Université Saad DAHLAB - Blida 1

Faculté des sciences

Département d'Informatique

Mémoire présenté par :

CHAHBOUB Thissas

SALMI Samir

Pour l'obtention du diplôme de Master

Domaine : Mathématique et Informatique

Filière : Informatique

Spécialité : Traitement Automatique de la Langue

Sujet :

Vers un Système d'Identification Automatique des Dialectes Algériens à partir des Vidéos YouTube

Soutenu le : 04/10/2021	**Devant les membres du jury composé de :**	
M. S. Ferfera	**Université de Blida 1**	**Président**
M. Riali	**Université de Blida 1**	**Examinateur**
Mme M. MEZZI	**Université de Blida 1**	**Promotrice**
M. ABBAS	**CRSTDLA**	**Encadreur**
M. M. LICHOURI	**CRSTDLA**	**Co-encadreur**

Résumé

Les dialectes sont un sujet qui intéresse plusieurs disciplines comme la linguistique, la phonétique, et même l'informatique, pour leurs caractéristiques qui ne se conforment pas toujours aux règles linguistiques. L'Algérie offre un cas d'étude assez intéressant avec sa richesse linguistique.

Le but de cette étude était de proposer une approche phonétique pour l'identification automatique des dialectes, basée sur les caractéristiques acoustiques et spectrales en utilisant des séquences d'audios récoltées à partir de YouTube.

Nous avons exploré deux méthodes : la première était l'extraction des paramètres acoustiques et la deuxième a consisté en la classification des spectrogrammes. Les expériences ont été menées sur 23 dialectes algériens en utilisant des modèles d'apprentissage automatique, profond et par transfert.

Nous avons obtenu des résultats de 90% d'identifications correctes avec les fichiers de 20s. Ces résultats peuvent être améliorés en utilisant d'autres modèles d'apprentissage automatique, profond ou une approche prosodique qui aura de meilleure performance.

Mots Clés : Identification, acoustique, spectrogramme, apprentissage automatique, apprentissage profond.

Abstract

Dialects are a subject that interests several disciplines such as linguistics, phonetics, or even computer science, for their characteristics which do not always conform to linguistic rules. Algeria offers a rather interesting case study with its linguistic wealth.

The aim of this study was to propose a phonetic approach for the automatic identification of dialects, based on acoustic and spectral characteristics using audio sequences collected from YouTube.

We have explored two methods: the first was the extraction of acoustic parameters and the second was the classification of spectrograms. The experiments were carried out on 23 Algerian dialects using machine, deep and transfer learning models.

We have obtained a result of 90% of correct identifications with the 20s files. These results can be improved using other machine learning models, deep or a prosodic approach that will have better performance.

Key Words: Dialect, Acoustic, spectrogram, Machine Learning, Deep Learning.

ملخص

اللهجات هي مادة تهم العديد من التخصصات مثل علم اللغة، علم الصوتيات أو حتى علوم الكمبيوتر، لخصائصها التي لا تتوافق دائمًا مع القواعد اللغوية. تقدم الجزائر دراسة حالة مثيرة للاهتمام إلى حد ما مع ثرائها اللغوي.

كان الهدف من هذه الدراسة اقتراح نهج لفظي للتعرف التلقائي على اللهجات، استنادًا على الخصائص الصوتية والطيفية باستخدام لقطات صوتية التي تم جمعها من موقع يوتيوب، يستشهد من بينها لهجات الشرق والغرب ووسط الجزائر.

اكتشفنا طريقتين: الأولى استخراج المعلومات الصوتية والثانية تصنيف المخططات الطيفية. أجريت التجارب على 23 لهجة جزائرية باستخدام نماذج التعلم الآلي والعميق والتحويل.

حصلنا على 90٪ من نتائج تحديد الهوية الصحيحة مع ملفات 20 ثانية. يمكن تحسين هذه النتائج باستخدام نماذج أخرى من التعلم الآلي، أو التعلم العميق أو النهج الإيجابي الذي سيعطي نتائج أحسن.

الكلمات المفتاحية: التعريف، الصوتيات، المخطط الطيفي، التعلم الآلي، التعلم العميق.

Remerciement

Au terme de ce Travail nous remercions le centre de recherche CRSTDLA pour leur confiance et leur accueil.

Nous adressons nos remerciements également à Mr Lichouri qui nous a confié le sujet de ce mémoire et qui nous a constamment guidé dans sa réalisation.

Nous remercions Mr Abbas qui a partagé avec nous son savoir et ses connaissances.

Nous tenons à remercier notre Promotrice Mme Mezzi de nous avoir donné la chance de bénéficier de la clarté de son enseignement et de l'étendue de son savoir. Nous la remercions pour sa générosité, son enthousiasme et sa passion. Ainsi que pour les conseils bienveillants qui nous ont souvent encouragé et aidé à mener ce travail à bout.

Nous remercions tous les enseignants qui ont contribué à notre formation.

Dédicace

Je dédie ce travail à tous ceux qui me sont chers

A mes chers parents

Aucune dédicace ne saurait exprimer mon respect et mon amour, je vous remercie d'avoir cru en moi et de m'avoir encouragé et offert l'environnement sain pour évoluer et réussir dans mes études.

A ma sœur racha

Ce travail a vu le jour grâce à son aide sans limite, à sa force, sa curiosité et son dévouement. Je la remercie pour tout et lui souhaite beaucoup de réussites dans sa vie.

A ma Chère grand-mère

Je remercie ma grand-mère qui m'a boosté et motivé pour perfectionner ce travail.

A mes sœurs Asma, Keyssa et Thiziri.

A mes amis les plus proches Yousra et Linda qui m'ont soutenu et qui étaient toujours à mes côtés.

A mon ami et Binôme Samir

Je le remercie pour le travail fait et pour son soutien moral.

A la mémoire de mes grands-pères que dieu les accueils dans son

Vaste Paradis

Chahboub Thissas.

Dedication

I dedicate this work to my parents, who have been present since the start of this journey.

To my siblings, may God guide them towards success.

To Nabil, Akram, Riadh my partners in crime.

To my friends especially Walid, Mohamed, Abdou who I walked this long journey with.

To my friend Thissas and her sister Racha.

To Bombina.

To my cats Minou, Fifi, Soussou.

And last but not least to our beloved advisor Madam Mezzi.

I thank God for blessing me with them, and I pray for them in good faith.

SALMI Samir.

Table des matières

Table Des figures

LISTE DES TABLEAUX :

Liste d'acronymes

TAL : Traitement Automatique du Langage

TAP : Traitement Automatique de la Parole

SVM : Support Vector Machine

CNN : Convolutional Neural Network

RAP : Reconnaissance Automatique de la Parole

ASM : Arabe Standard Moderne

ADI : Automatic Dialect Identification

IA : Intelligence Artificielle

biLSTM: LSTM bidirectionnel

MFCC: Mel-Frequency Cepstral Coefficients

GLF: Golf

EGY: Egypte

LAV: Levantin

NOR : Nord-africain

AA : Dialecte Algérien

MLP : Multi Layer Perceptron

Introduction Générale

1. Contexte global

L'humain entretient une relation spéciale avec le langage car il lui est essentiel pour communiquer avec l'extérieure, tout comme c'est le lien commun qu'il partage avec sa communauté. C'est néanmoins fascinant de constater qu'au sein d'un même groupe plusieurs manières de parler la même langue peuvent émaner chez certains.

Les dialectes sont omniprésents en Algérie grâce à la diversité de sa population. C'est le motif principal qui a motivé l'organisme de recherche qui nous a accueilli pour travailler sur ce projet.

Le centre de recherche scientifique et technique pour le développement de la langue arabe (C.R.S.T.D.L.A), œuvre pour associer les domaines de la linguistique, la phonétique, l'informatique et d'autres disciplines afin d'entreprendre des recherches théoriques et appliquées pour développer des techniques et les implémenter sur la langue arabe.

C'est en suivant cet objectif qu'ils se sont intéressés aux dialectes variés de l'Algérie.

2. Problématique

Le dialecte Algérien avec sa richesse et diversité ne se conforme pas à des standards linguistiques car il se diffère de l'arabe standard et il est constitué d'un lexique qui peut avoir plusieurs autres origines. C'est pour cette raison que nous n'avons pas opté pour une approche linguistique, nous avons plutôt choisi la conception phonétique du langage. Donc, l'identification automatique du dialecte est la première étape pour effectuer plusieurs taches en TAL et en TAP (Traduction automatique, fouille d'opinions…) et ce travail sera un premier pas pour briser les barrières de communications entre certaines régions d'Algérie.

3. Objectif

Notre but vise à créer un outil qui automatise la détection du dialecte parlé par un locuteur à partir d'un audio seulement, et ce en utilisant plusieurs techniques d'apprentissage automatique et profond, sur 23 classes de dialectes uniquement, collectés à partir de vidéos YouTube. On cite parmi eux les dialectes de l'est, l'ouest,

et le centre d'Algérie, ainsi par manque de ressources il manque les parlés du sud les différents dialectes de la langue Tamazight.

4. Organisation du mémoire

Dans ce mémoire, nous allons d'abord introduire les concepts essentiels en Traitement automatique de la parole.

Dans le chapitre qui suit nous abordons l'arabe d'un point de vue linguistique et nous détaillons les dialectes de cette langue.

La Deuxième partie de ce chapitre traite de tous les travaux que nous avons pu recueillir dans l'état de l'art et qui se rapprochent de notre objectif. Ensuite, nous consacrons le troisième chapitre pour développer théoriquement le schéma des solutions et approches que nous avons choisi.

Le quatrième chapitre est la continuité logique du précédent car nous y incluons l'implémentation des solutions et l'évaluation des résultats obtenus et l'application finale.

Finalement nous achevons ce travail avec une conclusion générale qui explore les perspectives que nous offrent ces résultats.

Chapitre I : Traitement automatique de la parole

1. Introduction

Dans tous les systèmes de communication, la parole est le moyen le plus naturel pour établir un contact direct entre les humains. Avec l'évolution de la technologie, la parole a connu un large intérêt par la communauté scientifique qui s'est mis à l'étude et l'analyse de la langue et du mécanisme de production de la parole. Le but est de développer des algorithmes d'apprentissage pour apprendre à la machine de comprendre le langage humain, ce qui est équivalant à développer des systèmes de traitement automatique de la parole. Dans ce chapitre, nous allons d'abord aborder les différents niveaux d'analyse de la langue, puis on passera à la définition d'un signal vocal ainsi que de ses paramètres acoustiques et spectrales, pour finir on va aborder aussi le mécanisme de la parole et les différentes approches de reconnaissance de la parole.

2. Les Niveaux d'Analyse de La Langue

Pour le traitement automatique de la parole, on passe par différents niveaux d'analyses :

Phonétique, phonologique, prosodique, lexicale, syntaxique, sémantique.

- ***Phonétique*** : Science qui étudie les caractéristiques physiques des sons sur trois plans complémentaires (articulatoire, acoustique, perceptif).
- ***Phonologique :*** Étudie l'aptitude linguistique en relation avec le son, en faisant abstraction de ses propriétés physiques. Elle définit une des unités de base (phonèmes) avec des contraintes de combinaison.
- ***Prosodique :*** Elle recouvre les aspects liés à la hauteur de la voix, à l'intensité et à la durée des segments syllabiques.
- ***Lexicale :*** Segmentation des phrases en unités lexicales.
- ***Syntaxique :*** Elle étudie la syntaxe d'une langue et ses règles grammaticales.
- ***Sémantique :*** La sémantique est définie d'un point de vue linguistique, comme la relation entre la forme des signes linguistiques, ou « signifiants »,

et ce qui est signifié, ou "signifiés". En reconnaissance de la parole la sémantique restreint la combinatoire syntaxique.

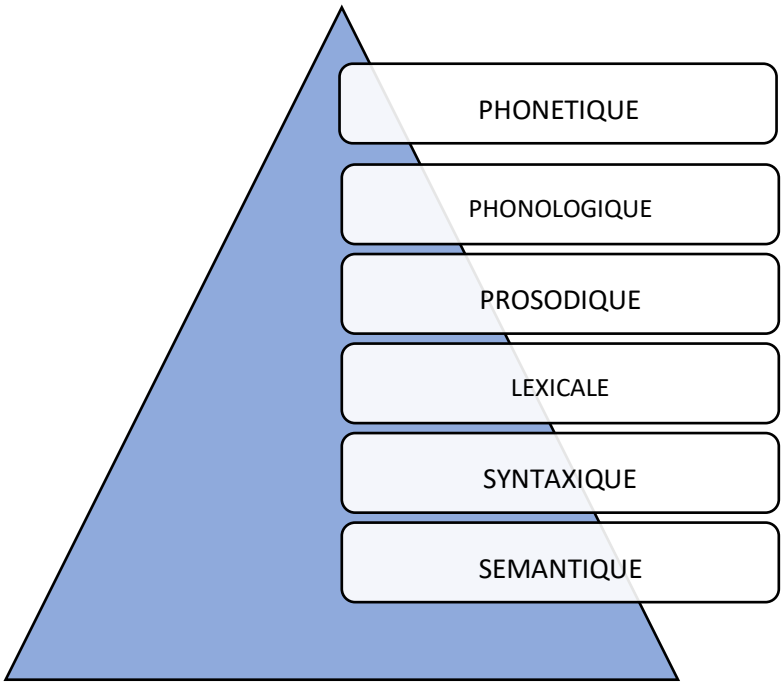


Figure 1 Les niveaux d'analyse de la parole.

-La figure 1 représente de manière descendante les niveaux d'analyse de la parole lors de l'analyse d'un signal vocal. Ce travail se focalise sur l'étude du niveau phonétique, plus précisément l'analyse acoustique des signaux vocaux.

3. Analyse Acoustique

La paramétrisation du signal de parole consiste en l'extraction d'un ensemble de vecteurs acoustiques. Le but de cette opération est d'obtenir une nouvelle représentation qui est plus compacte et plus appropriée à la modélisation statistique. Les paramètres les plus utilisés dans les systèmes de reconnaissance de la parole reposent sur une représentation cepstral du signal de parole. Avant l'extraction des paramètres, un prétraitement qui consiste à détecter les zones de silences est effectué afin de n'utiliser que les zones d'activité acoustique. Cette opération est très difficile à

mener à cause de la présence de bruit qui change les caractéristiques du signal de parole.

- ***Signal acoustique*** *:* Le signal est une variation dans le temps d'une grandeur physique de nature quelconque porteuse d'information. L'opération de numérisation du signal audio se réalise en théorie en trois étapes (échantillonnage, quantification, codage).

- ***Paramètre Acoustique*** *:* Pour chaque segment un vecteur de paramètres (traits acoustiques) est extrait, ces paramètres doivent être

  - pertinents : Extraits de mesures suffisamment fines, ils doivent être précis mais leur nombre doit rester raisonnable (éliminer la redondance des données) afin de ne pas avoir de coût de calcul trop important dans le module de décodage.

  - discriminants : Ils doivent donner une représentation caractéristique des sons de base et les rendre facilement séparables.

  - robustes : Ils ne doivent pas être trop sensibles à des variations de niveau sonore ou à un bruit de fond.

- ***Segmentation d'un signal vocal :*** Cette opération consiste à découper le signal en segments suffisamment homogènes pouvant être transcris en unités de base (phonème, syllabe…). Pour chaque segment un vecteur de paramètres (traits acoustiques) est extrait, ces paramètres doivent être pertinents et robustes.

3.1. Paramètres Acoustiques

L'étude acoustique du signal de parole correspond à l'évaluation de ses paramètres acoustiques à savoir : la durée, la fréquence fondamentale et l'intensité. Les modifications apportées à l'un d'eux peuvent altérer indéniablement les autres paramètres. Cependant, si nous voulons étudier ces paramètres d'un point de vue acoustique, nous pouvons les considérer comme étant parfaitement indépendants.

1. ***Intensité :*** L'intensité correspond au corrélat acoustique de la pression sub-glottique. Autrement dit, elle représente l'amplitude des vibrations des cordes vocales**.**

2. ***Durée :*** La durée est le paramètre acoustique le plus délicat à évaluer. La difficulté de mesure réside dans sa grande variabilité due au contrôle quasi impossible du système phonatoire. Chaque phonème se caractérise par ses propres durées intrinsèques et co-intrinsèques de même que le facteur de compressibilité/expansion. [1]
3. ***Fréquence fondamentale :*** Valeur de la fréquence fondamentale de l'émission glottale qui caractérise la hauteur de l'échantillon sonore. La connaissance de la fréquence f0 est nécessaire pour faire la différence entre la voix d'un homme, d'une femme ou d'un enfant : f0 moyen-homme 100 à 150 Hz, f0 moyen-femme 200 Hz à 300 Hz et f0 moyen-enfant 350 à 400 Hz. Elle est souvent associée au calcul de la variation relative temporelle de hauteur et de la variation relative temporelle d'intensité de l'objet sonore.

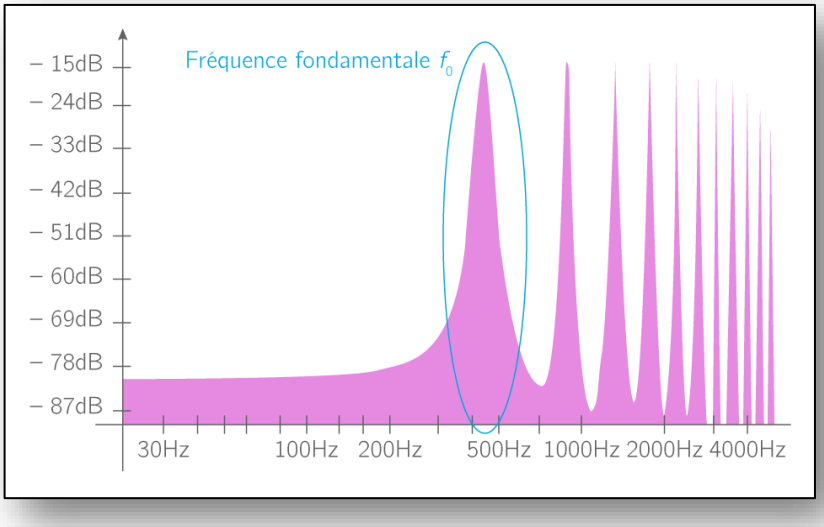


Figure 2 Fréquence fondamentale f0.

4. ***Formants :*** Valeurs des abscisses fréquentielles des pics qui décrivent la fonction de transfert du conduit vocal et qui différencient notamment les voyelles. Les formants sont au mieux au nombre de cinq (figure 3) mais les deux premiers (notés conventionnellement F1 et F2) les plus importants en intensité, sont souvent estimés suffisants pour discriminer les voyelles dans le plan du diagramme formantique.

-Chaque pique de la figure 3 représente les fréquences des formants (F1, F2, F3, F4, F5 respectivement) en fonction de l'intensité.

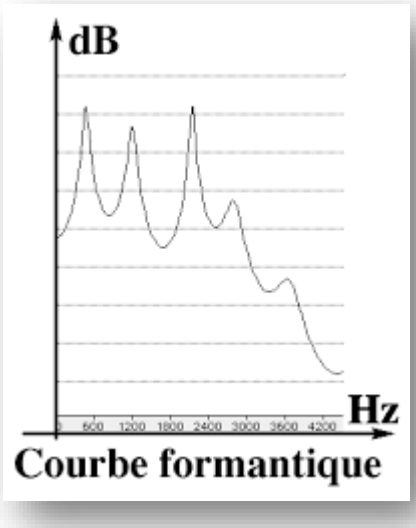


Figure 3 Courbe formantique.

4. Mécanisme de production de la parole

Le processus de production de parole est un mécanisme très complexe qui repose sur une interaction entre le système neurologique et physiologique. Il y a une grande quantité d'organes et de muscles qui entrent dans la production de sons des langues naturelles. Le fonctionnement de l'appareil phonatoire humain repose sur l'interaction entre les poumons, le larynx, et les cavités supra-glottiques.

Les poumons, le larynx fournissent ce qui est essentiel pour la production de n'importe quel son, qu'il soit musical ou langagier : une source d'air et une source de bruit. Les cavités supra-glottiques renferment les organes qui permettent de modifier le son qui est émis par le travail conjoint des poumons, larynx.

Lorsque l'air est expulsé des poumons, il passe à travers un tube formé de plusieurs cartilages appelé le larynx. Le larynx contient des muscles et des cartilages. Les cartilages les plus importants et les plus connus sont les cordes vocales qui peuvent s'ouvrir et se refermer très rapidement (jusqu'à 400 fois par seconde chez les enfants, par exemple), produisant ainsi des variations de pressions dans l'air. Ces variations de pressions sont perçues comme du son par l'oreille humaine. [1]

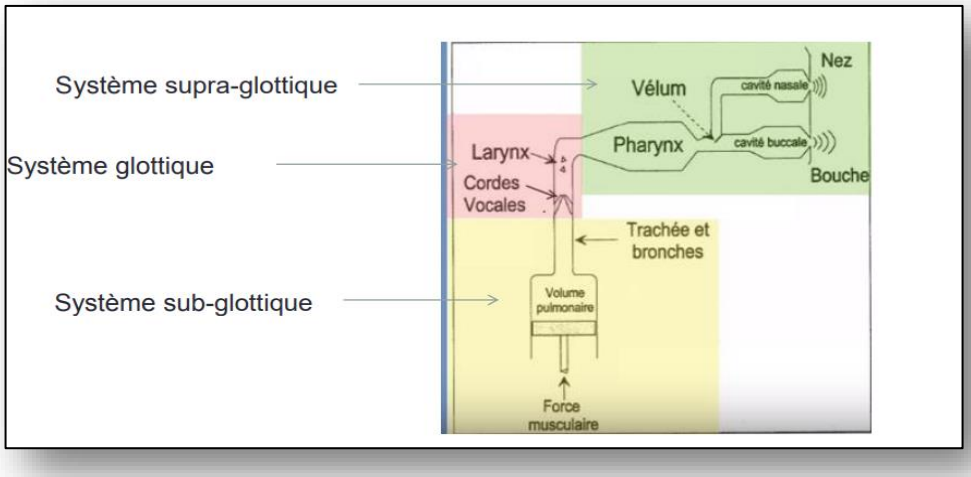


Figure 4 Mécanisme de production de la parole [1].

5. Les différentes approches de reconnaissance de la parole

Le principe général d'un système de RAP peut être décrit par la figure (5) :

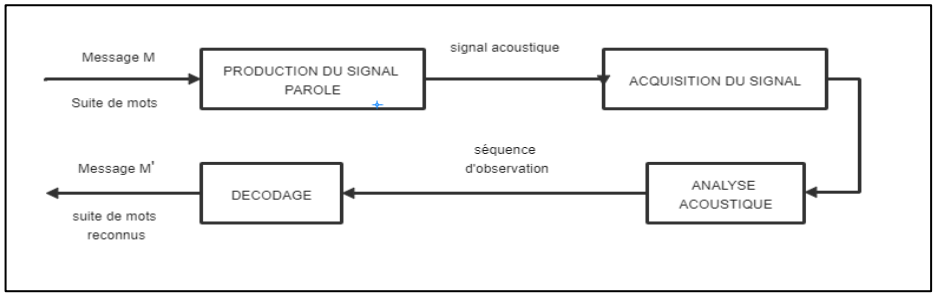


***Figure 5 Principe de reconnaissance de la parole** [2].*

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 acoustiques ou d'observations O (chaque vecteur est un ensemble de paramètres acoustiques).

