y, les affriquées dentales sourdes [ts, tts], les co-articulation vocalique furtive akw [u/w].

— La lettre b se lit soit b ou v. (v, [β], p[β], b (b)).

— La lettre d qui se lit d ou dh ([ð]).

— La lettre t qui se lit soit [ts] ou th[θ].

— La lettre g qui se lit gu ou g. (G[g],ḡ [ɟ]).

— La lettre k qui se lit k ou k. (K[ç]).

Tableau de translittération de la langue amazighe

Consonnes :

N°	Point & mode d'articulation	Notation latine	Graphie tifinaghe	Equivalent en arabe	Appellation en amazighe
1	Bilabiale occlusive sonore	b	ⴱ	ب	ⵉⴰⴱ (yab)
2	Palatale occlusive sonore	g	ⴳ	گ	ⵉⴰⴳ (yag)
3	Palatale occlusive sonore vélarisée	g^w	ⴳⵯ	-	ⵉⴰⴳⵯ (yag^w)
4	Dentale occlusive sourde	d	ⴷ	د	ⵉⴰⴷ (yad)
5	Dentale occlusive sourde emphatique	ḍ	ⴹ	ض	ⵉⴰⴹ (yaḍ)
6	Labiodentale spirante sourde	f	ⴼ	ف	ⵉⴰⴼ (yaf)
7	Palatale occlusive sourde	k	ⴽ	ك	ⵉⴰⴽ (yak)
8	Palatale occlusive sourde vélarisée	k^w	ⴽⵯ	-	ⵉⴰⴽⵯ (yak^w)
9	Laryngale spirante sourde	h	ⵀ	ه	ⵉⴰⵀ (yah)
10	Pharyngale spirante sourde	ḥ	ⵃ	ح	ⵉⴰⵃ (yaḥ)
11	Pharyngale spirante sonore	ɛ	ⵄ	ع	ⵉⴰⵄ (yaɛ)
12	Vélaire spirante sourde	x	ⵅ	خ	ⵉⴰⵅ (yax)
13	Ovulaire occlusive sourde	q	ⵇ	ق	ⵉⴰⵇ (yaq)
14	Palatale spirante sonore	j	ⵊ	ج	ⵉⴰⵊ (yaj)
15	Apicodentale latérale	l	ⵍ	ل	ⵉⴰⵍ (yal)
16	Bilabiale nasale sonore	m	ⵎ	م	ⵉⴰⵎ (yam)
17	Dentale nasale sonore	n	ⵏ	ن	ⵉⴰⵏ (yan)
18	Apicale vibrante sonore	r	ⵔ	ر	ⵉⴰⵔ (yar)
19	Apicale vibrante sonore emphatique	ṛ	ⵕ	-	yaⴹ (yaṛ)
20	Vélaire spirante sourde	γ	ⵖ	غ	ⵉⴰⵖ (yaγ)
21	Alvéolaire spirante sourde	s	ⵙ	س	ⵉⴰⵙ (yas)
22	Alvéolaire spirante sourde emphatique	ṣ	ⵚ	ص	ⵉⴰⵚ (yaṣ)
23	Palatale spirante sourde	c	ⵛ	ش	ⵉⴰⵛ (yac)
24	Dentale occlusive sourde	t	ⵜ	ت	ⵉⴰⵜ (yat)
25	Dentale occlusive sourde emphatique	ṭ	ⵟ	ط	ⵉⴰⵟ (yaṭ)
26	Alvéolaire spirante sonore	z	ⵣ	ز	ⵉⴰⵣ (yaz)
27	Alvéolaire spirante sonore emphatique	ẓ	ⵥ	-	ⵉⴰⵥ (yaẓ)

Voyelles :

28		a	ⴰ	◌ⵯ
29		u	ⵓ	◌ⵓ
30		i	ⵉ	◌ⵉ
31		e	ⵄ	-

Semi-voyelles :

32		w	ⵡ	و
33		y	ⵢ	ي

FIGURE 2.4 – Tableau des lettres de la langue amazighe [8].

2.3 Les phonèmes spécifiques de la langue amazighe en écriture latine

Les changements dans la façon dont nous prononçons les sons peuvent affecter des éléments isolés ou ceux qui se trouvent à proximité les uns des autres. On peut les diviser en deux types : les changements simples, où un son évolue directement en un autre (il s'agit du passage d'une consonne ou une voyelle à une autre), et les changements dus à des assimilations phonétiques. Ces derniers se produisent lorsque deux sons se rencontrent et influent l'un sur l'autre, créant ainsi une nouvelle prononciation.
Voici une représentation des changements phonétiques mentionnés : Les variations du phénomène

Phonème	Exemple
[j]->[g]	[waji]->[wagi],[ajjur]->[aggur]
[d]->[t]	[nniden]->[adar]
[g]->[k]	[waki]->[wagi]
[w]->[m]	[imumi]->[iwumi]
[l]->[r]	[armi]->[almi]
[θ]->[h]	[nihni]->[niθni]
[t]->[ts]	[θamellalt]->[θamellalts]

TABLE 2.2 – Tableau représentatif des changements phonétiques [61].

d'assimilation peuvent se produire lorsqu'il y a une rencontre entre deux consonnes. Le tableau ci dessous illustre ces changements :

L'assimilation aux frontières des morphèmes)	Réalisation	Exemple	Transcription phonétique	Equivalent en français	Région
/n + t/	[n+ɸ] [tt]	/n tmttot /	[nməttoθ] [ttməttoθ]	«de la femme ».	Kabylie extrême Occidentale-Ait Yahia Moussa. Béni-Douala, Tigzirt…
/n + w/	[bbʷ] [ppʷ] [ggʷ] [ww]	/n wrgaz/	[bbʷ ərgaz] [ppʷ ərgaz] [ggʷ ərgaz] [wwərgaz]	« de l'homme ».	Ait Yahia Moussa-Draa El Mizan, Tigzirt, … (variante sexuelle : Ouadhias, Béni-Douala…) Michelet, Ighil-Ali, Béjaia, … Kabylie-Extrême Orientale-Béjaia.
/d+t/	[ts] [tt]	/d tamttot/	[tsaməttoθ] [ttamətto θ]	« c'est une femme ».	Tigzirt, Draa El Mizan. Michelet-Akbil, Ighil-Ali-Béjaia…
/l + t/	[tt] [lt]	/ wltma / / Tiʁilt /	[wəttma] [θiʁitt] [wəltma] [θiʁilt]	« ma sœur ». « col ». « ma sœur ». « col ».	Kabylie extrême Occidentale -Draa El Mizan… Kabylie Occidentale-Michelet, Bouzeguene..
/ad + t/	[ts] [tt]	/ad troh/	[atsroh] [attroh]	« elle va partir ».	Tigzirt, Draa El Mizan, Mâatkas (Souk El-Tenine), Ouadhias (Tizi n Tlata)… Michelet-Akbil, Ath-Ouacif, Ath-Yenni…

FIGURE 2.5 – Tableau illustrant les variations du phénomène d'assimilation [61].

2.4 Structures de promotion de la langue amazighe

Les structures de promotion de la langue amazighe jouent un rôle crucial dans la préservation et la valorisation de cette langue. Parmi ces structures, on cite le Haut Commissariat à l'Amazighité (HCA), l'Institut Royal de la Culture Amazighe (IRCAM) et le Centre de Recherche en Langue et Culture Amazighes (CRLCA).

— Le Haut Commissariat à l'Amazighité (HCA) en Algérie, créé par le décret 147-95 du 27 mai 1995 et placé sous la tutelle de la Présidence, est dirigé par un Haut Commissaire et un Secrétaire Général.

  Ses missions principales sont la réhabilitation et la promotion de l'Amazighité, ainsi que l'introduction de la langue amazighe dans l'enseignement et la communication. Depuis 1995, le HCA a édité près de 200 ouvrages, accordé 350 subventions à des associations culturelles et scientifiques, et organisé de nombreuses rencontres scientifiques et culturelles [20].

— L'Institut Royal de la Culture Amazighe (IRCAM), basé à Rabat, est chargé de sauvegarder la culture amazighe en préservant ses traditions, sa langue et ses expressions culturelles.

  Il promeut l'amazighe en la valorisant comme une richesse nationale et une source de fierté pour tous les Marocains. De plus, il renforce la place de l'amazighe en l'intégrant dans le système éducatif, les médias et diverses sphères de la vie publique [37].

— En Algérie, le Centre de Recherche en Langue et Culture Amazighes (CRLCA), situé à Bejaïa, constitue une institution clé dédiée à l'étude et à la promotion de la langue et de la culture amazighes.

FIGURE 2.6 – Centre de Recherche en Langue et Culture Amazighes (CRLCA) [10].

Fondé le 26 février 2016, le CRLCA est un établissement public à caractère scientifique et technologique, spécialisé dans l'étude et la promotion de la langue et de la culture amazighes [10].

Le CRLCA emploie 20 chercheurs spécialisés, répartis en quatre divisions distinctes :

— **Langue, Métalangue et Didactique de Tamazight :** recherche sur la grammaire, la lexicologie, les variétés linguistiques amazighes, et développement d'outils pédagogiques.

— **Traitement Automatique des Langues (TAL) :** développement de technologies modernes, incluant la reconnaissance optique de caractères (OCR) pour les alphabets amazighs Latin et Tifinagh, utilisant l'intelligence artificielle.

— **Littérature, Arts et Patrimoine Amazighes :** études sur la littérature amazighe traditionnelle et moderne, ainsi que sur les expressions artistiques amazighes.

— **Civilisation Amazighe :** exploration anthropologique, archéologique et historique de la société amazighe et de ses pratiques culturelles et patrimoniales.

Le centre a pour objectif principal de mener des programmes de recherche dans les domaines de la langue et de la culture amazighe. Ses missions incluent [10] :

— Recherche appliquée sur différents aspects linguistiques.

— Gestion de la terminologie scientifique et technique.

— Promotion de la documentation pédagogique, scientifique et technique.

— Développement de méthodes de traduction.

— Recensement, valorisation et étude des coutumes, pratiques culturelles et expressions de la culture amazighe

— Contribution à la préservation du patrimoine immatériel.

— Étude de l'évolution du patrimoine amazighe à travers l'histoire.

Cet organigramme illustre la structure organisationnelle du centre, détaillant les divisions et les relations hiérarchiques au sein du CRLCA.

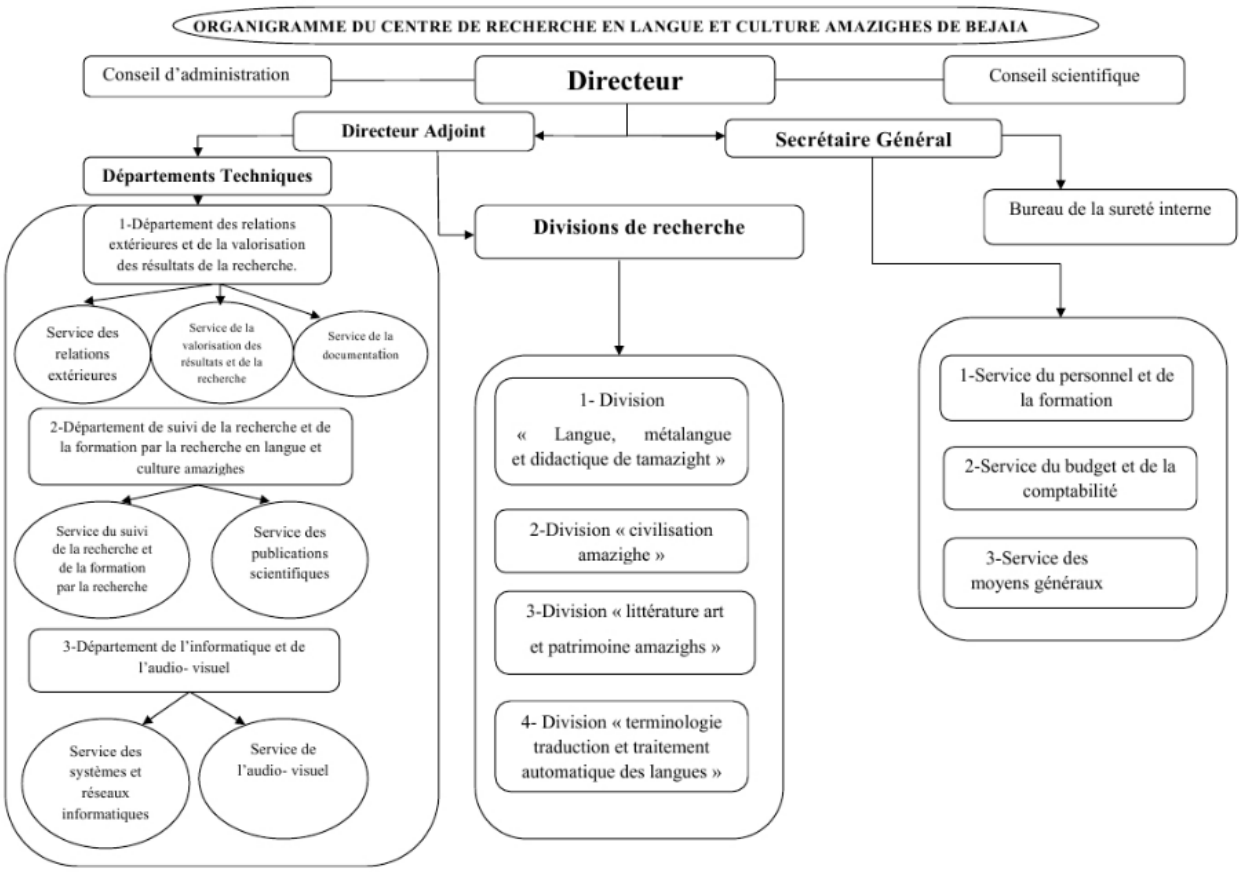


FIGURE 2.7 – Organigramme CRLCA [10].

Nous avons effectué un stage de cinq mois au Centre de Recherche en Langue et Culture Amazighes (CRLCA). Notre stage s'est déroulé au sein de la division Terminologie, Traduction et Traitement Automatique des Langues, intégrée à l'équipe de recherche numéro 3.

Ce stage était spécifiquement orienté dans le cadre du projet sur les outils de reconnaissance vocale avec traduction des emprunts, présidé par Monsieur Zaidi Ali.

Conclusion

Ce chapitre nous a permis de découvrir la langue amazighe, en mettant l'accent sur la variante kabyle. Nous avons étudié ses différents dialectes, sa répartition géographique, et son système d'écriture, y compris l'utilisation de l'alphabet latin. Nous avons également examiné les caractères spécifiques de la langue kabyle et les structures de promotion de la langue amazighe.

Cette exploration vise à offrir une compréhension approfondie de la richesse et de la complexité de la langue kabyle. En préparation à la création du corpus vocal kabyle, qui sera abordée dans le prochain chapitre.

3
Préparation de Corpus

Introduction

Dans ce chapitre, nous décrirons les étapes clés de la création d'un corpus vocal. Nous commencerons par expliquer la collecte des données, en détaillant la manière dont celle-ci a été effectuée. Ensuite, nous aborderons la préparation du corpus, suivie de son analyse selon divers critères.

Par la suite, nous effectuerons un prétraitement préliminaire des enregistrements audio. Enfin, nous détaillerons la création de notre dataset vocal, en incluant l'organisation des fichiers audio et leurs transcriptions.

Sommaire

3.1 Collecte de données

Définition 15. ***Collecte de données*** *:*

La collecte de données consiste à rassembler et mesurer des informations sur des variables d'intérêt de manière systématique et établie, afin de répondre à des questions de recherche spécifiques [54].

Ces données peuvent être obtenues à partir de diverses sources telles que des enquêtes, des groupes de discussion, des entretiens, des questionnaires, des observations et des bases de données existantes, etc.

3.1.1 Types de données

Les données peuvent être catégorisées de différentes manières, notamment en données quantitatives et qualitatives.

— **Données qualitatives :** les données qualitatives sont principalement non numériques et généralement descriptives ou nominales par nature. Ce type de données est présenté sous forme de mots ou de phrases [54]. Les données qualitatives répondent généralement aux questions "How and Why", dans une étude de recherche et couvrent principalement des informations relatives aux sentiments, aux perceptions et aux émotions [22].

— **Données quantitatives :** les données quantitatives sont de nature numérique et peuvent être calculées mathématiquement. Elles utilisent différentes échelles, qui peuvent être classées comme échelle nominale, échelle ordinale, échelle intervalle et échelle de ratio [54]. Les approches quantitatives traitent du "What" du programme, elles utilisent des méthodes structurées de collecte de données et sont basées sur un échantillonnage aléatoire [22].

3.1.2 Les méthodes de collecte de données

En général, on divise les méthodes de collecte de données en deux principales catégories : les méthodes de collecte de données primaires (Primary Data Collection Methods) et les méthodes de collecte de données secondaires (Secondary Data Collection Methods) [22].

Définition 16. ***Données primaires*** *:*

Les données primaires proviennent directement d'une expérience de première main et n'ont pas encore été rendues publiques. Elles sont considérées comme plus fiables, authentiques et objectives, car elles n'ont pas été altérées par des individus.

Par conséquent, leur validité est généralement jugée plus élevée que celle des données secondaires. Dans les enquêtes statistiques, il est essentiel d'utiliser des informations provenant de sources primaires pour garantir la qualité et la précision des résultats [54].

Méthodes de collecte de données primaires

La collecte de données primaires implique la collecte directe des données en utilisant à la fois des méthodes qualitatives et quantitatives.

Parmi les nombreuses méthodes de collecte de données primaires, on retrouve notamment les questionnaires, les entretiens, l'observation, la technique d'échantillonnage des activités, ainsi que des méthodes statistiques, entre autres [54]. Chaque méthode offre des avantages et des limitations, et le choix dépend souvent de la nature de la recherche, des objectifs spécifiques et des ressources disponibles.

Définition 17. ***Données secondaires*** *:*

Les données secondaires proviennent de sources déjà publiées, ce qui signifie qu'elles ont été rassemblées par quelqu'un d'autre dans un autre contexte et peuvent être utilisées à d'autres fins dans une recherche.

Elles jouent un rôle essentiel dans la recherche en fournissant des informations provenant d'études antérieures, qui peuvent servir de base à la mise en œuvre d'une nouvelle recherche ou de données de référence nécessaires [22].

Méthodes de collecte de données secondaires

Les données secondaires désignent les données extraites de sources déjà publiées. Elles sont obtenues à l'aide de méthodes de collecte de données secondaires. Ces données peuvent provenir aussi bien de sources qualitatives, telles que les rapports d'entretiens, que de sources quantitatives, comme les données du recensement.

Les méthodes de collecte de données secondaires peuvent être généralement catégorisées en sources publiées imprimées, livres, journaux/périodiques, revues électroniques, sites Web officiels et de plateformes de datasets [22].

3.2 Échantillonage

Définition 18. ***L'échantillonage*** *:*

L'échantillonnage (ou sampling en anglais) est le processus de sélection d'un groupe d'individus qui va être interrogé dans le cadre d'une étude et qui symbolise une population de référence.
Il permet de mener des enquêtes à grande échelle en utilisant un échantillon de la population pour remplacer l'ensemble et ainsi mener le sondage de manière réaliste.

3.2.1 Types d'échantillonage

Pour effectuer un échantillonage il existe deux options :

1. **L'échantillonnage aléatoire** : l'échantillonnage aléatoire (ou méthode d'échantillonnage probabiliste) est déterminé à partir d'une procédure de tirage aléatoire statistique. Malgré le hasard, la représentativité de l'échantillon aléatoire est assurée par les lois statistiques de la probabilité. Un échantillonage aléatoire peut se faire à travers l'utilisation des techniques suivantes [18] :

   — **Échantillonnage aléatoire simple** : consiste à sélectionner des éléments de la population de manière aléatoire, de telle sorte que chaque élément ait une probabilité égale d'être choisi.

   — **Échantillonnage systématique** : est une méthode dans laquelle seul le premier élément est choisi de manière aléatoire, tandis que les éléments suivants sont sélectionnés selon une règle prédéfinie par les chercheurs.

   — **Échantillonnage stratifié** : est une méthode qui implique la sélection aléatoire d'individus à l'intérieur de groupes prédéfinis appelés strates.

   — **Échantillonnage en grappe** : est une méthode où des groupes d'individus, appelés grappes, sont sélectionnés de manière aléatoire parmi la population. Ensuite, tous les individus au sein des grappes sélectionnées sont inclus dans l'échantillon.

2. **L'échantillonnage représentatif** : un échantillon représentatif est souvent utilisé dans une étude quantitative (questionnaire ou sondage), il est défini comme représentatif lorsqu'il a les mêmes caractéristiques que la population étudiée (population mère). Un échantillonage représentatif peut se faire à travers l'utilisation des techniques suivantes [18] :

   — **Échantillonnage de commodité** : les personnes ou les éléments de l'échantillon sont sélectionnés en fonction de leur disponibilité et de leur accessibilité.

   — **Échantillonnage par quotas** : vise à diviser la population en segments spécifiques en déterminant qui doit être inclus dans l'enquête en fonction de critères prédéfinis.

   — **Échantillonnage raisonné** : les chercheurs sélectionnent les participants en fonction de leur expertise, de leur compréhension de la question de recherche ou de leurs objectifs spécifiques.

   — **Échantillonnage référence** : les participants recrutés sont encouragés à recommander d'autres personnes qu'ils connaissent pour rejoindre l'échantillon.

3.3 Préparation du corpus

Définition 19. ***Corpus*** *:*

Selon ***Sinclair*** *et* ***Habert*** *: un corpus est une collection de données langagières qui sont sélectionnées et organisées selon des critères linguistiques et extra-linguistiques explicites pour servir d'échantillon d'emplois déterminés d'une langue [17].*

Variété des contenus d'un corpus

— **Texte** : un corpus écrit se compose principalement de textes et peut inclure une variété de documents tels que des extraits, des tableaux, etc. Il peut être présenté sous diverses formes telles qu'un livre, une page web ou un document.
— **Parole** : un corpus de parole est un ensemble constitué d'enregistrements de données orales.
— **Visuel** : un corpus visuel consiste en des vidéos ou des images, accompagnées éventuellement de textes pour fournir des compléments, des descriptions ou des présentations des objets visuels contenus dans le corpus.

3.3.1 Approche de collecte de données pour la constitution d'un corpus vocal

Notre projet repose entièrement sur une approche de collecte de données primaires qui se concentre seulement sur des enregistrements vocaux comme unique source d'informations. Nous avons opté pour une approche d'échantillonnage de commodité car elle permet de collecter des données sans nécessiter de planification complexe, en interrogeant les personnes disponibles sur place sans avoir à organiser des rencontres spécifiques.

Dans le cadre de cette collecte de données, 36 listes ont été préparées en colaboration avec Mme S.Matoub du centre de recherche en langue et culture amazighe, chacune représentant un phonème spécifique. Chaque liste comprend 15 noms et 15 verbes, accompagnés de leurs phrases associées.

ɛ

phonème	mots		phrases
ɛ (ɛ) [ɛ]	verbes	ɛebbi	Iɛebba I wayen ur yezmir
		ɛemmeɣ	Iɛemmer-d snat n terbuyin
		ɛiwen	Win yuḥwaḏen kra ad t-iɛawen
		ɛeddi	Ilḥeq-d d aneggar, Iɛedda d amezwar
		ɛawed	Yettɛawed ayen yeḥfeṭ akken ad yecfu
		cɛel	Yecɛel times ad iseḥmu
		sɛuḍer	Mazal yesɛuḍur deg uṭṭar-is
		dɛu	Tedɛa-yas ad t-iweffeq rebbi
		lɛeb	Teɣli deg berra almi telɛeb
		steɛfu	Seg-mi i d-yekker netta d axeddim
		nazeɛ	Seg sbaḥ tettnazeɛ; iqerreḥ-it lǧerḥ-is
		ɛaned	Ayen yexdem gma-s ad t-iɛaned
		beyyeɛ	Ibeyyen-as amek ara yexdem
		ceyyeɛ	Iceyyeɛ-it as-d-yaɣ lqahwa
		fqeɛ	Laɛi-tt kan ad tefqeɛ
	Noms	Lmaɛna	Yeḥrec meɛna txuṣṣ-it lmaɛna
		Taɛasit	Yeɛya deg tɛassit
		Laɛyaṭ	Icuf uqerruy-iw seg laɛyaṭ
		Lɛic	Yewwi-d lɛic i lmal-is
		Sɛaya	ṣeḥḥa tif sɛaya
		Aɛiban	ilul-d akken d aɛiban
		ṣṣenɛa	yal ṣṣenɛa yessen-it
		tiɛiqert	yebra-yas wergaz-is im tiɛiqqert i tella
		taɛekkazt	bla taɛekkazt-is; ur yezmir ad yeddu
		taɛmamat	ixdem taɛmamt ɣef uqerruy-is
		ṭmaɛ	ṭmaɛ yesseṭmeɛ
		aɛeṛbun	tessers aɛerbun ad ttaɣ taxatemt
		Leɛdez	ikcem-iyi leɛdez ur zmireɣ ad kreɣ
		Aɛejmi	yezla aɛejmi deg tmeɣra-s
		Aɛerjun	yuɣ-d tlata iɛerjunen n tmer

FIGURE 3.1 – Exemple de liste illustrant l'utilisation d'un phonème.

