REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Saad Dahleb, Blida 1

Faculté des Sciences

Département d'informatique

Mémoire de fin d'études en vue de l'obtention du diplôme de Master2

Option : traitement automatique de la langue.

Thème :

Reconnaissance automatique de la parole arabe continu.

- **Présenté par** : Mr HAMMADECHE Abdel Hakim.
  Mr TAKI Mohamed.
- **Encadré par :** Mr ABBAS Mourad
- **Promoteur :** M GHABGHOUB Yasmine

Promotion : 2018/2019

Remerciements

Avant tout, nous remercions Dieu le Tout Puissant de nous avoir donné la force, le courage, la santé et la patience pour pouvoir accomplir ce travail.

Nous adressons nos sincères remerciements et notre gratitude la plus profonde à tous ceux qui nous ont aidé à l'accomplissement de ce modeste travail.

Nous remercions chaleureusement notre promotrice ***Mme yassmine ghebghoub****, et notre encadreur* ***Mr mourad abbas****, pour avoir accepté de diriger ce travail, leurs encouragements, leur disponibilité et pour leur rigueur scientifique, pour leurs nombreux conseils qui ont contribué à la réalisation de ce travail, pour la confiance qu'ils nous ont accordé, et leur soutien scientifique et moral.*

Nous remercions également les membres du jury pour l'honneur qu'ils nous ont fait, en acceptant d'examiner ce travail et d'apporter leurs critiques enrichissantes, veuillez trouver ici l'expression de notre sincère reconnaissance.

Dédicaces

A mes chers parents

Je remercie mes très chers parents, qui m'ont toujours apporté le meilleur, vous avez su me guider et me conseiller tout au long de mon parcours.

Merci mes chers parents, qu'ALLAH vous bénisse et vous accorde une longue et heureuse vie.

A mes chers frères et sœurs

Pour leur amour et leur soutien inestimable.

A tous mes amis

Abdelhakim

Dédicaces

À mes chers parents

Aucune dédicace ne saurait exprimer mon respect, mon amour éternel et ma considération pour les sacrifices que vous avez consenti pour mon instruction et mon bien être.

Je vous remercie pour tout le soutien et l'amour que vous me portez depuis mon enfance et j'espère que votre bénédiction m'accompagne toujours.

A mes chers frère et sœurs

Je vous remercie pour votre soutien, vous m'avez remonté le moral lorsque ma détermination flanchait, j'espère que vous appréciez le fruit de tant d'effort

Et a toute ma famille et mes amis

Qui ont toujours était la quand j'avais besoin d'eux

Mohamed

Abstract

Speech is the most natural form of human communication and speech processing has been one of the most exciting areas of the signal processing. Speech recognition technology has made it possible for computer to follow human voice commands and understand human languages. The main goal of speech recognition area is to develop techniques and systems for speech input to machine and treat this speech to be used in many applications. As Arabic is one of the most widely spoken languages in the world.

Statistics show that it is the first language (mother-tongue) of 295 million native speakers ranked as fiveth after Mandarin, Spanish ,English and India. In spite of its importance, research effort on Arabic Automatic Speech Recognition (ASR) is unfortunately still inadequate.

This thesis proposes and describes an efficient and effective framework for designing and developing a speaker-independent continuous automatic Arabic speech recognition system based on a phonetically rich and balanced speech corpus. The developing Arabic speech recognition system is based on the Carnegie Mellon university Sphinx tools.
To build the system, we develop three basic components. The dictionary which contains all possible phonetic pronunciations of any word in the domain vocabulary.
The second one is the language model such a model tries to capture the properties of a sequence of words by means of a probability distribution, and to predict the next word in a speech sequence. The last one is the acoustic model which will be created by taking audio recordings of speech, and their text transcriptions, and using software to create statistical representations of the sounds that make up each word. The system use the rich and balanced database that contains 620 sentences, a total of 2863 words. The phonetic dictionary contains about 3720 definitions corresponding to the database words. And the language model contains 2866 mono-gram and 4049 bi-grams and 4673 tri-grams. The engine uses 3-emmiting states Hidden Markov Models (HMMs) for tri-phone-based acoustic models..