Finalement le module de décodage consiste à associer à la séquence d'observations O une séquence de mots reconnus M'.

Un système Traitement Automatique de la parole transcrit la séquence d'observation O en en une séquence de mots M'en se basant sur le module d'analyse acoustique et celui de décodage.

Le problème du TAP est généralement abordé selon deux approches que l'on peut opposer du point de vue de la démarche : L'approche globale et l'approche analytique. [2]

5.1. L'approche Globale

L'approche globale considère l'énoncé entier comme une seule unité indépendamment de La langue. Elle consiste ainsi à abstraire totalement les phénomènes linguistiques et ne retenir que l'aspect acoustique de la parole. Cette approche est destinée généralement pour la reconnaissance des mots isolés séparés par au moins 200 ms (voir figure6), ou enchainés appartenant à des vocabulaires réduits.

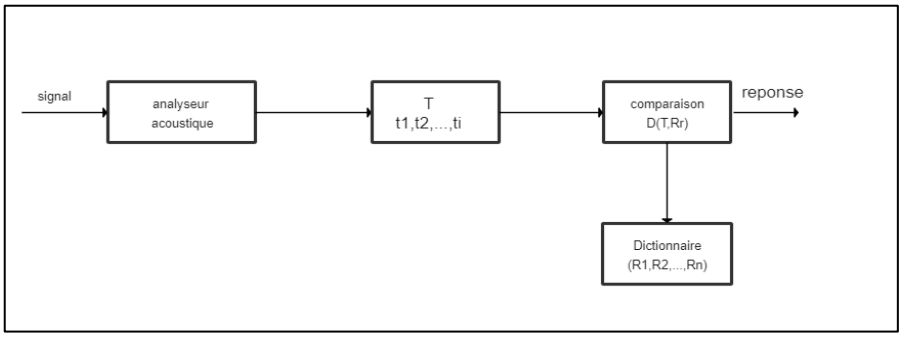


***Figure 6 Reconnaissance des mots isolés** [2].*

Dans les systèmes de reconnaissance globale, une phase d'apprentissage est nécessaire pendant laquelle l'utilisateur prononce la liste des mots du vocabulaire de son application. Pour chacun des mots prononcés, une analyse acoustique est effectuée permettant d'extraire les informations pertinentes sous forme de vecteurs de paramètres acoustiques. Le résultat est stocké ensuite en mémoire.

5.2. L'approche Analytique

L'approche analytique cherche à trouver des solutions au problème de la reconnaissance de la parole continue ainsi qu'au problème du traitement de grands vocabulaires. Cette approche consiste à segmenter le signal vocal en constituants élémentaire (mot, phonème, biphone, triphone « séquence de 3 phonèmes », syllabe)

puis à identifier ces derniers, et enfin à reconstituer la phrase prononcée par étapes successives en exploitant des modules d'ordre linguistique (niveaux lexical, syntaxique ou sémantique). Le processus de la reconnaissance de la parole dans une telle méthode peut être décomposé en deux opérations :

1. Représentation du message (signal vocal) sous la forme d'une suite de segments de Parole, c'est la segmentation. [2]

2. Interprétation des segments trouvés en termes d'unités phonétiques, c'est l'identification.

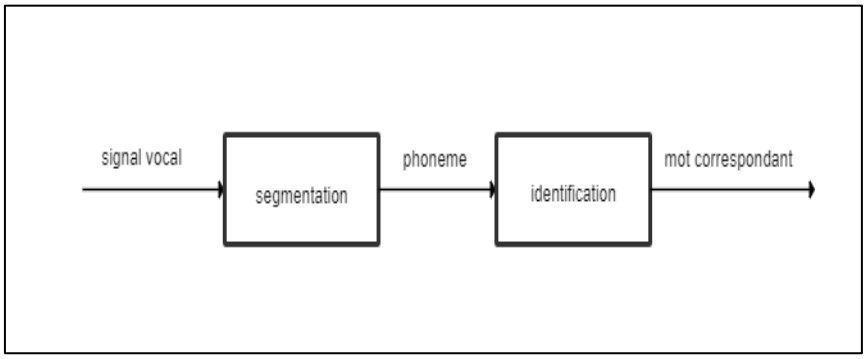
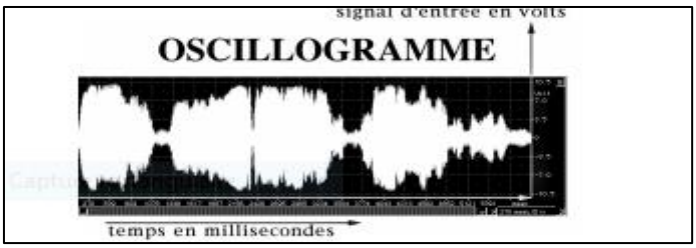


***Figure 7 Identification des mots** [2].*

6. Visualisation du son

L'étude du son a donné naissance à différents modèles de représentations ayant chacun un intérêt particulier.

6.1. Oscillogramme



Oscillogramme [2]

L'oscillogramme est l'une des plus anciennes représentations, il montre l'évolution temporelle de l'amplitude du signal (figure8). C'est une simple fonction du temps qui

ne dévoile pas la structure interne du son (sa composition fréquentielle) et qui se révèle peu intéressante pour des objets sonores complexes et notamment pour l'étude de la parole.

6.2. Représentation temporelle

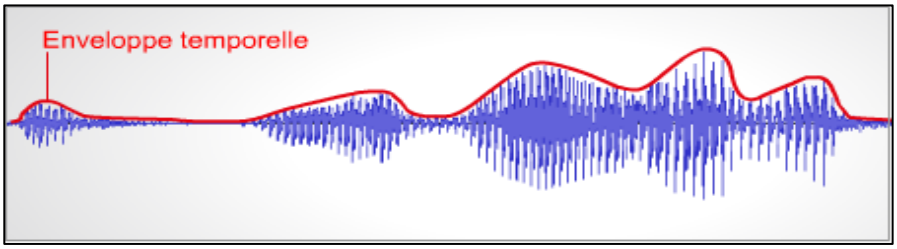


Figure 8 Représentation du son **[3].**

La figure 8 représente l'évolution de l'intensité du signal sonore dans le temps.

- La partie bleue montre l'évolution de l'intensité d'un son de parole dans le temps.
- Cette vue temporelle permet d'apprécier l'évolution de l'enveloppe temporelle (la ligne rouge) qui joue un rôle important dans la perception de la parole.

6.3. Représentation fréquentielle ou spectrale

Ce mode permet de visualiser la composition fréquentielle d'un son mais également l'intensité de chaque fréquence.

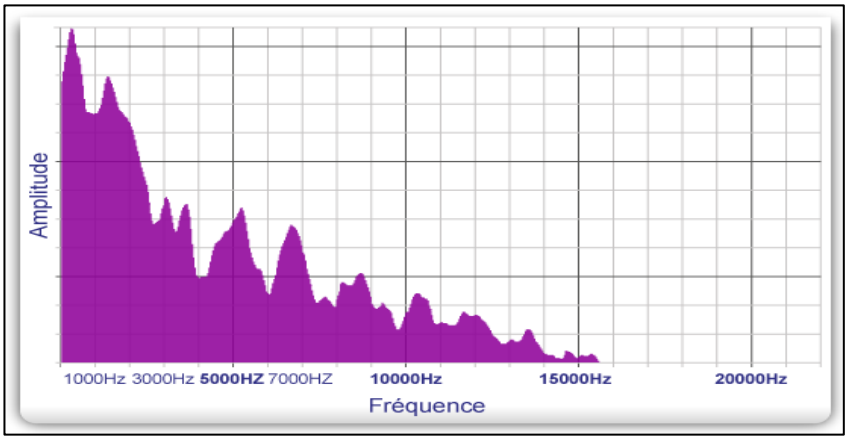


Figure 9 Représentation spectrale [3]

La figure 9 représente la composition spectrale de l'échantillon sonore (voir figure 8). On peut voir que la fréquence de son s'étend de 80HZ à 15000HZ.

6.4. Représentation Tridimensionnelle : Spectrogramme

Il s'agit de la représentation temps-fréquence du son. On trace la répartition énergétique du son en fonction du temps et des fréquences. Le sonagramme est très utilisé pour étudier le signal de parole.

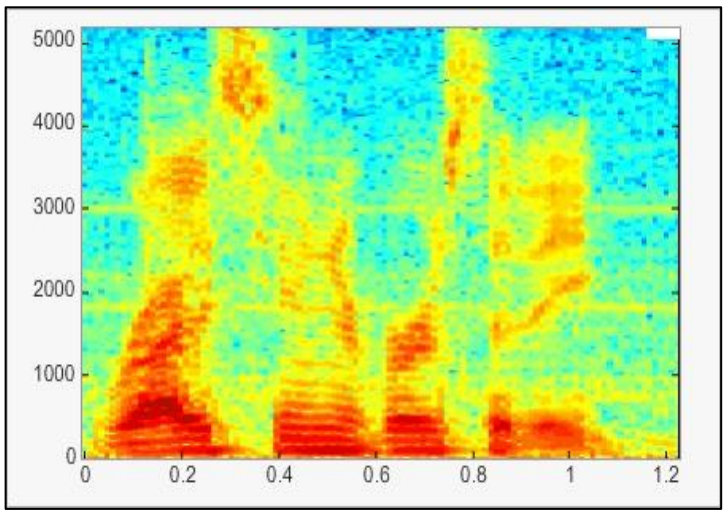


Figure 10 Spectrogramme.

-La figure 10 montre l'évolution de la fréquence et l'intensité dans le temps.

-L'intensité est définie par la couleur, plus la couleur évolue vers le rouge, plus l'intensité est importante. Les traits noirs soulignent les formants des voyelles.

-Il est possible d'associer d'autres palettes de couleurs telles que les niveaux de gris dans PRAAT (logiciel conçu pour la manipulation, la synthèse, et le traitement des sons vocaux).

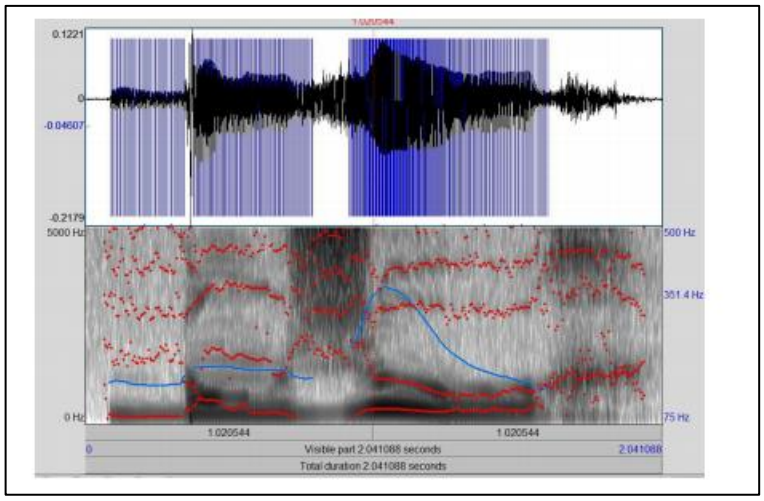


Figure 11 Spectrogramme du mot /bonjour/ parole de femme par PRAAT [4].

Dans la (Figure 11) l'énergie forte correspond à du noir et l'énergie faible à du blanc.

Dans la portion de son correspondant au /on/ de /bonjour/, on remarque une périodicité du son, c'est à-dire une forme qui se répète de façon régulière au cours du temps, même si de légères variations sont observées entre chaque cycle successif. Cette période vaut ici environ 5 ms, calculée entre deux pics du signal, dans le sens horizontal qui représente le temps. En inversant cette valeur, on trouve la fréquence correspondante notée F0, égale ici à 200 Hz environ. Il s'agit de la fréquence fondamentale, tout à fait dans la norme d'une voix de femme en parole spontanée. La hauteur est reportée dans le spectrogramme dans l'axe vertical. Il est donc normal de voir apparaître un trait de couleur chaude (donc d'énergie élevée) à environ 200 Hz.

Le trait en bleu dans PRAAT représente l'évolution de la fréquence fondamentale.

Les point en rouge nous donne le résultat des différents formants des voyelles.

7. Conclusion

Le traitement automatique prend beaucoup de paramètres en considération pour pouvoir automatiser la langue et créer des systèmes hommes machines robustes. Dans ce chapitre on a abordé les étapes et les outils importants pour le traitement automatique pour avoir un système performant. Dans le chapitre suivant nous allons voir le côté linguistique de la langue arabe et les difficultés rencontrées lors de son automatisation.

Chapitre II : Langue arabe Dialectale et Travaux Connexes

PARTIE 01 Description de la Langue Arabe et de l'arabe dialectal

1. Introduction

Le traitement automatique de la langue naturelle est un domaine vaste qui regroupe plusieurs disciplines, la linguistique, l'informatique, les mathématiques et même les sciences cognitives. Son but est d'apprendre à la machine de comprendre le langage humain en prenant en compte les règles grammaticales, syntaxiques et morphologiques de la langue.

Dans la 1ére partie de ce chapitre nous allons parler de la linguistique de la langue arabe standard et des problèmes rencontré lors du traitement automatique de la langue, puis on va voir les différents dialectes arabes qui existent et à la fin on va voir la spéciéité du dialecte algérien. La 2éme partie va aborder les travaux connexes, des techniques utilisés et à la fin nous allons récapituler les résultats de chaque travail.

2. Bref historique du TAL

Le traitement automatique des langues naturelles est né à la fin des années quarante du siècle dernier, dans un contexte scientifique très précis. Entre 1951 et 1954 ZELLING Harris publie ses travaux les plus importants en linguistique distributionnaliste. Au cours de cette année le premier traducteur automatique a été mis au point, en prenant quelques phrases en russes qui étaient sélectionnées à l'avance pour faire le test et pour les traduire en anglais [3].

En 1957 N. Chomsky publie ses premiers travaux importants sur la syntaxe des langues naturelles, et sur les relations entre grammaires formelles et grammaires naturelles qui sont très importantes pour le domaine du traitement automatique du langage naturelle. Les lois de Chomsky ont aidé à faciliter l'automatisation de la langue naturelle et à la création du premier vrai système automatique de traduction appelé Systran crée en 1957, qui traduit du russe vers l'anglais.

• l'absence de voyellation de la majorité des textes arabes écrits : ce phénomène entraîne un nombre important d'ambiguïtés morphologiques.

2.1 La Langue Arabe et ses variantes

L'arabe est une langue parlée par plus de 200 millions de personnes. Elle est la langue officielle d'au moins 22 pays. Cette langue s'est imposée en tant que langue de la littérature, la culture, de la pensé, des dictionnaires, des traités des sciences et des techniques. C'est une langue riche et élégante. Elle a connu un profond développement dans la syntaxe et l'enrichissement de son lexique. L'arabe peut être considérée comme un terme générique rassemblant plusieurs variétés [4]:

• *l'arabe classique* : Il s'agit d'une forme linguistique ancienne dont la grammaire a été fixée entre le 8e et le 10e siècle. L'arabe classique (dit aussi arabe « coranique ») n'est plus que la langue du patrimoine culturel passé avec ses œuvres classiques et son livre sacré : le Coran. L'arabe classique est appris dans les établissements d'enseignement à travers la littérature arabe classique.

• *l'arabe standard moderne (l'ASM)* : une forme un peu différenciée de l'arabe classique, et qui constitue la langue écrite de tous les pays arabophones. L'ASM reste le langage de la presse, de la littérature et de la correspondance formelle, alors que l'arabe classique appartient au domaine religieux et est pratiqué par les membres du clergé.

• *les dialectes arabes* : malgré l'existence d'une langue commune, chaque pays a développé son propre dialecte. Issus de l'arabe classique, leurs systèmes grammaticaux respectifs affichent de nettes divergences avec celui de l'ASM. On peut regrouper ces dialectes en quatre grands groupes :

1.les dialectes arabes, parlés dans la Péninsule Arabique : dialectes du Golfe, dialecte du Nadjd, yéménite.

2. les dialectes maghrébins : algérien, marocain, tunisien, hassanya de Mauritanie.

3. les dialectes proche-orientaux : égyptien, soudanais, syro-libano-palestinien, irakien (nord et sud).

4. la langue maltaise est également considérée comme un dialecte arabe.

3. Dialectes Arabes

3.1 L'arabe dialectal

L'arabe dialectal est une autre forme de la langue arabe utilisée dans les communications quotidiennes, généralement appelée 'āmmiyya "langue commune" ou dārija "langue courante". Cette variété possède également d'autres noms, parmi lesquels nous citons " l'arabe vernaculaire " et "l'arabe parlé". Elle est définie selon Al-Toma, 1969 comme étant "la langue courante des activités quotidiennes, elle est généralement parlée, bien qu'elle soit parfois écrite. Elle varie non seulement d'un territoire arabe à un autre, mais aussi d'une région à une autre au sein du même territoire". Les dialectes populaires sont également bien définis ; non pas parce qu'ils sont entièrement codifiés, mais parce qu'ils sont acquis naturellement par leurs locuteurs natifs. Ainsi, presque tous les pays arabes ont leurs propres dialectes qui sont plus ou moins différents les uns des autres au sein du même pays, et plus naturellement, de ceux des autres pays. Ces différences dépendent considérablement de l'histoire de chaque pays et de son emplacement géographique. Prenons par exemple l'Algérie qui était une colonie française après avoir été placée sous souveraineté de l'Empire ottoman. En dialecte algérien, le mot « table » emprunté du français est dit طابلة TaAblaħ en dialecte algérien, de même pour le mot سكارجي sukaArjiy emprunté du turque qui signifie 'ivrogne'. Le dialecte algérien comprend également plusieurs termes qui dérivent du berbère comme par exemple قرجومة Qarjuwmaħ pour dire 'gorge'. Les systèmes grammaticaux des différents dialectes affichent de nettes divergences avec celui du MSA. Cependant, nous signalons que pour deux pays arabes frontaliers, les populations qui vivent des deux côtés de la frontière parlent des dialectes très proches partageant une bonne partie de leur syntaxe et lexique. Par exemple, dans la région qui se situe au Nord-Est de l'Algérie, regroupant les villes de Souk Ahras, Tébessa et Annaba ; utilise un dialecte plus proche du dialecte tunisien que du dialecte algérien. *[3]*

3.2 Les variétés dialectales de la langue arabe

La classification des dialectes arabes a intéressé les chercheurs et les observateurs depuis plusieurs années. Plusieurs classifications ont été proposées pour la répartition de ces dialectes au cours des années selon certains critères à savoir le critère

géographique (horizontal) et le critère social (vertical). De ce fait, plusieurs grands groupes de dialectes, correspondant environ aux divers principes linguistiques, ont été proposés. Ces groupes répondent souvent à des divisions géographiques naturelles. Ce dernier constat est appuyé aussi par Versteegh, qui avance que : 'les critères des classifications courantes ne sont pas toujours clairs. Dans une certaine mesure, ils semblent souvent ne refléter qu'une répartition géographique'[1]. Cette classification géographique, selon Embarki est relativement récente par rapport à d'autres classifications, comme la classification sociologique. La dialectologie arabe distingue généralement deux grandes zones ou familles principales de dialectes *[4]* :

• La zone occidentale (l'Afrique du Nord, le Maghreb) : contient le groupe du Maghreb qui comporte l'Algérie, le Maroc, la Tunisie, la Libye et la Mauritanie,

• La zone orientale (le Machrek) : contient le groupe du Machrek comportant l'Égypte, la Syrie et les autres pays du Moyen-Orient (l'Irak, les Etats du Golfe, Yémen, Oman, Jordanie, etc.). Selon Baccouche ces groupes sont séparés géographiquement et approximativement par l'Est libyen (du Sallûm au Tchad) et présentant plusieurs traits distinctifs morphophonologiques et lexico-sémantiques.

- Les dialectes maghrébins **:** Les dialectes de cette catégorie sont caractérisés par une forte influence des langues française et berbère. La plupart des dialectes considérés peuvent être inintelligible par l'orateur dans d'autres régions du Moyen-Orient, en particulier sous forme orale. La géographie du Maghreb lui procure une grande région, de ce fait elle présente une plus grande variation de dialecte, plus importante que celle perçue dans d'autres régions comme le Levant ou le Golfe. Elle peut être aussi divisée en d'autres sous-catégories

[1] Versteegh , 2011.

Figure 12 Classification des parlers dialectal.

En plus de la géographie, le critère social est aussi proposé par certains chercheurs pour la stratification des dialectes, comme celle qui répartie les dialectes en deux groupes : groupes citadins et groupes bédouins. Cette classification est soutenue dans [5] qui explique : 'les linguistes et autres observateurs de l'aire arabophone ont montré depuis longtemps que la plus petite localité comme la région la plus étendue sont traversées par une division entre ʿarab (nomades) vs ḥaḍar (sedentaires). Le terme ḥaḍar correspond à une population sédentaire, de type citadin ou villageois ; quant à ʿarab, il englobe des populations nomades et semi-nomades'. Ceci porte le nombre de classes dialectales à trois : parlers bédouins nomades, parlers bédouins sédentaires, et parlers citadins. [3]

3.3 Aperçu historique du dialecte Algérien

Le dialecte algérien, noté AA, est l'un des dialectes du Maghreb parlé en Algérie. Ce dialecte est aussi appelé دارجة daArjaħ, جزايري jazaAyri ou دزيري dziyriy signifiant simplement 'algérien'. Ce dialecte est considéré comme un langage de basse variété (Faible variété). Ceci signifie que l'AA est faiblement normalisé et standardisé. Il est utilisé dans la presse, la télévision, la communication sociale, les échanges Internet, SMS, etc. Il est à mentionner que seules les communications officielles en lecture et en écriture n'utilisent pas le dialecte AA. Cependant, même si AA est parlé par la population de l'Algérie, estimée à 40 millions de personnes, il est caractérisé par une variation de ce même dialecte en fonction de l'emplacement géographique des

locuteurs de l'AA. Ces variations ne créent généralement pas d'obstacles à comprendre le dialecte. En plus de AA, la population algérienne parle aussi le tamazight mais avec des rapports différents : AA est utilisé par 70 à 80% de la population, cependant la langue berbère est la langue maternelle d'une communauté importante de la population algérienne : Un pourcentage important d'algériens sont des natifs Amazigh. La langue Amazigh est utilisée principalement dans le centre de l'Algérie (Alger et la Kabylie), l'Est de l'Algérie (Bejaïa et Sétif), dans les Aurès (le chaoui), dans le Mzab (nord du Sahara) et il est utilisé par les Touaregs basés dans le sud du Sahara (Hoggar).