Nous avons pris grand soin de sélectionner divers lieux afin de refléter les différents dialectes de cette langue.

En explorant des endroits tels que l'université Abderrahmane Mira, la Maison de la Culture et la bibliothèque Principale de la Lecture Publique, nous avons rencontré une variété de locuteurs, chacun avec sa propre variation phonologique. Parallèlement, nous avons inclus des résidences universitaires et des quartiers représentatifs pour assurer une diversité des variables choisies.

Grâce à ces interactions enregistrées, nous avons créé un corpus vocal , témoignant de la richesse linguistique de notre région. Chaque locuteur a été guidé pour prononcer au moins l'un des 36 phonèmes répertoriés, et utiliser deux mots spécifiques ainsi que deux phrases contenant ce phonème.

De plus, chaque locuteur est invité à fournir des informations supplémentaires telles que son âge, sa région, sa commune, sa daira et son statut social, offrant ainsi une compréhension plus approfondie du contexte sociolinguistique de chaque enregistrement.

FIGURE 3.2 – Lieux de collecte.

Les critères de création du corpus

La sélection des participants à notre corpus vocal, bien que réalisée de manière aléatoire, a été effectuée avec soin et réflexion. Nous avons pris en compte une variété de critères tels que l'âge, le sexe et la situation sociale et région :

— **L'âge** : la parole évolue avec l'âge. Les enfants, les adolescents, les adultes et les personnes âgées présentent des schémas de parole distincts en termes de vocabulaire, de syntaxe, d'intonation et de débit. Ainsi, pour garantir que les modèles de traitement vocal soient efficaces et adaptables à une multitude d'utilisateurs, il est indispensable de les entraîner avec des données provenant de différents groupes d'âge.

— **Sexe** : les hommes et les femmes ont des caractéristiques anatomiques distinctes qui se traduisent par des différences dans la tonalité et la qualité de la voix.

— **Statut Sociale** : le statut social peut influencer le choix du vocabulaire, le style de parole, l'accent et d'autres aspects linguistiques. Par exemple, les personnes issues de milieux sociaux différents peuvent avoir des schémas de parole distincts.

— **Région** : la collecte d'enregistrements vocaux de diverses régions géographiques permet de saisir la diversité linguistique régionale, incluant les dialectes, les accents et les variations linguistiques propres à chaque région. Cette diversité est cruciale pour développer des modèles de reconnaissance vocale précis et adaptés à une variété de populations linguistiques.

Une fois la collecte des données terminée, nous avons rassemblé un total de 2641 enregistrements vocaux. Ces enregistrements seront ensuite soumis à un processus de prétraitement afin d'analyser et d'extraire les informations pertinentes.

3.3.2 Outils & Moyens de création de corpus

Pour créer notre corpus vocal, nous avons utilisé divers outils offerts par le Centre de Recherche en Langue et Culture Amazighe (CRLCA), qui nous ont facilité la collecte de données, l'enregistrement et le prétraitement des informations. Ces outils nous ont permis d'optimiser notre travail et d'assurer une gestion efficace de notre projet.

Dictaphone

Pour effectuer des enregistrements, nous avons utilisé des dictaphones pour capturer de manière précise et claire les voix des personnes enregistrées. Ces dictaphones permettent d'enregistrer sur des cartes mémoire, ce qui facilite le stockage et la gestion des enregistrements audio.

FIGURE 3.3 – Dictaphone.

TALN-RV Corpus

L'application mobile TALN-RV Corpus a été développée par le Centre de Recherche en Langue et Culture Amazighe.

Elle intègre un formulaire comprenant neuf champs : Wilaya, Commune, Région, Âge, Sexe, PH/API (code phonémique), Code ENR (code d'enregistrement), Enregistrement et Statut social. TALN-RV Corpus est interconnectée à TALN-RV-DATA.

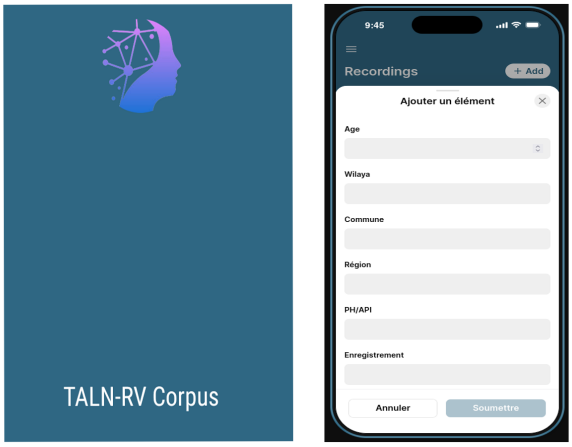


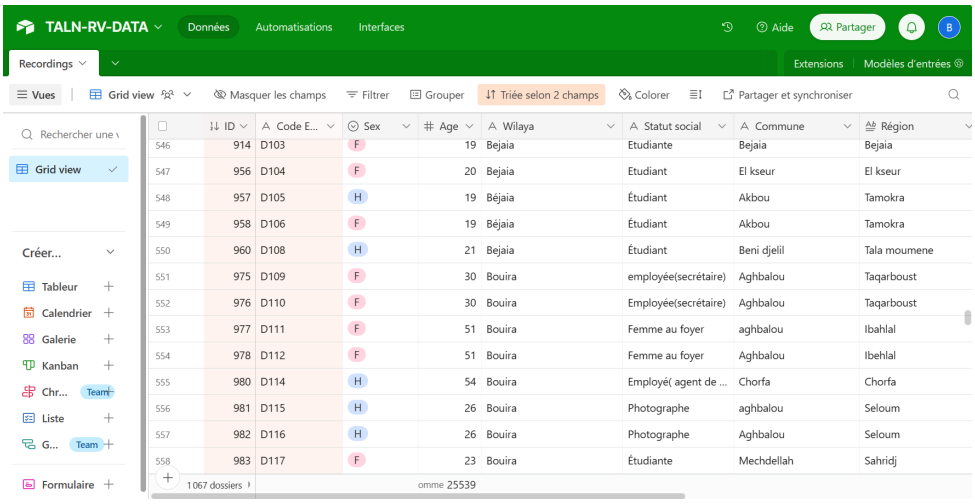
FIGURE 3.4 – TALN-RV Corpus.

TALN-RV-DATA

Nous utilisons Airtable pour stocker les données collectées, car c'est une plateforme en ligne qui combine les fonctionnalités d'une base de données relationnelle et permet aux utilisateurs de créer des bases de données personnalisées.

Notre base de données TALN-RV-DATA comprend 10 champs : ID, code ENR, sexe, âge, wilaya, statut social, commune, région, PH/API, et un champ pour inserer les enregistrements.

Cette base de données est liée à l'application TALN-RV Corpus, de sorte que chaque nouvelle donnée enregistrée est automatiquement ajoutée à la base de données.



	ID	Code E...	Sex	Age	Wilaya	Statut social	Commune	Région
546	914	D103	F	19	Bejaia	Etudiante	Bejaia	Bejaia
547	956	D104	F	20	Bejaia	Etudiant	El kseur	El kseur
548	957	D105	H	19	Béjaia	Étudiant	Akbou	Tamokra
549	958	D106	F	19	Béjaia	Étudiant	Akbou	Tamokra
550	960	D108	H	21	Bejaia	Étudiant	Beni djelil	Tala moumene
551	975	D109	F	30	Bouira	employée(secrétaire)	Aghbalou	Taqarboust
552	976	D110	F	30	Bouira	Employée(secrétaire)	Aghbalou	Taqarboust
553	977	D111	F	51	Bouira	Femme au foyer	aghbalou	Ibahlal
554	978	D112	F	51	Bouira	Femme au foyer	Aghbalou	Ibehlal
555	980	D114	H	54	Bouira	Employé(agent de ...	Chorfa	Chorfa
556	981	D115	H	26	Bouira	Photographe	aghbalou	Seloum
557	982	D116	H	26	Bouira	Photographe	Aghbalou	Seloum
558	983	D117	F	23	Bouira	Étudiante	Mechdellah	Sahridj

FIGURE 3.5 – TALN-RV-DATA.

Audacity

Pour traiter les fichiers vocaux que nous avons, nous utilisons Audacity, un logiciel gratuit et open-source d'édition audio.

Il permet aux utilisateurs d'enregistrer, de modifier et de mixer des fichiers audio dans divers formats. Audacity offre de nombreuses fonctionnalités, notamment l'application d'effets audio tels que la réduction du bruit, l'égalisation et l'ajustement du volume.

FIGURE 3.6 – Audacity Logo.

3.4 Analyse du corpus

Comme mentionné précédemment, notre corpus vocal a été constitué en prenant en compte plusieurs critères tels que l'âge, le sexe et les zones géographiques qui indiquent l'origine des locuteurs. Une analyse globale des enregistrements par rapport à ces critères se trouve ci-dessous :

3.4.1 Âge

D'après les données présentées dans la figure ci-dessous, les individus âgés de 19 à 30 ans sont les plus nombreux dans notre collecte. Leur forte participation s'explique par leur disponibilité et leur volonté de contribuer activement à l'étude.

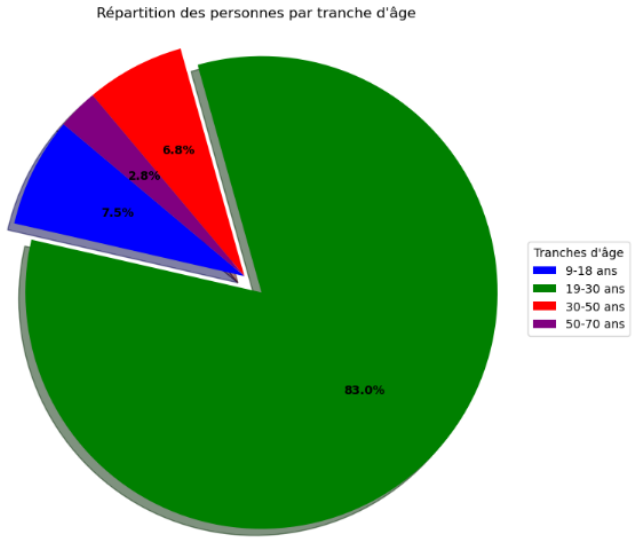


FIGURE 3.7 – Répartition des personnes par tranche d'âge.

3.4.2 Sexe

D'après la figure ci-dessous, on observe une présence plus importante de femmes par rapport aux hommes, car elles participent d'avantage et sont plus nombreuses dans la population ciblée.

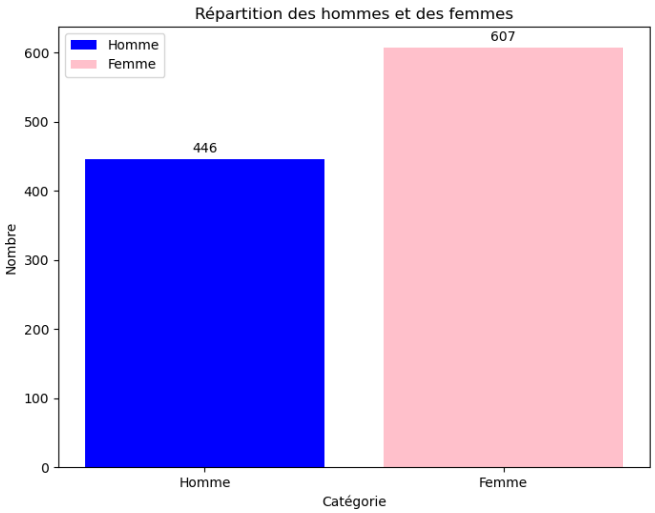


FIGURE 3.8 – Répartition des personnes par sexe.

3.4.3 zones géographiques

La figure ci-dessous illustre les origines géographiques des locuteurs qui ont participé à notre collecte, fournissant ainsi une vue d'ensemble détaillée de la diversité régionale des participants.

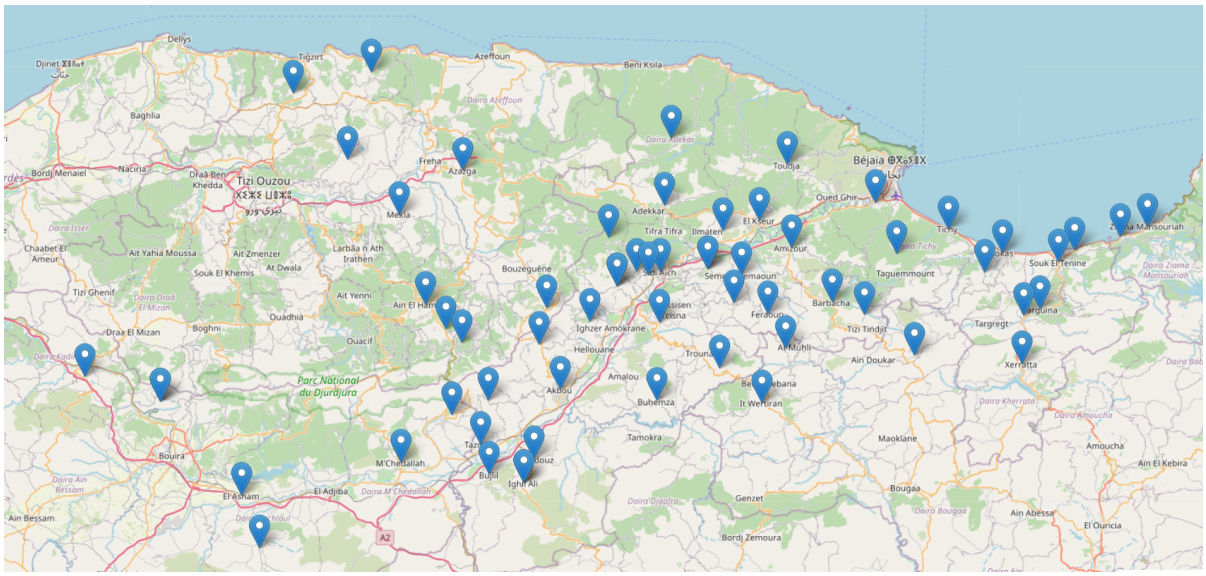


FIGURE 3.9 – Zones géographiques.

3.5 Prétraitement

Définition 20. ***Prétraitement*** *:*

Le prétraitement de la voix englobe les différentes étapes nécessaires pour améliorer la qualité des enregistrements vocaux.

Après la collecte des données vocales, ce processus comprend la réduction du bruit pour éliminer les sons indésirables, la correction de la tonalité pour ajuster la hauteur et la fréquence de la voix, et d'autres techniques comme la normalisation du volume et l'égalisation pour équilibrer les différentes fréquences audio.

3.5.1 Les étapes du prétraitement

Les enregistrements vocaux sont essentiels pour la reconnaissance vocale. Cependant, pour les exploiter efficacement, il est souvent indispensable de les soumettre à un processus de prétraitement afin d'en extraire des données pertinentes et d'assurer leur qualité.

3.5.2 Réduction du bruit

Lors d'un enregistrement vocal dans une pièce bruyante, le bruit de fond peut fortement altérer la compréhension de la parole. Pour remédier à cela, on utilise une technique essentielle

appelée réduction du bruit.

Cette technique vise à éliminer les sons indésirables tels que les bourdonnements, les sifflements, les grésillements, et autres interférences qui peuvent être présents dans l'enregistrement et nuire à sa qualité. Pour réduire le bruit dans un enregistrement, nous avons choisi d'utiliser l'application Audacity, qui offre les outils nécessaires pour effectuer cette tâche.

— La première étape consiste à importer le fichier de type wav que nous avons enregistré.

— Après avoir importé le fichier dans Audacity, nous avons identifié et sélectionné une partie de l'enregistrement contenant uniquement du bruit. Ensuite, nous avons accédé au menu "*Effets*" et choisi "*Réduction de bruit et réparation*", puis "*Prendre le profil de bruit*" afin qu'Audacity puisse analyser et enregistrer la signature sonore du bruit.

— Après cela, nous avons sélectionné l'ensemble de la piste audio et appliqué les paramètres optimaux de réduction et de sensibilité. Pour finir, nous avons cliqué sur "*Valider*" pour confirmer les modifications et écouté le résultat afin de vérifier la qualité de la réduction de bruit.

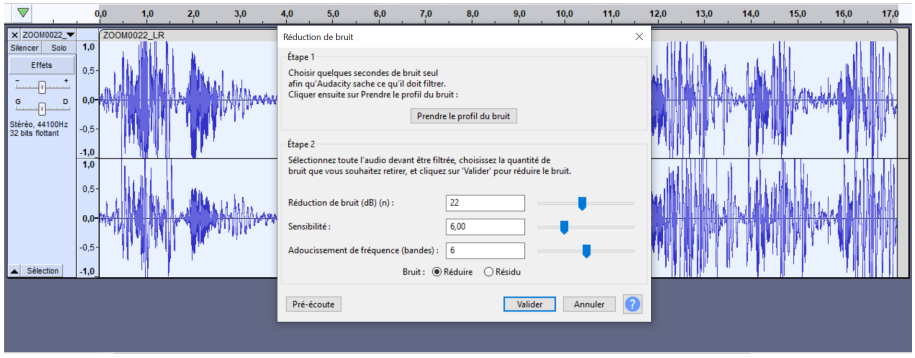


FIGURE 3.10 – Élimination du bruit.

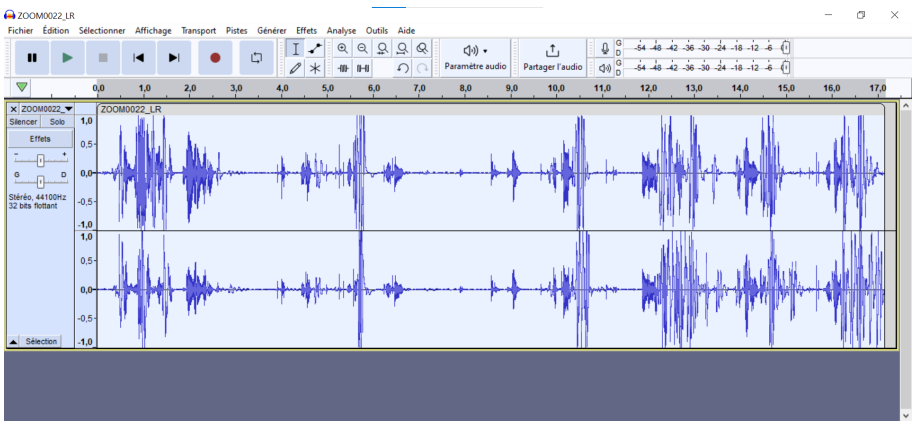


FIGURE 3.11 – Fichier de type wav aprés la réduction du bruit.

3.5.3 La segmentation

La segmentation des enregistrements vocaux consiste à diviser le flux audio en unités linguistiques significatives telles que les phonèmes , les mots ou les phrases. Cette étape est fondamentale car elle permet de découper l'enregistrement en parties plus faciles à gérer et à analyser.

Pour effectuer cette tâche nous utilisons Audacity en se basant sur les pauses et les variations d'intonation.

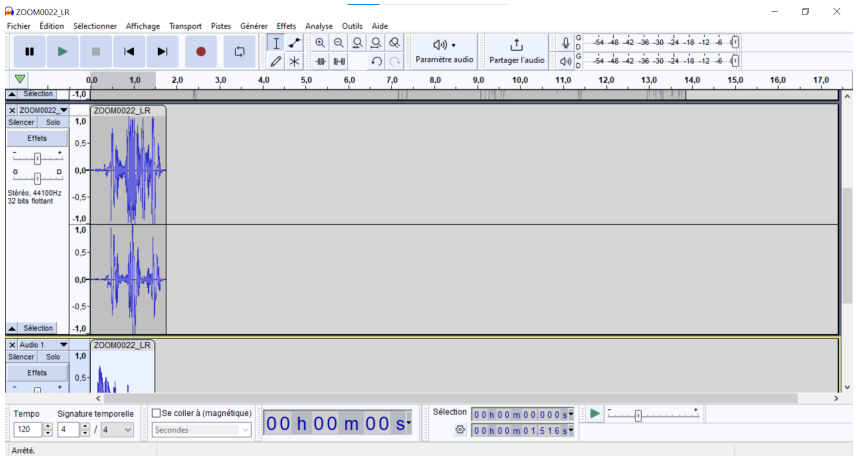


FIGURE 3.12 – Segmentation des enregistrements vocaux.

3.5.4 Les défis liés au prétraitement

Lors du prétraitement , nous avons rencontré plusieurs défis. Le principal problème était le taux élevé de bruit dû au vent et les voix environnantes, car la collecte de données vocales a été réalisée dans des lieux publics. Cela a nécessité une étape conséquente d'élimination du bruit.

Le deuxième défi était de devoir segmenter chaque phonème, mot et phrase individuellement. Cela prenait beaucoup de temps en raison du grand nombre d'enregistrements à traiter et de la période limitée pour le faire. De plus, la plupart des participants n'ont pas laissé d'espaces entre les mots lorsqu'ils lisaient les phrases, ce qui compliquait la segmentation.

Cependant, ces défis n'ont pas empêché de mener à bien cette étape et d'obtenir les meilleurs résultats possibles.

3.6 Synthèse vocale

Définition 21. ***Synthèse vocale*** *:*

la synthèse vocale est un terme générique qui désigne la sortie vocale d'un système informatique. Son principal objectif est de produire des sons de parole à partir d'une représentation phonétique du message et permet la transmission d'informations sous forme orale soit en l'absence d'écran, soit en complément avec celui-ci [35].

Nous avons choisi d'utiliser Festival pour enrichir notre base de données et obtenir plus de données vocales.

3.6.1 Présentation de Festival

Festival est un outil complet pour la création de systèmes de synthèse vocale, avec divers modules disponibles. Il offre une conversion texte-parole via plusieurs API : ligne de commande, interpréteur Scheme, bibliothèque C++, Java et interface Emacs. Multilingue, il supporte principalement l'anglais (britannique et américain) et l'espagnol, avec d'autres langues ajoutées par différentes communautés. Festival est un logiciel libre, permettant une utilisation commerciale et non commerciale sans restriction [11].

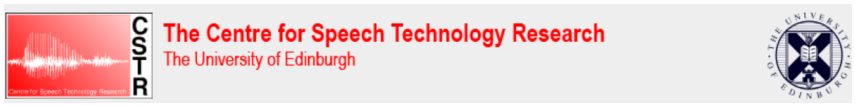


FIGURE 3.13 – Festival.

3.6.2 Mise en place de Festival

Avant de pouvoir utiliser Festival sous Linux, nous devons d'abord l'installer sur notre système. Une fois l'installation terminée, nous devons vérifier si Festival a été installé correctement en exécutant la commande **festival** dans le terminal pour afficher sa version ou son statut d'installation. Cette vérification est importante pour s'assurer que le logiciel est prêt à être utilisé conformément à nos besoins.

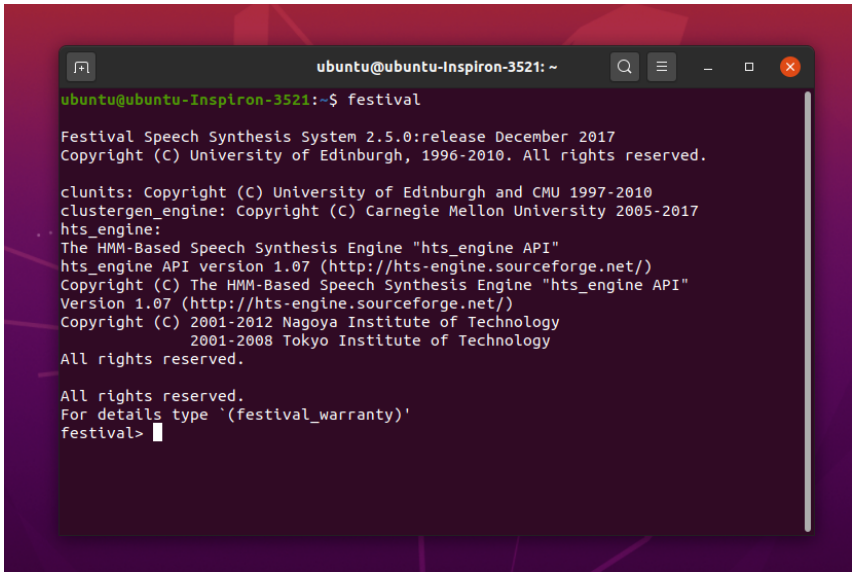


FIGURE 3.14 – Vérification de l'installation de Festival.

— Pour afficher la liste des voix disponibles dans Festival et choisir celle qui nous convient, nous devons utiliser la commande **(voice.list)**.