Keywords: *Arabic automatic speech recognition, acoustic model, and language Model*

Résume

La parole est la forme la plus naturelle de communication humaine et le traitement de la parole a été l'un des domaines les plus intéressants du traitement du signal La technologie de reconnaissance vocale a permis à un ordinateur de suivre les commandes de la voix humaine et de comprendre les langages humains. L'objectif principal de la zone de reconnaissance vocale est de développer des techniques et des systèmes permettant à la parole d'entrer dans la machine et de traiter cette parole dans de nombreuses applications. L'arabe est l'une des langues les plus parlées au monde Les statistiques montrent qu'il s'agit de la première langue (langue maternelle) de 295 millions de locuteurs natifs classés cinquième après le mandarin, l'espagnol, l'anglais et l'Inde Malgré son importance, les efforts de recherche sur la reconnaissance vocale automatique en arabe sont encore insuffisants.

Cette thèse propose et décrit un cadre efficace pour la conception et le développement d'un système de reconnaissance automatique de la langue arabe, continu et indépendant du locuteur, basé sur un corpus de parole riche, riche et équilibré. Le système en cours de reconnaissance de la parole en arabe est basé sur les outils Sphinx de l'Université Carnegie Mellon.

Pour construire le système, nous développons trois composants de base Le dictionnaire qui contient toutes les prononciations phonétiques possibles de n'importe quel mot du vocabulaire du domaine, Le second est le modèle de langage, Un tel modèle tente de saisir les propriétés d'une séquence de mots au moyen d'une distribution de probabilité et de prédire le mot suivant dans une séquence vocale Le dernier est le modèle acoustique qui sera créé en prenant des enregistrements audio de la parole et leurs transcriptions de texte, et en utilisant un logiciel pour créer des représentations statistiques des sons qui composent chaque mot. Le système utilise la base de données riche et équilibrée qui contient 620 phrases, un total de 2863 mots. Le dictionnaire phonétique contient environ 3720 définitions correspondant aux mots de la base de données. Le modèle de langage contient 2866 mono-grammes et 4049 bi-grammes et 4673 tri-grammes. Le moteur utilise des modèles de Markov cachés (HMM) à 3 états émetteurs pour les modèles acoustiques à trois téléphones.

Mots clés : Reconnaissance automatique de la parole arabe, model de language,model acoustique

الملخص

الكلام هو النمط الأكثر استخداما في التواصل الإنساني ومعالجة الكلام كان واحدا من أكثر المواضيع المهمة في مجال معالجة الإشارات. ولقد حققت تكنولوجيا التعرف على الكلام إمكانية متابعة الأوامر الصوتية للإنسان وفهم لغات البشر باستخدام جهاز الكمبيوتر فكان الهدف الرئيسي لمجال التعرف على الكلام تطويرالتقنيات والنظم لإدخال الكلام إلى الآلة ومنها جهاز الكمبيوتر ومن ثم معالجته لاستخدامه في مجالات متعددة. وبماأن اللغة العربية هي واحدة من أكثر اللغات انتشارا في العالم وكما تظهر الإحصائيات أنها اللغة الأولى –اللغة الأم- ل 295 مليون ناطق أصلي باللغة العربية كما أنها تحتل المرتبة الخامسة بعد الماندرين والإسبانية والإنكليزية و الهندية على الرغم من أهميتها إلا أن الجهود التي تبذل في أبحاث التعرف على الكلام في اللغة العربية للأسف ما زال غير كافي .

هذه الأطروحة تقترح وتصف كيفية تصميم وتطوير نظام التعرف على الكلام العربي التلقائي ، المعتمد على المتحدث المستقل والخطاب المستمر ، وذلك باستخدام قاعدة كلام غنية صوتيا ومتوازنة من جامعة كارنيجي ميلون sphinx ويتم الاستناد في تطوير النظام على أدوات

ولبناء هذا النظام تم إنشاء ثلاث وحدات أساسية وهي القاموس اللفظي وهو يحتوي على النطق الصوتي للكلمات ، ووحدة نموذج اللغة وهو يحوي احتمالات ورود الكلمات معا، وأخيرا وحدة المحرك وهو يعمل على مقارنة الصوت المدخل مع المعلومات الصوتية المخزنة للكلام في القاموس اللفظي .