De plus, le dialecte AA est influencé principalement par trois langues : l'arabe, le tamazight et le français. A ce titre, nous citons la définition du célèbre humoriste et comédien algérien, Mohamed Fellag, qui décrit le AA comme suit : « L'algérien de la rue est une langue trilingue, un mélange de français, d'arabe et de berbère. ». Cette diversité a contribué à avoir un paysage linguistique à la fois complexe et riche en Algérie comme l'avance Taleb Ibrahimi « le paysage linguistique de l'Algérie, produit de son histoire et de sa géographie, est caractérisé par la coexistence de plusieurs variétés langagières – du substrat berbère aux différentes langues étrangères qui l'ont plus ou moins marquée en passant par la langue arabe, vecteur de l'islamisation et de l'arabisation de l'Afrique du Nord. ». De ce fait, le dialecte algérien ne peut pas être présenté comme un système linguistique homogène, mais il possède de multiples variétés linguistiques. Selon Queffelec et al… Nous distinguons quatre variétés linguistiques pour le dialecte algérien :

3.3.1 L'Oranais

Cette variété est parlée dans l'ouest de l'Algérie, précisément depuis la frontière algéro-marocaine jusqu'aux limites de la ville de Ténès,

3.3.2 L'Algérois

Cette variété est largement répandue dans la zone centrale de l'Algérie jusqu'à Bejaia,

3.3.3 Le rural

Les locuteurs de cette variété sont situés dans l'est de l'Algérie comme Constantine, Annaba ou Sétif. Nous signalons aussi que les locuteurs situés plus à l'est, c'est-à-dire

de Constantine à la frontière algéro-tunisienne, sont aussi considérés dans cette catégorie. Il est aussi à signaler qu'il existe des déclinaisons de cette variante propre à certaines villes, comme c'est le cas pour les villes d'Annaba et de Constantine.

3.3.4 Le Saharien

Il est considéré comme le dialecte de la population algérienne habitant la partie sud de l'Algérie, à partir de l'Atlas saharien.

Par ailleurs, nous signalons aussi que le dialecte AA est enrichi par les langues des groupes ayant colonisé ou géré la population algérienne au cours de l'histoire du pays. Parmi les langues de ces groupes, nous citons : le turc, l'espagnol, l'italien et plus récemment le français. Nous pouvons considérer de ce fait le dialecte AA comme une fertilisation croisée de nombreuses langues avec l'arabe du fait de l'histoire de l'Algérie, qui a fait de cette dernière un carrefour de multiples civilisations et a donné lieu à une grande palette de variété pour les dialectes existants en Algérie.

Cette palette prend des couleurs régionales, provinciales voir même locales. Ces variétés sont matérialisées par la présence de mots étrangers dans le dialecte et de systèmes de prononciation différents variant sensiblement d'une région à une autre. En plus des mots d'emprunt et l'intégration de certains d'entre eux dans la morphophonologie du dialecte algérien, l'influence des langues sur le AA a été matérialisée également par l'alternance codique (le code switching) souvent dans les conversations quotidiennes, en particulier du français, par exemple, 'lycée', 'salon', 'quartier', 'normal', etc. L'utilisation de ces mots est réalisée sans aucune adaptation de la phonologie. Ceci crée une situation linguistique assez complexe. Ce mélange dans la langue a été étudié par de nombreux sociolinguistiques, qui ont décrit le paysage linguistique de l'Algérie comme « multilinguisme ou « poly-glossique » où plusieurs variétés de langues coexistent. [3]

4. Synthèse

On peut catégoriser le dialecte algérien dans la catégorie du « code-switching » vu sa diversité linguistique, et l'utilisation de plusieurs langues pour construire une phrase. On se retrouve face à une langue complexe et compliquée à automatiser.

Partie 02 Travaux connexes

Travail 01 Construction d'un corpus de discours basé sur un podcast arabe pour l'identification de la langue et du dialect **[6]** .

1.1. Problématique

Les auteurs travaillent sur un corpus de multi-langues et multi-dialectes qui contiennent : MSA (arabe standard moderne), anglais et 4 dialectes arabes : Saudia, Egyptien, libanais et syrien pour un système d'identification de la langue et du dialecte. Quels sont les techniques utilisées pour l'identification de ces langues ?

1.2. Approche utilisée

La Dataset de parole utilisée contient plus de 8H de podcast arabe téléchargé. Cette data (ArPod) contient le MSA et un peu de ses dialectes de ces régions : Arabie-Saoudite (KSA), Syrie (SYR), Égypte (EGY), Lybon (Lyb) en plus de l'anglais. Les langages/dialectes ont une durée entre 50mn à 1h30.Notons que les dialectes LYB, EGY et KSA incluent quelques expressions en anglais dans leurs conversations.

Le système comprend deux types de représentation data : acoustique et spectral. Plusieurs caractéristiques ont été utilisées comme MFCC et entropie de l'énergie, Taux de passage à zéro, centroïde spectral et d'autres. Dans ce travail deux schèmes ont été utilisés. En utilisant le 2éme type de représentation data et en utilisant les spectrogrammes. Les classifieurs utilisés dans ces expériences sont : KNN, SVM, MLP et Extra arbres.

1.2.1 Classification basée sur les caractéristiques acoustiques

Scheme 1

Dans ce schème 34 caractéristiques sont sélectionnées :

1. Coefficients MFCC (13)
2. Energie (1) et énergie d'entropie (1)
3. Taux de passage à zéro (1) et centroïde spectrale (1)
4. Atténuation spectrale (1) et vecteur chroma (12)
5. Propagation spectrale (1) et entropie spectrale (1)
6. Flux spectral (1) et déviation chroma (1)

Scheme 2

Dans ce schème un Framework sur la base de Librosa avec un total de 193 composants sont sélectionnés :

1. Coefficients MFCC (40)
2. Mel spectrogramme (128) et vecteur chroma (12)
3. Central spectral (7) et Tonnetz (6)

1.2.2. Classification basée sur spectrogramme

C'est un procédé de reconnaissance d'images pour résoudre le problème de l'identification de la langue parlée. L'idée est d'extraire le spectrogramme du discours de jeu de données au format .wav. Ensuite, appliquer un Classificateur CNN pour identifier les langues et les dialectes en fonction de leurs spectrogrammes respectifs.

1.3. Résultats

1.3.1. Résultats de langage et dialecte sans code-switching

La première expérience a été consacrée pour identifier les langues et dialectes qui ne contiennent pas de code-switching. Il s'agit de MSA, d'anglais, dialectes syrien et saoudien. Les expériences ont été réalisé sur des segments d'audios de 6,30 et 60 secondes.

Sur la base des résultats SVM basé sur le schème 2 surpasse le schème 1 et approches basées sur le spectrogramme, avec une mesure F1 égale à 96%, par de courts durée (6 sec). L'approche basée sur spectrogramme a donné un score F1 de 56 % pour les énoncés d'une durée de 1 min. On souligne que la performance basée sur les schèmes 1 et 2 est inversement proportionnelle à la durée, et elle est meilleure lorsqu'il s'agit d'énoncés plus courts. Ceci est vrai pour les classificateurs KNN, SVM et Extratrees, sauf pour les performances MLP qui augmentent légèrement avec la durée.

Le tableau qui suit (Tableau 1) résume les résultats de l'algorithme SVM schème 1.

Durée /Mesure	**Précision**	**Recall**	**F-score**
6s	69%	65%	65%
30s	59%	55%	50%
1min	78%	67%	52%

Tableau 1 Résultats de SVM schème 1.

Le tableau (2) donne les résultats de l'algorithme SVM et MLP avec schème 1

Tableau 2 Résultats de SVM et MLP avec schème 2

	Précision		***Recall***		***F-score***	
Algorithmes	*MLP*	*SVM*	*MLP*	*SVM*	*MLP*	*SVM*
6S	*100%*	*95%*	*25%*	*95%*	*40%*	96%
30S	*100%*	*93%*	*29%*	*93%*	*45%*	*95%*
1Mn	*100%*	*89%*	*33%*	*93%*	*49%*	*95%*

	Précision	**Recall**	**F-score**
6s	60%	45%	52%
30s	59%	55%	51%
1mn	65%	56%	56%

Tableau 3 approche basée sur spectrogramme CNN.

1.3.2. *Résultats des dialectes avec code-switching*

Les corpus sélectionnés pour être utilisés sont en égyptien, saoudien et Dialectes libanais où les locuteurs alternent entre l'anglais et ces dialectes. Le meilleur résultat a été obtenu par SVM en utilisant le deuxième schème avec un F1 de 98%, pour les énoncés les plus courts (6 secondes).

Cependant, contrairement aux expériences portant sur les langues et dialectes sans code-switching, les performances obtenues en utilisant les deux schèmes et l'approche basée sur le spectrogramme n'est pas influencée par la durée des énoncés de test.

Travail 02 identification automatique de la langue pour les langues berbères et arabes à l'aide de caractéristiques prosodiques **[7]**

2.1. Problématique

Ce travail décrit un système d'identification automatique pour distinguer entre deux langages communs en Algérie qui sont MSA (arabe standard moderne) et kabyle qui est un dialecte de tamazight algérienne. Les caractéristiques utilisées sont les caractéristiques spectrales et prosodiques. Alors quels sont les algorithmes utilisés pour l'étapes de la classification pour identifier les deux dialectes ?

2.2. Approches utilisées :

Deux bases de données ont été utilisées pour la construction du système d'identification une en MSA et une en kabyle, dans le but de concevoir une base de données bilingue. Dix phrases en MSA ont été prises, enregistrées par 13 locuteurs et réitérées 10 fois (ce qui donne un total de 13*10*10= 1300 phrases). Chaque phrase est échantillonnée en 44100Hz et codé en 16 bits. Par conséquent, les phrases du corpus ont été sélectionné pour satisfaire un critère : phrase phonétiquement équilibrée, dans le but d'avoir le plus grand nombre de phonèmes tout en conservant la fréquence d'occurrence de chaque phonème. Pour le kabyle, toute les phrases ont été intégrées dans un dialogue significatif. Chaque dialogue (24 phrases) est répété cinq fois par 6 locuteurs (4 hommes et 2 femmes) de la Kabylie ce qui donne un total de (6*24*5=720 phrases).

2.2.1. Extractions des caractéristiques

Les performances d'un système d'identification de langage dépendent essentiellement des vecteurs de caractéristiques qui représentent la parole. Dans ce travail, on étudie les caractéristique prosodique et acoustiques.

a. Fréquence fondamentale (pitch) : la mélodie est un trait prosodique qui représente la fréquence fondamentale du signal vocal en fonction du temps ; il représente la période de vibration des cordes vocales.
b. Stress : Caractéristique prosodique qui représente l'intensité du signal vocal.

c. Fréquence de coefficient cepstral : Les MFCCs sont des scripteurs spectraux. Ils représentent l'information acoustique d'un signal vocal. C'est la caractéristique la plus utilisée car elle résout le problème de la sensibilité de fréquence de l'audition.

2.2.2. Classification en utilisant SVM

Les SVM reposent sur le principe de la minimisation des erreurs quadratique moyenne qui leur permet d'améliorer les performances du système. Puisque l'objectif de notre système est de classer entre deux langues nous utiliserons SVM car ils sont bien adaptés pour cette tâche.

2.3. Résultats

Le diagramme suivant (fig13) montre un système d'identification automatique de langage.

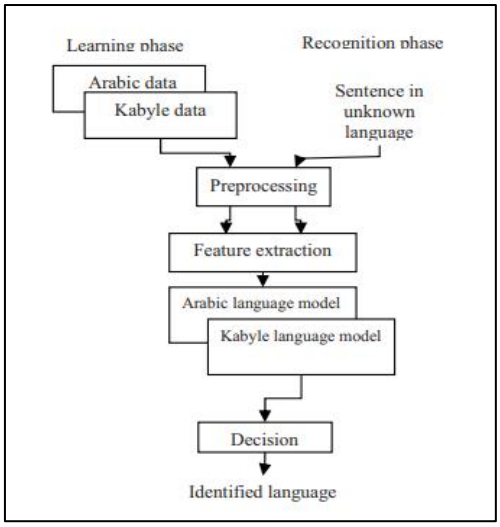


Figure 13 Système d'identification automatique de la langue

-Ce tableau (Tableau 4) montre une analyse comparative utilisant les caractéristiques prosodiques.

	MSA	Kabyle	Average
F0	90.41	82.08	86.25
Stress	98.75	91.67	95.2
Fusion	99.17	91.67	95.42

Tableau 4 Évaluer la Reconnaissance correcte à l'aide des caractéristiques prosodiques.

D'après le tableau 4 on remarque que la fusion des deux caractéristiques prosodiques nous donne un bon résultat de 95.42%. La matrice de confusion montre dans le tableau 5 les détails des erreurs faites lors de la fusion de stress et f0. La première colonne montre les langages de l'entrée des fichiers de tests, et la première ligne montre les langages identifiés avec le système d'identification.

	MSA	Kabyle
MSA	238	2
Kabyle	20	220

Tableau 5 Matrice de confusion de reconnaissance utilisant les paramètres de vecteurs combinés (melodie+stress).

Les Tableaux 4 et 5 montrent qu'une meilleure reconnaissance est obtenue en combinant les deux paramètres prosodiques comparé aux paramètres utilisé séparément.

Les résultats pour les paramètres acoustiques

Langage	*Précision*
MSA	97.91
Kabyle	93.75
Moyenne	95.83

Tableau 6 Précision et moyenne du système d'identification pour chaque langue.

Les résultats obtenus montrent que les performances sont similaires et légèrement meilleures que ceux obtenues en utilisant les paramètres prosodiques. Ce qui montre que les MFCCs sont des descripteurs robustes pour distinguer entre les langages. Ça donne sens car les deux langages ont différents phonèmes et séquences phonémiques. Les MFCCs vont aider à réduire la confusion entre MSA et kabyle et à améliorer le taux d'identification.

	MSA	**Kabyle**
MSA	235	5
Kabyle	15	225

Tableau 7 Matrice de confusion de reconnaissance utilisant un vecteur de paramètres MFCC et ses dérivants.

2.4. Synthèse

Dans ce travail, un système d'identification automatique a été conçu pour les deux langages les plus parlés en Algérie, MSA et kabyle, utilisant une combinaison entre les paramètres acoustiques et prosodiques. Ce qui a donné des résultats encourageant en termes de taux moyens de reconnaissance correct de 97.5%. Inclure les informations prosodiques a aidé à améliorer les résultats d'identification comparant au système utilisant les informations acoustiques seulement.

Travail 03 Architectures de réseaux de neurones pour l'identification du dialecte arabe [8]

3.1. Problématique

Le but dans ce travail est de classifier 5 dialectes arabes différents qui sont : l'arabe du golf, égyptien, Afrique du nord, levantin et MSA en entrainant plusieurs modèles de réseaux de neurones alors que la quantité des données d'entrainement disponible pour l'apprentissage est limité.

3.2. Approches utilisées

Deux jeux de données de tests ont été publiés en un seul pendant la phase de test, et ont ensuite été mis à disposition séparément : l'un est le jeu de test du challenge MGB-3 composé d'extraits de vidéos YouTube multi-domaine, l'autre est un ensemble de données de test surprise YouTube. La distribution des 5 dialectes égyptien (EGY), golfe (GLF), levantin (LAV), arabe standard moderne (MSA) et nord-Africain (NOR) dans les ensembles de données est présentée dans le tableau. Notons quelques faits concernant les transcriptions de mots : premièrement, la transcription est vide pour une proportion significative des phrases comme indiqué dans le tableau. Pour certains fichiers audios, la phrase est vraiment à peine intelligible alors que pour d'autres, la phrase est claire mais le fichier son semble avoir un volume inférieur. Deuxièmement, les transcriptions de mots ont tendance à contenir moins de mots inconnus par rapport aux années précédentes, mais de nombreux mots manquent dans les transcriptions. Enfin, les transcriptions semblent dominées par le MSA, avec quelques phrases dialectales transcrites en séquences MSA.

Dialecte		EGY	GLF	VBL	MSA	NI	Le total
Nombre d'énoncés	former	3177	2873	3117	2219	3205	14591
	développeur	315	265	348	238	355	1566
	Essai MGB-3	302	250	334	262	344	1492
	Test Youtube	1143	1147	1131	944	980	5345
Transcriptions de mots vides (%)	former	2,90	5.78	5.68	0,50	10,58	5.38
	développeur	5.71	2.26	6.03	1.06	2,54	3,64
	Essai MGB-3	1,66	1,20	1,50	0,00	1,45	1.21
	Test Youtube	4.37	2.09	12.82	1.06	8.47	5.84

Tableau 8 Répartition des dialectes et pourcentage de transcriptions de mots vides dans les ensembles de données ADI 2018.

Les éditions précédentes de la sous-tâche ADI ont montré comment des classificateurs plus traditionnels, comme SVM ou KRR surpassait les approches de réseau de neurones. Dans cette édition, SYSTRAN participe en entrainant plusieurs modèles de réseaux de neurones pour montrer que l'on peut également obtenir des résultats compétitifs par rapport à de tels classificateurs.

Les algorithmes utilisés pour ce système d'identification sont : SVM (*Machine à vecteur des supports) multi-classes à l'aide d'une fonction de base radiale)*, Réseau neuronal convolutif à multi-entrées, Réseau neuronal convolutif biLSTM, classification binaire avec CNN-biLSTM et classification des mel spectrogramme en utilisant CNN-biLSTM.

3.3. Résultats

Comparable aux résultats SVM (0.5270), le Multi Input CNN obtient le score F1 le plus élevé de 0. 5289.Contrairement aux attentes, le CNN-biLSTM fonctionnant directement sur la donnée audio n'a pas réussi à apprendre de meilleures représentations acoustiques pour l'identification des dialectes avec un score F1 de 0,3894. Cependant, l'utilisation de CNN-biLSTM pour la classification binaire et la prise de probabilité maximale ont amélioré les performances pour 0,4339. Les deux ensembles de test proviennent de vidéos YouTube, mais tous systèmes ont nettement mieux fonctionné

Sur l'ensemble de test MGB-3, qui est de taille beaucoup plus petite. En ce qui concerne les résultats de classification de la meilleure exécution résumés dans la figure 14, le taux d'erreur le plus élevé concerne les énoncés levantins souvent prédits comme appartenant au dialecte nord-africain, alors que ces dialectes ne sont ni géographiquement ni typologiquement proches.

Model	Test sets		
	MGB-3	Youtube	Total
Only lexical features	0.4493	0.4494	0.4502
Only phonetic features	0.3035	0.2839	0.2891
Only acoustic features	**0.5656**	0.5097	0.5239
Lexical + acoustic features	0.5630	0.5138	0.5267
Lexical + phonetic + acoustic features	0.5632	**0.5143**	**0.5270**
Only new acoustic features	0.7334	0.5379	0.5823
Lexical + new acoustic features	0.7350	0.5439	0.5873
Lexical + phonetic + new acoustic features	**0.7376**	**0.5470**	**0.5905**

Tableau 9 Résultats pour la tâche ADI (scores F1 macro-moyennés)

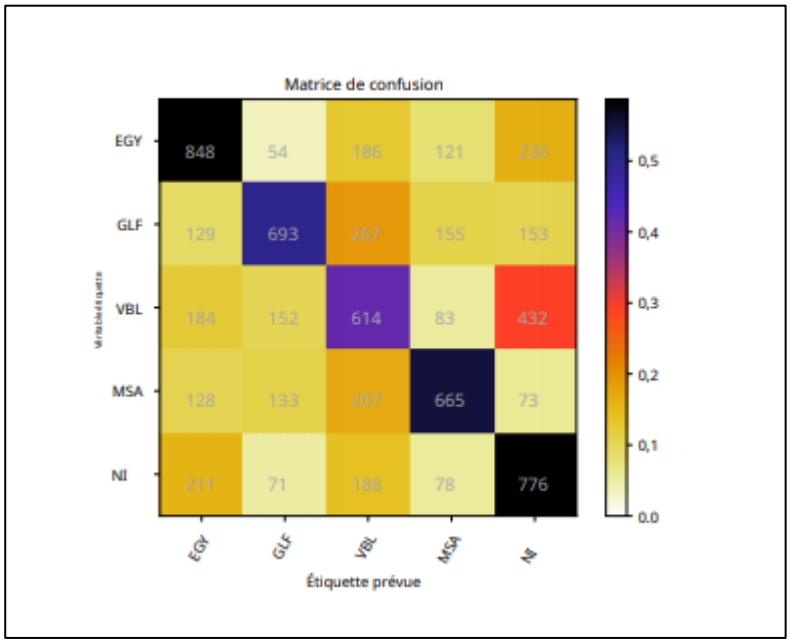


Figure 14 Matrice de confusion pour CNN à entrées multiples (série 1).

3.3.1. Résultats SVM

Dans le tableau 9, on présente les scores F1 qu'on a obtenus avec des SVM en utilisant une seule ou une combinaison des fonctionnalités. La classification avec des traits lexicaux donne de meilleurs résultats (F1 : 0,4502) qu'avec les traits phonétiques (F1 : 0,2891), mais classification avec caractéristiques acoustiques uniquement (F1 : 0.5239) les surpasse tous les deux. Les caractéristiques acoustiques semblent donc la représentation la plus utile des données pour cette tâche d'identification dialectale. Les combinaisons de fonctionnalités n'obtiennent que des résultats légèrement supérieurs (F1 : 0.5270), et pas pour le test MGB-3 ensemble. Les nouveaux encastrements acoustiques formés sur les trains et les dev sets (publié après la date de soumission) plus loin améliorer les performances jusqu'à 0,5905.

3.3.2. Résultats CNN multi-entrée (execution1)

Model	Test sets		
	MGB-3	Youtube	Total
Only lexical features	0.2988	0.3635	0.3505
Only phonetic features	0.3347	0.3251	0.3307
Only acoustic features	0.5495	0.5100	0.5209
Lexical + acoustic features	0.5483	0.5075	0.5184
Lexical + phonetic + acoustic features (run 1)	**0.5552**	**0.5186**	**0.5289**
Only new acoustic features	**0.7260**	0.5363	**0.5791**
Lexical + new acoustic features	0.7105	**0.5374**	0.5767
Lexical + phonetic + new acoustic features	0.7212	0.5258	0.5697

Tableau 10 Résultats des modèles CNN Multi-Input (scores F1 macro-moyennés).

Les auteurs ont sélectionné comme la meilleure configuration pour les intégrations de caractères CNN multi-entrées et les intégrations de téléphone de taille 32, appris séparément par convolutions 1D avec des filtres [5*8, 3*8] (taille de filtre*nombre), tanh fonction d'activation, décrochage 0,5 et mise en commun maximale globale ; puis une fois concaténé, une couche entièrement connecter de taille 32, Fonction d'activation ReLu, décrochage 0,5. On obtient la meilleure précision sur le dev set après 7 époques(epoch), mais on constate que sur le train la perte est déjà très faible et la précision très élevée dès le début de la 2ème époque, signalant un probable surajustement. Nos tests de plus haut (64) ou taille inférieure (16, 8) des inclusions, abandon plus élevé (0,7), tailles de filtre plus élevées [3, 5, 7] et deux couches de taille entièrement connectées 16 tous ont conduit à des résultats comparables, légèrement inférieurs. Dans le tableau 10, On présente les scores F1 qu'on a obtenus avec des systèmes reposant sur cette configuration, en utilisant sélectivement une, plusieurs ou toutes les fonctionnalités. Comparant à SVM, on remarque que les caractéristiques acoustiques à elles seules atteignent des performances similaires à la combinaison de caractéristiques, ce qui suggère que la classification dans ce système repose principalement sur des informations acoustiques. On observe une nette augmentation des performances lors de l'utilisation des nouvelles intégrations acoustiques formées sur les ensembles de train et de développement, en particulier sur l'ensemble de test MGB-3.