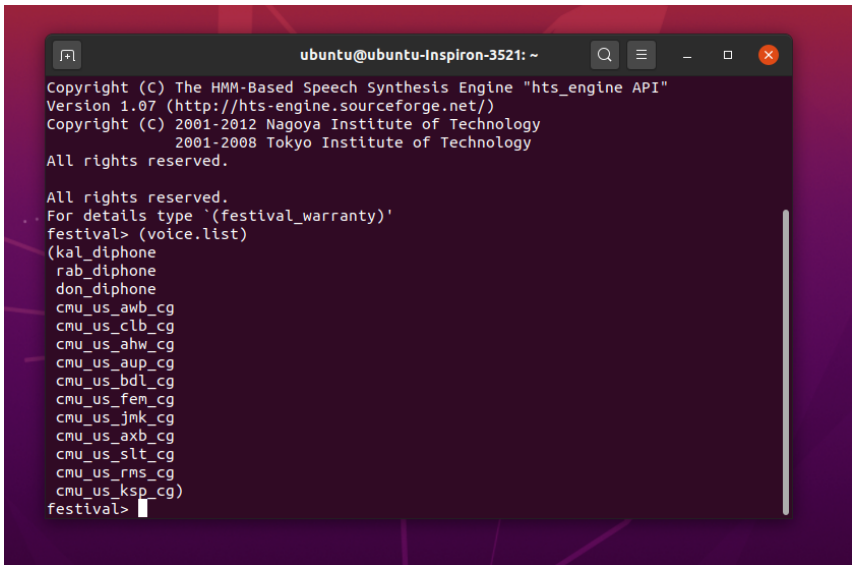


FIGURE 3.15 – Liste des voix.

— Une fois que nous avons obtenu la liste des voix disponibles, nous pouvons choisir celle qui nous convient en utilisant simplement la commande **festival (la voix que on veux utiliser**). Pour notre cas, nous avons choisi d'utiliser ***cmu_us_ksp_cg***.

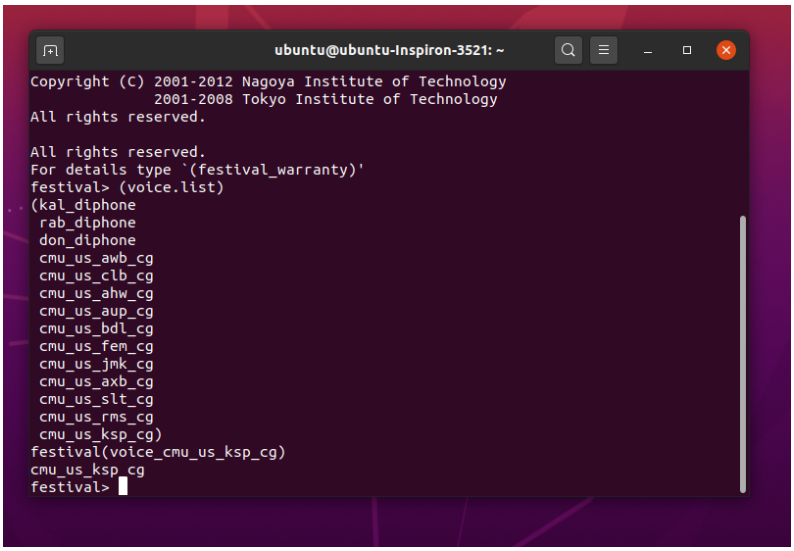


FIGURE 3.16 – La sélection d'une voix.

— Après avoir identifié la voix qui nous convient, nous procédons à l'insertion du texte que nous voulons transformer en parole. Pour ce faire, il suffit d'écrire la commande ***sayText(le text que on veux lire).***

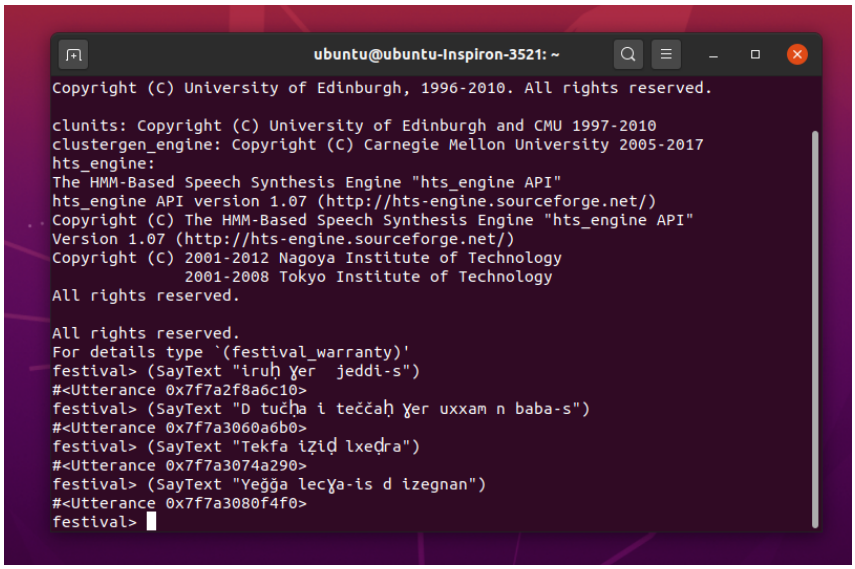


FIGURE 3.17 – Le texte à lire.

3.6.3 Limitations linguistiques de Festival

Bien que Festival soit un logiciel convivial, même pour les débutants, il présente des limites en termes de langues prises en charge. En effet, il ne propose que des accents américain et britannique, ainsi que l'espagnol. Lorsque nous avons essayé d'insérer du texte en kabyle, Festival n'a pas réussi à lire les caractères spéciaux et n'a pas pris en charge l'accent kabyle, rendant ainsi les phrases incompréhensibles.

Cette limitation nous a empêchés d'utiliser Festival pour augmenter notre corpus vocale.

3.7 Dataset vocale

Afin d'assurer une interprétation précise du contenu audio lors de la phase de reconnaissance vocale, nous avons élaboré un dataset vocal nommé TALN.

Le dataset (fichier csv) est organisé en deux colonnes : une colonne contenant les identifiants et une autre contenant les transcriptions textuelles correspondantes. Un dossier, nommé identiquement au fichier CSV, héberge les fichiers audio au format wav. Chaque fichier audio est nommé d'après son identifiant correspondant dans le dataset, assurant ainsi une correspondance directe entre les enregistrements vocaux et leurs transcriptions.

3.7.1 Étapes de creation d'un dataset vocale

La création du dataset s'est déroulée en plusieurs étapes méthodiques.

— Après avoir intégré les enregistrements déjà traités dans la base de données en ligne TALN-RV-DATA, nous avons procédé à une écoute minutieuse de chaque enregistre-

ment. Notre objectif était d'extraire ceux qui étaient clairs et ne présentaient pas un niveau de bruit excessif.

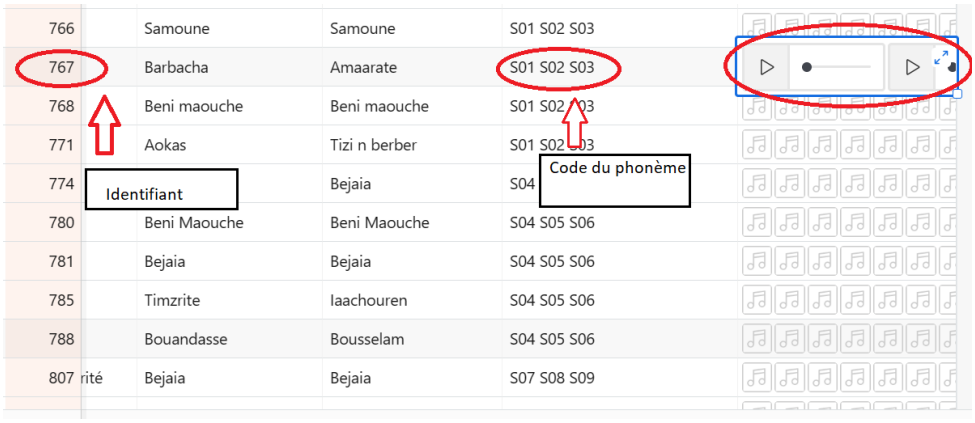


FIGURE 3.18 – Extraction des données.

— Après avoir identifié les enregistrements à utiliser ainsi que leur identifiant et le code du phonème de chaque enregistrement, nous avons fait usage de LibreOffice, une suite bureautique libre et gratuite, offrant une gamme variée d'outils pour la création, l'édition et la gestion de documents, de feuilles de calcul et de présentations.

À l'aide de l'application de feuilles de calcul présente dans LibreOffice, nous avons saisi les identifiants et les transcriptions de chaque enregistrement wav sélectionné, un par un. Cette approche nous a permis d'organiser efficacement les données et de les associer de manière précise, facilitant ainsi leur manipulation ultérieure.

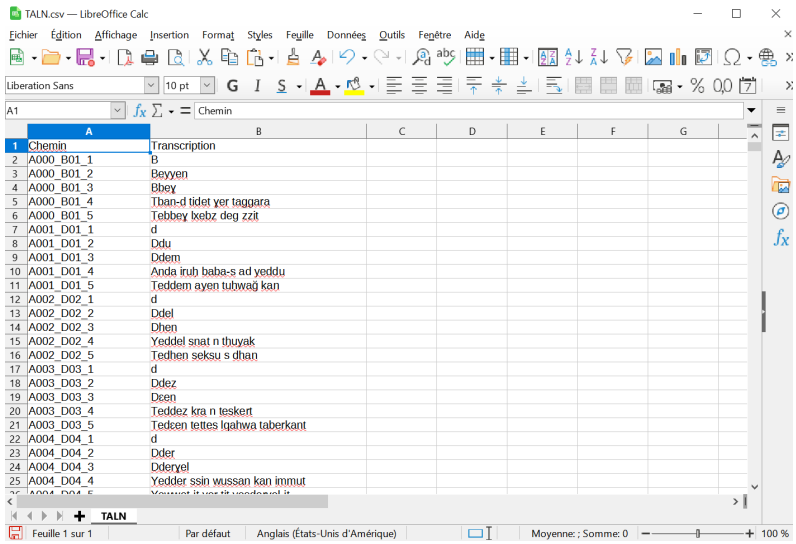


FIGURE 3.19 – TALN.csv.

— Ensuite, nous avons téléchargé ces fichiers audio et les avons stockés dans un dossier spécifique nommé TALN, garantissant ainsi une organisation claire et structurée de notre base de données vocale.

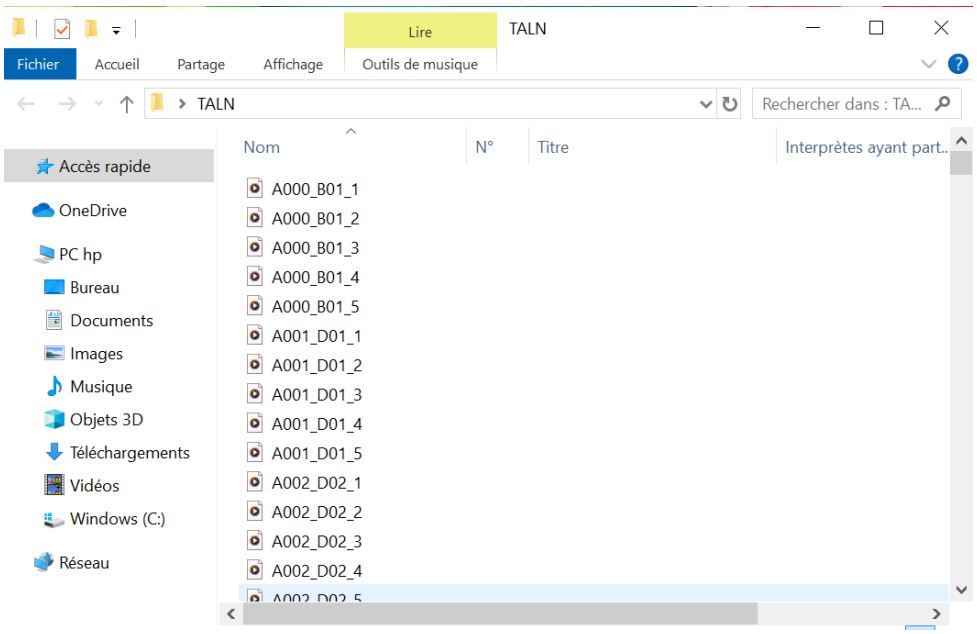


FIGURE 3.20 – Dossier wav.

Une fois que les enregistrements au format wav ont été soigneusement saisis, sélectionnés et accompagnés de leurs transcriptions, notre base de données contient désormais 9886 enregistrements au total, comprenant chacun sa transcription.

Étant donné que notre étude se concentre sur des lettres de la langue kabyle, nous avons choisi d'extraire uniquement les phonèmes correspondants dans un nouveau dataset vocal. Cela nous permet de focaliser notre analyse linguistique sur les aspects pertinents à notre recherche spécifique.

3.8 Dataset vocale phonème

Une fois que le jeu de données vocales a été préparé, nous avons extrait tous les enregistrements contenant les phonèmes ainsi que leurs transcriptions, en suivant les étapes similaires à celles utilisées pour créer le dataset TALN.

À l'issue de ce processus, notre nouveau dataset est nommé PH, comprenant 2265 enregistrements avec leurs transcriptions respectives.

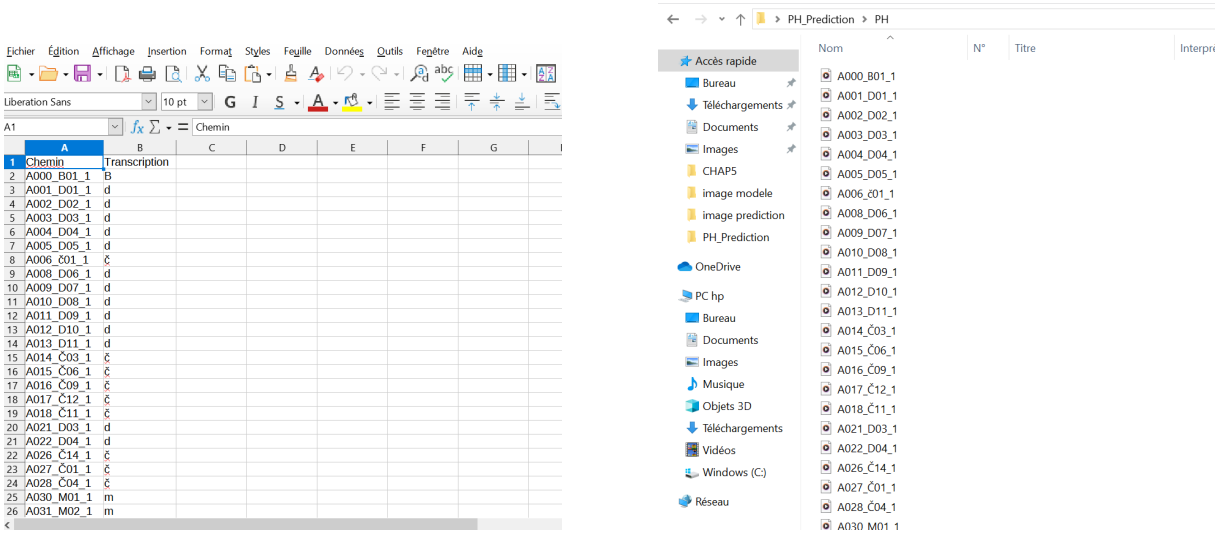


FIGURE 3.21 – Dataset vocale phonème.

Conclusion

Ce chapitre a présenté les étapes fondamentales de la création de notre corpus vocal. Nous avons détaillé le processus de collecte des données, la préparation du corpus, ainsi que le prétraitement des enregistrements vocaux.

Bien que nous ayons exploré la synthèse vocale pour illustrer son potentiel dans la génération de voix artificielles, nous avons rencontré des limitations liées à l'absence de la langue kabyle.

Enfin, nous avons mis en évidence l'importance d'une organisation rigoureuse des fichiers audio et de leurs transcriptions dans la création de notre dataset vocale.

Dans le chapitre suivant, nous aborderons les étapes du prétraitement des données. Ensuite, nous discuterons des différentes techniques de reconnaissance vocale permettant de transcrire les enregistrements audio en texte.

4

Traitement du Signal Vocal & Apprentissage Profond

Introduction

Dans ce chapitre, nous examinerons les étapes du prétraitement du signal audio, qui améliorent sa qualité, ainsi que les méthodes d'extraction des caractéristiques acoustiques, notamment les coefficients MFCC (Mel Frequency Cepstral Coefficients), qui visent à transformer les données vocales en séries de chiffres exploitables par la machine. Enfin, nous étudierons les techniques de reconnaissance vocale, telles que les réseaux de neurones, qui permettent à la machine de comprendre et de transcrire la parole humaine.

Sommaire

4.1 Outils de développement

Cette section est dédiée à la présentation des divers outils mis en œuvre dans le développement de notre modèle. Ces outils ont joué un rôle crucial à chaque étape du processus, permettant une optimisation et une efficacité accrues.

1. Jupyter Notebook :

C'est un fichier, à l'extension .ipynb (ipython notebook), qui est un format de texte brut basé sur JSON. Il peut être converti en divers formats tels que des présentations, des pages web, des documents PDF ou des scripts exécutables. Gratuit et open-source, il est utilisé dans l'environnement Jupyter pour la création, le partage et l'exécution de codes Python et d'autres langages.

2. Python :

Python, langage de programmation open-source, est réputé pour sa simplicité et sa lisibilité. Avec une vaste bibliothèque et une portabilité étendue, il convient à différents systèmes d'exploitation. Son orientation objet et son typage dynamique offrent une flexibilité et une rapidité de programmation.

3. TensorFlow :

TensorFlow, développé par Google, est une plateforme open-source dédiée à l'apprentissage automatique et profond. Elle utilise des tenseurs pour représenter les données et adopte une architecture basée sur des graphes de flux de données, ce qui facilite la distribution du calcul sur des clusters de machines. Bien qu'il soit compatible avec plusieurs langages, Python reste largement préféré [12].

4. Librosa :

Librosa est un package Python pour l'analyse musicale et audio. Elle fournit les outils nécessaires pour créer des systèmes de recherche d'informations musicales et pour explorer les caractéristiques des signaux audio. Elle permet aussi de visualiser ces derniers [63].

5. Scikit-learn :

Scikit-learn est une bibliothèque open-source en Python qui offre des outils simples et efficaces pour l'apprentissage automatique, le prétraitement des données et l'évaluation des modèles.

6. Keras :

Keras est une API de réseaux neuronaux en Python, conçue pour une expérimentation rapide. Elle permet de créer facilement des modèles grâce à son interface simple et modulaire. Keras supporte les réseaux CNN et RNN, ainsi que leurs combinaisons, et peut s'exécuter sur CPU et GPU, facilitant ainsi le développement et le déploiement de modèles de machine learning.

4.2 Prétraitement des signaux audio

Le prétraitement des signaux audio est un domaine vaste qui intègre des concepts de physique, de mathématique et d'informatique. Théoriquement, il s'agit de la manipulation et de l'analyse des signaux acoustiques enregistrés pour diverses applications, telles que la reconnaissance vocale et l'analyse de la parole, etc. Voici les principales étapes de ce processus :

4.2.1 Rééchantillonnage

Définition 22. ***Rééchantillonnage :***

Le rééchantillonnage est le processus de modification de la fréquence d'échantillonnage d'un signal audio numérique [3]. Cela permet d'adapter le signal à une nouvelle fréquence d'échantillonnage, soit en augmentant, soit en diminuant le nombre d'échantillons par seconde. Le rééchantillonnage est souvent utilisé pour assurer la compatibilité avec différents dispositifs ou pour optimiser la qualité et l'efficacité du traitement audio.

Il existe plusieurs fréquences d'échantillonnage disponibles, mais on a effectué le rééchantillonnage de nos différents signaux audio à une fréquence de 16 kHz pour plusieurs raisons :

— D'après le théorème de **Nyquist-Shannon**, pour bien reproduire un signal à partir de ses échantillons, la fréquence d'échantillonnage doit être au moins deux fois plus élevée que la fréquence maximale du signal [50]. Ainsi, en choisissant une fréquence d'échantillonnage de 16 kHz, nous nous assurons de capturer tous les détails importants de la voix humaine, qui s'étend généralement jusqu'à environ 8 kHz.

— Rééchantillonner à 16 kHz au lieu de fréquences plus élevées permet de réduire la taille des fichiers audio, ce qui économise de l'espace de stockage ou de la bande passante lors du stockage ou de la transmission des données audio.

— Une fréquence d'échantillonnage de 16 kHz rend le traitement audio plus rapide et efficace tout en préservant une qualité audio satisfaisante.

Processus du rééchantillonnage

Le rééchantillonnage à une fréquence de 16 kHz implique des opérations mathématiques spécifiques pour ajuster la fréquence d'échantillonnage d'un signal audio à cette valeur :

— **Interpolation** : pour augmenter la fréquence d'échantillonnage d'un signal en dessous de 16 kHz jusqu'à 16 kHz, on ajoute de nouveaux échantillons entre les échantillons existants.

— **Décimation** : si le signal d'origine est échantillonné à une fréquence supérieure à 16 kHz et qu'on souhaite le réduire jusqu'à 16 kHz, la méthode de décimation permet de sélectionner les échantillons appropriés à cette nouvelle fréquence d'échantillonnage pour reconstruire le signal audio.

Algorithme de Rééchantillonnage

Cet algorithme automatise le processus de rééchantillonnage des fichiers audio de type wav situés dans un dossier spécifié. Pour chaque fichier audio, il charge le signal, le rééchantillonne à une fréquence de 16 kHz via la fonction librosa.resample(), puis écrit le résultat dans un nouveau fichier wav dans un dossier de sortie désigné.

Algorithm 1: Rééchantillonnage des fichiers audio

input :
↓ input_directory : Chemin vers le dossier contenant les fichiers de type wav ;
↓ Fréquence d'échantillonnage cible ;
output:
↑ Fichiers wav rééchantillonnés ;
Initialization :
Obtenir la liste des fichiers dans le dossier `input_directory` ;
foreach *fichier dans la liste des fichiers* **do**
Étape 1 : Construire le chemin complet du fichier audio ;
Étape 2 : Charger le fichier audio ;
Obtenir le signal audio et la fréquence d'échantillonnage originale ;
Étape 3 : Rééchantillonner le signal audio à la fréquence d'échantillonnage cible ;
Obtenir le signal rééchantillonné ;
Étape 4 : Construire le chemin complet du fichier de sortie ;
Étape 5 : Enregistrer le signal rééchantillonné dans le fichier de sortie ;
end

La figure ci−dessous représente deux graphiques : le premier montre le signal original et le second montre le signal rééchantillonné.

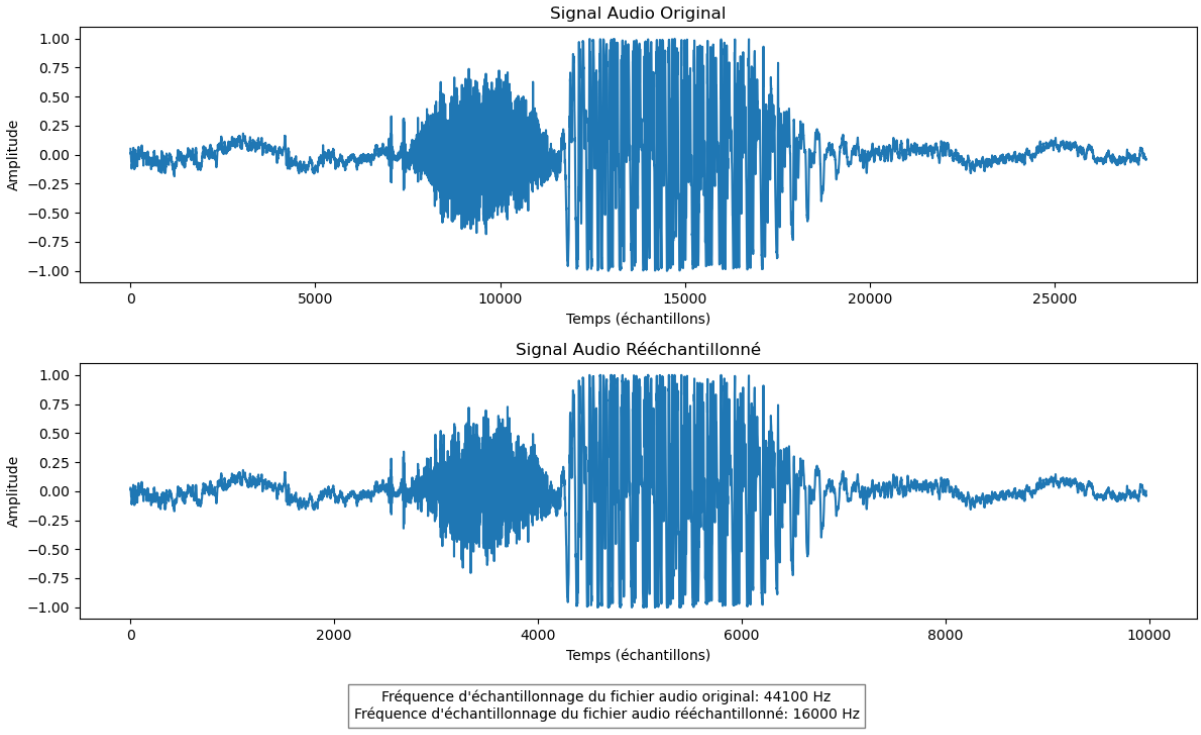


FIGURE 4.1 – Avant & Aprés Rééchantillonnage

4.2.2 Normalisation

Définition 23. ***Normalisation***

La normalisation des signaux audio, signifie ajuster le volume d'un signal audio pour qu'il ne soit ni trop faible ni trop fort. Cela se fait en trouvant le niveau de volume maximal dans le signal, puis en modifiant le volume de tout le signal pour qu'il atteigne un niveau souhaité, souvent proche de 1. Cela garantit que le son reste clair et agréable à écouter, quel que soit le système audio utilisé.