النظام يستخدم اثنين قاعدة بيانات القاعدة الاولى متوازنة تحتوي على 620 جملة تتضمن ما مجموع 2863 كلمة أما القاموس اللفظي فيحتوي على حوالي 3720 تعريف يقابل كل منها كلمة في قاعدة البيانات بينما نموذج اللغة المستخدم يحتوي على 2866 أحادية المقطع و 4049 ثنائي المقطع و 4673 متعدد. وتم استخدام محرك مبني على نماذج ماركوف المخفية مستندا إلى النماذج الصوتية الثلاثية المقاطع؛ و تم تدريب النظام على 3.5 ساعات من خطاب البيانات باللغة العربية .

القاعدة الثانية تحتوي على 388 جملة تتضمن ما مجموع 3528 كلمة أما القاموس اللفظي فيحتوي على حوالي 388 تعريف يقابل كل منها كلمة في قاعدة البيانات بينما نموذج اللغة المستخدم يحتوي على 3534 أحادية المقطع و 8290 ثنائي المقطع و 9622 متعدد. وتم استخدام محرك مبني على نماذج ماركوف المخفية مستندا إلى النماذج الصوتية الثلاثية المقاطع؛ و تم تدريب النظام على 1.5 ساعات من خطاب البيانات بالدارجة الجزائرية .

كلمات البحث: المستمر التعرف على الكلام العربي, نموذج اللغة ,القاموس الفظي

Liste des abréviations

ASR Automatic Speech Recognition

CFG Context-Free Grammars

CHMM Continuous Hidden Markov Model

CMN Cepstral Mean Normalization

CMU Carnegie Mellon University

DCT Discrete Cosine Transform

ELRA European Language Resource Association

HMMs Hidden Markov Models

JSGF JavaTM Speech API Grammar Format

KACST King Abdulaziz City of Science and Technology

LDC Linguistic Data Consortium

LMGrammar Language Model Grammar

LPC Linear Predictive Encoding

MFCC Mel Frequency Cepstral Coefficients

MSA Modern Standard Arabic

PCI Peripheral Component Interconnect

PLP Prediction Coefficient Extraction

SAPI Speech API

SCHMM Semi-Continuous Hidden Markov Model

WER Word Error Rate

WRR Word Recognition Rate

Sommaire

Listes des figures

Listes des tableaux

INTRODUCTION GENERALE

Depuis presque trente ans, la reconnaissance automatique de la parole est un domaine qui a captivé le public ainsi que de nombreux chercheurs. À ses balbutiements, les projections sur ses applications étaient très optimistes : quoi de plus naturel que de parler à une machine, sans avoir à s'encombrer d'un clavier ?
Malheureusement, malgré l'incroyable évolution des ordinateurs et des connaissances, la reconnaissance automatique de la parole n'en demeure pas moins un sujet de recherche toujours actif et les résultats obtenus sont encore loin de l'idéal qu'on aurait pu en attendre, il y a vingt ans.

Cependant, si le système de reconnaissance idéal n'existe pas encore, des applications concrètes émergent petit à petit. La reconnaissance automatique de la parole commence à équiper certains téléphones ou GPS qui, en identifiant certains mots clefs, permettent d'effectuer les tâches demandées. Les systèmes de reconnaissance sont également utilisés pour indexer de grandes bases de données audiovisuelles, pour rechercher des termes dans des flux audio ou encore comme interface de dialogue homme-machine. Dans la pratique, quand les conditions d'utilisation sont correctes, ces systèmes s'avèrent efficaces. Néanmoins, les principales limites des systèmes actuels sont relatives à leur robustesse : les conditions d'utilisation doivent être similaires à celles utilisées pour entraîner le système, l'environnement sonore peu bruité, les locuteurs ne peuvent pas parler simultanément. Souvent, l'utilisateur a dû s'adapter pour utiliser les logiciels.

De plus en plus, la technologie de la reconnaissance de la parole fait son chemin vers des applications réelles. Actuellement un changement qualitatif dans l'état de l'art est apparu promettant d'apporter des capacités de reconnaissance et de les mettre à la portée de chaque personne.

La reconnaissance de la parole continue pour un vocabulaire moyen (quelques milliers de mots) est actuellement possible dans un logiciel de reconnaissance de la parole. La reconnaissance de la parole humaine se situe à l'intersection de nombreux domaines tels que l'acoustique, l'électronique, la phonétique...Pour atteindre un haut niveau, un système de reconnaissance de la

parole doit s'inspirer des travaux d'une vaste gamme de disciplines scientifiques : Mathématique, informatique, technologie...etc.