3.3.3. *Résultats CNN-biLSTM (série 2)*

Conv	Filters and Options	Test sets		
		MGB-3	Youtube	Total
1D	8*200, 4*400 (run 2)	0.4380	0.3711	0.3894
1D	8*200, 4*400 with masking	0.3587	0.2843	0.3013
1D	8*200, 4*400 with masking + batch normalization	0.2506	0.1932	0.2062
1D	**8*64, 4*64**	**0.4614**	**0.4098**	**0.4235**
1D	8*64, 4*64 with masking + balanced batch	0.3542	0.3046	0.3174
1D	8*64, 4*64 with batch normalization	0.1421	0.1390	0.1398
1D	3*3, 3*3, 3*3, 3*3	0.0749	0.0620	0.0649
2D	[3x3]*3, [3x3]*3, [3x3]*3, [3x3]*3	0.0652	0.0620	0.0763
2D	[7x7]*16, [5x5]*32, [3x3]*64, [3x3]*128, [3x3]*256	0.2507	0.2771	0.2721

Tableau 11 Résultats pour les modèles CNN-biLSTM (scores F1 macro-moyennés).

3.3.4. *Résultats CNN biLSTM (exécution 3)*

	F1 dans les systèmes binaires		F1 dans le système final
	Ce dialecte	Autres dialectes	Ce dialecte
EGY	0,50	0,78	0,49
GLF	0,53	0,84	0,54
VBL	0,39	0,82	0,31
MSA	0,38	0,92	0,37
NI	0,47	0,84	0,47
F1 (macro)			0,43

Tableau 12 Résultats par dialecte dans les 5 systèmes binaires et système final (scores F1).

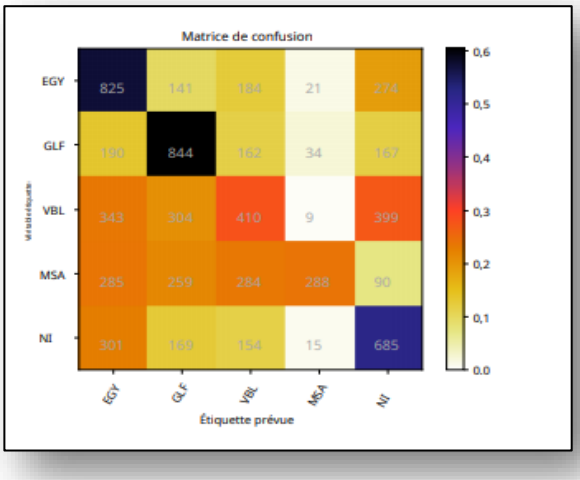


Figure 15 Matrice de confusion pour les CNN-biLSTM binaires (Run 3).

3.4. Synthèse

Les essais 2 et 3 donnent des profils d'erreur similaires, ce dernier étant illustré à la figure 15 : les dialectes égyptiens, du Golfe et d'Afrique du Nord sont en majorité correctement classés, mais ce qui est clair c'est la haute précision mais la mémorisation médiocre de la classification de l'arabe standard moderne et la grande incertitude du modèle pour le Levantin. Cela suggère que le CNN-biLSTM peut utilement reconnaître que les 4 dialectes sont différents de MSA mais ne parvient pas à reconnaître les énoncés levantins ou MSA, attribuant une étiquette au niveau du hasard. Une tentative d'explication de cette confusion est la forte présence de MSA dans les énoncés d'autres dialectes, notamment dans les fichiers audio égyptiens, du Golfe et du Levant. Comme le montre le tableau 12, le score F1 de chaque dialecte dans le système final de CNN-biLSTM est pratiquement identique au score F1 dans les systèmes binaires, ce qui n'apporte aucun avantage réel à la combinaison des classificateurs binaires. Ainsi, le levantin et le MSA pour lesquels le système présente la confusion la plus élevée et qui sont en fait typologiquement les dialectes les plus proches, présentent le score F1 le plus bas dans les systèmes binaires.

Conclusion

La langue arabe est une langue riche linguistiquement, elle est complexe et compliquée ce qui rend son automatisation difficile. Dans ce chapitre, nous avons abordé la linguistique de la langue arabe et les problèmes rencontrer lors de son automatisation. Ensuite, on a vu les techniques utilisées dans la littérature pour l'identification du dialecte arabe et de la langue arabe.

Dans le chapitre qui suit, nous allons voir les deux approches et les techniques utilisées pour l'identification automatique du dialecte algérien.

Chapitre III : Modélisation de La Solution

1 Introduction

Le dialecte algérien diffère d'une région a une autre ce qui nous donne la possibilité de les distinguer.

Dans ce chapitre, nous présenterons la modélisation de notre solution pour l'identification automatiques des dialectes Algériens. Nous commencerons par présenter le corpus utiliser pour que notre travail arrive à bout, ensuite nous passons à l'architecture globale de notre solution. A la fin nous allons définir notre approche pour la classification des audios et des spectrogrammes.

2 Schéma global

Le but de notre travail est d'identifier les dialectes algériens à partir des données fournies par la structure d'accueil. Pour ce fait, nous avons commencé par convertir les vidéos en audios. Après la conversion, nous passons au prétraitement des audios pour supprimer le bruit, cette étape est faite pour l'approche acoustique seulement.

Les audios vont être segmenté en fichier de 5s, 10s et 20s.

Nous exploitons les données segmentées pour effectuer deux méthodes différentes :

La première consiste à extraire les paramètres acoustiques des dialectes et les affecter dans un seul vecteur numérique pour les classifier en utilisant deux modèles, *Machine à Vecteur Des Supports et un modèle de Réseau de Neurone Convolutif*.

La deuxième méthode consiste à convertir les audios de chaque durée de segment en spectrogrammes et les images obtenus vont passer par un prétraitement puis les classifier en utilisant un modèle *d'Apprentissage Par Transfert pré-entrainé VGG16* qui a une très bonne accuracy pour la classification d'image. Finalement, nous évaluons le modèle entrainé en utilisant des mesures d'évaluations.

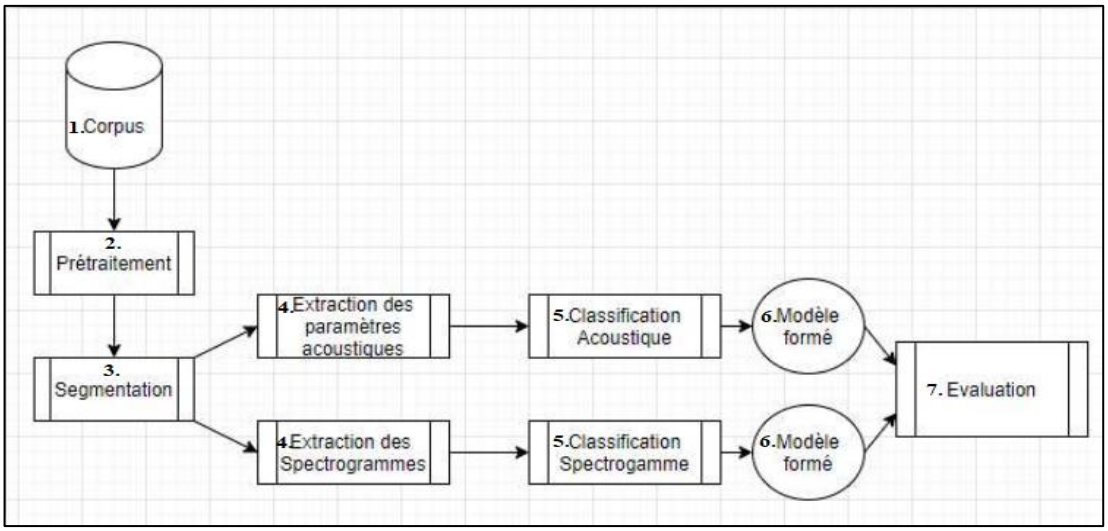


Figure 16 Schéma global de la solution.

La figure 16 représente le shéma global de notre implémentation de la solution pour l'identification automatique des dialectes algeriens.

2.1. Présentation du corpus

Pour concevoir les ensembles de données vocales algériens, avec diversité accentuel et dialectales, nous avons adopté YouTube comme une source pour notre travail, qui contient des vidéos avec des sujets variées créé par des auteurs de différentes régions du territoire nationale.

Notre corpus a été fournie par la structure d'accueil, il se compose de vidéos téléchargés à partir de YouTube sous format (MP4). Nous l'avons enrichie avec d'autre vidéo de la même source. Nous présentons le contenu de notre corpus dans la figure suivante (figure 20) :

Notre corpus contient 23 wilayas, donc 23 dialectes seulement. Nous avons quelques dialectes du centre, de l'ouest et de l'est qui sont cités dans la figure 17 avec des vidéos de durées qui varient entre 2h7mn et 2h42 mn.

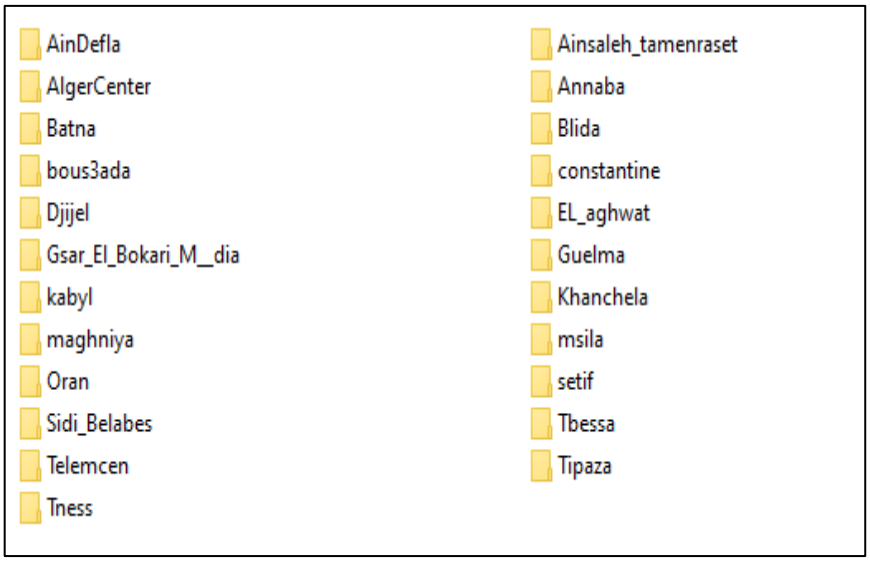


Figure 17 Contenu du Corpus.

2.2. Prétraitement Des données

Le prétraitement des données dans l'Apprentissage Automatique est une étape cruciale qui contribue à améliorer la qualité des données pour promouvoir l'extraction d'informations significatives à partir des données. Il fait référence à la technique de préparation (nettoyage et organisation) des données brutes pour les rendre adaptées à la construction et à la formation de modèle d'Apprentissage Automatique et de Deep Learning. En termes simples, le prétraitement des données est une technique d'exploration des données qui transforme les données brutes en un format compréhensible et lisible.

Prétraitement des audios : on supprime le bruit en utilisant la fonction **«** *NoiseReduce* **»** avec python qui réduit le bruit dans les signaux temporels comme la parole. Il s'appuie sur une méthode appelée **«** *spectral gate* **»** qui est une forme de Noise Gate, Il fonctionne en calculant un spectrogramme d'un signal et en estimant un seuil de bruit (ou porte) pour chaque bande de fréquence de ce signal/bruit. Ce seuil est utilisé pour calculer un masque, qui bloque le bruit en dessous du seuil de variation de fréquence [9].

Comme le montre la figure 18, le processus de NoiseReduce nettoie notre signal d'un grand nombre d'impuretés.

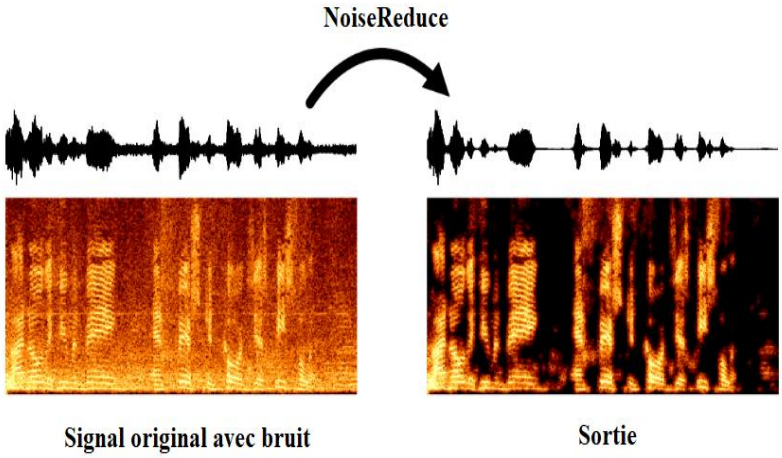


Figure 18 Reduction de bruit avec NoiseReduce [11].

Prétraitement des images : le traitement d'image peut être des tâches simples comme le redimensionnement d'image. Afin de fournir un ensemble de données d'images à un réseau convolutif, elles doivent toutes être de la même taille. D'autres tâches de traitement peuvent avoir lieu comme la transformation géométrique et de couleur ou la conversion de couleur en niveaux de gris et bien d'autres.

L'objectif du prétraitement des images est : Les données acquises sont généralement désordonnées et proviennent de différentes sources. Pour les alimenter au modèle ML (ou au réseau de neurones), ils doivent être standardisés et nettoyés. Le plus souvent, le prétraitement est utilisé pour effectuer des étapes qui réduisent la complexité et augmentent la précision de l'algorithme appliqué. Il n'est pas possible d'écrire un algorithme unique pour chacune des conditions dans lesquelles une image est prise, ainsi, lorsque nous acquérons une image, nous avons tendance à la convertir en une forme qui permet à un algorithme général de la résoudre.

3. Classification

La classification est une technique utilisée dans l'analyse des bases de données, elle permet d'apprendre des modèles de décision qui permettent de prédire le comportement des exemples futurs.

3.1. Classification audio

La classification audio est le processus d'écoute et d'analyse d'un enregistrement audio, ce processus est au cœur de la technologie d'IA moderne, y compris les assistants virtuels, les traducteurs, la reconnaissance vocale et plus encore.

Les projets de classification audio comme le nôtre commencent par des données audios annotées. Les machines ont besoin de ces données pour apprendre comment entendre et quoi écouter. En utilisant ces données, ils développent la capacité de différencier les sons pour accomplir des tâches spécifiques, dans notre cas l'identification des dialectes.

Il existe différents types de classifications audio, dans notre travail nous nous intéressons à deux d'entre elles :

3.1.1. Classification basée sur les paramètres acoustiques

Également connu sous le nom de détection d'événement acoustique, ce type de classification identifie l'endroit où un signal audio a été enregistré. Cela signifie différencier les environnements tels que les restaurants, les écoles, les maisons, les bureaux, les rues, etc. Une utilisation de la classification des données acoustiques est la création et la maintenance de bibliothèques de sons pour le multimédia audio. Il joue également un rôle dans la surveillance des écosystèmes. Un exemple en est l'estimation de l'abondance des poissons dans une partie particulière de l'océan sur la base de leurs données acoustiques.

Nous avons adopté cette technique dans notre implémentation pour les performances satisfaisante qu'elle a démontré dans la littérature. [6]

Dans la section suivante, nous allons présenter les paramètres acoustiques que nous avons extrait pour cette classification.

3.1.1.1. Paramètres acoustiques

A. Mel-Spectrogramme :

Mel-spectrogramme calcule un coefficient de spectrogramme de puissance à l'échelle Mel. Un objet de type spectrogramme mel montre une représentation acoustique

temps-fréquence du son, comme la montre (la Figure 19 : Mel-spectrogramme). La densité spectrale de puissance est échantillonnée en un certain nombre de points autour de temps et de fréquences également espacés (sur une échelle de fréquence mel) [10].

L'échelle de fréquence mel est définie comme :

Mel = 2595 * log10((1+hertz/700) [10].

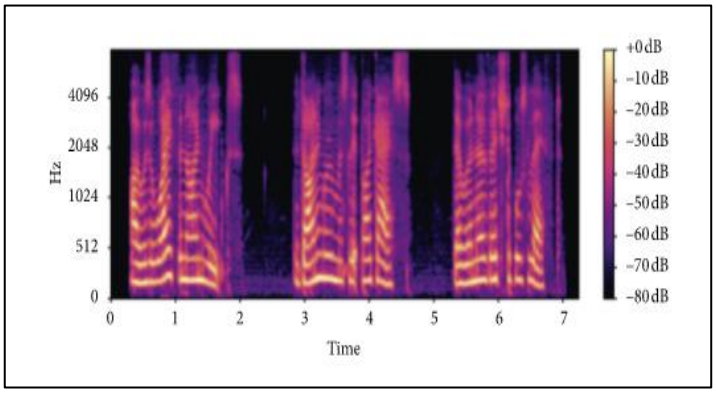


Figure 19 Mel-spectrogramme.

B. MFCC

MFCC représente avec précision le conduit vocal qui est une forme filtrée d'une voix humaine et se manifeste ainsi dans l'enveloppe d'un spectre de puissance à court terme, comme la montre (la Figure 20 : MFCC). Afin de calculer les MFCC [10], un ensemble d'étapes séquentielles doivent être suivies :

- Encadrement du signal en trames courtes. Le signal audio est encadré en trames de 20 à 40 ms (25 ms est standard) pour surmonter les changements dans l'échantillon sur une courte période de temps, car il est constamment modifié sur une longue période de temps [10].

- Périodogramme du spectre de puissance. Celui-ci calcule pour chaque trame l'estimation du périodogramme du spectre de puissance, qui identifie les fréquences dans la trame [10].

- Application du Mel Filterbank aux spectres de puissance (ou addition de l'énergie dans chaque filtre). Un filtre est nécessaire pour estimer les énergies dans diverses régions de fréquence qui apparaissent dans un groupe de cases de périodogramme agrégé en raison d'informations inutiles dans l'estimation spectrale du périodogramme. Par conséquent, le Mel Filterbank estime l'énergie près de 0 Hz, puis pour des fréquences plus élevées car il y a moins de souci pour les variations [10].

- Logarithmes de toutes Energie de Filterbank. Les grandes variations d'énergies sont mises à l'échelle à l'aide d'une échelle logarithmique car il n'y a pas de sons différents dans les grandes énergies. L'échelle logarithmique est une

technique de normalisation de canal qui est également exploitée pour la soustraction de la moyenne cepstrale [10].

- DCT du Log Filterbank Energies. En raison de la corrélation des énergies Filterbank qui conduit au chevauchement, le DCT est utilisé pour décorréler les énergies. Cela génère des matrices de covariance diagonales en tant que caractéristiques [10].

- Coefficients DCT, Des coefficients DCT plus élevés sont choisis pour réduire les changements rapides des énergies du Filterbank et éliminer le reste [10].

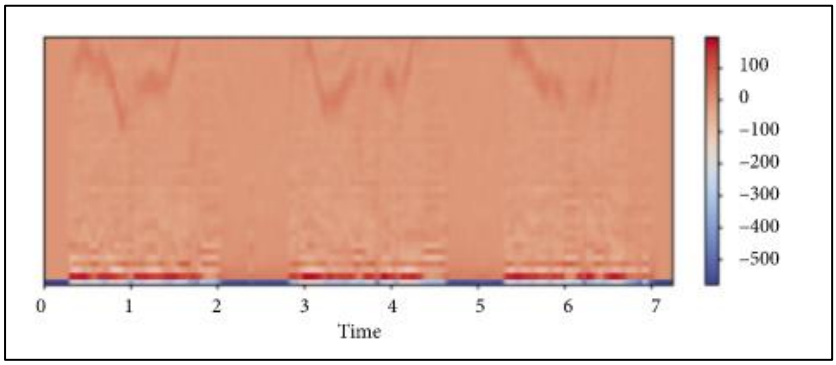


Figure 20 MFCC.

C. Chroma-STFT (Short-Time Fourier Transform)

Chroma-STFT calcule un Chromagramme à partir d'une forme d'onde ou d'un spectrogramme de puissance, comme la montre (la Figure 21 : Chroma-STFT). Les caractéristiques chroma sont de puissants représentants d'un son musical dans lequel l'ensemble du spectre est projeté sur 12 cases représentant les 12 demi-tons distincts (ou Chroma) de l'octave musicale [10].

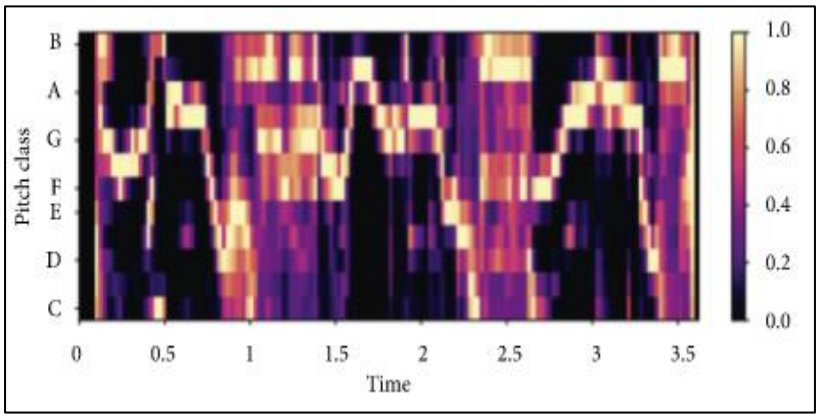


Figure 21 Chroma-STFT.

D. Spectral Contrast

Spectral Contrast calcule le contraste spectral. Il représente la distribution spectrale relative au lieu de l'enveloppe spectrale moyenne [10].

E. Tonnetz

Tonnetz calcule les caractéristiques centroïdes tonales (ou Tonnetz), qui détecte les changements dans le contenu harmonique des signaux audio musicaux [10].

Nous avons utilisé ces caractéristiques comme données d'entrée pour l'apprentissage d'un model SVM et un autre CNN.

3.1.2. Classification basée sur les spectrogrammes

Nous avons choisi d'implémenter cette méthode en plus de la précédente, pour comparer la performance de la technique se basant sur l'extraction des caractéristiques acoustiques et celle utilisant les spectrogrammes en tant que caractéristique représentative du signal vocal.

La classification des sons est l'une des applications les plus utilisées dans l'apprentissage audio en profondeur. Il s'agit d'apprendre à classer les sons et à prédire la catégorie de ce son. Dans le cas des spectrogrammes, on classe les sons à partir de l'image qui l'identifie donc à partir du spectrogramme qui associe à chaque fréquence une intensité ou une puissance. Dans ce cas la classification est une classification d'image. Ce type de problème peut être appliqué à de nombreux scénarios pratiques, par exemple la classification de clips musicaux pour identifier le genre de musique, ou la classification de courts énoncés par un ensemble de locuteurs pour identifier le locuteur sur la base de la voix.

4. Algorithmes utilisés pour la classification

Dans cette section nous allons voir l'architecture des algorithmes de classification que nous avons implémenter dans notre travail.

Nous commençons par L'algorithme d'Apprentissage automatique que nous avons utilisés dans la classification des paramètres acoustique, ensuite l'architecture du CNN et l'algorithme d'apprentissage par transfert utilisé pour la classification des spectrogrammes.

4.1. Machine à vecteurs des supports binaire (SVM binaire)

C'est une classe d'algorithmes de classification supervisée qui nécessite la création d'une base d'apprentissage, leur principe est de séparer les données en classes à l'aide d'une frontière, de telle façon que la distance entre les différents groupes de données et la frontière qui les sépare soit maximale. Cette distance est aussi appelée "marge", les "support du vecteur" étant les données les plus proches de la frontière. L'idée des SVM est de trouver un hyperplan qui sépare le mieux ces deux classes.

Le cas le plus simple, est celui où les données d'entrainement viennent uniquement de deux classes différentes aussi appelé SVMs binaire comme la montre la figure 22.