Technique de normalisation choisie

Pour normaliser nos fichiers audio, on a opté pour la méthode de "normalisation par amplification". Cette méthode consiste à ajuster l'amplitude du signal audio en le multipliant par un facteur d'amplification calculé en divisant chaque échantillon par l'amplitude maximale du signal.

Ce processus garantit que l'amplitude maximale du signal normalisé soit égale à 1. Cette méthode est choisie pour diverses raisons :

— En amplifiant le signal audio, on peut régler les niveaux de volume pour qu'ils restent dans une plage idéale, tout en conservant la variation entre les parties silencieuses et les plus bruyantes.

— Elle permet d'ajuster les niveaux de volume pour qu'ils soient uniformes à travers différentes pistes audio, ce qui évite les variations de volume désagréables lors de la lecture.

Algorithme de normalisation

Algorithm 2: Normalisation des fichiers audio

input :
↓ `input_directory` : Chemin vers le dossier contenant les fichiers audio à normaliser ;
↓ `output_directory` : Chemin vers le dossier où les fichiers normalisés seront enregistrés ;
output:
↑ Fichiers audio normalisés enregistrés dans le dossier de sortie ;
Fonction normalize_audio(signal) :
begin
  Calculer l'amplitude maximale du signal ;
  Normaliser le signal en divisant par l'amplitude maximale ;
  return Retourner le signal normalisé ;
foreach *fichier* **do**
  Construire le chemin complet du fichier audio ;
  Charger le fichier audio ;
  Normaliser le signal audio ;
  Construire le chemin complet du fichier audio normalisé :
  Enregistrer le signal audio normalisé ;

Cet algorithme normalise les fichiers audio dans un répertoire donné. Il commence par définir une fonction pour normaliser les amplitudes des signaux audio ”normalize_audio(signal)” qui prend un signal audio en entrée et normalise ses amplitudes en divisant chaque échantillon par la valeur maximale de son amplitude absolue. Ensuite il parcourt tous les fichiers audio du répertoire d'entrée, charge chaque fichier, normalise son amplitude, puis écrit le fichier normalisé dans un répertoire de sortie.

La figure ci-dessous nous permet de distinguer visuellement la différence entre le signal original et le signal normalisé. Pour ce faire, nous commençons par calculer les valeurs maximales absolues des deux signaux, original et normalisé. Ensuite, nous vérifions si le signal normalisé a une amplitude maximale proche de 1,0. Cela suggère que le signal a été correctement normalisé.

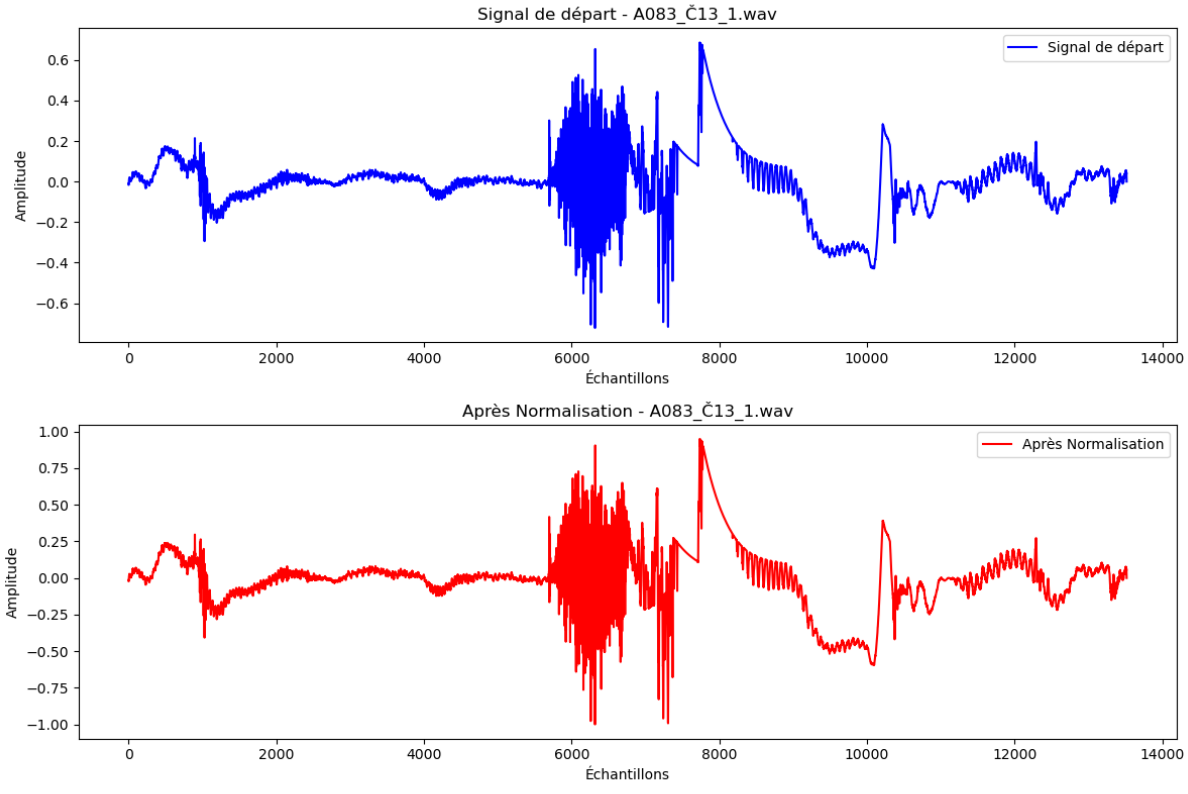


FIGURE 4.2 – Avant & Aprés Normalisation

4.2.3 Réduction du bruit

Définition 24. ***Réduction du bruit*** *:*

Réduire le bruit dans un fichier audio, c'est enlever les bruits gênants, comme les fonds bruyants ou les perturbations, tout en gardant les parties importantes du son. Cela rend l'audio de meilleure qualité en supprimant les distractions et en faisant ressortir les parties essentielles du son.

Téchnique de réduction du bruit choisie

On a employé la bibliothèque "noisereduce" pour atténuer les bruits indésirables dans les fichiers audio en utilisant la fonction "nr.reduce_noise". Ce choix est justifié par plusieurs raisons, notamment :

— La bibliothèque "noisereduce" est réputée pour sa capacité à supprimer efficacement le bruit des fichiers audio tout en préservant les parties essentielles du son.

— La fonction "nr.reduce_noise" de la bibliothèque "noisereduce" est conçue pour être simple à utiliser, avec une interface claire et des paramètres ajustables selon les besoins.

— La fonction "nr.reduce_noise" est capable de traiter des fichiers audio de différentes tailles et longueurs tout en maintenant une qualité sonore acceptable.

Processus de réduction de bruit

La fonction "nr.noise_reduce" de la bibliothèque "noise_reduce" applique des principes mathématiques avancés de prétraitement du signal pour réaliser la réduction de bruit :

1. **Transformée de Fourier à Court Terme (TFCT) :** la fonction "nr.noise_reduce" commence par utiliser la transformée de fourier à court terme pour transformer le signal audio du domaine temporel au domaine fréquentiel.

   Mathématiquement [9] :

$$X(t,f) = \sum_{n=0}^{N-1} x[n] \cdot w(n-t) \cdot \exp\left(-\frac{j2\pi fn}{N}\right) \tag{4.1}$$

   Où :
   — $X(t,f)$: le spectrogramme, représentant la composante fréquentielle f à l'instant t.
   — $x[n]$: le signal d'entrée.
   — w$(n-t)$: une fonction de fenêtre centrée autour de t.
   — t : le temps.
   — f : la fréquence.
   — $\exp\left(-\frac{j2\pi fn}{N}\right)$: est le noyau de la Transformée de Fourier, avec j l'unité imaginaire.

2. **Estimation du Spectre de Bruit :** une fois le signal transformé en domaine fréquentiel, la fonction estime le spectre de bruit, soit en analysant les segments silencieux du signal, soit en utilisant des méthodes statistiques pour évaluer le bruit de fond constant. Le spectre de bruit $B(t, f)$ est généralement obtenu en calculant la moyenne des sections du spectrogramme où il n'y a que du bruit et pas de signal utile. Cela permet d'estimer les composantes de bruit présentes à chaque fréquence et à chaque instant, fournissant ainsi une référence pour la réduction du bruit.

3. **Application du Filtrage Spectral :** la réduction de bruit repose sur la soustraction des composantes bruitées du spectrogramme. Cela peut être formulé par :

$$Y(t, f) = H(t, f) \cdot X(t, f) \tag{4.2}$$

Où :

— $Y(t, f)$: le spectrogramme filtré.
— $H(t, f)$: une fonction de gain qui est déterminée en fonction du rapport signal/bruit estimé. Elle est obtenue par :

$$H(t, f) = \frac{|X(t, f)|^2 - |B(t, f)|^2}{|X(t, f)|^2} \tag{4.3}$$

Où :

— $|X(t, f)|^2$: la puissance spectrale du signal bruité.
— $|B(t, f)|^2$: la puissance spectrale du bruit estimé.

4. **Transformée de Fourier Inverse à Court Terme (TFICT) :** le spectrogramme filtré est reconverti dans le domaine temporel en utilisant la Transformée de Fourier Inverse à Court Terme (TFICT). Cela permet de reconstituer le signal audio avec le bruit réduit.

$$y(n) = \sum_{m=0}^{M-1} \sum_{k=0}^{N-1} Y(m, k) \cdot \exp\left(-\frac{j2\pi kn}{N}\right) \cdot w[n - mR] \tag{4.4}$$

Où :

— $y[n]$: le signal reconstruit dans le domaine temporel.
— $Y(m, k)$: le spectrogramme filtré du signal. Il représente les composantes fréquentielles du signal après avoir été filtré à l'instant m et à la fréquence k.
— N : la taille de la fenêtre.
— $w[n - mR]$: une fenêtre temporelle appliquée aux échantillons du signal. Elle est utilisée pour lisser et pondérer les parties du signal pendant sa reconstruction.

Algorithme de réduction du bruit

Algorithm 3: Réduction du bruit des fichiers audio

input :
↓ `input_directory` : Chemin vers le dossier contenant les fichiers audio bruités ;
↓ `output_directory` : Chemin vers le dossier où les fichiers réduits du bruit seront enregistrés ;
output:
↑ Fichiers audio débruités enregistrés dans le dossier de sortie ;
foreach *fichier dans input_directory* **do**
chargement du fichier audio ainsi que sa fréquence d'échantillonage;
réduire le bruit dans le signal audio original avec la méthode *noisereduce*;
enregistrement du fichier audio débuité dans output_directory ;

La fonction reduce_noise de la bibliothèque noisereduce est utilisée dans cet algorithme pour réduire le bruit des fichiers audio. Cette fonction prend en entrée le signal audio original ainsi que sa fréquence d'échantillonnage, et elle retourne une version du signal avec le bruit réduit.

À l'intérieur de la fonction, le bruit est réduit en utilisant des algorithmes adaptés de prétraitement du signal, qui peuvent inclure des techniques telles que la soustraction spectrale ou la suppression adaptative de bruit. Une fois le prétraitement terminé, la fonction renvoie le signal audio avec le bruit réduit.

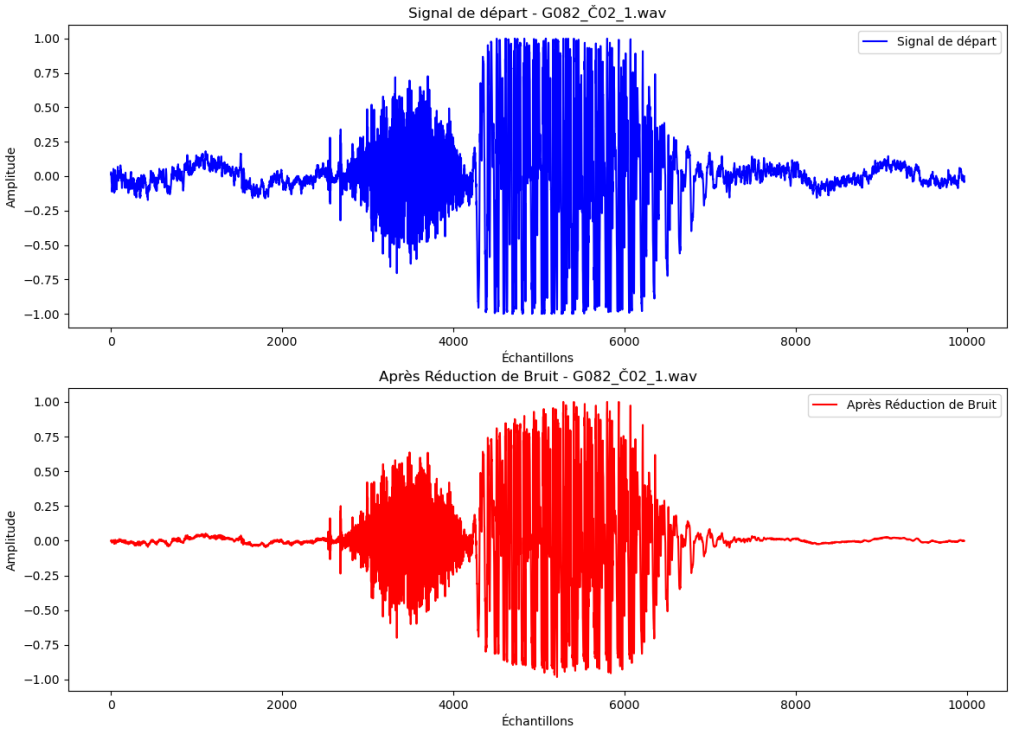


FIGURE 4.3 – Avant & Aprés Réduction du bruit.

Mesures de performance de la réduction du bruit

— **Rapport signal sur bruit (Signal-to-Noise Ratio , SNR) :** il représente la mesure de la puissance du signal par rapport à celle du bruit qui l'entoure. Le SNR peut être déterminé en utilisant une formule fixe qui compare les deux niveaux et renvoie le rapport, ce qui montre si le niveau de bruit impacte le signal souhaité. Ce ratio est généralement exprimé comme une seule valeur numérique en décibels (dB).

Le ratio peut être de zéro, un nombre positif ou un nombre négatif. Un rapport signal sur bruit supérieur à 0 dB indique que le signal est plus fort que le bruit. Plus le ratio est élevé, plus le signal est fort par rapport au bruit de fond [52].

l'équation du SNR est exprimée comme suit :

$$\text{SNR} = 10 \cdot \log_{10}\left(\frac{P_{\text{signal}}}{P_{\text{bruit}}}\right) \tag{4.5}$$

Où :

— $P_{\textbf{signal}}$: représente la puissance du signal désiré.
— $P_{\textbf{bruit}}$: représente la puissance du bruit de fond.

Pour calculer la puissance du signal (P_{signal}), on prend la moyenne des carrés de chaque échantillon de l'audio propre :

$$P_{\text{signal}} = \frac{1}{N}\sum_{i=1}^{N}(\text{audio propre}[i])^2 \tag{4.6}$$

Pour calculer la puissance du bruit (P_{bruit}), on prend la moyenne des carrés de la différence entre chaque échantillon de l'audio propre et de l'audio débruité :

$$P_{\text{bruit}} = \frac{1}{N}\sum_{i=1}^{N}(\text{audio propre}[i] - \text{audio débruité}[i])^2 \tag{4.7}$$

— N : représente le nombre total d'échantillons dans l'audio.

— **L'erreur quadratique moyenne (Mean Squared Error, MSE) :** le MSE est une mesure de l'erreur entre deux ensembles de valeurs. Pour deux ensembles de données, réels et estimés, de longueur N, le MSE est calculé en prenant la moyenne des carrés des différences entre chaque valeur réelle et sa valeur estimée correspondante [29].

Mathématiquement, cela peut être exprimé comme suit :

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(\text{real}[i] - \text{estimate}[i])^2 \tag{4.8}$$

Où :

— real[i] : représente la valeur réelle à la position i, c'est à dire le signal débruité.
— estimate[i] : représente la valeur estimée à la position i, c'est à dire le signal bruité.
— N : est le nombre total de valeurs dans les ensembles de données.

la figure ci−dessous contient les valeurs du SNR et du MSE calculées pour 9 fichiers audio.

```
SNR pour A000_B01_1.wav: 11.686553955078125 dB
MSE pour A000_B01_1.wav: 0.04325530305504799
SNR pour A000_B01_2.wav: 6.82589054107666 dB
MSE pour A000_B01_2.wav: 0.04325530305504799
SNR pour A000_B01_3.wav: 13.231750726699829 dB
MSE pour A000_B01_3.wav: 0.04325530305504799
SNR pour A000_B01_4.wav: 4.5531392097473145 dB
MSE pour A000_B01_4.wav: 0.04325530305504799
SNR pour A000_B01_5.wav: 6.042962074279785 dB
MSE pour A000_B01_5.wav: 0.04325530305504799
SNR pour A001_D01_1.wav: 9.927513599395752 dB
MSE pour A001_D01_1.wav: 0.04325530305504799
SNR pour A001_D01_2.wav: 10.977141857147217 dB
MSE pour A001_D01_2.wav: 0.04325530305504799
SNR pour A001_D01_3.wav: 3.754744529724121 dB
MSE pour A001_D01_3.wav: 0.04325530305504799
SNR pour A001_D01_4.wav: 3.838323652744293 dB
MSE pour A001_D01_4.wav: 0.04325530305504799
```

FIGURE 4.4 – SNR & MSE.

4.2.4 Extraction des caractéristiques

L'extraction des caractéristiques distinctives est une phase fondamentale du système de reconnaissance vocale. Elle permet de créer des vecteurs caractérisant les signaux audio segmentés.

Plusieurs algorithmes, d'extraction de caractéristiques peuvent être utilisées pour effectuer cette tâche, à titre d'exemples on peut citer les méthodes suivantes : Linear Predictive Coefficients (LPC), Linear Predictive Cepstral Coefficients (LPCC), Mel Frequency Cepstral Coefficients (MFCC), Perceptual Linear Predictive (PLP), PLP− Relative Spectra (PLP− RASTA) et Human Factor Cepstral Coefficient (HFCC).

Depuis quelques années, les coefficients cepstraux de fréquence à échelle de Mel (MFCC) sont les caractéristiques acoustiques les plus couramment employées dans les systèmes de reconnaissance automatique de la parole, à cause de leurs bonnes performances de précision [59] [51].

Les coefficients MFCC (Mel Frequency Cepstral Coefficients)

La parole étant un signal constitué d'une infinité d'informations, il faut en extraire les informations les plus importantes. Le processus d'extraction peut se résumer comme suit :

1. **Décomposition du signal en trames** : Le signal audio est découpé en petites portions de taille fixe, souvent avec une partie commune entre chaque portion pour éviter les discontinuités brusques entre les trames.

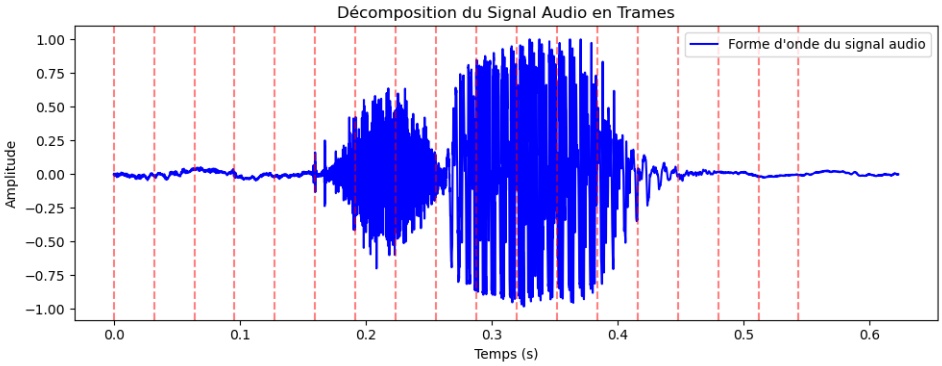


FIGURE 4.5 – Décomposition du signal en trames dans le domaine temporel

2. **Fenêtrage** : pour minimiser les discontinuités aux frontières des trames, on applique une fenêtre de Hamming à chaque trame avant d'effectuer la transformation de Fourier à court terme (STFT). Cela permet de lisser les bords du signal.

$$w(n) = 0.54 - 0.46 \cdot \cos\left(\frac{2\pi n}{N-1}\right) \tag{4.9}$$

Où :
— $w(n)$: la valeur de la fenêtre de Hamming à l'indice n.
— N : la longueur totale de la fenêtre.
— n : représente l'indice de l'échantillon dans la fenêtre.

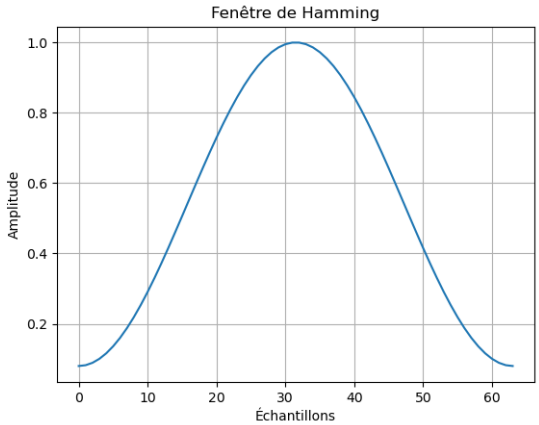


FIGURE 4.6 – Fenêtre de Hamming

3. **Transformation de Fourier à court terme (TFCT)** : Pour chaque trame du signal, une opération appelée Transformation de Fourier à Court Terme (TFCT) est effectuée, elle permet de convertir le signal audio du domaine temporel au domaine fréquentiel.

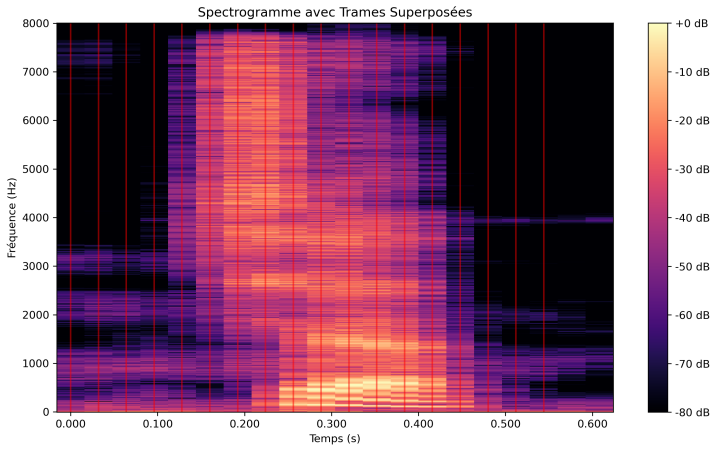


FIGURE 4.7 – Décomposition du signal en trames dans le domaine fréquentiel

4. **Calcul des coefficients MFCC** : les coefficients MFCC sont calculés pour chaque partie du signal audio, appelée trame. Tout d'abord, le spectre de puissance est obtenu à partir de chaque trame après avoir effectué une transformation de Fourier cité precedemment. Ensuite, pour rendre compte de la façon dont l'oreille humaine perçoit les différentes fréquences, le spectre est converti à une échelle spéciale appelée échelle de Mel. Pour ce faire, on applique une série de filtres triangulaires qui segmentent le spectre en bandes de fréquences. Ils sont plus larges dans les basses fréquences et se resserrent progressivement à mesure que les fréquences augmentent [51].

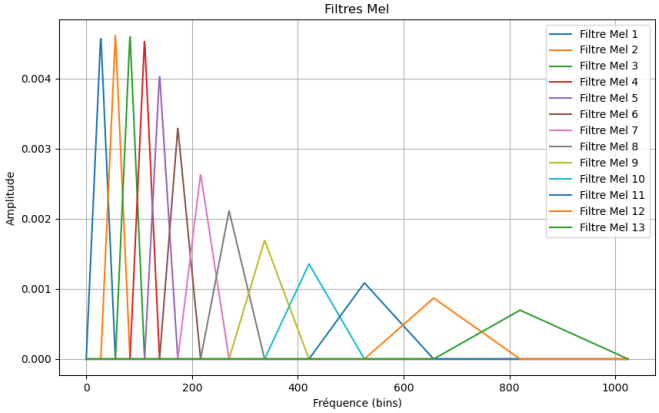


FIGURE 4.8 – Banc de filtres de Mel.

pour convertir une fréquence donnée en Hertz (f) en une valeur sur l'échelle de Mel (m) On utilise l'équation suivante :

$$m = 2595 \cdot \log_{10}\left(1 + \frac{f}{700}\right) \tag{4.10}$$

5. Pour finir, nous travaillons avec le spectre de Mel et le convertissons en domaine temporel en utilisant la transformée en cosinus discrète (DCT). Cela permet de réduire le nombre de données caractérisant le signal. En général, nous extrayons les 13 premiers coefficients MFCC, ce qui capture les informations essentielles du signal audio [51].