Le signal de la parole est l'un des signaux les plus complexes, il n'est pas facile de le caractériser par un modèle simple. L'un des problèmes de la reconnaissance de la parole est la variabilité du signal. Les précurseurs dans le domaine du traitement de la parole, pensaient que la parole était une simple juxtaposition d'éléments appelés "caractéristiques distinctives" ou invariants (phonèmes), et qu'à partir d'échantillons représentant les invariants d'une langue, on pouvait reconstituer ou reconnaître n'importe quelle phrase. Cette idée théorique n'est pas toujours vraie dans la reconnaissance de la parole continue à cause du problème de coarticulation. Pour surmonter ces difficultés, de nombreuses méthodes et modèles mathématiques originaux ou adaptés d'autres domaines ont été développés, parmi lesquelles on peut citer : la comparaison dynamique, les systèmes experts, les réseaux de neurones, les modèles stochastiques et en particulier les modèles de Markov cachés, etc.

Notre travail s'inscrit dans le cadre général de la reconnaissance automatique de la parole. Elle consiste à réaliser un Système de Reconnaissance Automatique de la Parole (SRAP) basé sur le CMU Sphinx qui est basé sur les Modèles de Markov Cachés (MMC) pour la reconnaissance des 620 phrases dont 2863 mots de l'Arabe standard et 388 phrases dont 3528 mots de l'Arabe dialectale (algérien).
Notre mémoire comporte 3 chapitres :

- Le premier chapitre, nous présentons une brève description de la reconnaissance vocale, qui contient les sujets suivants: technologie vocale, concepts de base, types de parole, utilisations et applications de la parole, matériel nécessaire pour manipuler la parole, fonctionnement du programme de reconnaissance, illustrer la langue arabe et une vue approfondie de la technique d'extraction de caractéristiques.
- Le deuxième chapitre contient la modélisation du système proposé.
- Le dernier chapitre contient la réalisation et l'expérimentation du système et le résultat.

1 CHAPITRE 1 : NOTION DE BASE

1.1 Introduction

La parole est un moyen de communication très efficace et naturel utilisé par l'humain. Depuis longtemps, il rêve de pouvoir s'adresser par ce même moyen à des machines ce qui les rendre plus intelligentes.

Cependant, malgré les énormes efforts de recherches consacrés dans l'essai de créer une telle machine intelligente qui peut reconnaître le mot parlé et comprend sa signification, on est loin d'atteindre le but désiré d'une machine qui peut comprendre le discours parlé par tous les locuteurs dans tous les environnements, ce qui est due aux limites du système de reconnaissance de la parole. Alors, qu'est-ce que l'on entend par la reconnaissance automatique de la parole (RAP) ?

1.2 Reconnaissance automatique de la parole

La reconnaissance automatique de la parole (ASR) est une technologie qui permet à un ordinateur d'identifier les mots qu'une personne dit dans un microphone ou un téléphone. Elle a un large domaine d'applications: reconnaissance des commandes (interface utilisateur vocale avec l'ordinateur), dictée, réponse vocale interactive, et elle peut être utilisée pour apprendre une langue étrangère. ASR peut également aider les personnes handicapées à interagir avec la société. C'est une technologie qui rend la vie plus facile [1].

Beaucoup de ses systèmes sont développés, les plus connus étant: Dragon Naturally Speaking, IBM via la voix, Microsoft SAPI. De nombreux systèmes de reconnaissance vocale open source sont également disponibles, qui sont basé sur les modèles de Markov cachés (HMM) [1].

L'arabe est l'une des six langues officielles des Nations Unies (ONU) et l'une des plus parlées au monde. Les statistiques montrent qu'il s'agit de la langue maternelle de 295 millions de locuteurs et se classe au cinquième rang après le mandarin, l'espagnol, l'anglais et l'hindi. Malgré son importance, les efforts de recherche sur la reconnaissance automatique de la parole en arabe sont malheureusement toujours insuffisants.