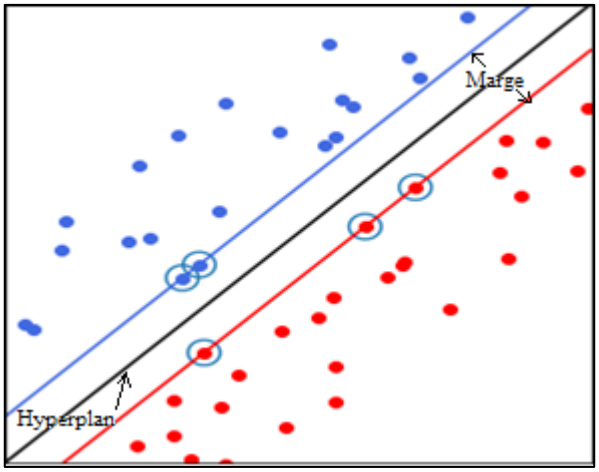


Figure 22 Support Vector Machine.

Si un tel hyperplan existe c'est-à-dire les données sont linéaments séparables, c'est une machine a vecteur support a marge dure.

L'hyperplan séparateur peut être représenter par la fonction suivante :

$$H(x) = \text{ax} + b$$

Et la fonction de décision pour un exemple x peut être représenter comme suite :

$$\begin{cases} Class = bleu\ si\ H(x) > 0 \\ Class = rouge\ si\ H(x) < 0 \end{cases}$$

Puisque nous parlons de données linéaments séparables, il n'existe aucun exemple qui se situe sur l'hyperplan c'est-à-dire H(x)=0.

Mais la réalité est loin de séparer le bleu du rouge, c'est-à-dire un hyperplan séparateur n'existe pas toujours ou ne représente pas toujours la meilleure solution pour la classification.

Dans le cas où les données ne sont pas linéament séparable ou contient du bruit ou tout simplement des erreurs dans l'étiquetage des données d'entrainements, cela nécessite des méthodes et des fonctions bien plus complexe de ce qu'on a vu précédemment.

Dans notre cas, nous avons 23 classes donc nous allons opter pour une classification multi-classes, pour ce fait nous allons régler le paramètre SVM « *Decision_function_shape = « ovo »* ». Pour notre classification multi-classe qui est le principe de one-to-one.

4.2. Machine à vecteur des supports multi-classe

Les machines à vecteur support sont dans leur origine binaires. Cependant, les problèmes du monde réel sont dans la plupart des cas multi-classe, donc la décision n'ai pas binaire et un seul hyperplan ne suffit pas [11].

Il existe deux méthodes pour contourner ce problème,

4.2.1. One-to-Rest

Tracer de multiple frontière de décision entre les différentes classes, ces méthodes décomposent l'ensemble d'exemple en plusieurs sous-ensembles représentant chacun un problème de classification binaire, et un hyperplan de séparation est déterminer par la méthode SVM binaire, On construit lors de la classification une hiérarchie des hyperplans binaires qui est parcourue de la racine jusqu'à une feuille pour décider de la classe d'un nouvel exemple. Comme la montre la figure suivante (figure 23) [11].

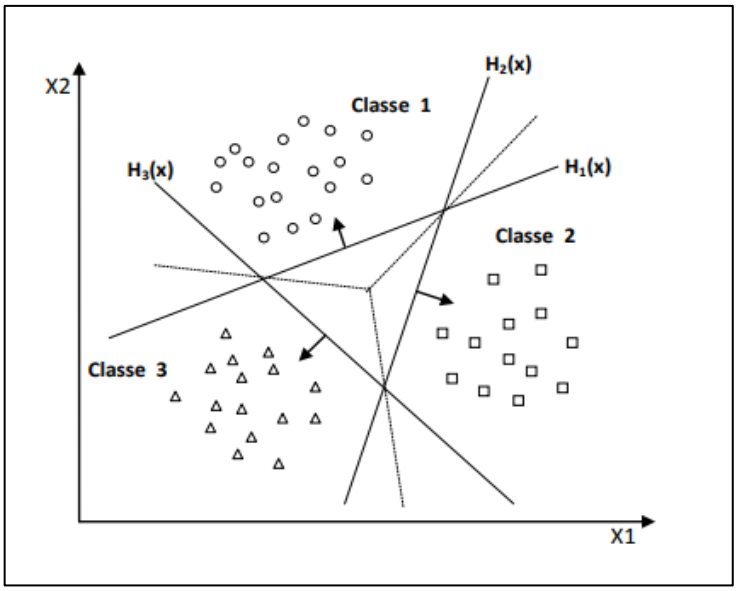


Figure 23 SVM OVR.

4.2.2. One-to-One

Nous avons besoin d'un hyperplan pour séparer toutes les deux classes, en négligeant les points de la troisième classe. Cela signifie que la séparation ne prend en compte que les points des deux classes dans la répartition actuelle [11].

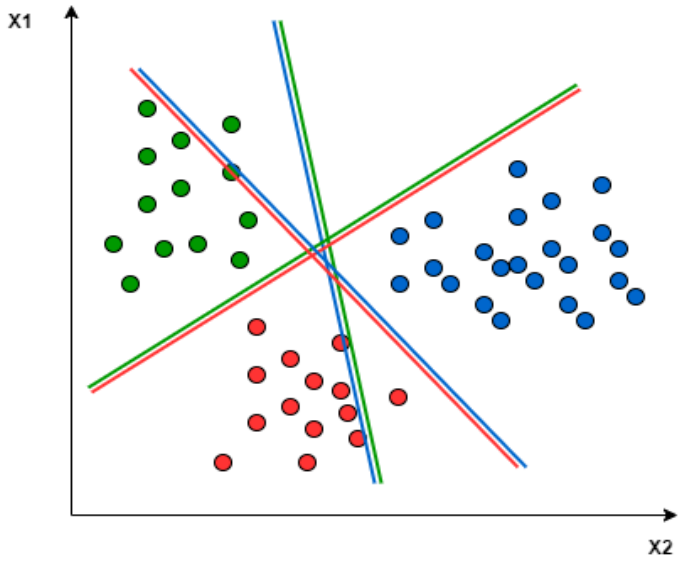


Figure 24 SVM OVO.

4.3. Réseau neuronal convolutif (CNN)

Les réseaux neuronaux convolutifs (Convolutional Neural Network, CNN ou ConvNet) sont utilisés avec succès dans un grand nombre d'applications. La tâche de reconnaissance de l'écriture manuscrite a été l'une des premières applications de l'analyse d'image par réseaux de neurones convolutifs. En plus de fournir des bons résultats sur des tâches de détection d'objet et de classification d'images, ils réussissent également bien lorsqu'ils sont appliqués à la reconnaissance faciale, à l'analyse vidéo, audios ou encore à la reconnaissance de texte [12].

4.3.1. Principe des réseaux de neurones convolutifs

Les réseaux neuronaux convolutifs ou ConvNets sont conçus pour traiter des données qui se présentent sous la forme de tableaux de valeurs en N dimensions pour N ∈ N +∗. Par exemple, une image couleur se compose de trois tableaux 2D contenant des intensités de pixels dans les trois canaux de couleur RVB (rouge, vert, bleu). Mais de

nombreux autres types de données se présentent sous la forme de tableaux à multiples dimensions :

-1D pour les signaux et les séquences, y compris la langue ;

- 2D pour images ou spectrogrammes audios ;

- et 3D pour les images vidéo ou volumétriques.

Le principe des ConvNets repose sur quatre idées clés qui exploitent les propriétés des signaux naturels :

- les connexions locales.

-les poids partagés (expliqué dans la section qui suit).

- la couche de regroupement (pooling) et les couches convolutifs (expliquée dans la section suivante).

4.3.2. Architecture Globale du CNN

Les CNN sont composés de trois types de couches : des couches convolutives, des couches de regroupement et des couches entièrement connectées. Comme la montre la figure 25 architecture de base (CNN)

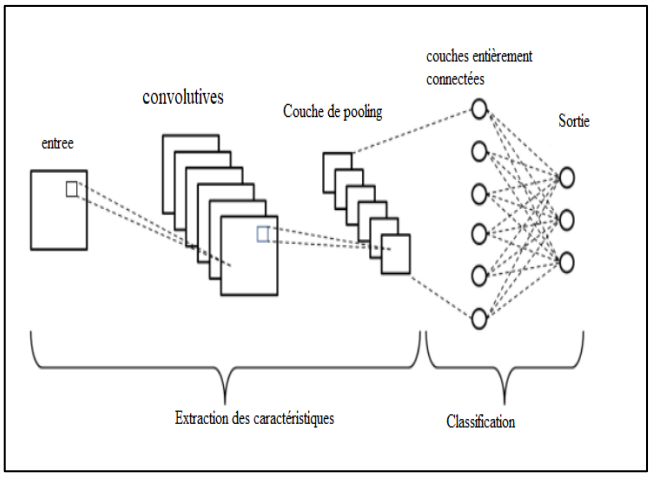


Figure 25 architecture de base (CNN).

Les différentes couches du CNN sont :

a. Couche convolutive :

Les couches convolutives constituent le noyau du réseau convolutif. Ces couches se composent d'une grille rectangulaire de neurones qui ont un petit champ réceptif étendu à travers toute la profondeur du volume d'entrée. Ainsi, la couche convolutive est juste une convolution d'image de la couche précédente, où les poids spécifient le filtre de convolution.

La couche convolutive détermine la sortie des neurones qui sont connectés aux régions locales de l'entrée par le calcul du produit scalaire entre leurs poids et la région connectée au volume d'entrée. ReLu vise à appliquer une fonction d'activation « élémentaire » telle qu'une fonction sigmoïde à la sortie de l'activation produite par la couche précédente.

Son objectif est de détecter la présence de caractéristiques (features) dans les images d'entrée. Cela est réalisé grâce à un filtrage par convolution qui consiste à faire glisser une fenêtre représentative de la caractéristique sur l'image d'entrée et à calculer le produit de convolution entre la caractéristique et chaque portion de l'image balayée. Dans ce contexte, le concept de caractéristique est assimilé au filtre. Dans chaque couche convolutive, chaque filtre est répliqué sur tout le champ visuel. Ces unités répliquées partagent la même paramétrisation (vecteur de poids et biais), c'est à dire

que les poids sont partagés, et forment une carte de caractéristiques (features map). Cela signifie que tous les neurones d'une même couche convolutive répondent aux mêmes caractéristiques. Cette réplication permet ainsi de détecter les caractéristiques quelle que soit leur position dans le champ visuel. C'est l'invariance par translation et c'est une caractéristique fondamentale des réseaux neuronaux à convolution. [12]

b. Couche de pooling

Ce type de couche est souvent placé entre deux couches de convolution : elle reçoit en entrée plusieurs caractéristiques, et applique à chacune d'entre elles l'opération de pooling. Qui réduit la taille des images, tout en préservant leurs caractéristiques importantes. On obtient en sortie le même nombre de caractéristiques qu'en entrée, mais celles-ci sont bien plus petites.

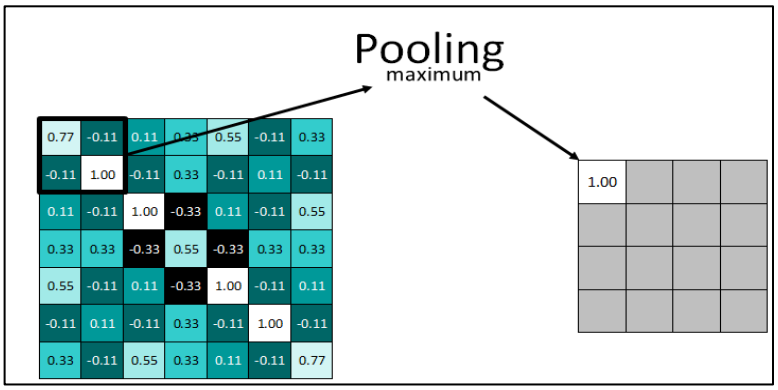


Figure 26 Réseaux de neurones convolutif couche de Pooling.

c. Couches entièrement connectée

La couche entièrement connectée constitue toujours la dernière couche d'un réseau de neurones, convolutif ou non. Elle n'est donc pas caractéristique d'un CNN. Ce type de couche reçoit un vecteur en entrée et produit un nouveau vecteur en sortie. Pour cela, elle applique une combinaison linéaire puis éventuellement une fonction d'activation aux valeurs reçues en entrée.

La dernière couche entièrement connectée permet de classifier l'image en entrée du réseau : elle renvoie un vecteur de taille N, où N est le nombre de classes dans notre problème de classification d'images. Chaque élément du vecteur indique la probabilité pour l'image en entrée d'appartenir à une classe.

4.3.3. Architecture générale de notre CNN pour la classification audio

Nous avons cité dans les sections précédentes que les CNN ont d'excellent résultats pour la classification des images, donc pour prendre un avantage maximal de ces classifieurs, nous pouvons transformer nos vecteurs acoustiques et les façonner sous forme d'image, nous avons sélectionnées 193 composants acoustiques sur la base Librosa qui seront pris comme paramètres d'entrée de notre modèle CNN.

La figure 27 représente l'architecture générale minimisée de notre modèle CNN.

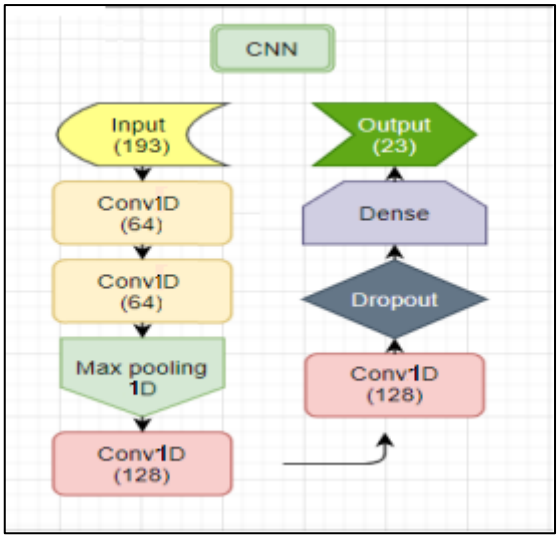


Figure 27 représentation graphique minimisée du réseau de neurone utilisé.

4.4. L'apprentissage par transfert

Le principe de base de l'apprentissage par transfert est simple : prendre un modèle formé sur un grand ensemble de données et transférer ses connaissances vers un plus petit ensemble de données. Pour la reconnaissance d'objets avec CNN, geler les premières couches convolutives du réseau et n'entrainer que les dernières couches qui font une prédiction. L'idée est que les couches convolutives extraient des caractéristiques générales de bas niveau applicables à toutes les images telles que les bords, les motifs, les dégradés et les couches ultérieures identifient les caractéristiques spécifiques d'une image telles que les yeux ou les roues par exemple. Ainsi, nous pouvons utiliser un réseau formé sur des catégories non liées dans un ensemble de données massif comme ImageNet et l'appliquer à notre propre problème car il existe des fonctionnalités universelles de bas niveau partagées entre les images.

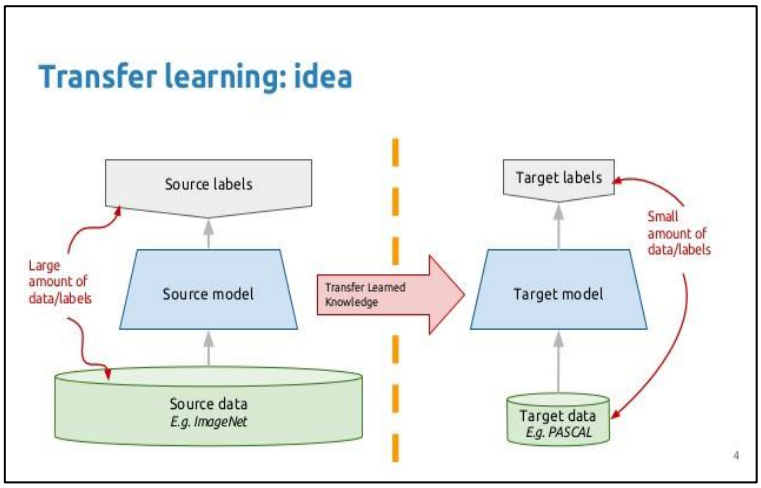


Figure 28 L'idée derrière le Transfer Learning.

Voici le schéma général de l'apprentissage par transfert pour la reconnaissance d'objets :

- Charger dans un modèle CNN pré-entraîné sur un grand ensemble de données
- Geler les paramètres (poids) dans les couches convolutives inférieures du modèle
- Ajouter un classificateur personnalisé avec plusieurs couches de paramètres entraînables pour modéliser

- Entraîner les couches de classificateur sur les données d'entraînement disponibles pour la tâche
- Ajustez les hyperparamètres et libérez plus de couches au besoin.

4.4.1. Architecture du Modèle Pré-entrainé VGG-16

Nous avons utilisé ce modèle pour la classification des spectrogrammes pour ses excellentes performances en classification d'images.

VGG-16 est un modèle de réseau neuronal convolutif proposé par K. Simonyan et A. Zisserman de l'Université d'Oxford [13].Le modèle atteint une précision de test de 92,7% dans le top 5 dans ImageNet, qui est un ensemble de données de plus de 14 millions d'images appartenant à 1000 classes. Il apporte une amélioration par rapport à AlexNet en remplaçant les grands filtres de la taille du noyau (11 et 5 dans la première et la deuxième couche convolutive, respectivement) par plusieurs filtres de la taille du noyau 3 × 3 les uns après les autres. VGG16 a été entraîné pendant des semaines et utilisait les GPU NVIDIA Titan Black.

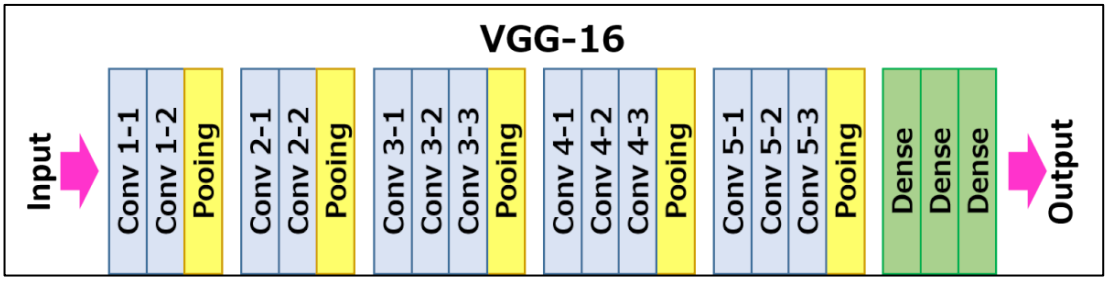


Figure 29 Architecture de VGG-16.

Les modèles pré-entraînés ImageNet sont souvent de bons choix pour l'apprentissage par transfert de vision par ordinateur, car ils ont appris à classer différents types d'images. Ce qui signifie qu'ils ont appris à détecter de nombreux types de caractéristiques différentes qui pourraient être utiles pour la reconnaissance d'images. C'est pour cette raison que nous avons choisis d'utiliser ce modèle dans notre classification basée sur spectrogrammes.

5. Modélisation de l'application

5.1. Diagramme de cas d'utilisation

Le diagramme des cas d'utilisation (Use Case Diagram) constitue la première étape de l'analyse UML en : Modélisant les besoins des utilisateurs, identifiant les grandes fonctionnalités et les limites du système et représenter les interactions entre le système et ses utilisateurs.[2]

Le diagramme suivant représente les fonctionnalités et les interactions entre le système et l'utilisateur de l'application web d'identification des dialectes.

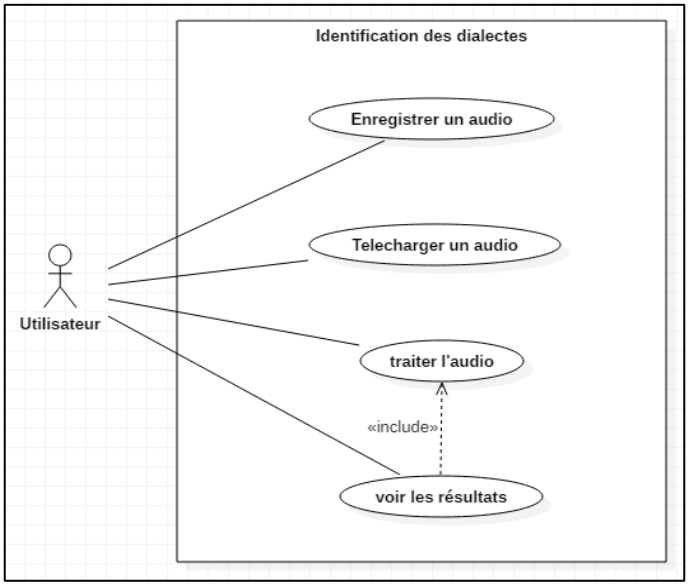


Figure 30 Diagramme des cas d'utilisation de l'application finale.

La figure 30 est un diagramme des cas d'utilisation représentant les fonctionnalités de l'utilisateur dans l'application finale.

Les modèles implémenter dans l'application sont les modèles de la classification des paramètres acoustiques. Les modèles de classification des spectrogrammes n'ont pas été implémenté dans l'application à cause du problème de version de TensorFlow qui est différent 2.6 dans collab et 2.5 dans nos machines.

[2] http://remy-manu.no-ip.biz/UML/Cours/coursUML2.pdf

Les diagrammes représentés dans cette section représentent les diagrammes de la 1ere approches utilisée.

Cas	Description
Enregistrer un audio	L'utilisateur peut enregistrer un audio et arrêter l'enregistrement après la fin de l'enregistrement.
Télécharger un audio	L'utilisateur peut choisir un audio à partir du disque dur.
Traiter l'audio	L'utilisateur clique sur le bouton Go pour traiter l'audio avec les modèles implémenter.
Voir les résultats	Après le traitement d'audio, une page de résultats s'affiche avec le dialecte prédit.

Tableau 13 Description du diagramme de cas d'utilisation.

Le tableau 13 est une description des cas d'utilisation représenté dans le diagramme.

5.2. Diagramme de Séquence

Le diagramme de séquence est un diagramme d'interaction qui présente les échanges de messages entres les acteurs et le système selon un ordre chronologique.

Le diagramme suivant est le diagramme de séquence qui va représenter les différents cas d'utilisation et leurs méthodes.

1. Enregistrer () : après la demande d'autorisation du microphone l'utilisateur enregistre un record vocal.
2. Télécharger () : l'utilisateur choisit un **fichier.wav** déjà enregistrer dans la machine.
3. Ecouter audio () : Donne la possibilité à l'utilisateur d'écouter l'enregistrement choix ou enregistrer.
4. Ext.characteritics () : l'utilisateur commence le traitement en appuyant sur un bouton qui déclenche l'extraction des caractéristiques acoustiques.
5. Prédiction () : Introduire les caractéristiques à un model x pour 12 différents modelés.
6. Clac. Probabilité () : Calcule la probabilité et classe par ordre croissant les résultats de la prédiction.
7. Affichage résultats () : affiche les top 5 résultats de la prédiction pour chaque modèle.