Algorithme de MFCC

L'algorithme de MFCC extrait les coefficients MFCC à partir de fichiers audio dans un répertoire spécifié. Il construit le chemin complet de chaque fichier, charge le signal audio et son taux d'échantillonnage (sr), puis calcule les MFCCs en utilisant une taille de fenêtre spécifiée (n_fft) et un chevauchement (hop_length).

Le signal est transformé en spectre de puissance, converti en échelle de Mel avec des filtres triangulaires. Les amplitudes des bandes de fréquence de Mel sont logarithmées, et une transformée en cosinus discrète (DCT) est appliquée pour obtenir les coefficients MFCC.

Algorithm 4: Extraction des MFCC des fichiers audio

input :
↓ `input_directory` : Chemin vers le dossier contenant les fichiers audio ;
output:
↑ MFCCs extraits pour chaque fichier audio ;
Fonction extract_mfcc (signal, sr, n_mfcc, n_fft, hop_length) :
begin
  Calculer les MFCC en utilisant la fonction `librosa.feature.mfcc` ;
  Retourner la matrice des MFCCs transposée ;
foreach *fichier dans* `input_directory` **do**
  Construire le chemin complet du fichier audio ;
  Charger le fichier audio et obtenir le signal ainsi que son taux d'échantillonnage ;
  Extraire les MFCC ;
  Afficher les MFCC pour le fichier ;
  Afficher un message de confirmation du traitement réussi pour le fichier ;

4.3 Méthodes de reconnaissance vocale

Les algorithmes de reconnaissance vocale sont essentiels pour transformer la parole humaine en texte. Ils s'appuient sur divers concepts issus du traitement du signal, de l'apprentissage automatique et de la linguistique informatique. Ces algorithmes peuvent être classés en deux grandes catégories : les méthodes basées sur l'apprentissage automatique et les méthodes basées sur l'apprentissage profond.

4.3.1 Méthodes basées sur l'apprentissage automatique

1. **Comparaison dynamique (dynamic time warping DTW) [4] :**

   Le Dynamic Time Warping (DTW) est une méthode efficace pour la reconnaissance de la parole, qui compare dynamiquement un vecteur de référence, extrait d'un échantillon audio connu, avec un vecteur de test, extrait d'un échantillon audio à reconnaître. En alignant tous les points des deux vecteurs et en utilisant la programmation dynamique pour minimiser le coût total de cet alignement, DTW permet de trouver la meilleure correspondance. Cette méthode est simple et nécessite peu de données d'apprentissage, mais elle est limitée par la taille du vocabulaire à reconnaître.

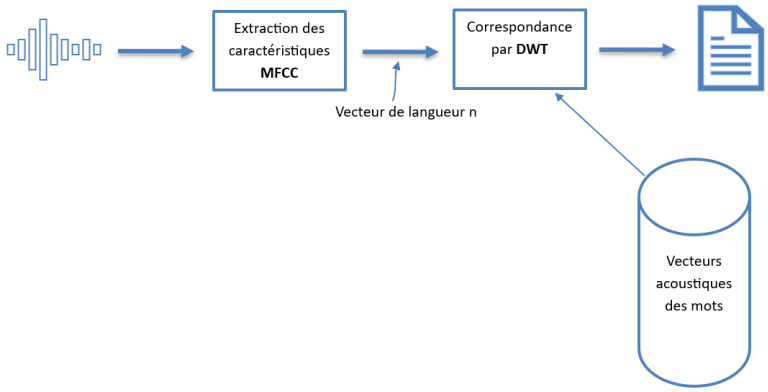


FIGURE 4.9 – Comparaison dynamique.

2. **Modèle de Markov caché (MMC) [5] [55] [4] :**

   Un modèle de Markov caché (MMC) est un outil statistique utilisé pour modéliser des séquences de données, avec des états observables et cachés, des matrices de transition et d'émission, et des distributions initiales. Dans le contexte de la reconnaissance vocale, chaque état représente une partie du signal audio, avec une transition entre les états basée sur des probabilités. Les MMC sont utilisés pour des petits vocabulaires, mais leur utilisation devient difficile pour des vocabulaires plus vastes.

   La figure ci-dessous représente un phonème ainsi que les coarticulations avec les phonèmes adjacents :

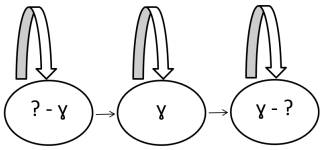


FIGURE 4.10 – Représentation d'un phonème.

3. **Modèles Gaussiens Mixtes (MGM) [13] :**

   Les modèles gaussiens mixtes (MGM) sont des modèles statistiques utilisés pour représenter la distribution d'une variable aléatoire en combinant plusieurs distributions normales. Chaque composante gaussienne est caractérisée par sa moyenne, sa variance et son poids. En reconnaissance vocale, les variations sonores sont décomposées en plusieurs composantes gaussiennes, et ces modèles sont entraînés sur des ensembles de

données associant des traits acoustiques à des transcriptions textuelles pour reconnaître et décoder la parole. Cependant, les MGM sont sensibles à la qualité et à la quantité des données d'entraînement, nécessitant une quantité importante de données pour estimer précisément les paramètres de chaque composante gaussienne.

4. **Machines à vecteurs de support (SVM)**

   Les SVM sont des algorithmes principalement utilisés pour la classification binaire, mais adaptables à la classification multiclasse via les approches "Un-contre-Tous" et "Un-contre-Un". Dans la première, un modèle est construit pour chaque classe par rapport à toutes les autres classes. Dans la seconde, des classifieurs binaires sont créés pour chaque paire de classes possibles.

   Les SVM sont appréciés pour leur capacité à bien généraliser et à classifier avec précision, mais ils peuvent rencontrer des problèmes avec des données contenant de nombreux termes non significatifs. De plus, ils ne fournissent pas directement des estimations de probabilité pour chaque classe, ce qui peut être limitant dans certains contextes de reconnaissance vocale.

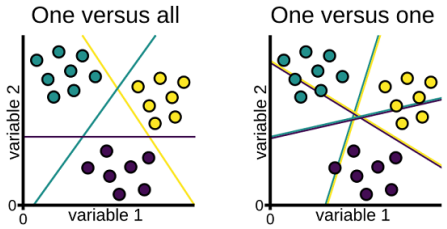


FIGURE 4.11 – Approches du SVM [19].

4.3.2 Méthodes basées sur l'apprentissage profond

Les méthodes modernes de reconnaissance vocale, en particulier celles basées sur le deep learning, reposent largement sur l'utilisation de réseaux de neurones. Ces réseaux de neurones, inspirés du fonctionnement du cerveau humain, sont des structures computationnelles capables d'apprendre des modèles complexes à partir de données audio brutes.

Définition 25. ***Neurone biologique :***

Un neurone est une cellule vivante du système nerveux, fondamentale pour le traitement de l'information dans le cerveau. Sa structure varie et peut adopter différentes formes telles que pyramidale, sphérique ou étoilée [40].

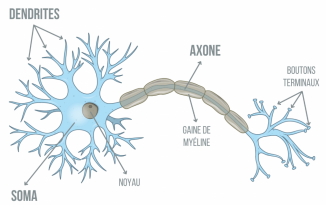


FIGURE 4.12 – Neurone biologique.

Définition 26. ***Neurone formel :***

Selon ***Mc Culloch*** *et* ***Pitts*** *un neurone formel est un modèle mathématique très simple, dérivé de la réalité biologique. Il reçoit plusieurs entrées $x_1, x_2, \ldots, x_n$ et les pondère avec des coefficients appelés poids synaptiques. Ensuite, il applique une fonction d'activation à la somme pondérée des entrées pour produire une sortie y [21].*

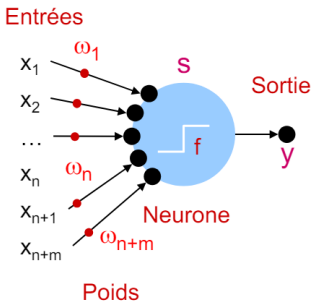


FIGURE 4.13 – Neurone artificiel.

où :

— x_i : un ensemble d'entrées.
— w_i : un ensemble de poids.
— S : un seuil ou potentiel d'activation.
— f : une fonction d'activation.
— y : une sortie.

Définition 27. ***Réseau de neurones artificiel (RNA) :***

Les réseaux de neurones artificiels sont constitués de plusieurs couches de calcul, avec des couches cachées entre l'entrée et la sortie. Ces couches sont ainsi nommées car les opérations qu'elles effectuent ne sont pas directement visibles pour l'utilisateur. Dans ces réseaux, chaque nœud d'une couche est connecté à tous les nœuds de la couche suivante, déterminant ainsi l'architecture du réseau en fonction du nombre de couches et de nœuds.

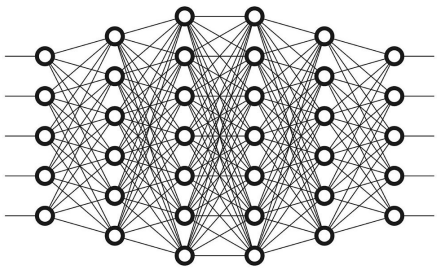


FIGURE 4.14 – Réseau de neurones artificiel.

Composants du réseau de neurones artificiel

Un réseau de neurones artificiel est composé de plusieurs éléments essentiels :

— **Fonction d'activation [49] :** la fonction d'activation est une opération mathématique qui normalise les signaux des neurones et introduit la non-linéarité dans le réseau. Ce qui permet de modéliser des relations complexes et de résoudre des problèmes plus difficiles. Les réseaux de neurones peuvent utiliser de nombreuses fonctions d'activation différentes, nous aborderons les fonctions d'activation les plus courantes :

1. **Fonction d'activation Softmax [67] :** la fonction d'activation softmax attribue une probabilité à chaque classe possible pour une entrée donnée. Elle assure que la somme de toutes ces probabilités est égale à 1. Ainsi, le neurone avec la plus haute probabilité représente la classe prédite. Cette approche permet une interprétation probabiliste claire des prédictions du réseau, facilitant ainsi la classification précise des données.

$$\Phi_i = \frac{e^{z_i}}{\sum_j e^{z_j}} \tag{4.11}$$

où :
— Φ_i : la sortie du neurone i.

— z_i : la valeur de sortie brute du neurone i.

— j : indices de tous les neurones du groupe (niveau).

— z_j : les valeurs de sortie brutes de tous les neurones du groupe (niveau).

2. **Fonction d'activation sigmoïde [67] :** la fonction sigmoïde, ou fonction logistique, est fréquemment choisie dans les réseaux de neurones. Les valeurs positives sont transformées vers 1 et les valeurs négatives vers 0. La fonction sigmoïde est souvent privilégiée dans les modèles où la sortie doit représenter une probabilité, car les probabilités se situent uniquement entre 0 et 1.

$$f(x) = \frac{1}{1 + e^{-x}} \tag{4.12}$$

Où :
— x : est la somme pondérée des entrées du neurone.

3. **Fonction de la tangente hyperbolique (tanh) [67] :** la fonction tanh est similaire à la fonction sigmoïde logistique mais avec des propriétés améliorées. La principale différence est que tanh transforme les valeurs d'entrée en une plage allant de -1 à 1, tandis que la sigmoïde logistique les transforme en une plage allant de 0 à 1.

$$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{4.13}$$

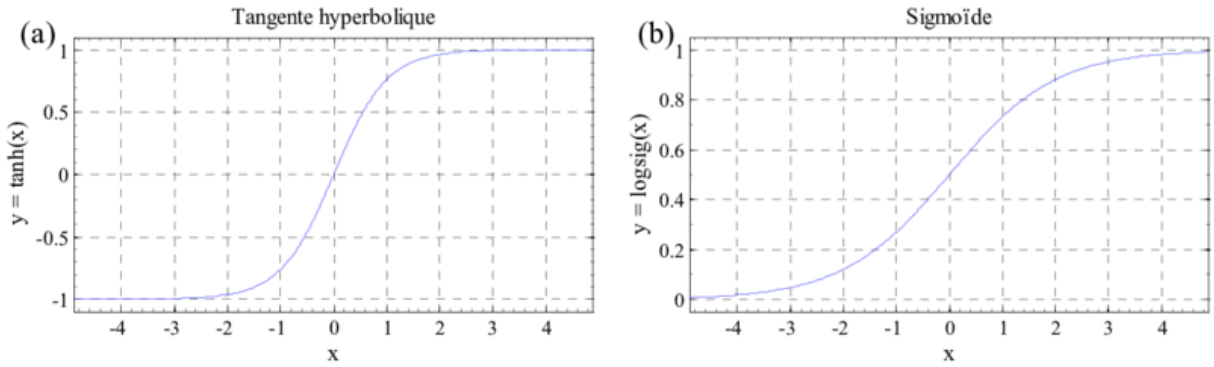


FIGURE 4.15 – Sigmoide VS Tanh.

4. **Fonction d'activation ReLU [67] :** la fonction ReLU (Rectified Linear Unit) est largement utilisée aujourd'hui en raison de ses performances supérieures lors de l'entraînement des réseaux de neurones. La plupart des réseaux de neurones utilisent ReLU pour les couches cachées en raison de sa capacité à accélérer l'apprentissage, tandis que la fonction softmax est souvent utilisée pour la couche de sortie.
La fonction ReLU est une fonction linéaire non saturante, c'est-à-dire qu'elle ne sature pas à -1, 0 ou 1.
L'équation de la fonction est :

$$\Phi(x) = \max(0, x) \tag{4.14}$$

— $\Phi(x) = 0$, lorsque x est inférieur à 0.

— $\Phi(x) = x$, lorsque x est supérieur ou égal à 0.

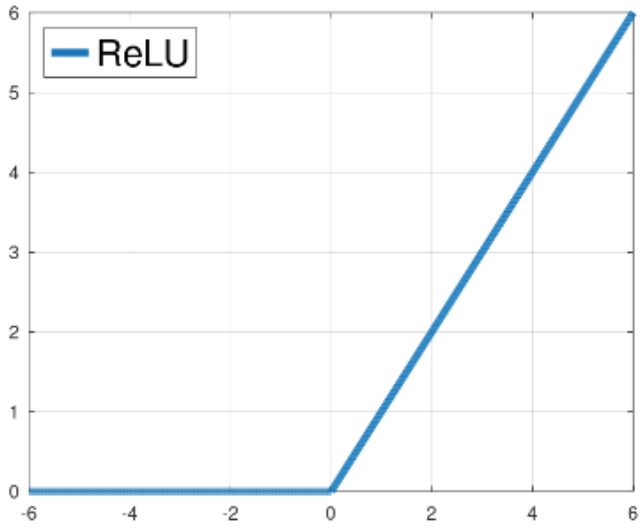


FIGURE 4.16 – ReLU.

4.4 Réseaux de neurones convolutifs

Définition 28. ***Réseaux de neurones convultifs :***

Un réseau de neurones convolutif (ou Convolutional Neural Network en anglais, CNN) est un type de modèle d'apprentissage profond spécialisé dans le traitement de données structurées, principalement utilisé pour des tâches comme la classification d'images. Ce modèle a été développé par ***Yann LeCun*** *en 1989. Sa conception s'inspire biologiquement du système visuel humain [24].*

4.4.1 Couches de CNN

Un CNN est simplement un empilement de plusieurs couches. Chaque son reçu en entrée va donc être filtré, réduit et corrigé plusieurs fois, pour finalement former un vecteur.

— **Couche de convolution :** la couche de convolution est centrale dans les réseaux neuronaux convolutifs. Elle utilise des filtres pour transformer un son en feature map qui indique l'emplacement et l'intensité des caractéristiques acoustiques détectées.

— **Couche de pooling :** la couche de pooling est généralement placée entre deux couches de convolution. Elle réduit la dimensionnalité des caractéristiques extraites après la convolution initiale, préservant les informations importantes [34].

— **Flattening :** intervient après les phases de convolution et de pooling. Il transforme les "feature maps" multidimensionnelles en un vecteur unidimensionnel en réorganisant toutes les valeurs sans tenir compte de leur position spatiale. Ce vecteur devient la couche d'entrée pour les couches entièrement connectées du réseau.

— **ReLU :** est une fonction d'activation qui prend une valeur en entrée et si cette valeur est négative, elle la remplace par zéro ; si elle est positive ou nulle, elle la laisse telle quelle [34].

— **Couche fully-connected :** est la dernière couche d'un réseau de neurones qui combine les features extraites par les couches précédentes pour classifier le son. Elle produit un vecteur de taille N, où N représente le nombre de classes dans le problème de classification sonore. Chaque composant de ce vecteur indique la probabilité que le son appartienne à une classe spécifique. La classe avec la probabilité la plus élevée est désignée comme la prédiction pour le son d'entrée [34].

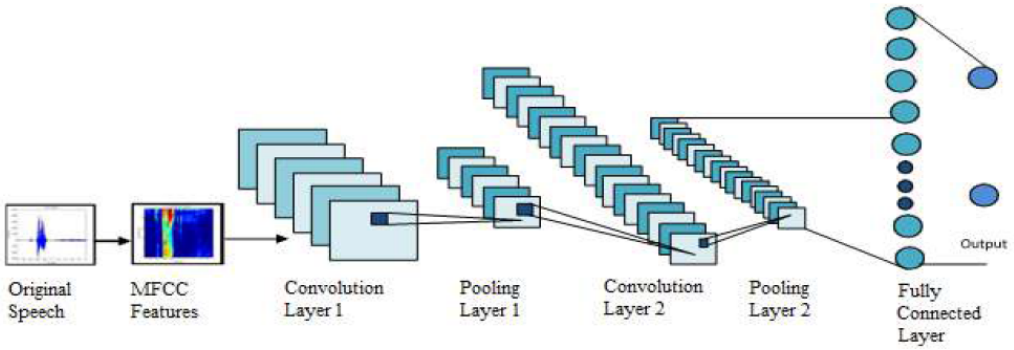


FIGURE 4.17 – Architecture d'un CNN [34].

4.5 Réseaux de neurones récurrents

Définition 29. ***Réseau de neurones récurrent :***

Un réseau de neurones récurrent (RNN), développé par ***Rumelhart*** *et d'autres en 1986 [25], est une famille de réseaux neuronaux caractérisée par des boucles internes permettant aux informations de persister et d'être utilisées dans les itérations suivantes. Contrairement aux réseaux de neurones traditionnels où l'information se propage uniquement de couche en couche, dans un RNN chaque étape de traitement des données dépend non seulement de l'entrée actuelle mais aussi de la sortie obtenue à partir de l'étape précédente. Les RNN sont très utilisés dans la reconnaissance vocale, où la nature séquentielle des données est dominante [45].*

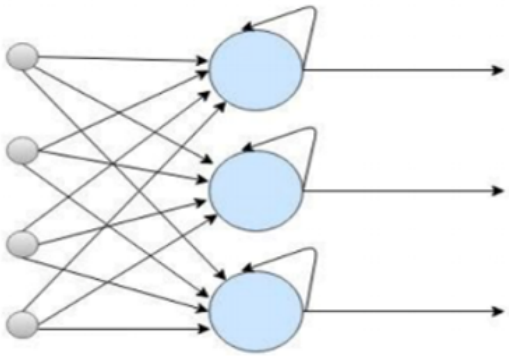


FIGURE 4.18 – RNN.

4.5.1 Fonctionnement d'un RNN

1. **Propagation avant :** considérons une séquence d'entrée $\{x_t\}_{t\in\{1,T\}}$ de dimension d. La cellule neuronale récurrente est définie par la séquence de ses états internes $\{h_t\}_{t\in\{1,T\}}$ de dimension l. La séquence des états internes est définie par l'équation récurrente :

$$h_t = \phi_t(x_t, h_{t-1}) = \sigma(W_h h_{t-1} + W_x x_t + b) \tag{4.15}$$

où :
— σ : la fonction d'activation.
— h_t : l'état interne au temps t.
— x_t : l'entrée au temps t.
— h_{t-1} : l'état interne au temps $t-1$.
— W_h : la matrice de poids pour l'état précédent.
— W_x : la matrice de poids pour l'entrée actuelle.
— b : le biais.

Autrement dit, l'état interne h_t au temps t dépend à la fois de la valeur de la séquence à ce même pas de temps et de son état interne au pas de temps précédent. L'état interne permet donc de modéliser l'historique de la séquence jusqu'au pas de temps t considéré. En général, la fonction de transfert ϕ_t permet de combiner l'entrée actuelle et l'état précédent de la cellule pour calculer l'état actuel, elle est identique pour tous les pas de temps.

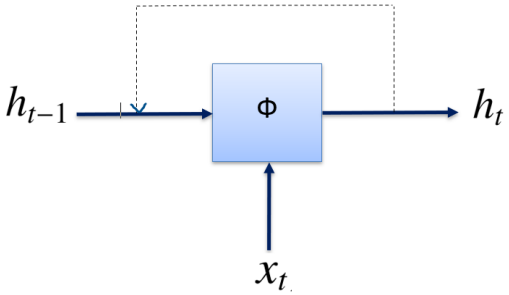


FIGURE 4.19 – Cellule récurrente.

La figure *ci-dessus* montre une version déroulée du schéma de la cellule récurrente, illustrant les étapes du réseau à chaque instant de temps.

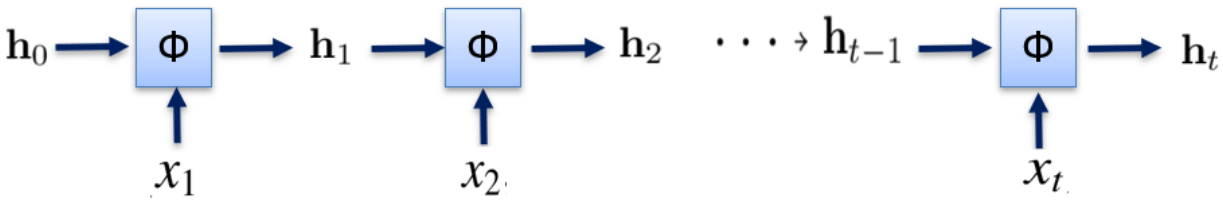


FIGURE 4.20 – Version déroulée d'un cellule récurrente

2. **Propagation Arrière :** après la propagation avant et la production des sorties, une fonction de perte $L(y, \hat{y})$ est calculée pour mesurer l'écart entre les valeurs prédites $\hat{y}$ et les valeurs réelles y, c'est-à-dire l'erreur. Cette fonction est utilisée sur les données d'entraînement pour minimiser l'erreur en ajustant les poids w du réseau.
La fonction de perte la plus courante est :

$$L(y, \hat{y}) = -\sum_i y_i \log(\hat{y}_i) \tag{4.16}$$

où :
— y_i : la valeur réelle de la classe i.
— $\hat{y}_i$: la probabilité prédite par le modèle que l'échantillon appartienne à la classe i.

Après avoir calculé la fonction de perte, on calcule la dérivée de cette dernière par rapport à chaque sortie prédite $\hat{y}_i$ pour quantifier l'impact de chaque prédiction sur la perte globale :

$$\frac{\partial L}{\partial \hat{y}_i} = -\frac{y_i}{\hat{y}_i} \tag{4.17}$$

L'étape suivante consiste à mettre à jour les poids, pour un poids w dans une couche donnée, le gradient est calculé en utilisant la régle de dérivation en chaine. Pour chaque couche du réseau :

— **Calcul du gradient par rapport à la sortie de la couche $\hat{y}$:** $\frac{\partial L}{\partial \hat{y}}$ c'est la dérivée de la fonction de perte par rapport à la sortie prédite $\hat{y}$.
— **Gradient par rapport à la somme pondérée des entrées z :** $\frac{\partial \hat{y}}{\partial z}$ c'est la dérivée de la sortie de la couche par rapport à la somme pondérée des entrées z.

$$z = \sum_{i=1}^{n} (w_i \cdot x_i) + b \tag{4.18}$$

— **Gradient de la somme pondérée des entrées par rapport au poids w :** $\frac{\partial z}{\partial w}$ c'est la dérivée de la somme pondérée des entrées z par rapport au poids w.

Ainsi, pour mettre à jour les poids w, on utilise la règle de dérivation en chaîne :

$$\frac{\partial L}{\partial w} = \frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial z} \cdot \frac{\partial z}{\partial w} \tag{4.19}$$

Le poids mis à jour est calculé ainsi pour chaque poids du réseau :

$$w_{\text{nouveau}} = w_{\text{actuel}} - \alpha \cdot \frac{\partial L}{\partial w} \tag{4.20}$$

α : le taux d'apprentissage.

4.5.2 Architectures du RNN

1. **Réseaux de neurones récurrents bidirectionnels (BRNN)**

   Dans le contexte de la reconnaissance vocale, comprendre ce qui vient après dans la séquence est aussi crucial que de comprendre ce qui s'est déjà passé pour prendre une décision précise [42].