L'arabe moderne standard (MSA) est la norme linguistique officielle de la langue arabe, qui est largement enseignée dans les écoles et les universités et utilisée au bureau et dans les médias. C'est l'objet et l'intérêt principal de nombreux pays récents et précédents recherches comparées à l'arabe dialectal [8, 9, 10, 11]. Le manque de données de formation parlées et écrites est l'un des principaux problèmes rencontrés par les chercheurs arabes en ASR. Une liste des corpus les plus populaires (de 1986 à 2005) est fournie [12] et ne répertorie que 19 corpus (14 écrits, 2 parlés, 1 écrit et parlé et 2 conversationnels). Ces corpus ne sont pas facilement accessibles au public et nombre d'entre eux ne peuvent être achetés qu'auprès du Consortium de données linguistiques (LDC) ou de l'Association de ressources linguistiques européennes (ELRA). Il est clair qu'il existe une pénurie de données parlées par rapport aux données écrites, ce qui nécessite un plus grand nombre de corpus de parole afin de desservir différents domaines de la RSA arabe. Les corpus parlés disponibles ont été principalement collectés dans les journaux télévisés (radios et télévisions) et les conversations téléphoniques. Ce type de données parlées ne sert pas

nécessairement la recherche ASR de qualité en arabe, en raison de la qualité des données parlées elles-mêmes en termes d'attributs d'enregistrement et de paramètres utilisés (par exemple, taux d'échantillonnage). Ils sont également limités à certaines applications et domaines. La couverture de tout corpus ne peut contenir des informations complètes sur tous les aspects du lexique et de la grammaire des langues [13], en raison du nombre limité de données de formation écrites et, partant, de données de formation orales insuffisantes. En outre, une relation claire et forte entre les formes écrites et parlées doit être clarifiée.

L'écriture est dite plus complexe, plus élaborée, plus explicite, plus organisée et plus planifiée que la parole [14]. Ces différences mènent généralement à l'idée que la forme écrite des corpus doit être créée avec soin avant de produire et d'enregistrer la forme parlée. Par conséquent, les linguistes et les phonéticiens produisent soigneusement des corpus écrits avant de les traiter à des spécialistes de l'enregistrement de la parole.

C'est ce qui s'observe également au cours des dernières années, qui ont produit un grand nombre de corpus riches en poids et / ou équilibrés pour de nombreuses langues. De nombreuses recherches ASR reposent désormais sur des corpus riches en poids et / ou équilibrés, par exemple anglais [15 - 17], mandarin [18], japonais [19], indonésien, coréen, hindi cantonais, turc et bien d'autres, obtenant des résultats relativement compétitifs. En ce qui concerne la langue arabe, les tâches de reconnaissance automatique de la parole sont principalement destinées aux chiffres arabes, aux nouvelles radiodiffusées, au commandement et au contrôle, au coran et aux recherches sur les proverbes arabes. Ils ont exploré diverses techniques et outils de pointe pour la reconnaissance de la parole en arabe [20, 21, 9, 11].

Le développement de systèmes de reconnaissance vocale automatique (ASR) précis est confrontés à deux problèmes majeurs. Le premier problème est lié à la discrétisation: les symboles diacritiques désignent les phonèmes de voyelles dans les mots désignés. Les textes arabes ne sont presque jamais complètement vocalisés: ce qui implique que les traits courts placés au-dessus ou au-dessous de la consonne, indiquant la voyelle suivant cette consonne, sont généralement absents. Cela limite la disponibilité du matériel de formation en arabe ASR. L'absence de ces informations conduit à de nombreuses formes de mots similaires et, par conséquent, diminue la prévisibilité dans le modèle de langage. Le deuxième problème est lié à la complexité morphologique car l'arabe possède un riche potentiel de formes de mots, ce qui augmente le taux de vocabulaire en sortie [22, 23]

1.3 Techniques ASR

La technique ASR basée sur HMM a conduit à de nombreuses applications nécessitant un vocabulaire étendu, une reconnaissance de la parole indépendante du locuteur et continue. HMM est un modèle statistique dans lequel le système modélisé comporte des paramètres inconnus. Le défi consiste à déterminer les paramètres cachés à partir des paramètres observables. Les paramètres de modèle extraits peuvent ensuite être utilisés pour effectuer d'autres tâches.

L'analyse, par exemple les applications de reconnaissance de formes. Son extension aux langues étrangères (l'anglais est la norme) représente un véritable défi pour la recherche [24].

Le système basé sur HMM consiste essentiellement à reconnaître la parole en estimant la vraisemblance de chaque phonème à de petites trames contiguës du signal de parole [25, 26]. Les

mots du vocabulaire cible sont modélisés en une séquence de phonèmes, puis une procédure de recherche est utilisée pour trouver, parmi les mots de la liste de vocabulaire, la séquence de phonèmes qui correspond le mieux à la séquence de phonèmes des mots parlés.