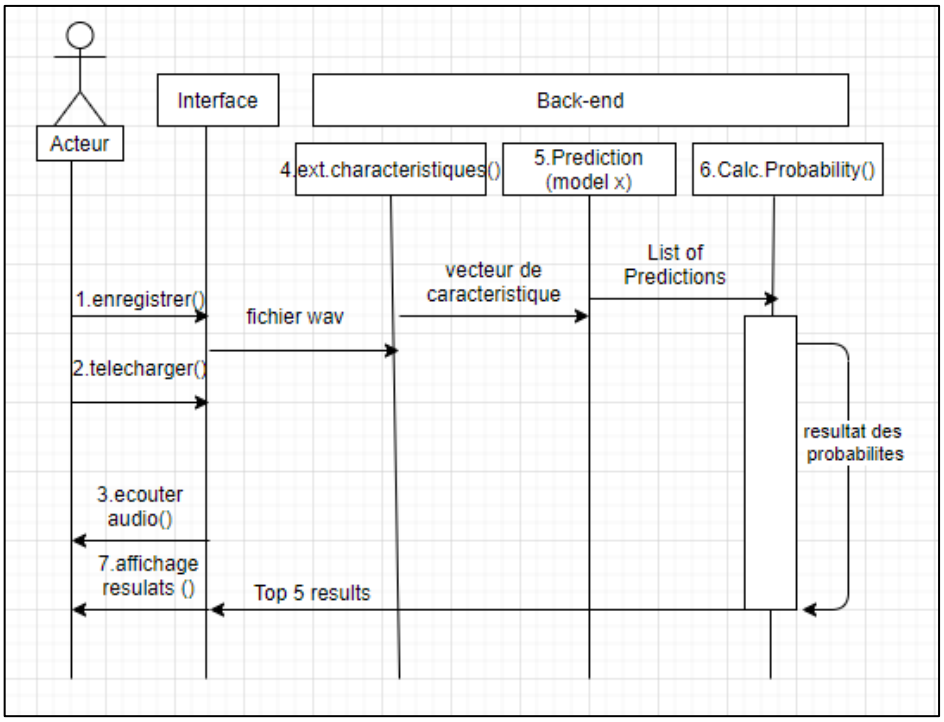


Figure 31 Diagramme de Séquence de l'application finale.

6. Evaluation

Le comportement de chaque modèle de classification est évalué à partir de certains paramètres permettant de mesurer son efficacité.

Les performances du modèle sont influencées par la taille des données d'apprentissage, la qualité des enregistrements vocaux et de manière plus significative, le type de modèle utilisé.

Il existe plusieurs mesures de performance pour évaluer les modèles de classification. Nous avons choisi les matrices de mesure suivantes qu'on va calculer sur 23 classes pour évaluer l'efficacité des modèles de notre classification multi-classe

6.1. Matrice de confusion

En apprentissage automatique supervisé, la matrice de confusion est une matrice qui mesure la qualité d'un système de classification. Chaque ligne correspond à une classe réelle, chaque colonne correspond à une classe prédite. Un des intérêts de la matrice de confusion est qu'elle montre rapidement si un système de classification parvient à

classifier correctement. La matrice de confusion de la figure 32 est une matrice qui représente la classification binaire.

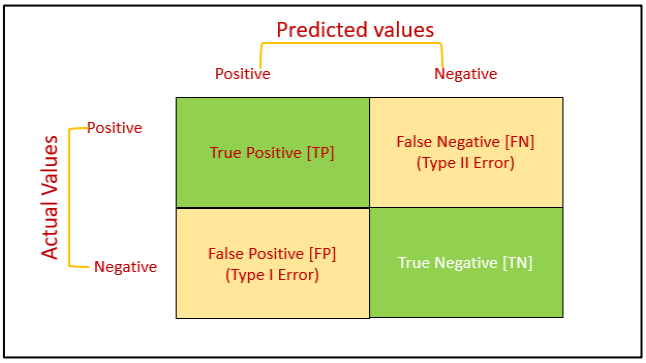


Figure 32 Matrice de Confusion.

Dans notre travail, nous avons un système de classification multi-classe, la matrice de confusion multi-classe est n*n, nous aurons n lignes pour chaque classe réelle et n colonnes correspondantes à chaque classe prédite. Donc 23 lignes pour les dialectes et 23 colonnes correspondantes aux dialectes prédits.

Nous présentons dans la figure suivante la matrice de confusion multi-classe.

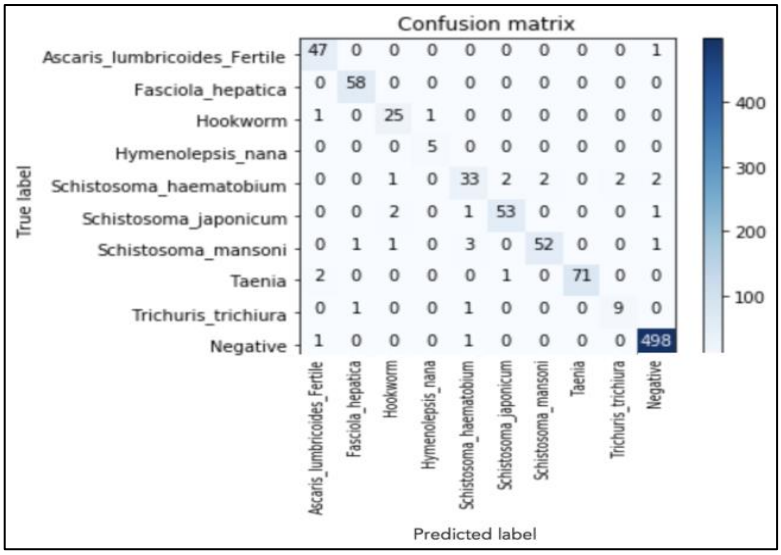


Figure 33 Matrice de confusion multi-classe.

L'axe y représente les classes réelles, et l'axe x les classes prédites. Il y a un dégradé de couleur de 0 à 1 qui va du plus clair (0) vers le plus foncé (1). Plus la couleur vire vers le foncé meilleur sont les

résultats, quand la couleur est claire c'est une confusion dans la classification (une classe n'a pas été prédite correctement).

6.2. Accuracy

Il montre à quelle fréquence le classificateur prédit les valeurs correctes et peut être calculé comme suit :

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$$

6.3. Précision

La précision est le rapport entre les prévisions positives correctes et les prévisions positives totales. Sur tous les positifs prédits, combien sont réellement positifs. La précision peut être calculée comme suit :

$$\text{Précision} = \frac{TP}{TP+FP}$$

Par exemple dans notre cas, après l'entrainement du modèle nous passons à son évaluation et nous prédissions les dialectes en utilisant le modèle entrainé. Le dialecte prédit positifs donne un taux de précision élevé. Ce qui signifie que ce dialecte même a été correctement prédit.

6.4. Rappel

Le rappel est une mesure du nombre de points positifs que votre modèle est capable de rappeler à partir des données. Sur tous les enregistrements positifs, combien d'enregistrements sont prédits correctement.

$$\text{Rappel} = \frac{TP}{TP+FN}$$

6.5. F-mesure

C'est la moyenne harmonique de la précision et du rappel. Il peut être exprimé mathématiquement comme ci-dessous :

$$\text{F-mesure} = \frac{2 * Precision * Rappel}{Precision + Rappel}$$

Où :

- ***TP*** est le nombre d'échantillons prédits positifs qui sont réellement positifs.
- ***FP*** est le nombre d'échantillons prédits comme positifs qui sont en réalité négatifs.
- ***TN*** est le nombre d'échantillons prédits comme négatifs qui sont réellement négatifs.
- ***FN*** est le nombre d'échantillons prédits comme négatifs qui sont réellement positifs.

7. Conclusion

Dans ce chapitre nous avons vu notre approche globale pour développer un modèle de détection du dialecte algérien, les algorithmes de classification, ainsi que les mesures d'évaluations.

Dans le prochain chapitre nous allons voir l'implémentation de notre solution, les modèles utilisés et les résultats obtenus ainsi que leur analyse. Finalement, nous allons présenter l'application web finale de notre système d'identification et une conclusion générale.

Chapitre IV : Implémentation de la solution

1 Introduction

Dans le chapitre précédent nous avons présenté les techniques utilisées pour l'implémentation de notre solution. Dans ce chapitre nous allons nous baser sur le côté pratique de l'implémentation en présentons des fragments de code pour illustrer chaque étape du schéma global de la modélisation.

Nous allons d'abord présenter notre environnement de travail, puis nous expliquerons les étapes des deux approches vues dans le chapitre précédent. Ensuite, nous résumerons les résultats d'évaluations de nos méthodes avec une analyse comparative. Finalement, nous présenterons l'application web finale avec des tests Lives.

2. Environnement de travail

Cette section sera dédiée à la mention des ressources matérielles et logicielles utilisées afin d'atteindre l'objectif initial du projet.

2.1. Ressources matérielles

Pour le développement complet de notre système, nous avons exploité nos machines personnelles dont les spécifications sont les suivantes :

	Machine 01	**Machine 02**	**Machine 03**
Processeur	Intel(R) Core (TM) i5-9400F CPU @ 2.90GHz 2.90 GHz	Intel(R) Core (TM) i7-4510U CPU @ 2.00GHz 2.60 GHz	Intel® Core ™ i3-6006U CPU @ 2.00GHz
Ram	16Gb	6Gb	4Gb

Tableau 14 Ressources Matérielle.

2.2. Ressources logicielles

- **Python 3.7.0 :** initialement développé en 1969 par Guide Van Rossem.

Python est l'un des langages de programmation les plus populaires utilisés par les développeurs aujourd'hui, et c'est le langage le plus utilisé dans le domaine de l'intelligence artificielle. Grâce à ses bibliothèques spécialisées, ce langage peut être utilisé dans divers contextes et s'accommoder à tout type d'application.[3]

Figure 34 Python logo.

- **TensorFlow 2.5.0 :** est une plate-forme end-to-end open source pour l'apprentissage automatique. Il dispose d'un écosystème d'outils complet et flexible, de bibliothèques et de ressources communautaires qui permet aux chercheurs de créer et de déployer facilement des applications basées sur le ML.[4]

Figure 35 TensorFlow logo.

[3] https://www.python.org/
[4] https://www.tensorflow.org/api_docs?hl=nb

- **Scikit-learn / Sklearn 0.24.2 :** Sklearn est probablement la bibliothèque la plus utile pour l'apprentissage automatique en Python. La bibliothèque Sklearn contient de nombreux outils efficaces pour l'apprentissage automatique et la modélisation statistique, notamment la classification, la régression, le clustering et la réduction de la dimensionnalité.[5]

Figure 36 Sckit-learn logo.

- **Librosa 0.8.1 :** est un package python pour l'analyse musicale et audio. Il fournit les blocs de construction nécessaires pour créer des systèmes de recherche d'informations musicales.

  Librosa aide à visualiser les signaux audios et à en extraire les caractéristiques à l'aide de différentes techniques de traitement du signal.[6]

Figure 37 Librosa logo.

- **Visual Code studio :** Visual Studio Code est un éditeur de code source léger mais puissant qui s'exécute sur votre bureau et est disponible pour Windows, MacOs et Linux. Il est livré avec une prise en charge intégrée de JavaScript et Node.js et dispose d'un riche écosystème d'extensions pour d'autres langages (tels que C++, C#, Java, Python, PHP, Go) et runtimes (tels que .NET et Unity).[7]

Figure 38 Visual code studio logo.

[5] https://scikit-learn.org/stable/
[6] https://librosa.org/
[7] https://code.visualstudio.com

- **Google Colaboratory :** Colaboratory, souvent raccourci en "Colab", est un produit de Google Research. Colab permet à n'importe qui d'écrire et d'exécuter le code Python de son choix par le biais du navigateur. C'est un environnement particulièrement adapté à l'Apprentissage Automatique, à l'analyse de données et à l'éducation. En termes plus techniques, Colab est un service hébergé de notebooks Jupyter qui ne nécessite aucune configuration et permet d'accéder gratuitement à des ressources informatiques, dont des GPU.[8]

Figure 39 Google colab logo.

- **Flask :** est un Framework d'applications Web WSGI (Web Server Gateway Interface) léger. Il est conçu pour rendre le démarrage rapide et facile, avec la possibilité d'évoluer vers des applications complexes.[9]

Figure 40 Flask logo.

[8] https://colab.research.google.com/notebooks/intro.ipynb
[9] https://flask.palletsprojects.com/

3. Classification et Evaluation des modèles

3.1. Classification basée sur les paramètres acoustiques

3.1.1. Préparation des données d'entrées

Après la collection des échenillions, On les a convertis en format **« .wav »** a l'aide de la fonction suivante :

- ffmpeg 16000 video.mp4 audio.wav

-16000 représente une fréquence d'échantillonnage audio.

a. Prétraitement

Dans le prétraitement nous avons utilisé l'algorithme de "NoiseReduce" pour la suppression du bruit avec la fonction suivante [9]:

```
def noisereduce(path,filename):
    try:
        data ,rate = librosa.load(path+'\\'+filename)
        reduced_noise = nr.reduce_noise(y=data, sr=rate)
        savefile=path+'\\'+filename
        sf.write(savefile, reduced_noise, rate)
    except:
        print('Error returned')
```

Figure 41 La fonction NoiseReduce pour le prétraitement.

b. Segmentation

C'est le partitionnement de donnée en segment de même taille, pour garder un coefficient similaire pour chaque catégorie de donnée.

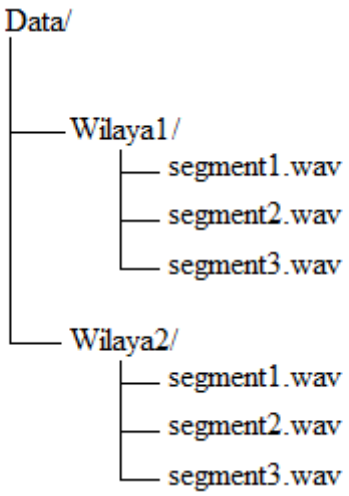
Dans notre travail nous avons partitionné nos données vocales en segments de trois tailles différentes 5s, 10s et 20s, qui seront après les données d'entre pour trois modelés différents, pour faire la comparaison entre trois cas d'études différents.

```
def splitPart(self, Start, End, split_filename):
    t1 = Start  * 1000
    t2 = End  * 1000
    split_audio = self.audio[t1:t2]
    filename=split_filename+'['+str(int(Start))+'-'+str(int(End))+']'+'.wav'
    split_audio.export(self.Destination + '\\' + filename, format="wav")
```

Figure 42 Fonction de segmentation.

3.1.2. Extraction des caractéristiques

Avant de commencer l'extraction de caractéristique il est très important d'organiser nos fichiers dans la structure suivante :



Pour que les caractéristiques soient étiquetées correctement par les noms des dossiers de chaque wilaya.

Pour extraire les caractéristiques mentionnées dans le chapitre 3 nous utilisons les fonctions prédéfinis dans la bibliothèque **Librosa**:

```
stft=librosa.stft(X)
mfccs=librosa.feature.mfcc(y=X, sr=sample_rate, n_mfcc=40).T, axis=0
chroma=librosa.feature.chroma_stft(S=stft, sr=sample_rate).T, axis=0
melspectrogram=librosa.feature.melspectrogram(X, sr=sample_rate).T, axis=0
spectral_contrast=librosa.feature.spectral_contrast(S=stft, sr=sample_rate).T, axis=0
tonnetz=librosa.feature.tonnetz(y=librosa.effects.harmonic(X), sr=sample_rate).T, axis=0
```

Figure 43 Les fonctions d'extraction des caractéristiques acoustiques.

Ces caractéristiques seront alors empilées verticalement dans un vecteur numérique, et les étiquettes dans un autre, ce processus s'exécute pour chaque segment de notre corpus.

A la fin du processus d'extraction des caractéristiques les deux vecteurs seront enregistrés dans deux fichiers pour utilisation future :

« Features.npy » pour les caractéristiques et « Labels.npy » pour les étiquettes (Nom des wilaya).

3.1.3. Partitionnement des données d'apprentissage et de test

Après la phase de l'extraction des caractéristiques, nous partitionnons nos données, en données d'apprentissage et donnée de test 80%, 20% respectivement en utilisant la fonction suivante :

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=20)
```

Figure 44 Fonction de partitionnement des données.

- X : Features.npy
- Y : Labels.npy
- Random_state : c'est un argument qui contrôle l'assemblement des données avant d'appliquer la partition.

3.1.4. Initialisation des modèles

a. Model Support vecteur machine (SVM)

Pour la création de notre modèle initial SVM nous avons utilisé la bibliothèque **Sklearn.svm** avec la fonction :

```
svm_clf = SVC(C=1, decision_function_shape="ovo",kernel='rbf',probability=True)
```

Figure 45 Fonction de création du modèle initial SVM.

- C : c'est un paramètre de régularisation qui prend la valeur 1 par défaut.
- Decision_function_shape : c'est une fonction de décision qui prend la valeur {'ovo', 'ovr'} (one vs one, one vs rest), ovo est toujours utiliser comme stratégie pour multi-class.
- Kernel : est une méthode utilisée pour prendre des données en entrée et les transformer en la forme requise de traitement des données, Le type de fonction de noyau le plus préféré est RBF.
- Probability : Définit l'argument de probabilité sur TRUE pour que la prédiction retourne une matrice de probabilité pour chaque classe.

b. Réseau neuronal convolutif (CNN)

Pour le CNN notre model sera initialiser comme suite :

```
model.add(Conv1D(64, 3, activation='relu', input_shape = 193))
model.add(Conv1D(64, 3, activation='relu'))
model.add(MaxPooling1D(3))
model.add(Conv1D(128, 3, activation='relu'))
model.add(Conv1D(128, 3, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(23, activation='softmax'))
```

Figure 46 Initialisation du modèle CNN

- MaxPooling1D : divise par deux la taille des entités en les sous-échantillonnant à la valeur maximale à l'intérieur d'une fenêtre, Cette couche est la raison pour laquelle un CNN peut gérer les énormes quantités de données dans les images
- Dropout : protège contre le surapprentissage en réglant aléatoirement les poids d'une partie des données à zéro
- Dense : est une couche de réseau neuronal qui est connectée en profondeur, ce qui signifie que chaque neurone de la couche dense reçoit des entrées de tous les neurones de sa couche précédente.

3.1.5. Apprentissage des modèles

Après le long travail de préparation nous somme en fin arrivés à l'apprentissage de nos modèles. Dans notre cas on a 6 modèles de classification à entrainer.

3x SVM pour les données segmenter en (5s, 10s et 20s) et 3x CNN pour les mêmes données.

Pour vérifier si la perte d'information causée par le prétraitement n'a pas un grand impact. Nous avons décidé d'entrainer les mêmes modèles de classification en utilisant des données d'entrées brute c.à.d. sans prétraitement. Ce qui nous laisse à la formation de 12 modèles de classification.

a. Apprentissage de modèle SVM

Après le partitionnement des données en test et train, on commence notre apprentissage par l'exécutons cette fonction :

```
model=svm_clf.fit(X_train, y_train)
```

Figure 47 Fonction pour l'apprentissage du modèle.

Et on le sauvegarde dans notre disc-dur avec la fonction :

```
import pickle
with open(path,"wb") as f:
    pickle.dump(model,f)
```

Figure 48 Fonction pour la sauvegarde du modèle.

b. *Apprentissage de modèle CNN*

On exécute les fonctions suivantes pour façonner nos données sous forme d'image et entrainer notre modèle CNN

```
X_train = np.expand_dims(X_train, axis=2)
X_test = np.expand_dims(X_test, axis=2)
model.fit(X_train, y_train, batch_size=32, epochs=300)
```

Figure 49 Fonction d'entrainement du modèle CNN.

- Batch_size : nombre d'échantillons par mise à jour du gradient.
- Epochs : itération sur l'ensemble de données fournies `x` et` y`qui aide à améliorer les résultats de notre modèle de classification finale.

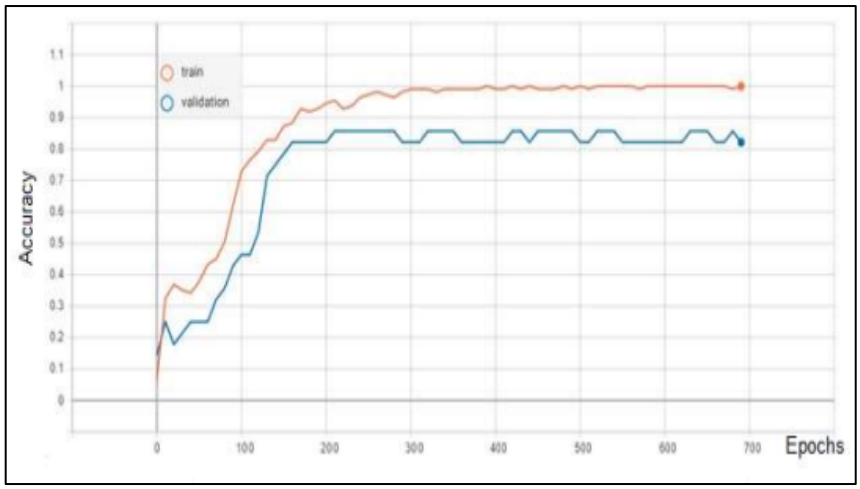


Figure 50 Graphe des résultats d'apprentissage des données en fonction du nombre d'époques.

Les résultats que nous avons obtenus de l'entrainement et de la validation d'un exemple de nos réseaux de neurones sont présentés dans la figure 50.

On remarque que la courbe de développement se redresse environ les 300 epochs que nous avons choisi par expérimentation.

3.1.6. Evaluation et analyse des résultats

Nous sommes arrivés finalement à entrainer nos 12 modèles de classification pour comparer les modèles, nous avons résumer les résultats dans le tableau suivant :

	Mesures de performance	Classifieur					
		5S		10S		20S	
		CNN	SVM	CNN	SVM	CNN	SVM
données prétraitées	Accuracy	0.9529	0.9036	0.9648	0.917	0.9767	0.9194
	Précision	0.9547	0.9053	0.9656	0.918	0.9774	0.9204
	Rappel	0.9529	0.9036	0.9648	0.917	0.9767	0.9194
	F-mesure	0.9533	0.9037	0.9649	0.9171	0.9769	0.9192
	Mesures de performance	Classifieur					
		5S		10S		20S	
		CNN	SVM	CNN	SVM	CNN	SVM
données brutes	Accuracy	0.9684	0.9552	0.9627	0.9663	0.9808	0.9577
	Précision	0.9703	0.9568	0.9641	0.9673	0.9815	0.9598
	Rappel	0.9684	0.9552	0.9627	0.9663	0.9808	0.9577
	F-mesure	0.9689	0.9555	0.9631	0.9665	0.9809	0.9579

Tableau 15 Mesures de performances de la classification des paramètres acoustiques.

a. Analyse des résultats de la classification acoustique

Pour les segments de 5s : Nous remarquons que les résultats des données brutes pour les deux algorithmes sont meilleurs que pour les données prétraitées.

Pour les segments de 10s : Nous remarquons que les résultats des données prétraitées avec le modèle CNN sont légèrement meilleurs que ceux des données brutes. Par contre, le modèle SVM avec les données brutes dépassent les résultats des données pré-entrainées de 5%.

Pour les segments de 20s : Les résultats de données brutes sont meilleurs que ceux des données pré-entrainées.

Nous constatons une légère diminution des performances pour les données pré-entrainées ce qui est dû à la perte d'informations lors des suppressions du bruit.

3.2. Classification basée sur les spectrogrammes

Corpus :

Le corpus utilisé est le même que celui que nous avons cité dans le chapitre 03. Nous prenons les audios segmentés et nous les convertissons en spectrogrammes.

3.2.1. Préparation des données

Après avoir segmenté les audios en segments de 5s,10s et 20s, nous convertissons chaque data en spectrogramme en utilisant la fonction suivante :