   Un RNN bidirectionnel (**Schuster** et **Paliwal**, 1997) combine deux RNN indépendants. L'un traite l'entrée du début à la fin, et l'autre de la fin au début. Nous concaténons ensuite les deux représentations calculées par les réseaux en un seul vecteur qui capture les contextes gauche et droit d'une entrée à chaque instant [14].

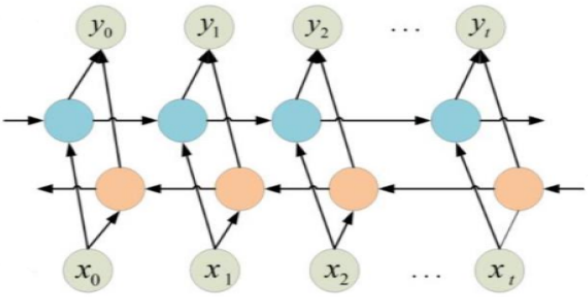


FIGURE 4.21 – BRNN.

(a) **Équation de l'état caché pour la propagation avant :**

$$\overrightarrow{h_t} = \sigma(W_x \cdot x_t + W_h \cdot \overrightarrow{h_{t-1}} + b) \tag{4.21}$$

— $\overrightarrow{h_t}$: est l'état caché à l'instant t pour la couche avant.

(b) **Équation de l'état caché pour la propagation arrière :**

$$\overleftarrow{h_t} = \sigma(W_x \cdot x_t + W_h \cdot \overleftarrow{h_{t+1}} + b) \tag{4.22}$$

— $\overleftarrow{h_t}$: est l'état caché à l'instant t pour la couche arrière.

Puis, les deux états résultants sont concaténés en un état h_t qui est ensuite utilisé comme entrée pour une couche de sortie, pour produire la prédiction finale.

Problème de la disparition du gradient [7] :

Bien qu'en théorie les RNN devraient conserver la mémoire à travers les étapes temporelles, en pratique, ils ont eu de mauvaises performances. **Hochreiter** (1991) et **Bengio** et **Al** (1994) ont exploré pourquoi les algorithmes d'apprentissage basés sur le gradient rencontrent des problèmes accrus avec la durée des dépendances.

Un problème majeur est la disparition du gradient qui est la dérivée de la perte par rapport aux poids. Si le gradient est trop petit, les poids ne peuvent pas être ajustés efficacement, empêchant le réseau d'apprendre.

Dans un RNN, les couches et les étapes temporelles sont reliées par multiplication. Multiplier un nombre légèrement supérieur à un peut entraîner une explosion des valeurs, et multiplier un nombre légèrement inférieur à un peut les faire disparaître.

Par conséquent, les gradients sont susceptibles de disparaître ou d'exploser. Bien que nous puissions résoudre les gradients explosifs en tronquant ou en mettant les valeurs au carré, les gradients qui disparaissent sont plus difficiles à résoudre.
Ci-dessous se trouve un diagramme des graphes de la fonction tanh et de sa dérivée :

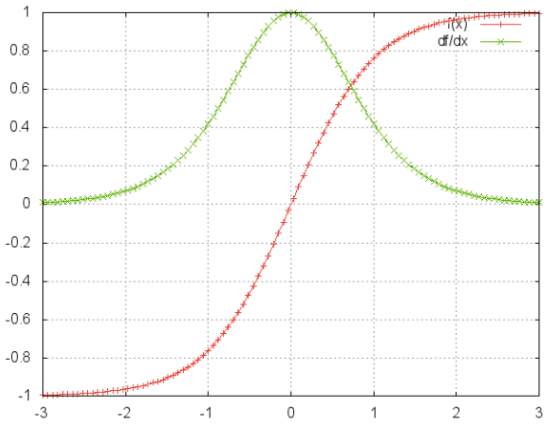


FIGURE 4.22 – Fonction tanh et sa dérivée [7].

2. **Mémoire à long et court terme**

Définition 30. ***Mémoire à long et court terme***

La solution de mémoire à long et court terme (Long Short Terme Memory ou LSTM) a été introduit par ***Schmidhuber*** *et* ***Hochreiter*** *en 1997. C'est un type spécial de RNN capable d'apprendre les dépendances à long terme [41].*

L'idée principale derrière les LSTM repose sur la "cellule mémoire", qui permet de retenir l'information pertinente pour la tâche au fil du temps lors du traitement des éléments de la séquence. Cette cellule mémoire est maintenue grâce à un système de portes [64].

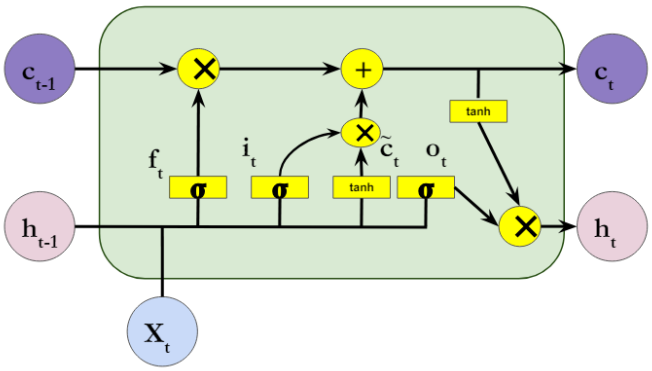


FIGURE 4.23 – Unité LSTM [58].

Fonctionnement d'un LSTM

Une unité LSTM commune comprend une cellule, une porte d'entrée (i_t), une porte de sortie (o_t) et une porte d'oubli (f_t). La cellule maintient et met à jour sa mémoire sur des intervalles de temps arbitraires, tandis que les trois portes contrôlent le flux d'informations entrant et sortant de la cellule [47], elles utilisent une fonction d'activation

segmoid, un résultat d'activation de "0" indique que la porte bloque l'information, tandis qu'un résultat d'activation de "1" permet à l'information de passer et d'être stockée dans la cellule [42].

— La porte d'oubli permet d'oublier une information qui était utile au temps $t-1$ mais qui ne l'est plus au temps t [47]. Sa formule est comme suit :

$$f_t = \sigma(W_f h_{t-1} + U_f x_t + b_f) \tag{4.23}$$

— La porte d'entrée permet à la cellule de stocker une information à l'instant t [47]. Sa formule est comme suit :

$$i_t = \sigma(W_i h_{t-1} + U_i x_t + b_i) \tag{4.24}$$

— La porte de sortie détermine quelle information sera transmise au temps $t+1$ en fonction de la mémoire C et d'une fonction d'activation [47]. Sa formule est comme suit :

$$o_t = \sigma(W_o h_{t-1} + U_o x_t + b_o) \tag{4.25}$$

— La cellule condiadate combine l'information provenant de la sortie précédente h_{t-1} et de l'entrée actuelle x_t pour former une nouvelle proposition d'information a_t, Ensuite, les valeurs sont compressées dans l'intervalle [-1, 1] en appliquant la fonction tanh [42] :

$$a_t = \tanh(W_c h_{t-1} + U_c x_t + b_c) \tag{4.26}$$

— A travers les deux portes précédentes f_t et i_t, une mémoire de répétition est créée pour LSTM [42] :

$$C_t = f_t \cdot C_{t-1} + i_t \cdot a_t \tag{4.27}$$

— La sortie h_t est déterminée en combinant la mémoire à court terme actualisée C_t avec la porte de sortie o_t [42] :

$$h_t = o_t \cdot \tanh(C_t) \tag{4.28}$$

Oú :

— f_t, i_t et o_t : vecteurs représentent les portes d'oubli, d'entrée et de sortie respectivement.

— $W_f, U_f, W_i, U_i, W_o, U_o, W_c, U_c$: sont les matrices de poids à régler durant l'entraînement.

— b_f, b_i, b_o, b_c : sont les vecteurs de biais à régler durant l'entraînement.

— x_t : est le vecteur en entrée de l'unité LSTM au pas t.

— h_t : est le vecteur de sortie (état caché) issu du module LSTM au pas t.

4.6 LSTM Bidirectionnel

Combiner les BRNN avec les LSTM donne les LSTM bidirectionnels (BLSTM), qui peuvent accéder au contexte à long terme dans les deux directions de l'entrée.

Dans l'évaluation automatique, où l'ensemble de la réponse est collecté en une seule fois, il n'y a aucune raison de ne pas exploiter à la fois le contexte futur et le contexte historique. De plus, il n'y a aucune preuve que l'une ou l'autre direction (vers l'avant ou vers l'arrière) soit plus appropriée pour notre tâche, nous modélisons donc la séquence dans les deux directions.

Récemment, les BLSTM introduits par **Schuster** et **Baldi** ont été utilisés dans de nombreux problèmes réels de traitement de séquences tels que la classification des phonèmes, la reconnaissance continue de la parole et la synthèse vocale.

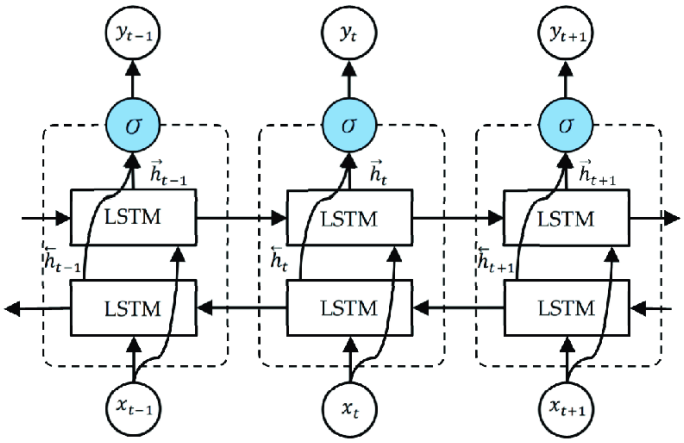


FIGURE 4.24 – BLSTM.

La figure ci-dessus montre que les réseaux de type BLSTM consistent à exécuter deux LSTM en parallèle : le premier réseau lit la séquence d'entrée de droite à gauche et le second réseau en sens inverse de gauche à droite.

Chaque LSTM engendre une représentation cachée : $\overrightarrow{h_t}$ (un vecteur allant de gauche à droite) et $\overleftarrow{h_t}$ (un vecteur allant de droite à gauche) qui sont ensuite combinés afin de calculer la séquence de sortie.

Dans le contexte d'un LSTM bidirectionnel, chaque porte est calculée à partir des états cachés des deux directions de la séquence (avant et arrière) et des entrées actuelles de chaque direction.

— **Porte d'oubli :**

$$f_t = \sigma(W_f^{(f)} h_{t-1}^{(f)} + U_f^{(f)} x_t^{(f)} + b_f^{(f)} + W_f^{(b)} h_{t+1}^{(b)} + U_f^{(b)} x_t^{(b)} + b_f^{(b)}) \tag{4.29}$$

— **Porte d'entrée :**

$$i_t = \sigma(W_i^{(f)} h_{t-1}^{(f)} + U_i^{(f)} x_t^{(f)} + b_i^{(f)} + W_i^{(b)} h_{t+1}^{(b)} + U_i^{(b)} x_t^{(b)} + b_i^{(b)}) \tag{4.30}$$

— **Porte de sortie :**

$$o_t = \sigma(W_o^{(f)} h_{t-1}^{(f)} + U_o^{(f)} x_t^{(f)} + b_o^{(f)} + W_o^{(b)} h_{t+1}^{(b)} + U_o^{(b)} x_t^{(b)} + b_o^{(b)}) \tag{4.31}$$

— **Cellule candidate :**

$$a_t = \tanh(W_c^{(f)} h_{t-1}^{(f)} + U_c^{(f)} x_t^{(f)} + b_c^{(f)} + W_c^{(b)} h_{t+1}^{(b)} + U_c^{(b)} x_t^{(b)} + b_c^{(b)}) \tag{4.32}$$

— **Mise à jour de l'état de la mémoire :**

$$C_t = f_t \cdot C_{t-1} + i_t \cdot a_t \tag{4.33}$$

— **État caché (sortie) :**

$$h_t = o_t \cdot \tanh(C_t) \tag{4.34}$$

Dans le cadre de notre projet de fin d'études, après avoir examiné différents modèles, nous avons conclu que l'implémentation d'un modèle RNN LSTM bidirectionnel est le meilleur choix pour la reconnaissance vocale des lettres de la langue kabyle. Ce choix de modèle nous permettra de capturer efficacement les relations contextuelles dans les séquences vocales, ce qui est crucial pour améliorer la précision de la reconnaissance des caractères dans cette langue.

Conclusion

Dans ce chapitre, nous avons exploré les outils de développement et les différentes étapes du prétraitement des signaux audio, notamment l'extraction des caractéristiques des signaux audio telles que les coefficients cepstraux de fréquence Mel (MFCC). Nous avons également étudié diverses méthodes de reconnaissance vocale qui permettent à la machine de transcrire et de comprendre le langage parlé. Cette exploration nous a permis de choisir le modèle approprié pour réaliser un système de reconnaissance vocale des lettres de la langue kabyle, qui sera implémenté dans le chapitre suivant.

5
Implémentation & Résultats

Introduction

Les systèmes de reconnaissance vocale des lettres de la langue kabyle rencontrent plusieurs défis, notamment la variation illimitée des prononciations humaines.

L'objectif de ce dernier chapitre est de présenter les étapes de l'implémentation du modèle proposé dans le cadre d'un système de reconnaissance vocale des lettres en kabyle. Nous allons implémenter un modèle BLSTM et détailler les différentes étapes de sa réalisation, ainsi qu'une discussion sur les résultats obtenus.

Sommaire

5.1 Environnement de développement

Afin de réaliser et développer notre projet de reconnaissance vocale des lettres de la langue kabyle, nous avons utilisé un ordinateur avec les caractéristiques techniques suivantes :

— **Processeur :** 11th Gen Intel(R) Core(TM) i7-1165G7 @ 2.80GHz 2.80GHz.

— **Mémoire RAM installée :** 8 Go.

— **Disque SSD :** 512 Go SSD.

— **Carte Graphique :** Intel Iris Xe Graphics.

— **IDE :** Jupyter Notebook.

— **Langage de programmation :** Python 3.12.

5.2 Principes clés de l'apprentissage profond

— **Nombre d'Itérations (Epochs) :** correspond au nombre de passages complets sur l'ensemble des données d'apprentissage. Il est définit comme un critère d'arrêt que ce soit les résultats, chaque itération implique une propagation avant à travers le réseau neuronal, suivie d'une propagation arrière pour ajuster les poids [49].

— **Learning Rate :** le taux d'apprentissage définit la rapidité avec laquelle un réseau met à jour ses paramètres. Un faible taux signifie que les poids sont ajustés lentement à chaque itération, prolongeant ainsi le processus d'apprentissage [34].

— **Le sur-ajustement (Overfitting) :** se produit lorsque le modèle se concentre trop sur les données d'entraînement, il finit par mémoriser même le bruit et les fluctuations aléatoires, ce qui l'empêche de bien fonctionner avec de nouvelles données [34].

— **Fonction d'optimisation :** les optimiseurs sont des techniques ou des algorithmes utilisés pour ajuster les paramètres d'un réseau de neurones, tels que les poids et le taux d'apprentissage [41].

— **Dropout :** est un paramètre additionnel qui consiste à ignorer des neurones aléatoirement en les mettant à 0 pour chaque mise à jour des poids pendant l'apprentissage ce qui permet d'éviter le sur-apprentissage.

— **Early stopping :** technique pour éviter le surapprentissage en surveillant les performances du modèle sur un ensemble de validation. Si les performances sur cet ensemble commencent à se détériorer alors que celles sur l'ensemble d'entraînement continuent de s'améliorer, l'entraînement est interrompu pour empêcher le modèle de devenir trop spécialisé aux données d'entraînement [49].

5.3 Métriques d'évaluation

En classification, chaque exemple possède une classe réelle associée, et le système fournit une classe prédite. Il est donc essentiel de définir les paramètres nécessaires, utilisés pour le calcul des métriques d'évaluation [45] :

— **Vrais positifs :** classe réelle = 1, classe prédite = 1.
— **Faux positifs :** classe réelle = 0, classe prédite = 1.
— **Faux négatifs :** classe réelle = 1, classe prédite = 0.
— **Vrais négatifs :** classe réelle = 0, classe prédite = 0.

Les paramètres "Vrais positifs" et "Vrais négatifs" désignent les exemples correctement prédits. Alors que les "Faux positifs" et "Faux négatifs" se produisent lorsque la classe réelle est en contradiction avec la classe prédite.

Pour évaluer les performances d'un modèle sur un ensemble de données. On utilise des métriques telles que :

— **Précision :** mesure la proportion des prédictions positives qui sont réellement correctes. Elle est exprimée par l'équation suivante [34] :

$$\text{Précision} = \frac{\text{Vrais positifs}}{\text{Vrais positifs} + \text{Faux positifs}} \tag{5.1}$$

— **Rappel :** mesure la capacité du modèle à retrouver tous les éléments réellement positifs. Elle est exprimée par l'équation suivante :

$$\text{Rappel} = \frac{\text{Vrais positifs}}{\text{Vrais positifs} + \text{Faux négatifs}} \tag{5.2}$$

— **F-mesure :** la F-mesure aussi appelée F1-score obtenue par la moyenne harmonique de la précision et du rappel. Elle est exprimée par l'équation suivante :

$$\text{F-mesure} = 2 \times \frac{\text{Précision} \times \text{Rappel}}{\text{Précision} + \text{Rappel}} \tag{5.3}$$

5.4 Préparation des données d'entrée du modèle

Dans cette section, nous allons décrire les étapes nécessaires pour préparer les données d'entrée pour notre modèle. Elles consistent en des enregistrements audio au format wav et leurs transcriptions correspondantes dans un fichier CSV.

Chaque fichier audio, une fois traité est transformé en une séquence de MFCC. Ensuite, la longueur maximale des séquences MFCC et des transcriptions est déterminée, afin de définir la longueur maximale pour le padding.

Dans notre cas nous avons déterminé que la plus longue séquence comporte 62 trames, et chaque trame est représenté par 13 coefficients MFCC. Ainsi, Les séquences les plus courtes seront complétées avec des vecteurs de zéros jusqu'à atteindre cette dimension (62, 13).

De la part des transcriptions, nous utilisons la tokenization pour diviser les textes en unités plus petites. Ensuite, nous adaptons le tokenizer aux données d'entrée pour comprendre les phonèmes utilisés et leur fréquence. Chaque phonème est encodé en un nombre selon le vocabulaire appris. Enfin, des zéros sont ajoutés à la fin de chaque séquence pour que toutes les

séquences aient la même longueur, égale à celle des séquences MFCC.

De plus, pour faciliter le processus d'apprentissage, les séquences cibles sont converties en représentation canonique, où chaque phonème du vocabulaire est représenté par un vecteur binaire composé de zéros (0), sauf la position réservée au phonème qui est mise à un (1).

Par exemple, pour le vocabulaire ["ḍ", "k", "l", "n", "ṭ", "ṣ", "ḥ", "s"] de taille égale à 8, la représentation canonique du phonème «ḥ» est [0, 0, 0 , 0, 0, 0, 1, 0].

La figure ci-dessous montre comment les données audio et leurs transcriptions phonémiques sont préparées et transformées en un format adapté à l'entraînement d'un modèle de reconnaissance vocale :

```
Dimensions de caractéristiques MFCC des échantillons audio, avec padding  : (2265, 62, 13)
Dimensions de Transcriptions phonémiques, avec padding : (2265, 62)
Dimensions de Transcriptions phonémiques, encodées en one-hot : (2265, 62, 32)
```

FIGURE 5.1 – Dimensions de données.

5.4.1 Répartition des données

Notre jeu de données est divisé en trois parties de tailles différentes :

— **Ensemble d'entraînement :** représente la plus grande partie des données. Cet ensemble sert à régler les paramètres du modèle par optimisation. Il contient 1812 fichiers de type wav, ainsi que leurs transcriptions initialement, mais après avoir réservé 20% pour la validation, il reste 1449 fichiers pour l'entraînement effectif.

— **Ensemble de test :** permet de mesurer les performances du modèle après l'entraînement. Il contient 453 fichiers de type wav, ainsi que leurs transcriptions.

— **Ensemble de validation :** est composé de données provenant de l'ensemble d'entraînement mais qui sont réservées spécifiquement pour l'évaluation pendant le processus d'entraînement. Il contient 363 fichiers de type wav, ainsi que leurs transcriptions.

```
Nombre d'échantillons dans l'ensemble d'entraînement (Avant la séparation pour la validation): 1812
Nombre d'échantillons dans l'ensemble d'entraînement (Après la séparation pour la validation): 1449
Nombre d'échantillons dans l'ensemble de validation: 363
Nombre d'échantillons dans l'ensemble de test: 453
```

FIGURE 5.2 – Ensemble d'entraînement, de validation et de test.

5.5 Architecture du modèle

Nous avons opté pour un modèle de reconnaissance vocale qui repose sur l'utilisation d'un réseau neuronal récurrent (RNN) bidirectionnel intégrant des mémoires à long terme et à court

terme (BLSTM). L'architecture du modèle comporte au total 10 couches différentes :

— **Couche de masquage (Masking) :** est utilisée pour prétraiter les séquences d'entrée après l'application du padding sur les MFCC et les transcriptions. Elle masque les valeurs spécifiées (les zéros) pour que le modèle ignore ces valeurs pendant l'entraînement.

— **Couches Bidirectional LSTM :** dans notre modèle, nous avons utilisé trois couches Bidirectional LSTM. Chaque couche est composée de deux sous-couches LSTM : l'une parcourt la séquence de gauche à droite et l'autre dans l'ordre inverse. Cela permet de mémoriser les informations sur de longues séquences de données dans les deux directions.

Couche	Sous-couche	Nombre de neurones
1ère couche LSTM bidirectionnelle (512 neurones)	1ère couche LSTM (direction avant)	256 neurones
	2ème couche LSTM (direction arrière)	256 neurones
2ème couche LSTM bidirectionnelle (512 neurones)	1ère couche LSTM (direction avant)	256 neurones
	2ème couche LSTM (direction arrière)	256 neurones
3ème couche LSTM bidirectionnelle (256 neurones)	1ère couche LSTM (direction avant)	128 neurones
	2ème couche LSTM (direction arrière)	128 neurones

TABLE 5.1 – Configuration des couches LSTM bidirectionnelles.

Les couches LSTM bidirectionnelles de notre modèle ont plusieurs paramètres importants. Voici ce que chaque paramètre fait :

— **return_sequences = True** : ce paramètre indique que chaque LSTM doit renvoyer une séquence complète de sorties à chaque étape temporelle, plutôt que seulement la sortie finale. Cela assure que chaque sous-couche LSTM suivante reçoit des informations détaillées pour chaque point de la séquence, préservant ainsi les informations temporelles sur toute la séquence.

— **recurrent_dropout = 0.3 :** ce taux de dropout récurrent de 30% est appliqué individuellement à chaque couche LSTM, permettant ainsi de désactiver aléatoirement environ 30% des connexions récurrentes. Lorsque les connexions récurrentes sont désactivées à une étape spécifique, cela signifie que les informations transmises par ces connexions à partir de l'état caché précédent h_{t-1} ne contribuent pas au calcul de l'état caché actuel h_t.

— **dropout = 0.3 :** appliqué aux entrées de chaque couche LSTM. Lorsque ce dropout est activé à 30%, environ un tiers des neurones dans la couche d'entrée sont désactivés aléatoirement à chaque itération d'entraînement.

— **Régularisation L2 :** ce paramètre est ajouté à chaque couche LSTM, ce qui signifie que chacune utilise la régularisation L2 pour ses poids limitant la capacité du modèle à s'ajuster trop précisément aux données d'entraînement. La régularisation L2 impose une pénalité proportionnelle à la somme des carrés des poids, forçant ainsi les poids à rester petits. Mathématiquement cela peut être formulé comme suit :

$$\text{L2} = \lambda \sum_i W_i^2 \tag{5.4}$$

Où :
— $\lambda = 0.01$: est le coefficient de régularisation L2.
— W_i : représente chaque poids individuel dans la couche LSTM.

— **Couches de normalisation :** pendant l'entraînement d'un réseau de neurones, les valeurs de sortie des neurones peuvent varier considérablement, ce qui rend l'entraînement plus difficile. La couche de normalisation intervient après chaque couche LSTM bidirectionnelle pour régulariser ces valeurs.

— **Couche TimeDistributed :** est utilisée dans les modèles séquentiels pour appliquer une même couche à chaque élément temporel d'une séquence de manière indépendante.

— **Couche Dense :** est une couche entièrement connectée qui comporte 128 neurones, chacun étant connecté à tous les neurones de la couche précédente. Chaque neurone utilise la fonction d'activation ReLU, qui transforme les valeurs négatives en zéro et laisse passer les valeurs positives.

— **Couche de Dropout :** appliquée après une couche Dense avec un taux de 0.4. À chaque passage d'une séquence de données à travers la couche dense, 40% des neurones sont

temporairement mis à zéro. Cela signifie qu'ils ne contribuent ni au calcul des valeurs de sortie, ni à la mise à jour des poids du réseau pendant cette itération spécifique.