```
def graph_spectrogram(wav_file):
    filename = wav_file.split('\\')[7][:-4]
    wilaya = wav_file.split('\\')[6]
    sound_info, frame_rate = get_wav_info(wav_file)
    pylab.figure(num=None, figsize=(10, 6))
    pylab.subplot(111)
    pylab.title('spectrogram of %r' % wav_file)
    pylab.specgram(sound_info, Fs=frame_rate, cmap='Spectral_r')
    pylab.savefig(chemin+"\\"+filename+'.png')
    pylab.close()
```

Figure 51 Fonction de conversion d'audio vers spectrogramme.

Cette fonction convertit les audios en spectrogramme, en réglant les paramètres de la taille de l'image et le nom. Puis, elle sauvegarde les spectrogrammes dans le chemin donné. Le résultat de quelques spectrogrammes des fichiers de 5s est dans la figure suivante :

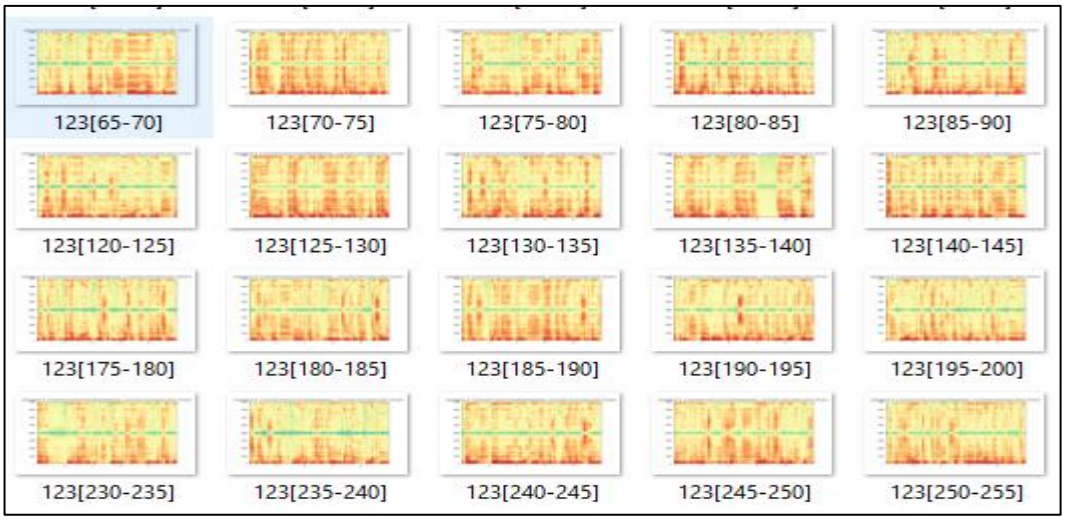


Figure 52 Spectrogramme obtenue après conversion des audios vers spectrogramme.

Après avoir converti les audios en spectrogramme, nous les importons dans drive pour entrainer notre classifieur sur cet ensemble de donnée et évaluer le modèle.

Accès aux données

Nous accédons aux données de notre drive pour importer les données de spectrogramme durée 5s,10s ou 20s avec la fonction suivante :

```
from google.colab import drive
drive.mount('/gdrive')
%cd  /gdrive

Mounted at /gdrive
/gdrive
```

Figure 53 Fonction pour accéder aux données dans drive.

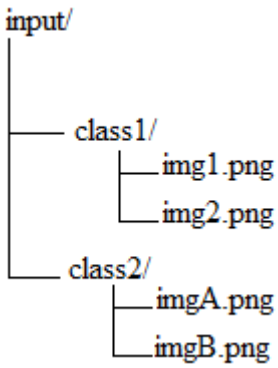
En utilisant cette fonction on a accès aux données de drive.

3.2.2. Partitionnement des données d'apprentissage et de test

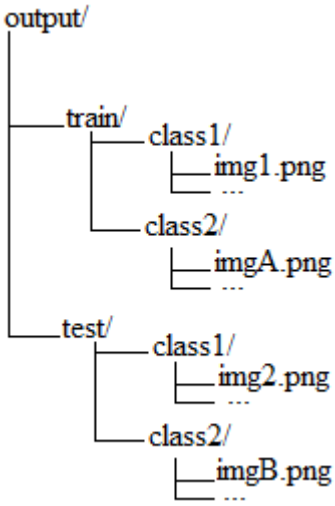
Installer le package split-folder qui partitionne les dossiers contenant des fichiers (dans notre cas des images) en dossiers de **Train**, de **validation** et de **test** (ensemble de données).

En utilisant la commande suivante : **pip install split-folders**

Le dossier de donnée doit avoir le format suivant :



Après l'utilisation de la fonction de partitionnement, nous obtenons les dossiers sous le format suivant :



Pour faire le split on utilise la fonction suivante :

```
[ ] splitfolders.ratio("/gdrive/MyDrive/Spectrogram_20s", output="/gdrive/MyDrive/Data", seed=1337, ratio=(.8, .2), group_prefix=None)
```

Figure 54 Fonction de partitionnement des données spectrogrammes.

Pour partitionner les données en donnée de train et validation on utilise ratio en définissant un tuple ratio (.8, .2) qui veut dire qu'on divise en 80% pour les données de Train et 20% pour les données de test.

- On définit le chemin des données Input pour les données d'entrée dans notre cas les données des spectrogrammes, et Output les données de sortie Data ou vont s'enregistrer les données de Train et validation.

- Seed : Définit la valeur de départ pour mélanger les éléments, la valeur par défaut est de 1337.
- Ratio : Le ratio pour le partitionnement en Train et test dans notre cas (.8, .2)
- Group_prefix : diviser les fichiers en groupes de taille égale en fonction de leur préfixe dans notre cas il est paramétrisé à None

3.2.3. Entrainement des données

La fonction keras **ImageDataGenerator.flow_from_directory ()** sera l'interface entre le jeu de donnée sur le disque et la boucle d'entrainement, elle permet de :

-Obtenir des images par lots : Batch 1, batch2, batch n

-Appliquer des opérations de prétraitement sur les images

-augmentation des images

a. Le prétraitement des images

Le prétraitement des images est fait comme suit :

```
batch_size = 32
image_size = (224,224)
```

Figure 55 Paramètres de prétraitement des images.

- Batch_size : Taille des lots de données, par défaut 32.
- Image_size : Taille avec laquelle on redimensionne les images après leur lecture à partir du disque. La valeur par défaut est (256,256) dans notre Travail on l'a défini à (224,224). Etant donné que le pipeline traite des lots d'images qui doivent toute avoir la même taille, cela doit être fournit.

Après cette étape on crée un générateur de donnée de train et validation et on fait une augmentation des images en utilisant les fonctions suivantes :

b. Générateur des données

La classe ImageDataGenerator est très utile dans la classification d'images. Il existe plusieurs façons d'utiliser ce générateur, selon la méthode que nous utilisons, ici nous allons nous concentrer sur flow_from_directory qui prend un chemin vers le répertoire

contenant les images triées dans des sous-répertoires et des paramètres d'augmentation d'image.

```
validation_datagen = ImageDataGenerator(rescale=1./255)

validation_generator = train_datagen.flow_from_directory(
                  '/gdrive/MyDrive/Data/val',
                  target_size=image_size,
                  batch_size=2*batch_size,
                  shuffle=False,
                  )

Found 1994 images belonging to 23 classes.
```

Figure 56 Fonction de génération des données de validation.

Nous avons un générateur de donnée avec une augmentation d'image pour les données de Train et de validation.

Dans la figure 48 nous présentons un générateur pour les données de validation, c'est la même fonction pour les données de train en changeant validation par train et les paramètres Batch_size = Batch_size pour train et le paramètre Shuffle n'est pas ajouté.

Les paramètres de flow_from_directory () :

- **Chemin** : chaîne, chemin d'accès au répertoire cible. Il doit contenir un sous-répertoire par classe. Toutes les images PNG, JPG, BMP, PPM ou TIF à l'intérieur de chacun des sous-répertoires de l'arborescence seront incluses dans le générateur (Dans notre cas le répertoire des donnée train et validation)
- **Target_size** : Tuple d'entiers, par défaut. Les dimensions auxquelles toutes les images trouvées seront redimensionnées. (Height, width)
- **Shuffle :** s'il faut mélanger les données (par défaut : true), si défini sur false, trie les données dans l'ordre alphanumérique.

3.2.4. Classification des spectrogrammes

a. Construction du modèle

On construit le modèle en utilisant les données de Train, comme le montre la fonction suivante :

```
class_number = train_generator.num_classes
```

Figure 57 Fonction de construction du modèle VGG16.

Cette fonction va nous donner le nombre de classe qu'on a dans nos données de training.

b. Téléchargement du modèle pré-entrainé VGG16

Commençons par télécharger le modèle pré-entraîné. Au fur et à mesure que nous téléchargeons, il va y avoir une différence importante. La dernière couche d'un modèle ImageNet est une couche dense de 1000 unités, représentant les 1000 classes possibles dans l'ensemble de données. Dans notre cas, nous souhaitons qu'il fasse un classement différent : 23 classes pour les 23 dialectes.

Nous allons supprimer la dernière couche du modèle. Nous pouvons le faire en définissant le paramètre (include_top=False) lors du téléchargement du modèle. Après avoir supprimé cette couche supérieure, nous pouvons ajouter de nouvelles couches qui donneront le type de classification que nous voulons :

La fonction du téléchargement du modèle et sa paramétrisation est démontré dans le fragment de code suivant :

```
base_model = VGG16(include_top =False,input_shape = (224,224,3))
```

Figure 58 Fonction de téléchargement du Modèle pré-entrainé VGG-16.

- **Input_shape = (224,224,3)** : spécifie la forme préférée des images dans notre jeu de données

c. Ajout de nouvelles couches

Nous pouvons maintenant ajouter les nouvelles couches pouvant être entraînées au modèle pré-entraîné. Ils prendront les caractéristiques des couches pré-entraînées et les transformeront en prédictions sur le nouveau jeu de données. Nous allons ajouter deux couches au modèle. Le premier sera une couche de mise en commun comme nous l'avons vu dans notre précédent réseau de neurones convolutifs. Nous devons ensuite

ajouter notre dernière couche, qui classera les dialectes. Ce sera une couche densément connectée avec une sortie. Comme il est présenté dans le fragment de code qui suit :

```
x = base_model.output
x = GlobalAveragePooling2D()(x)
x = Dropout(0.3)(x)
x = Dense(256, activation='relu')(x)
predictions = Dense(class_number, activation='softmax')(x)
model = Model(base_model.input, predictions)
```

Figure 59 Ajout des couches dans le modèle pré-entrainé.

- GlobalAveragePooling2D () : Ajouter une couche entièrement connectée
- Dropout : protège contre le surapprentissage en réglant aléatoirement les poids d'une partie des données à zéro
- Dense : est une couche de réseau neuronal qui est connectée en profondeur, ce qui signifie que chaque neurone de la couche dense reçoit des entrées de tous les neurones de sa couche précédente.
- Class_number : le nombre de classe.

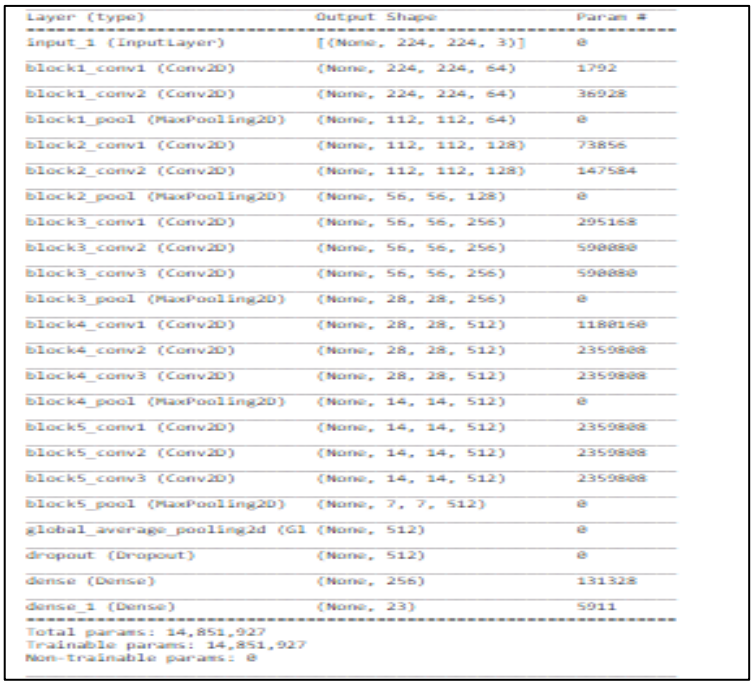
- Activation = ‘softmax’ : La couche soft max produira une valeur comprise entre 0 et 1 en fonction de la confiance du modèle à laquelle la classe appartient aux images.
- Après à la fin notre modèle est enfin préparé.

  On résume le contenu du modèle crée en utilisant la fonction suivante :

```
print(model.summary())
```

Figure 60 Fonction pour afficher les couches du modèle crée.

Le modèle s’affiche comme suit :



Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 224, 224, 3)]	0
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1792
block1_conv2 (Conv2D)	(None, 224, 224, 64)	36928
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0
block2_conv1 (Conv2D)	(None, 112, 112, 128)	73856
block2_conv2 (Conv2D)	(None, 112, 112, 128)	147584
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295168
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590080
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590080
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0
block4_conv1 (Conv2D)	(None, 28, 28, 512)	1180160
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2359808
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2359808
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2359808
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0
global_average_pooling2d (Gl	(None, 512)	0
dropout (Dropout)	(None, 512)	0
dense (Dense)	(None, 256)	131328
dense_1 (Dense)	(None, 23)	5911

Total params: 14,851,927
Trainable params: 14,851,927
Non-trainable params: 0

Figure 61 Affichage des couches du modèle de durée 20s.

d. Compilation du modèle

On compile le modèle avec les mesures de taux d'erreur et d'accuracy comme suit :

```
] model.compile(optimizer=optimizers.SGD(learning_rate=1e-3, momentum=0.9),
                loss='categorical_crossentropy' ,
                metrics = ['accuracy']
                )
```

Figure 62 Compilation du modèle.

Les paramètres de cette fonction :

- *Optimizers.SGD* : Optimiseur de descente de gradient
- *Learning rate* : Un Tensor, une valeur à virgule flottante ou une planification qui est un tf. keras.optimizer , ou un appelable qui ne prend aucun argument et renvoie la valeur réelle à utiliser. Le taux d'apprentissage. La valeur par défaut est 0,01.
- *Momentum* : hyperparamètre float >= 0 qui accélère la descente du gradient dans la direction appropriée et atténue les oscillations. La valeur par défaut est 0, c'est-à-dire la descente de gradient vanille.

- *Loss = 'categorical_crossentropy'* : une fonction de perte qui calcule la quantité qu'un modèle doit chercher à minimiser pour l'apprentissage d'un modèle.

e. Apprentissage du modèle

Après avoir préparé les données de training, de validation ainsi que notre modèle. On passe à l'apprentissage du modèle avec notre donnée de training et les vérifier en utilisant les données de validation.

```
nbr_epochs = 20
earlystopping = callbacks.EarlyStopping(monitor= "val_loss",
                                        mode ="min",
                                        patience = 5,
                                        restore_best_weights = True)
history=model.fit(train_generator,
          epochs=nbr_epochs,
          validation_data=validation_generator,
          callbacks =[earlystopping]
)
```

Figure 63 Apprentissage du modèle VGG-16 sur les données de spectrogramme.

-Toute ces étapes sont exécutées sur les données de spectrogrammes de durée 5s, 10s et 20s

Entrainer un model avec des données est aussi appelé ' ***fit a model*** 'en utilisant la fonction model. Fit qui prend en paramètre :

- Train_generator : les données de Train générées pour entrainer le modèle
- Epochs : Le nombre d'époque est le nombre d'itération faite sur les données d'entrainement et de validation. Nous l'avons défini comme suit pour les données des 3 segments :
    - Epoque = 20 pour les segments de 20s
    - Epoque = 5 pour les segments de 10s
    - Epoque = 3 pour les segments de 5 s

Epoque permet au modèle de mieux s'entrainer, et d'avoir de meilleurs résultats. Plus le nombre d'époques est grand meilleur sont les résultats du modèle entrainé.

Quand le nombre d'époque est trop grand ça peut entrainer un sur-apprentissage(overfitting) comme ça peut entrainer un sous-ajustement (underfitting) si le nombre est trop petit.

- *Validation_generator :* les données de validation pour vérifier notre modèle.
- *Earlystopping* : est l'arrêts précoce qui permet de définir un grand nombre arbitraire d'époques d'entrainement et d'arrêter l'entrainement une fois que les performances du modèle ne s'améliorent plus sur un ensemble de validation suspendu.

f. Enregistrement du modèle entrainé et des résultats

Enregistrement du modèle entrainé :

En utilisant la fonction model. Save :

```
target_dir = '/gdrive/MyDrive/Model_pré_entrainé20s'
model.save(os.path.join(target_dir+'/my_model_20s.h5'))
```

Figure 64 Fonction d'enregistrement du modèle entrainé.

Enregistrement des résultats d'entrainement dans une data frame :

```
history_df = pd.DataFrame(history.history)
history_df.to_csv(os.path.join(target_dir+'/hystory.csv'),index=False)
```

Figure 65 Fonction d'enregistrement des résultats d'entrainement.

Afficher les résultats :

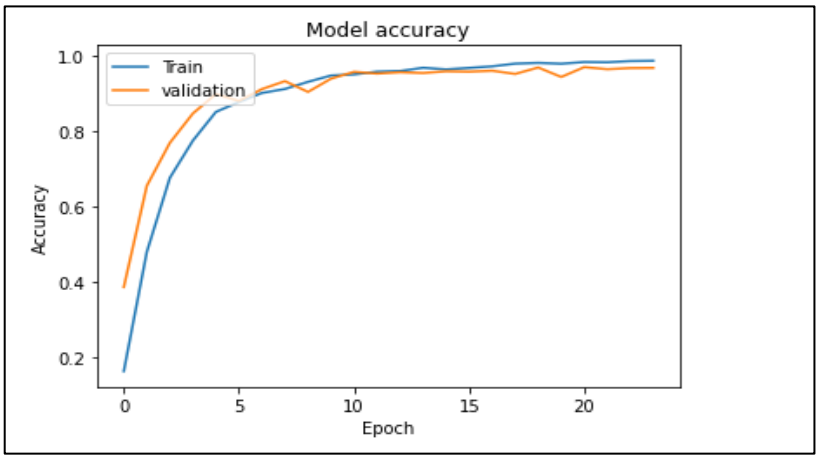


Figure 66 Résultat du Training du modèle 20s.

Ce graphe (figure 66) représente les résultats du training de notre modèle de spectrogramme de durée 20s en fonction du nombre d'époques.

Nous remarquons que la courbe des données de train est en développement continu, ce qui veut dire que le résultats d'accuracy se rapproche vers 90%. Notre modèle a été bien entrainé, et vérifié par les données de validation avec une accuracy de 88%

3.2.5. Evaluation des modèles

Modèle	Mesure de performance	Donnée 5s	Donnée 10s	Donnée 20s
Modèle VGG-16	**Précision**	0.71	0.86	0.90
	Recall	0.57	0.84	0.88
	F1-score	0.54	0.84	0.88
	Accuracy	0.57	0.84	0.88

Tableau 16 Résultat de classification des spectrogrammes.

a. Matrice de confusion

Les matrices de confusions suivantes sont une visualisation des résultats obtenus après la prédiction des données de tests sur les modèles entrainés.

Nous présentons la matrice de confusion des spectrogrammes de durée 10s,20s,5s dans la figure 67, 68, 69 respectivement et une analyse des résultats obtenus de chaque donnée.

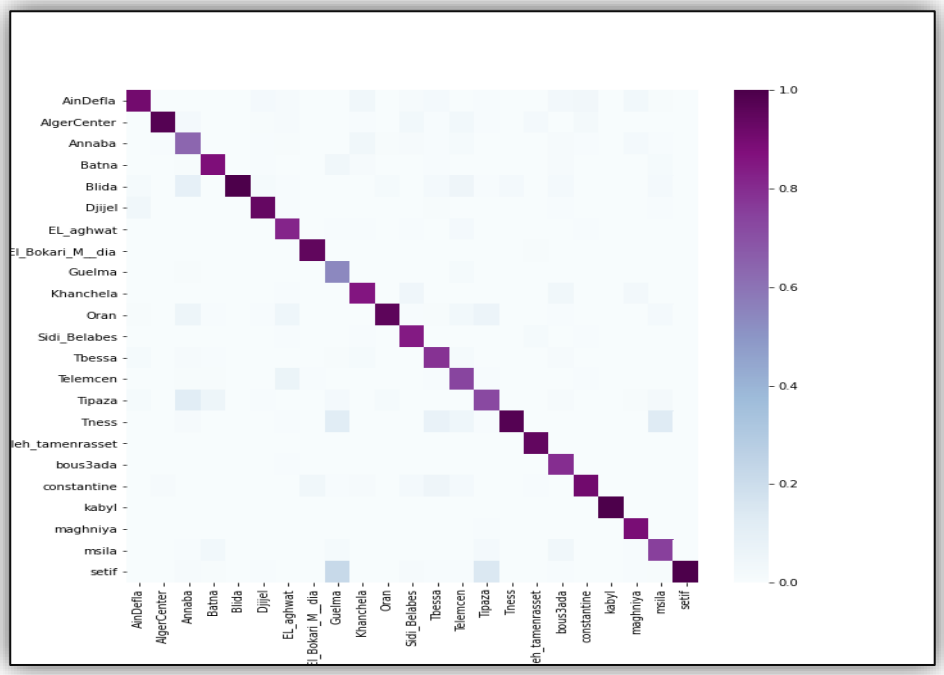


Figure 67 Matrice de Confusion des prédictions des spectrogrammes de durée 10s.

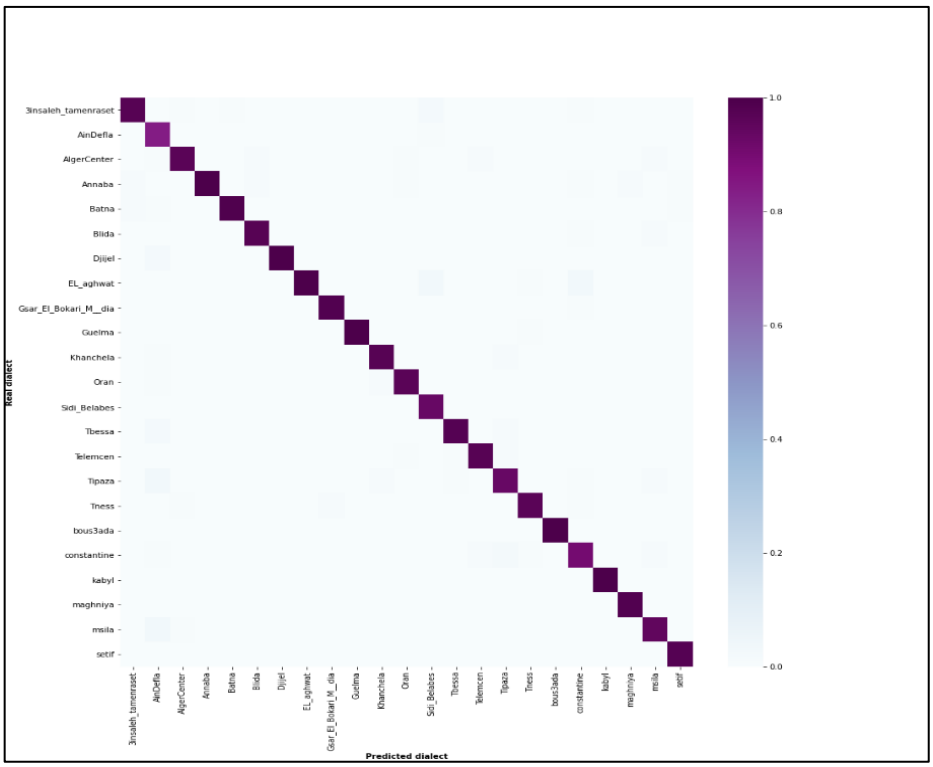


Figure 68 Matrice De Confusion de prédictions des dialectes spectrogrammes de durée de 20s.

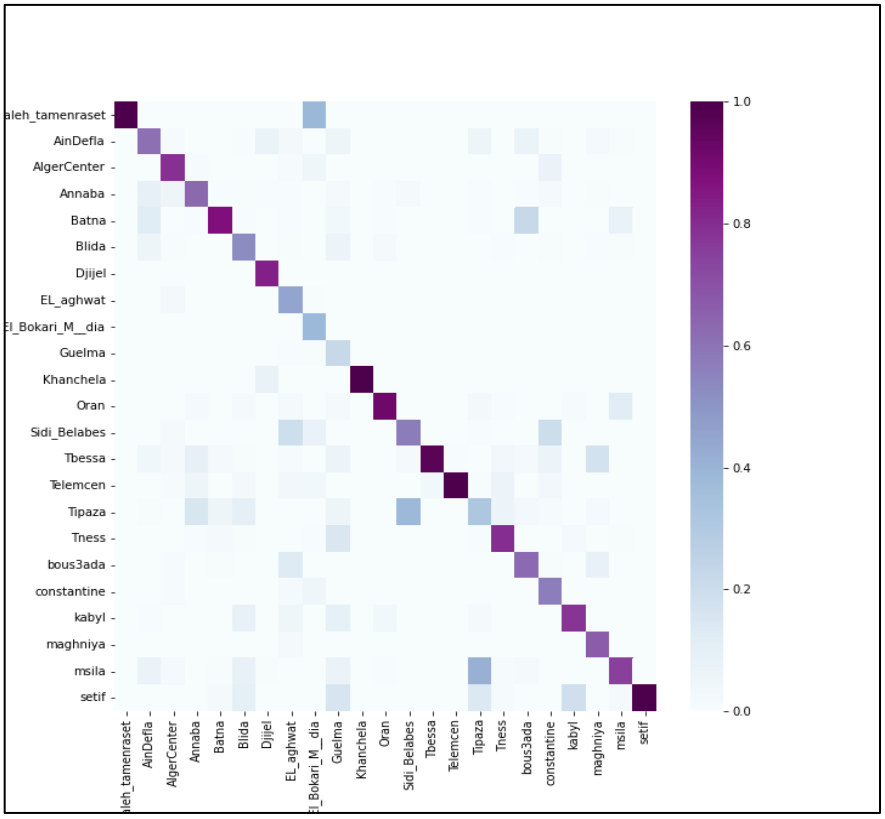


Figure 69 Matrice de Confusion de prédiction des dialectes des spectrogrammes de durée 5s

3.2.6. Analyse des résultats de classification des spectrogrammes

D'après les résultats résumés dans le tableau 16, l'identification des dialectes en utilisant les spectrogrammes donne de meilleurs résultats avec les données de durée 20s. Le taux d'accuracy des dialectes correctement classés est de 88% alors que l'accuracy des données de 10s et 5s est de 84% et 57% respectivement.

Ces résultats obtenus sont raisonnables, car les audios de segments 20s contiennent plus d'information que ceux de 5s et 10s. Donc notre système d'identification est un système performant qui arrive à identifier les dialectes d'après les audios contenant un nombre d'informations suffisant.

Nous concluons, que plus le segment d'audio contient des informations meilleures sont les résultats pour l'approche basée sur spectrogramme car nous remarquons une légère augmentation de performance avec une durée d'audio plus lente.

a. Analyse des Matrices de Confusions

Analyse de la matrice de confusion figure 67 10s :

Dans la matrice de confusion des données de 10s, l'axe x est l'axe des wilayas prédites et l'axe y des wilayas réelles. Nous remarquons que l'axe x représente un dégradé de couleur qui va du clair vers le foncé. Le clair prend la valeur 0 et le plus foncé 1 qui représente la précision de notre prédiction.

Dans cette matrice nous avons pleins de carré qui sont clairs ce qui signifie qu'il y a une confusion dans la classification (un dialecte qui devait être prédit correctement a été prédit en tant qu'un autre dialecte).

On prend un exemple de la matrice, Alger centre a une couleur qui vire vers le noir ce qui signifie que le dialecte a été bien prédit avec une précision de 98%.

Analyse de la matrice de confusion figure 68 20s :

En analysant la matrice de confusion des spectrogrammes de durée 20s, nous remarquons que les classes qui ne sont pas correctement prédite ne sont pas très présente d'après les carrées qui ont une couleur qui donne une précision entre 80% et 95%. Ce qui signifie que les prédictions sont correctes (un dialecte a été correctement prédit). Le taux de confusion de prédiction est de 10% seulement.

Analyse de la Matrice de Confusion figure 69 5s :

Dans la matrice de confusion des données de 5s, nous remarquons que le dégradé des couleurs clairs est présent plus que dans les autres matrices ce qui est équivalant à une confusion dans l'identification des dialectes.

L'accuracy est de 57% c'est des résultats satisfaisant par apport à des audios de 5s qui ne contiennent pas beaucoup d'information pour une bonne identification. Le taux d'erreur de classification des fichiers de 5s est de 43% c'est un pourcentage élevé, ces résultats sont dû au manque de ressource et au manque d'informations dans l'audio.[10]

En résumé, le modèle VGG- 16 pré-entrainé nous a aidé à avoir de très bons résultats de classification. Il est connu pour avoir un taux d'accuracy élever dans la classification d'images. Dans notre travail, on est arrivé à avoir des résultats raisonnable et satisfaisant.

3.3. Synthèse des résultats :

Les résultats obtenus montrent que les performances de l'approche acoustique sont légèrement meilleures que ceux utilisant les spectrogrammes. Ce qui indique que les paramètres acoustiques sont des descripteurs plus fiables pour distinguer entre les dialectes, ils aident à réduire la confusion entre eux et à donner de meilleurs résultats.

Les résultats obtenus sont très bon comparant à ce qui existe dans la littérature, notre contribution avec ce travail est d'avoir travaillé avec 23 dialectes algériens. Les résultats obtenus peuvent être amélioré en utilisant d'autres approches.

[10] Note : Google Colab nous offre une Limite d'utilisation de son GPU, ce n'est pas possible d'utiliser le GPU longtemps pour entrainer les données d'images volumineuses.

4. Application web final

Après la classification, les modèles que nous avons créés seront implémentés dans une application web, qui a pour but de donner à l'utilisateur la possibilité de faire un test dialectal sur un enregistrement audio donnée.

4.1. Présentation de l'interface

L'interface web regroupe les fonctionnements nécessaires pour faciliter la tâche à l'utilisateur.

Au début, nous avons l'interface **''Home''** représenter dans la figure. Qui est l'accueil de notre application.

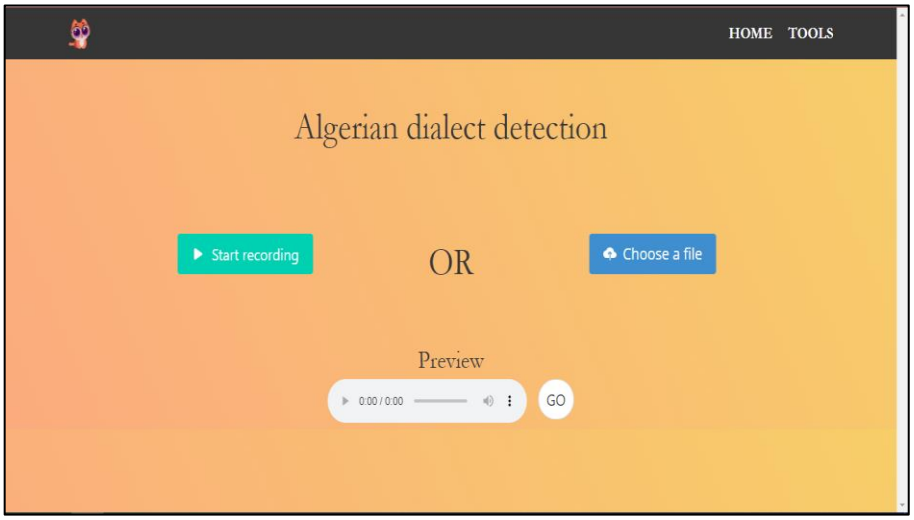


Figure 70 Interface Primaire.

La figure 70 représente l'interface d'accueil de notre application

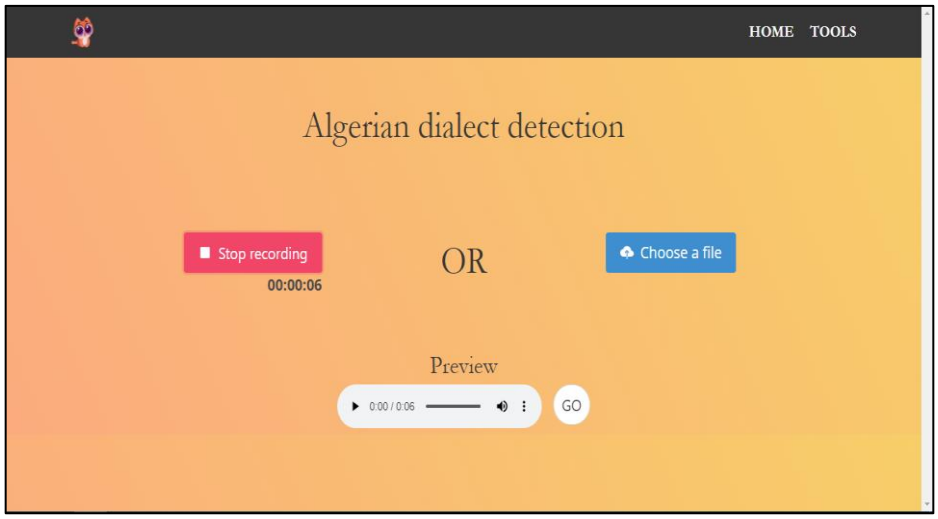


Figure 71 Interface Lors De L'enregistrement.

Dans l'accueil, l'utilisateur Télécharge un fichier audio sur lequel il veut faire un test de dialecte. Soit par un bouton **''Start recording''**, pour faire un enregistrement vocal en direct, affichant un compteur qui représente la taille de l'enregistrement pendant l'enregistrement un autre bouton **"Stop recording"** pour arrêter l'enregistrement.

Ou bien par un autre bouton **"Choose a file"** pour télécharger un fichier déjà enregistrer dans notre machine.

Nous avons aussi un bloque **"Preview"** pour écouter notre enregistrement vocal. Finalement un bouton **"GO"** pour commencer notre traitement et nous diriger vers la page d'affichage des résultats figure 72.

4.2. Présentation des résultats d'un cas concret

La page d'affichage des résultats se compose de 12 blocs, chaque bloc affiche 5 meilleurs résultats obtenus pas un model **X**, avec un pourcentage représentant le taux de prédiction correcte d'un dialecte donnée.

Dans cette section nous testons notre système abouti, sur un cas concret.

L'audio que nous avons utilisé est un audio enregistré par une personne d'Alger centre. Les résultats obtenus sont présentés dans la figure 72.

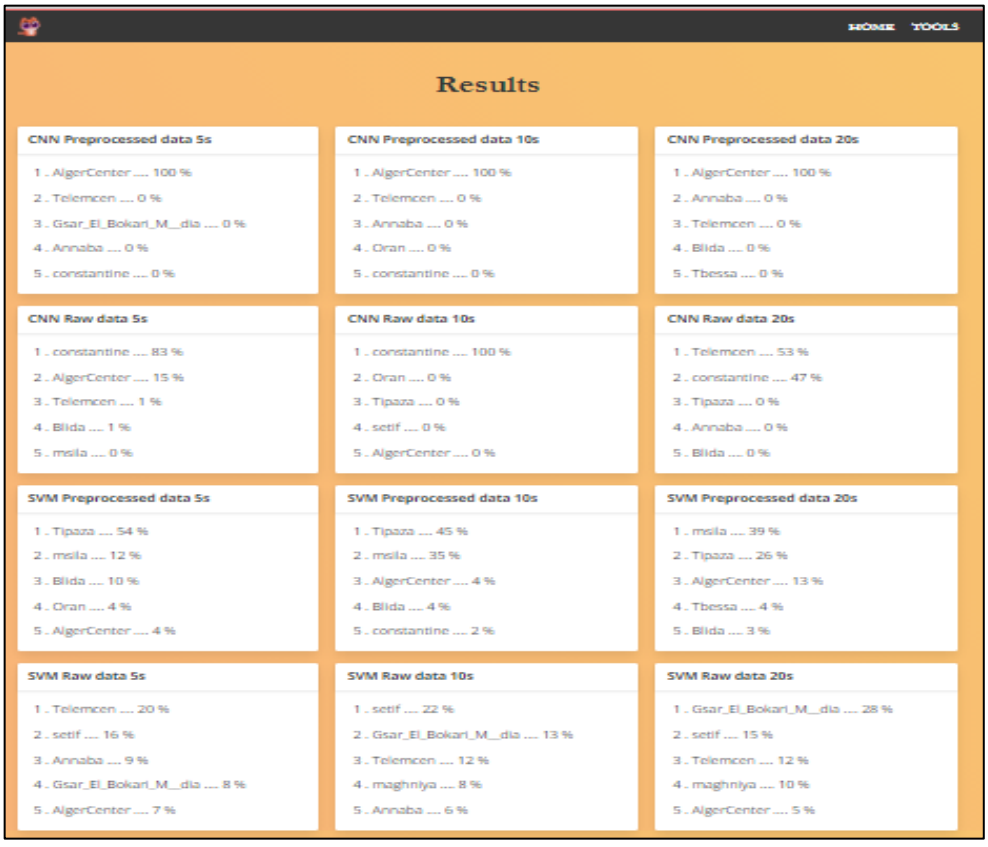


Figure 72 Interface des résultats d'un audio enregistré avec dialecte algérois.

Dans la figure 72, nous affichons la page des résultats de l'audio enregistré avec les méthodes de l'approche de classification acoustique.

Dans ce cas le dialecte est correctement prédit (100% Alger centre) avec les données prétraitées en utilisant le modèle CNN.

4.3. Analyses de la confusion des résultats

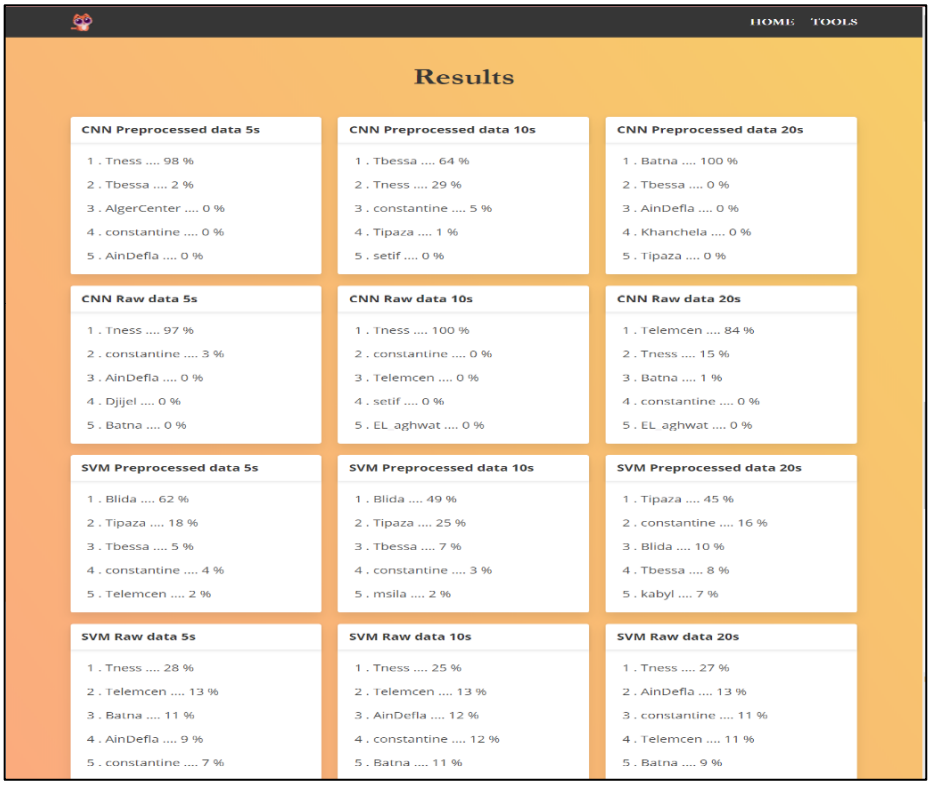


Figure 73 Interface des résultats d'un audio enregistré avec le dialecte Kabyle.

Dans la figure 73 nous avons un résultat obtenu à partir d'un audio choisi du disque dur d'une personne qui parle kabyle avec le dialecte de bouira, nous obtenons les résultats affichés dans la figure avec l'approche de classification des paramètres acoustiques. Avec les données brutes en utilisant le modèle CNN nous obtenons 100% Tness, il y a eu une confusion dans les résultats. Cela est dû au manque de données et de locuteurs dans notre donnée du dialecte kabyle.

5. Conclusion

Dans ce dernier chapitre, nous avons présenté et discuter les résultats que nous avons obtenus, ainsi que l'application finale. Ces résultats sont satisfaisants et démontre que la complexité et la diversité des dialectes peut être brisé par un système d'identification robuste.

Conclusion générale

Conclusion Générale

Le traitement automatique de la parole connait son essor depuis quelques décennies déjà. Aujourd'hui ses techniques sont même introduites dans notre vie quotidienne, comme dans les assistants vocaux qui offrent des fonctions de la reconnaissance de la parole, la synthèse de la parole, l'identification ou la vérification d'un locuteur et bien d'autres.

Cependant ces services ont leur limite et ne sont pas encore tout à fait au point, ils offrent de bons résultats pour les langues les plus parlées comme l'anglais. Ils ne sont pas très adaptés pour notre société qui a une très grande variété de dialectes parlés.

Le travail réalisé dans le cadre de ce mémoire est focalisé sur le développement d'un outil d'identification des dialectes algériens en se basant sur l'aspect phonétique. Nous avons utilisé deux approches pour atteindre notre objectif. La première est basée sur les caractéristiques acoustiques extraits à partir des audios, la deuxième sur la classification des spectrogrammes en utilisant les algorithmes d'apprentissage automatique, profond et par transfert. Ainsi, nous avons implémenté les modèles SVM, CNN et VGG-16 pour leurs très bonnes performances citées dans la littérature.

Nous avons entrainé ces modèles sur des données audios de durée 5s,10s et 20s. Les résultats que nous avons obtenus pour les données de durées 20s avec le modèle CNN, sont excellent avec une précision de 98% pour les données brutes et 97% pour les données prétraitées du modèle acoustique.

Pour les données de 10s et 5s nous avons obtenus avec le modèle CNN 97% et 95% de précision respectivement pour les données prétraité et 96% de précision pour les données brutes. Les résultats du modèle CNN dépasse de peu ceux du modèle SVM qui a eu une précision de 91%.

En ce qui concerne la deuxième approche, les résultats de classification des spectrogrammes de durée 5s, 10s et 20s sont de : 90%, 86% et 71% de précision respectivement.

Cette série d'expériences menées sur les 23 dialectes démontre la pertinence des paramètres acoustique ainsi que celle de l'utilisation des spectrogrammes pour l'identification des dialectes algériens.

Perspectives

Les travaux effectués dans ce mémoire ainsi que les résultats obtenus nous ont permis d'entreprendre de nouvelles pistes et perspectives de recherche dans le domaine du TAL et du traitement automatique des dialectes. Ces nouvelles ouvertures peuvent améliorer le résultat obtenu et donner une continuité à ce travail, on les cite dans les points suivants :

-Amélioration des résultats obtenus en utilisant d'autres techniques et modèles d'apprentissage automatique et profond.

-Enrichissement du corpus utilisé pour couvrir tous les dialectes algériens existants comme les dialectes du sud qui n'ont pas été traité et les différents dialectes de la langue tamazight.

-Utilisation d'une autre approche prosodique qui va traiter les paramètres rythmiques du dialecte. Cette approche peut être utile et peut donner des résultats intéressants pour l'identification des dialectes algériens qui ont des intonations différentes.

-Création d'un système de traduction automatique et de reconnaissance en se basant sur la première étape qui est l'identification du dialecte.

Bibliographie

[1] D. Hamdani.G, «Cour Des Principaux parametres du signal vocal,» Universite Saad Dahleb, Algerie, 2020.

[2] A. Hacine-Gharbi, «Sélection de paramètres acoustiques pertinents pour la reconnaissance de,» Université d'Orléans, France, 2012.

[3] H. Saadane, «Le traitement automatique de l'arabe dialectalisé : aspects méthodologiques et algorithmiques. Linguistique,» Université Genoble Alpes, France, 2015.

[4] 1. Cohen, 2. Barkat, 2. Embarki et 2. B. 1. Saâdane et al., La dialectologie arabe.

[5] E. Mohamed, «Les dialectes arabes modernes : état et nouvelles perspectives pour la classification géo-sociologique,» University of Franche-comté, 2008.

[6] M. A. ,. M. K.Lounnas, «Building a Speech Corpus based on Arabic Podcasts for Language,» Computational Linguistics Dept., CRSTDLA, Algeria, Algerie.

[7] L. K.Lounnas, «Automatic language identification for berber and arabic languages using prosodic features,» laboratory of spoken communication and signal processing USTHB, Algérie, 2018.

[8] M. Q. P. J. C. J. S. Elise Michon, «Neural Networks Architectures for Arabic Dialect Identification,» SYSTRAN, France, 2018.

[9] T. J. S. timsainb, «Noise reduction in python using spectral gating (speech, bioacoustics, audio, time-domain signals),» 2018. [En ligne].

[10] R. S. Alkhawaldeh, «DGR: Gender Recognition of Human Speech Using One-Dimensional Conventional Neural Network,» 2019. [En ligne].

[11] E. (Baeldung), «Multiclass Classification Using Support Vector Machines,» 25 august 2021. [En ligne].

[12] C. ETIENNE, «Apprentissage profond applique´ a` la reconnaissance des émotions dans la voix,» Ecole doctorale n ´ ◦580 Ecole Doctorale Sciences et Technologies de l'Information et, Paris, 2019.

[13] K. e. A.Zisserman, «Very Deep Convolutional Networks for Large-Scale Image Recognition,» Université d'Oxford.

[14] M. B.CAILLAUD, «Analyse sonographique et aspects de la phonetique appliqué.,» Revue l'epi N°93.