— La dernière couche du modèle utilise une couche Dense à l'intérieur de TimeDistributed. Cette couche transforme les sorties des couches LSTM précédentes en vecteurs de taille égale au nombre de classes possibles. Chaque élément de ces vecteurs de sortie correspond à la sortie d'un neurone représentant une classe potentielle. En appliquant l'activation softmax à la sortie de la couche Dense, ces valeurs sont converties en probabilités normalisées. Chaque valeur dans le vecteur de sortie indique la probabilité que l'entrée observée à ce pas de temps appartienne à chaque classe possible.

Remarque 1. *Les paramètres utilisés dans chaque couche du modèle ont été sélectionnés après de nombreux essais, afin d'obtenir les meilleurs résultats possibles.*

5.5.1 Paramètres d'apprentissage

— **Optimiseur d'erreur :** Adaptive Moment Estimation (Adam)
— **Fonction de perte :** categorical_crossentropy.
— **Nombre d'itérations :** 100.
— **Taux d'apprentissage :** 0.0001.

5.5.2 Sortie du modèle

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 masking (Masking)           (None, 62, 13)            0

 bidirectional (Bidirectiona  (None, 62, 512)          552960
 l)

 layer_normalization (LayerN  (None, 62, 512)          1024
 ormalization)

 bidirectional_1 (Bidirectio  (None, 62, 512)          1574912
 nal)

 layer_normalization_1 (Laye  (None, 62, 512)          1024
 rNormalization)

 bidirectional_2 (Bidirectio  (None, 62, 256)          656384
 nal)

 layer_normalization_2 (Laye  (None, 62, 256)          512
 rNormalization)

 time_distributed (TimeDistr  (None, 62, 128)          32896
 ibuted)

 dropout (Dropout)           (None, 62, 128)           0

 time_distributed_1 (TimeDis  (None, 62, 41)           5289
 tributed)

=================================================================
Total params: 2,825,001
Trainable params: 2,825,001
Non-trainable params: 0
_________________________________________________________________
```

FIGURE 5.3 – Sortie du modèle.

5.6 Analyse des résultats

Le jeu de données comprend 2265 enregistrements et leurs transcriptions. Chaque itération d'entraînement dure en moyenne 252 secondes, et l'entraînement complet du modèle a pris environ 7 heures.

5.6.1 Courbes d'entraînement et de validation

Pour évaluer la performance de notre modèle sur 100 itérations d'entraînement, les deux graphiques ci-dessous présentent respectivement l'évolution de la perte et de la précision pour les ensembles d'entraînement et de validation.

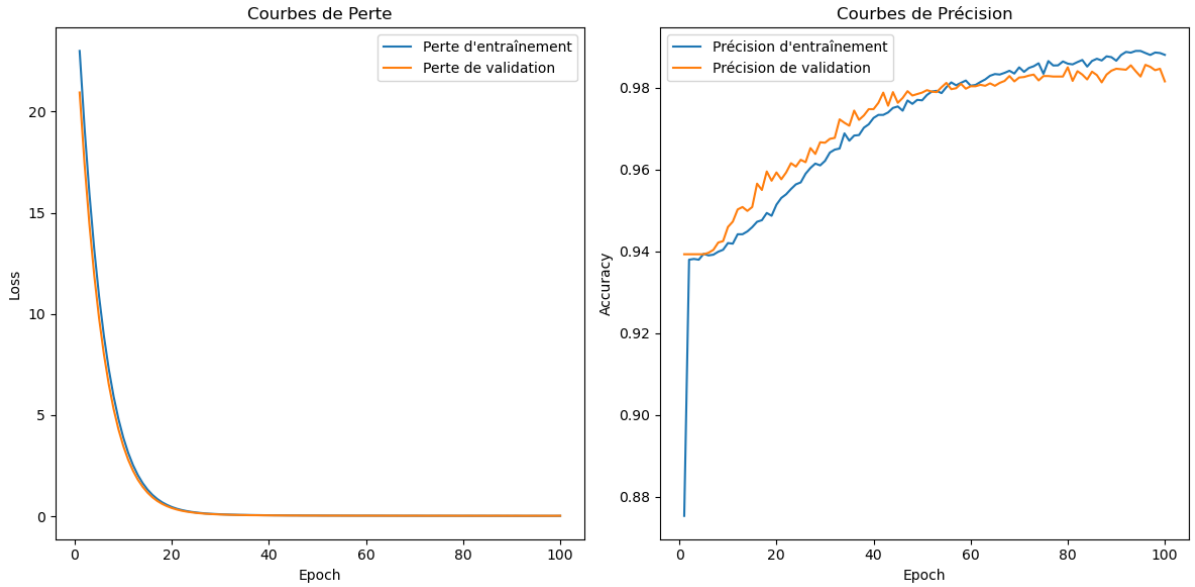


FIGURE 5.4 – Courbes d'entraînement et de validation.

— À gauche, la courbe de perte montre la diminution progressive de l'erreur (Loss) à la fois pour l'ensemble d'entraînement (courbe bleue) et l'ensemble de validation (courbe orange), indiquant que le modèle apprend efficacement à partir des données fournies.

— À droite, la courbe de précision reflète l'amélioration de la proportion de prédictions correctes, démontrant une augmentation continue de la performance du modèle pour les ensembles d'entraînement et de validation.

D'après ces courbes, on remarque qu'il n'y a pas de surapprentissage, car les courbes d'entraînement et de validation restent proches l'une de l'autre, indiquant que le modèle généralise bien aux nouvelles données. La perte finale atteint une valeur de 0.0295 et la précision de 0.9831.

5.6.2 Matrice de confusion

Cette matrice nous permet d'identifier plus précisément les classes pour lesquelles le modèle fonctionne bien et celles où il peut rencontrer des difficultés, en comparant les prédictions correctes et incorrectes pour chaque classe.

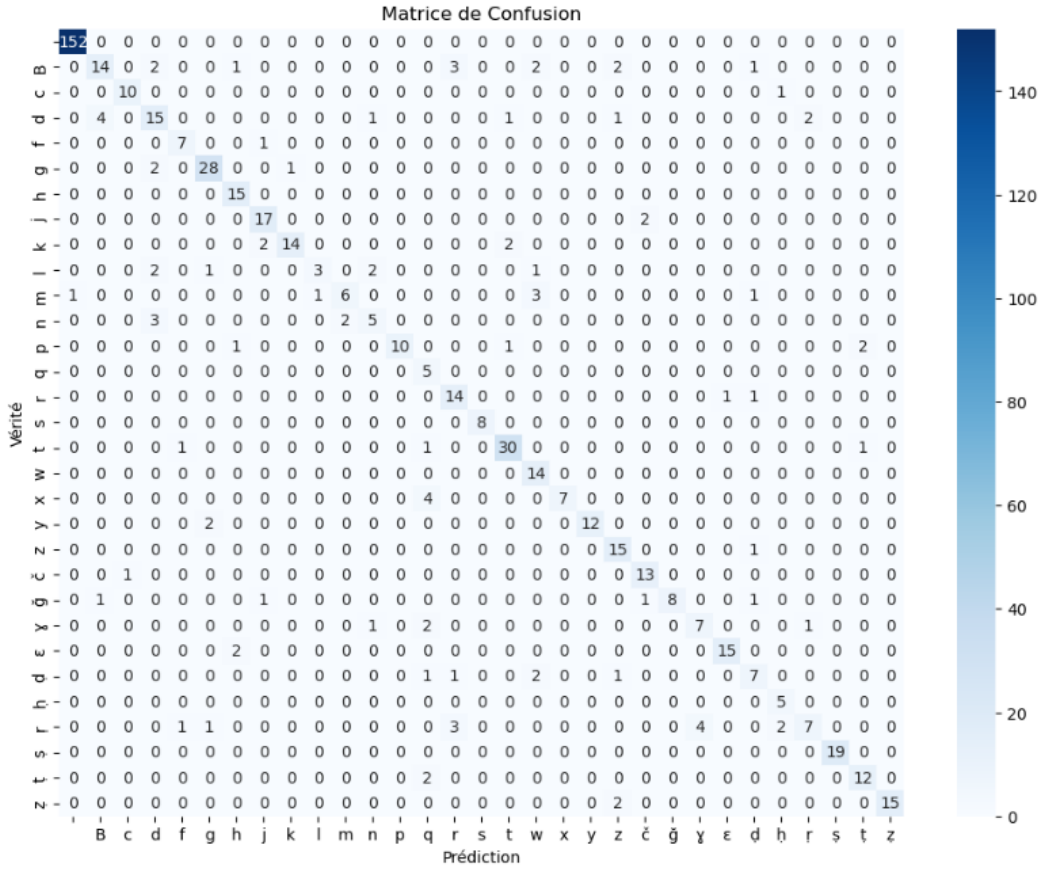


FIGURE 5.5 – Matrice de confusion.

La diagonale de cette matrice représente le nombre d'exemplaires correctement reconnus pour chaque classe lors de la prédiction, tandis que les autres valeurs indiquent le nombre d'exemplaires non reconnus pour chaque classe.

5.6.3 Courbe ROC

La courbe ROC (Receiver Operating Characteristic) est utilisé pour évaluer la performance d'un modèle de classification binaire. Elle trace le taux de vrais positifs contre le taux de faux positifs à différents seuils de classification.

— **Axe des Y (Ordonnée) :** représente le taux de vrais positifs (TVP), aussi appelé sensibilité ou rappel. Il est calculé comme suit :

$$TVP = \frac{VP}{VP + FN} \tag{5.5}$$

— **Axe des X (Abscisse) :** représente le taux de faux positifs (TFV), calculé comme suit :

$$TFP = \frac{FP}{FP + VN} \tag{5.6}$$

Où :

— **AUC (Area Under the Curve) :** la surface sous la courbe ROC est une mesure de la capacité du modèle à distinguer entre les classes positives et négatives. Une AUC de 1 indique une classification parfaite, tandis qu'une AUC de 0.5 indique une performance similaire à celle d'un modèle aléatoire.

Cette figure ci-dessous illustre les résultats obtenus pour la courbe ROC.

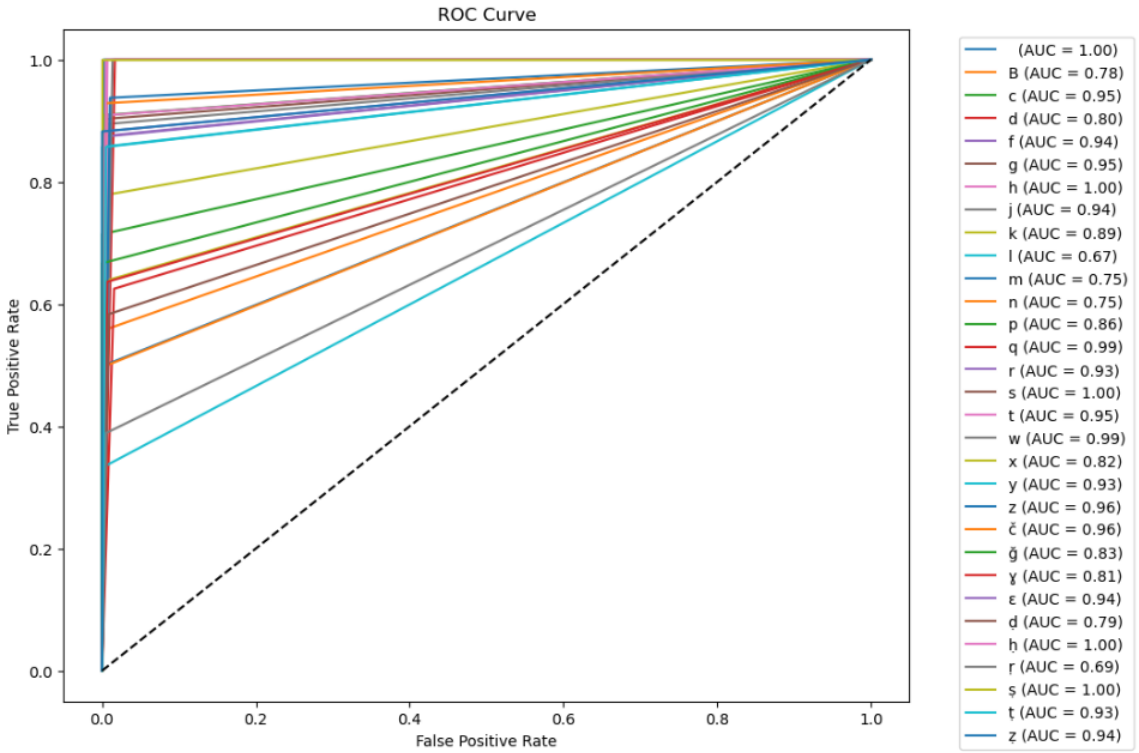


FIGURE 5.6 – Courbe ROC.

Les résultats mettent en évidence plusieurs modèles ayant obtenu des performances remarquables, avec des valeurs d'AUC supérieures ou égales à 0.95. Cela suggère que ces modèles sont très efficaces pour discriminer entre les classes positives et négatives.

Toutefois, pour certains autres modèles, il est nécessaire d'apporter des améliorations significatives afin d'atteindre des niveaux de performance comparables. Ces modèles pourraient présenter des AUC plus basses, indiquant une capacité moindre à distinguer correctement les exemples positifs et négatifs.

5.6.4 Rapport de classification

Le figure ci-dessous présente les résultats du rapport de classification, fournissant une analyse de la précision, du rappel, du F1-score et du support pour chaque classe de caractères.

	precision	recall	f1-score	support
	0.99	1.00	1.00	152
B	0.74	0.56	0.64	25
c	0.91	0.91	0.91	11
d	0.62	0.62	0.62	24
f	0.78	0.88	0.82	8
g	0.88	0.90	0.89	31
h	0.79	1.00	0.88	15
j	0.81	0.89	0.85	19
k	0.93	0.78	0.85	18
l	0.75	0.33	0.46	9
m	0.75	0.50	0.60	12
n	0.56	0.50	0.53	10
p	1.00	0.71	0.83	14
q	0.33	1.00	0.50	5
r	0.67	0.88	0.76	16
s	1.00	1.00	1.00	8
t	0.88	0.91	0.90	33
w	0.64	1.00	0.78	14
x	1.00	0.64	0.78	11
y	1.00	0.86	0.92	14
z	0.71	0.94	0.81	16
č	0.81	0.93	0.87	14
ǧ	1.00	0.67	0.80	12
ɣ	0.64	0.64	0.64	11
ɛ	0.94	0.88	0.91	17
ḍ	0.58	0.58	0.58	12
ḥ	0.62	1.00	0.77	5
ṛ	0.70	0.39	0.50	18
ṣ	1.00	1.00	1.00	19
ṭ	0.80	0.86	0.83	14
ẓ	1.00	0.88	0.94	17
accuracy			0.84	604
macro avg	0.80	0.79	0.78	604
weighted avg	0.86	0.84	0.84	604

FIGURE 5.7 – Rapport de classification.

5.6.5 Taux de reconnaissance globale

La figure présentée ci-dessous offre une vue détaillée du taux de reconnaissance globale et du taux d'erreur obtenus à partir de l'analyse du modèle.

Le taux de reconnaissance globale est 86,07 % indique la proportion des caractères correctement identifiés par rapport à l'ensemble des caractères testés. En parallèle, le taux d'erreur est de 13,93 % représente la proportion de caractères mal identifiés ou incorrectement reconnus par le modèle.

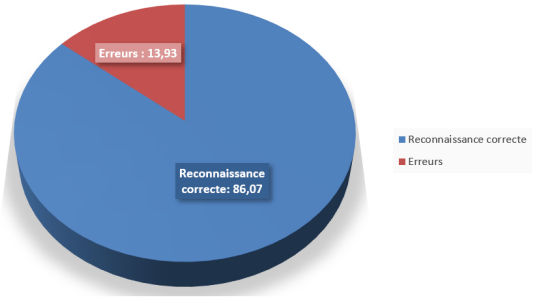


FIGURE 5.8 – Taux de reconnaissance globale.

Pour améliorer la performance et réduire le taux d'erreur du modèle, des ajustements sont essentiels afin d'optimiser sa précision et sa fiabilité. Cela nous permettra d'atteindre une reconnaissance plus précise des caractères de la langue kabyle.

5.6.6 Prédiction sur les données test

Après avoir réalisé une analyse approfondie du modèle en utilisant des visualisations telles que les matrices de confusion, la courbes de ROC, etc., nous avons effectué une prédiction sur les données de test. La figure ci-dessous montre les résultats obtenus lors de cette prédiction :

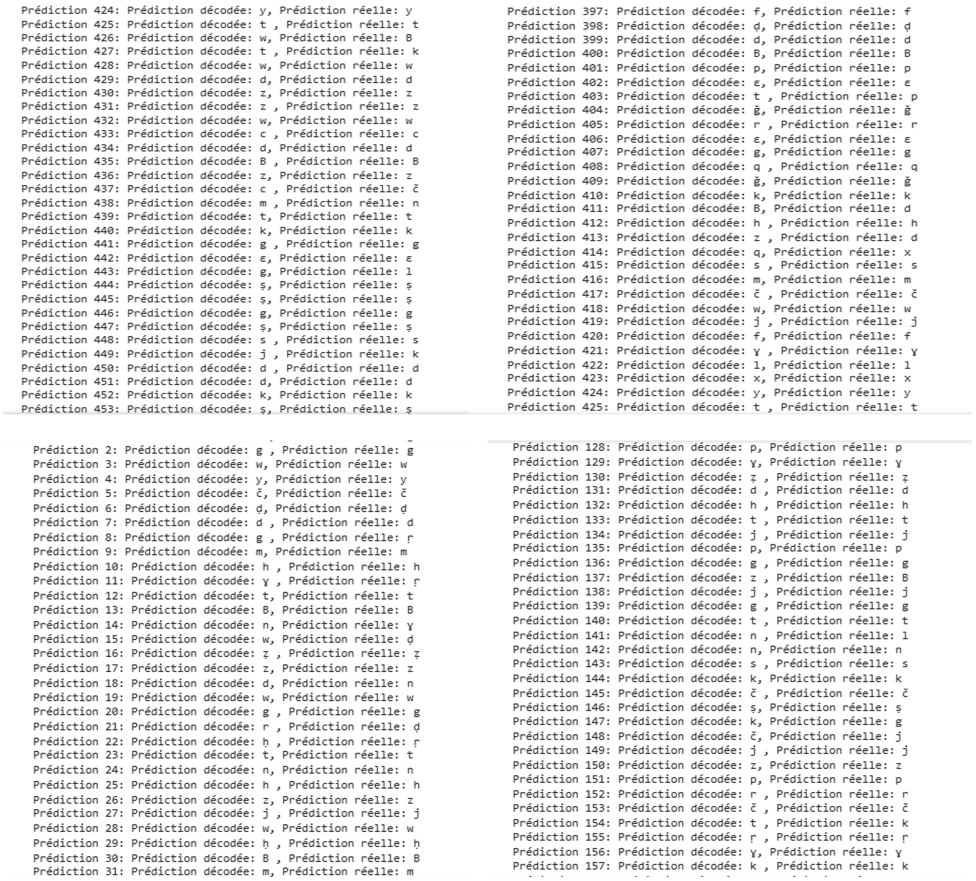


FIGURE 5.9 – Prédiction sur les données test.

Lors de la prédiction sur les données de test, nous avons constaté quelques erreurs, principalement en raison de la similitude de prononciation entre certains phonèmes lorsque le locuteur les a lus. De plus, le fait que notre dataset soit limité peut également contribuer à ces erreurs.

5.6.7 Erreurs de prédiction

Le graphique ci-dessous montre les erreurs de prédiction pour les phonèmes identifiés par notre modèle. Chaque barre représente le nombre d'occurrences où la prédiction ne correspond pas au phonème correct :

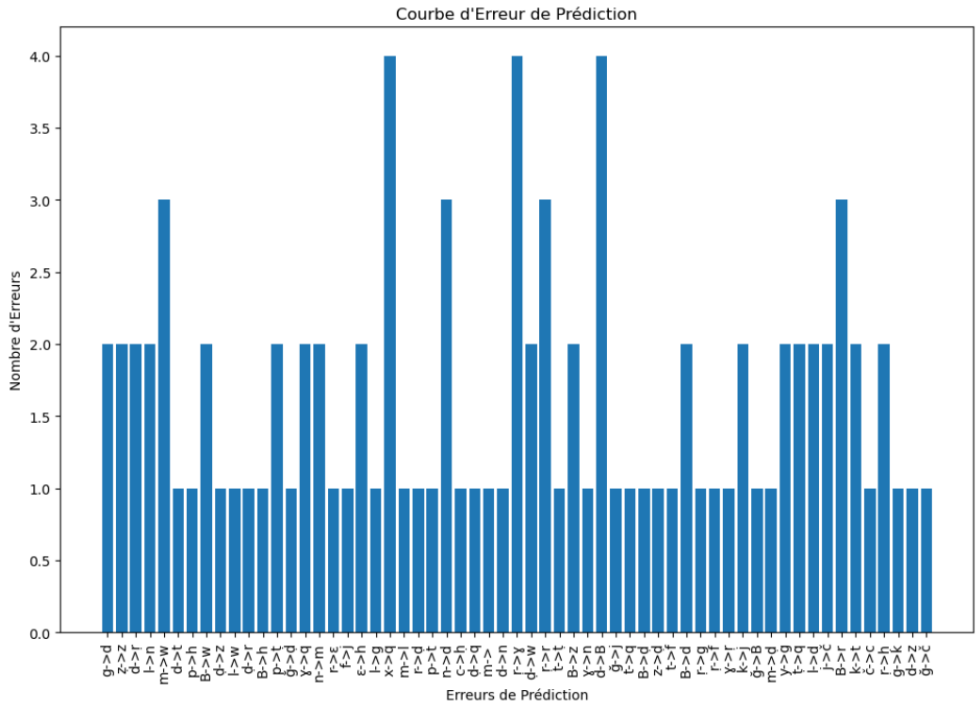


FIGURE 5.10 – Courbe d'erreur de prédiction.

Pour comprendre les raisons de ces erreurs, nous nous concentrons sur un cas spécifique où le phonème ɽ a été prédit 4 fois comme γ. Dans cette analyse, nous examinons deux enregistrements audio des deux phonèmes prononcés par la même personne :

Visualisation des MFCC

Les graphiques ci-dessous montrent les coefficients MFCC pour les deux fichiers audio. Les plages de temps sont similaires, indiquant une durée équivalente, une analyse visuelle montre des similitudes dans les motifs et les intensités de couleur, suggérant des caractéristiques spectrales communes. Cependant, une évaluation plus approfondie des MFCC est nécessaire pour préciser les similitudes entre les fichiers audio :

— **Distance euclidienne :** mesure la différence entre les vecteurs moyens des MFCC des deux fichiers audio. Nous avons obtenu une valeur de 43.54, bien qu'elle ne soit pas

très faible, reste relativement petite. Cela indique des caractéristiques sonores similaires malgré des différences potentielles dans l'amplitude des coefficients.

— **Similarité cosinus :** évalue la ressemblance entre deux vecteurs en calculant le cosinus de l'angle entre eux. Une valeur proche de 1 indique une forte similitude, les vecteurs étant alignés dans la même direction. Une valeur proche de 0 signifie qu'ils sont perpendiculaires et donc très différents. Une valeur de 0.99 entre les vecteurs moyens des coefficients MFCC suggère un quasi-alignement dans l'espace multidimensionnel. Cela signifie que les fichiers contiennent des sons ou des paroles très similaires.

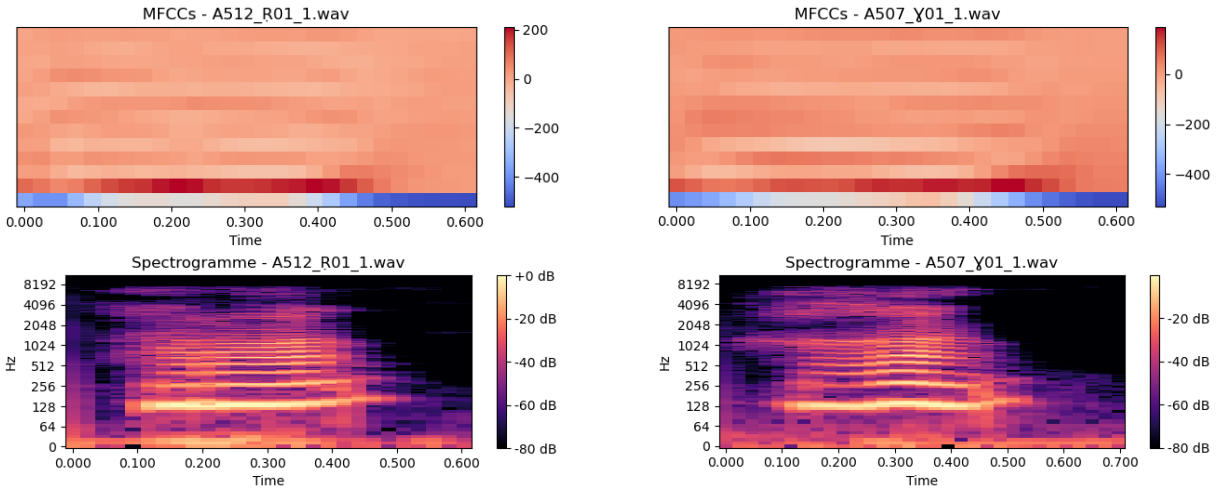


FIGURE 5.11 – Visualisation des MFCCs et des spectrogrammes pour les fichiers audio.

```
MFCCs moyens (A512_ʀ01_1.wav) : [-249.27396    107.961876     6.341356    13.579588    -4.2087126
   12.872485     26.037567    -5.170373    10.232536     7.739972
    1.2221445     9.445725    -9.401279 ]
MFCCs moyens (A507_ɣ01_1.wav) : [-2.7575388e+02  8.6753387e+01  1.8943680e+01  1.0594788e+01
 -3.0846572e+00  3.3147118e+01  2.4373251e+01 -3.3429971e+00
  1.5979853e-01  9.2676439e+00 -3.3341591e+00  1.5177796e+01
 -8.4128971e+00]
Distance euclidienne entre les MFCCs : 43.53773498535156
Similarité cosinus entre les MFCCs : 0.9905167
```

FIGURE 5.12 – Evaluation de la similtitude.

Comparaison des MFCC Triangles

Chaque graphique parmi les graphiques ci-dessous montre la variation d'un coefficient spécifique au fil du temps pour deux phonèmes, avec les lignes bleues représentant le phonème ʀ̣ et les lignes rouges représentant le phonème γ.

On remarque que dans plusieurs graphiques les lignes bleues et rouges se chevauchent ou sont très proches les unes des autres, cela indique une similarité entre les prononciations des deux phonèmes, car indépendamment ils trouvent des coefficients similaires.

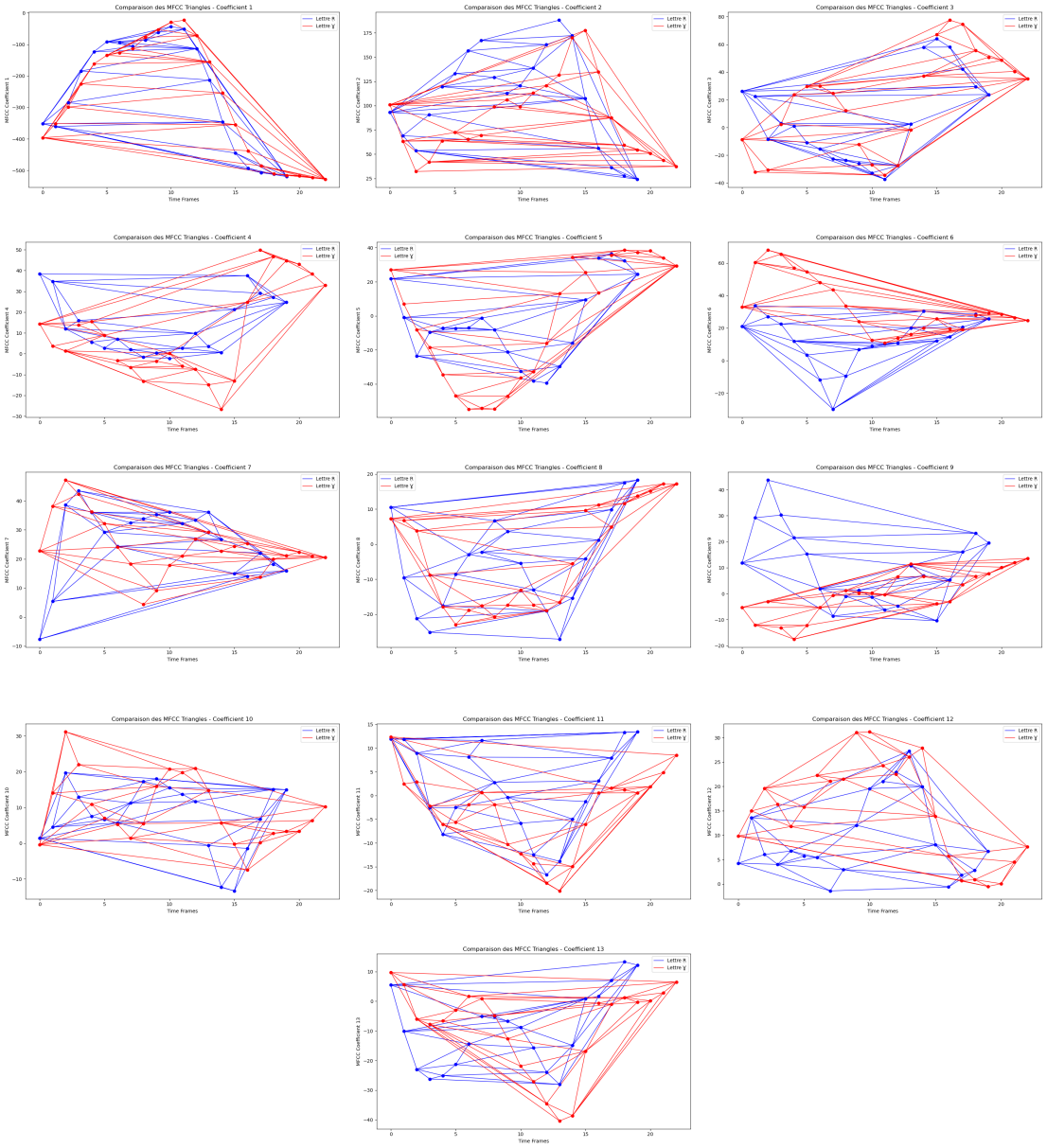


FIGURE 5.13 – Comparaison des MFCC Triangles.

5.6.8 Évaluation du modèle sur de nouveaux enregistrements

Pour évaluer plus précisément notre modèle, nous allons le tester sur de nouveaux enregistrements. Cette méthode nous permettra d'identifier les phonèmes nécessitant des améliorations.

La figure ci-dessous montre les résultats obtenus :

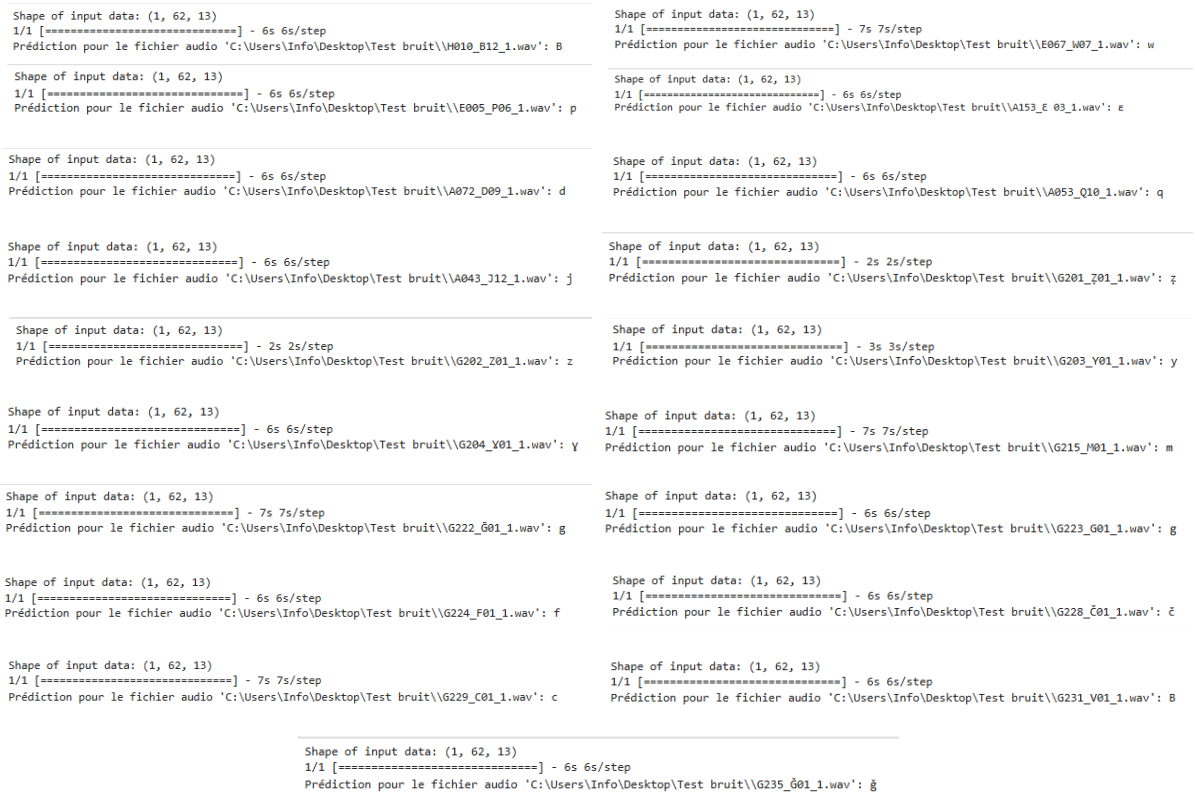


FIGURE 5.14 – Résultats de la prédiction.

Bien que le modèle soit capable de prédire certains phonèmes. Mais il rencontre des difficultés avec d'autres en raison de leurs similitudes phonétiques, ce qui conduit à des erreurs de prédiction.

Conclusion

Ce chapitre a exploré le processus de développement de notre modèle, couvrant l'environnement matériel et logiciel, la préparation des données, l'architecture du modèle, et les résultats obtenus.

Nous avons présenté une approche de reconnaissance vocale des lettres spécifiques de la langue kabyle, basée sur les réseaux de neurones récurrents (RNN). Cette méthode a donné des résultats montrant une précision notable et un taux d'erreur faibles.

Conclusion générale et perspectives

Ces dernières années, grâce aux progrès en traitement du signal, en algorithmes, en architectures et en matériel, la recherche en reconnaissance automatique de la parole a connu une forte avancée. Aujourd'hui, les systèmes de reconnaissance vocale sont utilisés dans de nombreuses applications, comme les logiciels de dictée, les services de transcription automatique, et les traducteurs vocaux.

Ce mémoire a exploré comment concevoir et développer un système de reconnaissance vocale pour identifier précisément les lettres de la langue kabyle. Cet objectif a été atteint grâce à la mise en place d'un système qui a nécessité la collecte préalable de 2641 enregistrements par divers locuteurs de différentes tranches d'âge, genres et régions. Ces données ont été soigneusement traitées pour extraire un total de 9986 enregistrements, répartis en phonèmes, mots et phrases. Chaque enregistrement a été analysé afin d'identifier et de capturer ces caractéristiques acoustiques.

Dans le cadre de notre étude sur la reconnaissance vocale des lettres de la langue kabyle, ces enregistrements ont été filtré pour extraire uniquement les enregistrements des phonèmes avec leurs annotations. Ainsi, nous avons formulé un nouveau dataset comprenant 2256 enregistrements et les transcriptions associées.

Nous avons appliqué notre modèle utilisant des réseaux de neurones récurrents, notamment un LSTM bidirectionnel, sur ces données. Notre choix s'est porté sur ce modèle en raison de sa capacité à capturer efficacement les dépendances temporelles dans les séquences audio, ainsi que sa flexibilité pour gérer des données complexes et variées.

Les résultats obtenus après l'application de notre modèle ont été très encourageants, avec une précision mesurée à 98,31 % c'est le meilleur résultat que nous pouvons obtenir. Ce qui démontre l'efficacité et la performance du modèle proposé.

En conclusion, ce projet nous a permis d'acquérir de nouvelles connaissances. Nous avons pu découvrir au cours de ce travail de nouvelles notions telles que la reconnaissance automatique de la parole, le traitement d'un signal vocal, les réseaux de neurone, le RNN, LSTM bidirectionnel. Ce travail nous a permis de mettre en pratique nos connaissances sur les réseaux de neurones et d'acquérir des connaissances supplémentaires.

Cependant, plusieurs perspectives sont envisagées pour enrichir et étendre cette recherche :

1. Envisager d'effectuer d'autres collectes de données pour enrichir notre corpus vocal.

2. Avant d'intégrer les données réellement collectées et traitées, utiliser un corpus vocal de référence pour entraîner le modèle de reconnaissance sélectionné.
3. Développer un système de reconnaissance vocale adapté aux mots et aux phrases.
4. Concevoir un système de transcription complet dédié à la langue kabyle.

Bibliographie

[1] La peur des maquis. *Courrier International, Sainte-Geneviève (France)*, 549 (Mai 2001), 46.

[2] Tout s'explique : La synthèse vocale. *aireslibres* (Décembre 2013).

[3] Reconnaissance vocale par pipeline comment traiter et analyser des donnees vocales et audio a l aide de votre pipeline. *FasterCapital* (february 2024).

[4] A.Benchenief. Reconnaissance voacle basée sur les svm. Mémoire de master, Université Mohamed Khider, Biskra, Décembre 2011.

[5] A.Guilbaut, and A.Belloti. Chaînes de markov cachées. Mémoire de master, Université de Lille, 2018.

[6] A.Kocabiyikoğlu. *Constitution d'un corpus de traduction de la parole : augmentation du corpus LibriSpeech*. Doctoral thesis, Université Grenoble Alpes, Grenoble, France, 2017.

[7] A.Mikami. *Architectures de réseaux neuronaux récurrents à mémoire à court et long terme pour la génération de musique et de paroles japonaises*. Doctoral thesis, Boston College, 2016.

[8] B.Elbarkani. *Le choix de la graphie tifinaghe pour pour enseigner, apprendre L'amazighe au Maroc : conditions, Représentation et Pratiques*. Doctoral thesis, Ecole doctorale (484) Lettres, Langues, Linguistique et Arts, 35,rue du Onze-Novembre 42023 Saint Etienne Cedex 2, Décembre 2010.

[9] B.Mohammed-Sadok. Transformée hilbert huang application a la détection des défauts du moteur asynchrone. Memoire de master, Université Mohamed Khider Biskra, Juin 2018.

[10] Centre de recherche en langue et culture amazighes (CRLCA-Algérie). https://crlca.dz/fr/. Consulté le 24 juin 2024.

[11] Centre for Speech Technology Research, University of Edinburgh. The festival speech synthesis system. https://www.cstr.ed.ac.uk/projects/festival. Consulté le 10 mai 2024.

[12] Ch.Basumallick. What is tensorflow ? meaning, working, and importance. *Technical writer spiceworks* (December 2022).

[13] C.Robert-Granié, and B.Servin. *Modèle Linéaire mixte gaussien*. Doctoral thesis, Institut national de la recherche agronomique, Mars 2012.

[14] D.Jurafsky, James, and H.Martin. *Speech and Language Processing*. 2. 2024.

[15] D.Medved. *Deep learning applications for biomedical data and natural language processing*. Doctoral thesis (compilation), Department of computer science lund university, Box 118 SE-221 00 Lund Sweden, 2018.

[16] F.Adjed. Vers une normalisation du kabyle : Alphabet. halshs-03171259v2.

[17] F.Oukina, and I.Amar. Elaboration d'un corpus de test pour un systéme d'évaluation automatique des réponses courtes. Mémoire de master, Universite Saad Dehlb Blida 1, 2019.

[18] G.Claude. Échantillonnage : tout ce que vous devez savoir pour vos recherches.

[19] H-I.Rhys, and J.Nowosad. *Machine learning with R, the tidyverse, and mlr*. 2. 2020.

[20] Haut commissariat à l'amazighité (HCA-Algérie). https://www.hcamazighite.dz/fr/. Consulté le 24 juin 2024.

[21] H.Battane, and Z.Benaissa. Détection et diagnostic de défauts d'un onduleur par la technique des réseaux de neurones. Mémoire de master, Université Ibn-Khaldoun de Tiaret, 2018.

[22] H.Taherdoost. *Data Collection Methods and Tools for Research, a step-by-step Guide to choose data collection technique for academic and business research projects*, vol. 10 of *1*. International journal of academic research in management (IJARM) hal-03741847f, Augest 2021.

[23] I-H.Sarker. Machine learning : Algorithms, real-world applications and research directions. *SN COMPUT SCI 2* (March 2021), 160.

[24] I.Bahri, and M.Laib. Combinaison svm et apprentissage profond pour la reconnaissance de carecteres arabe. Mémoire de master, Université Ahmed Draïa d'Adrar, 2021.

[25] I.Goodfellow, Y.Bengio, and A.Courville. *Deep Learning*. 2. MIT Press, 2016.

[26] I.Khalla, and B.Sedairia. Apport de la lecture dans l'apprentissage de la phonétique cas des élèves de la 5 éme année primaire. Mémoire de master, Université 8 mai 1945, Guelma, 2019.

[27] I.Zara. L'intelligence artificielle principe, outils et objectifs. Mémoire de master, Universite Badji Mokhtar, Annaba, 2019.

[28] J-P.Delahaye. Intelligence artiicielle et le test de turing.

[29] J.Benesty, J.Chen, Y.Huang, and I.Cohen. *Noise reduction in speech processing*, vol. 2 of *2*. Springer, Augest 2009.

[30] J.Brownlee. *Deep Learning Models for Natural Language in Python*, vol. 1.1. 2017.

[31] J.Han, M.Kamber, and J.Pie. *Data Mining Concepts and Techniques*, 3 ed. Morgan Kaufmann, 225 Wyman Street, Waltham, MA 02451, USA, 2011.

[32] J.Hu. Explainable deep learning for natural language processing. degree project in information and communication technology, Kth royal institute of technologie school of electrical engineering and computer science, Stockholm, Sweden, October 2018.

[33] J.Mariani. Reconnaissance automatique de la parole : progrès et tendances advances and trends in automatic speech recognition. *LIMSI-CNRS 7*, 5, 239–266.

[34] K.Amar, and N.Hammou. Reconnaissance vocale du genre basée sur l'apprentissage profond. Mémoire de master, Universite Abdelhammid Ibn Badis Mostaganem, 2022.

[35] L.Amiar. Un systeme hybride ag/pmc pour la reconnaissance de la parole arabe. Mémoire de magister, Universite Badji Mokhtar Annaba, 2005.

[36] L.Guechtal, R.Ouelbani, and Y.Talbi. Une approche basée sur le traitement automatique du langage naturel (taln) pour la classification taxonomique des séquences métagénomiques 16s rrna. Mémoire de master, Université fréres mentouri Constantine 1, 2023.

[37] L'institut royal de la culture amazighe (IRCAM-Maroc). https://www.ircam.ma/fr. Consulté le 24 juin 2024.

[38] L.Melis, and P.Desmet. *Modèles linguistiques*. No. 79-145. 2000.

[39] M.Amar. intelligence artificielle. Cours.

[40] M.Boutouiga, and N.Hor. Comportement d'isolateur capot et tige f160d-146dc artificiellement pollué sous tension alternative 50hz. Mémoire de master, Université Ibn Khaldoun Tiaret, 2019.

[41] M.Djaballah. Système de prédiction de la consommation d'énergie basé deep learning. Mémoire de master, Université de 8 Mai 1945 Guelma, 2021.

[42] M.Khene, and S.Souid. Détection d'activité vocale utilisant l'apprentissage profond. Mémoire de master, Université de Ghardaïa, 2019.

[43] M.Mohssen, M-B.Khan, and E-M.Bashier. *Machine learning algorithms and applications*. No. 13. Taylor et Francis Group, Boca Raton, July 2016.

[44] M.Mokri. Classification des images avec les réseaux de neurones convolutionnels. Mémoire de master, Université Abou Bakr Belkaid Tlemcen, 2017.

[45] N.ZERARI. *Intégration d'un module de reconnaissance de la parole au niveau d'un systéme audiovisuel-application téléviseur*. Doctoral thesis, Université Batna 2 Mostefa Ben Boulaïd, AVRIL 2021.

[46] O.Ould-Braham, and L.Souag. Pour une histoire de la langue berbère dans sa diversité et sa complexité. *Études et documents berbères*, 45-46 (Decembre 2022), 5–35.

[47] P.Hairy. Les réseaux de neurones récurrents pour les séries temporelles.

[48] R.Belbachir, and M.Azoug. L'impact de la langue maternelle sur l'enseignement-apprentissage du fle, cas des élèves de première année du collège 8 mai 1945, kherrata. Mémoire de master, Université abderrahmane mira, Bejaïa, 2020.

[49] R.Bosri, and I.Harbouche. Développement d'un modèle prédictif de durée de vie des polymères. Mémoire de master, Université Ibn Khaldoun Tiaret, 2019.

[50] R.Keim. The nyquist–shannon theorem : understanding sampled systems. *TECHNICAL ARTICLE* (May 2020).

[51] R.Ramdani. Commande vocale d'une plateforme mobile. Mémoire de master, Université Saad Dahlab de Blida, 2016.

[52] R.Sheldon, and J.Burke. signal-to-noise ratio (s/n or snr). *Nemertes Research* (August 2021).

[53] S-C.İleri, A.Karabina, and E.Kiliç. Comparison of different normalization techniques on speakers, gender detection.

[54] S-M-S.Kabir. *Basic guidelines for research : an introductory approach for all disciplines*. No. 201-275. Book Zone Publication,Chittagong-4203, Bangladesh, July 2016.

[55] S.Aouragh. *Les chaînes de Markov cachées*. Doctoral thesis, Université de Biskra, 2006.

[56] S.Arrami. Dictionnaire amazigh carte des parlers amazighs (berbères). https://amazigh24.com/qui-sont-les-amazighs. Consulté le 25 Avril 2024.

[57] S.Chaker. Le berbere de kabylie (algérie). *Inalco-Centre de Recherche Berbère, Encyclopédie berbère XXVI* (2004), 4055–4066.

[58] S.Chandar, and A.Parthipan. *On challenges in training recurrent neural networks*. Doctoral thesis, Université de Montréal, November 2019.

[59] S.Djemil. Evaluation des techniques de reconnaissance des sons des oiseaux. Mémoire de master, Université Badji Mokhtar Annaba, 2019.

[60] S.Harnard. Can a machine be conscious ? how ? *Journal of Consciousness Studies 67-75*, 10 (January 2003), 4–5.

[61] S.Hassani. Variation et mutation phonétiques en kabyle.

[62] S.Mcleod. Z-score : definition,calculation and interpretation. *SimplyPsychology* (May 2019).

[63] S.Salmi, and T.Chahboub. Vers un système d'identifiation automatique des dialectes algériens à partir des vidéos yuutube. Mémoire de master, Universite Saad Dahlb, Blida, 2021.

[64] Y.Amirat. Extraction d'entités nommées par apprentissage profond. Mémoire de master, Université du Québec à Montreal, Juin 2020.

[65] Y.Montenay. Que-connaissez-vous-de-la-culture-kabyle-de-la-langue-kabyle. https://fr.quora.com. Consulté le 25 Avril 2024.

[66] Z.Benblal, and F.Belouafi. Intégration d'un lemmatiseur arabe dans le cadre d'un système de recherche d'information. Mémoire de fin d'étude, Université ahmed draia-adrar département des mathématiques et informatique, 2014-2015.

[67] Z.Bennanni, and K.Khour. Classification des événements audio environnementaux à l'aide de réseaux de neurones convultifs. Mémoire de master, Université Saad Dahlb Blida, 2019.

[68] Z.Bobbit. Z-score normalization definition et examples. *Statology* (August 2021).

[69] Z.Gaci. Quel système d'écriture pour la langue berbère(le kabyle). Mémoire de fin d'étude, Université Mouloud Mammeri De Tizi Ouzou, 2011.

Résumé

Résumé

Ce travail vise à développer un système de reconnaissance vocale des lettres en langue kabyle, en suivant plusieurs étapes essentielles. Tout d'abord, des données ont été collectées sur le terrain, puis prétraitées en deux phases : avec Audacity pour réduire le bruit et segmenter les signaux audio, et avec Python pour rééchantillonner les signaux, les normaliser et réduire davantage le bruit. Les caractéristiques essentielles, telles que les coefficients cepstraux en fréquences mel (MFCC), ont été extraites pour capturer les aspects distinctifs des signaux vocaux. Ces caractéristiques ont été utilisées comme entrées pour entraîner des modèles de réseaux neuronaux récurrents (RNN), en particulier le modèle BLSTM (Bidirectional Long Short-Term Memory), qui a été choisi pour sa capacité à capturer les dépendances temporelles dans les données séquentielles. Le modèle BLSTM a atteint une précision de 98,31% pour la transcription des enregistrements vocaux, démontrant une adaptabilité efficace à différents locuteurs et conditions d'enregistrement.

Mots-clés : Reconnaissance vocale, lettres, langue kabyle, enregistrements vocaux, collecte de données, prétraitement des données, MFCC, réseau de neurones, RNN, BLSTM.

Abstract

This work aims to develop a speech recognition system for letters in the Kabyle language, following several essential steps. Firstly, data was collected in the field and then preprocessed in two phases : with Audacity to reduce noise and segment the audio signals, and with Python to resample the signals, normalize them, and further reduce noise. Key features, such as Mel-Frequency Cepstral Coefficients (MFCC), were extracted to capture the distinctive aspects of the vocal signals. These features were used as inputs to train recurrent neural network models, particularly the BLSTM (Bidirectional Long Short-Term Memory) model, which was chosen for its ability to capture temporal dependencies in sequential data. The BLSTM model achieved an accuracy of 98.31% for transcribing the vocal recordings, demonstrating effective adaptability to different speakers and recording conditions.

Keywords : Speech recognition, letters, Kabyle language, voice recordings, data collection, data preprocessing, MFCC, neural network, RNN, BLSTM.