Chaque phonème est modélisé comme une séquence d'états HMM. Dans les systèmes standard basés sur HMM, les probabilités (également appelées probabilités d'émission) d'une certaine observation de la base de sondage produite par un état sont estimées à l'aide de modèles de mélange gaussiens traditionnels. L'utilisation de HMM avec des mélanges gaussiens présente plusieurs avantages notables, tels qu'un cadre mathématique riche, des algorithmes d'apprentissage et de décodage efficaces et une intégration aisée de multiples sources de connaissances. Les systèmes HMM, connus sous le nom de HKM (Hidden Markov Model Toolkit), mis au point à l'Université de Cambridge (27); et le système Sphinx développé à la Carnegie Mellon University [28]. Les outils Sphinx peuvent être utilisés pour développer un large spectre de tâches de reconnaissance vocale.

Par exemple, le Sphinx-II [29] utilise les modèles SCHMM (modèle de Markov caché semi-continu) pour réduire le nombre de paramètres et les ressources informatiques nécessaires au décodage, mais présente une précision limitée et une procédure d'entraînement compliquée. D'autre part, Sphinx-III utilise le modèle de Markov continu caché (CHMM) avec des performances supérieures, mais requiert des ressources informatiques substantielles. Sphinx-4, développé en Java, peut être utilisé pour créer des applications de reconnaissance vocale indépendantes de la plateforme [30, 28].

1.4 Travaux existants dans Reconnaissance de la parole en arabe

Le développement d'un système de reconnaissance de la parole en arabe est un effort multidisciplinaire, qui nécessite l'intégration de l'arabe phonétique [31, 32], des techniques de traitement de la parole en arabe [33] et du langage naturel [34, 35]. La mise au point d'un système de reconnaissance de la parole en arabe a récemment été abordée par un certain nombre de chercheurs. La reconnaissance de la parole continue en arabe a été abordée par Al_Otaibi, [36]. Il a fourni un ensemble de données de parole pour l'arabe moderne standard (MSA). Il a étudié différentes approches pour la construction du corpus de la parole en arabe et a proposé une nouvelle technique d'étiquetage de la parole en arabe. Il a signalé un taux de reconnaissance de l'ASR dépendant du locuteur de 93,78% en utilisant sa technique.

L'ASR a été construit à l'aide du kit d'outils HTK. Bila et al. [37] ont abordé les problèmes d'indexation des informations en arabe et ont abordé un certain nombre de questions de recherche relatives à la reconnaissance de la parole en arabe. Il existe un certain nombre d'autres tentatives de construction de la RSA arabe (RSAA), mais elles ont considéré soit un vocabulaire limité, soit un système dépendant du locuteur [4, 8, 9, 23, 26].

Les problèmes les plus difficiles à résoudre pour développer des RSA extrêmement précis en arabe sont la prédominance de textes non vocalisés, l'énorme variété dialectale et la complexité morphologique [24]. Kirchhoff et All [10] ont étudié la reconnaissance de l'arabe dialectal et étudié les différences entre l'arabe dialectal et l'arabe formel du point de vue de la reconnaissance de la parole. Vergyri et al. [38] étudient l'utilisation d'un modèle de langage basé sur la morphologie à différentes étapes d'un système de reconnaissance de la parole en arabe

conversationnel; Il a également étudié le texte arabe de diacritisation automatique à utiliser dans la formation des modèles acoustiques pour ASR. Satori et autres [11] a introduit un système de reconnaissance vocale arabe dans lequel la formation et le processus de reconnaissance utilisent des caractères romanisés. La plupart des travaux antérieurs sur l'ASR arabe avaient été concentrés sur le développement de dispositifs de reconnaissance utilisant des caractères romanisés. Le système utilisé dans [9] utilise le moteur Sphinx-IV de la CMU (Carnegie Mellon University) est basé sur des modèles de Markov cachés (HMM), qui ont obtenu un taux de reconnaissance de mots de 99,21% pour environ 35 minutes de données de discours de formation et 7 minutes de test de parole. Les données.

Le système de [11] utilisait également le moteur CMU Sphinx-IV basé sur HMM pour la même tâche et obtenait un taux de reconnaissance des mots de 85,56% pour les locuteurs hommes et de 83,34% pour les locuteurs femmes. Dans [21], un type différent de données de parole a été présenté pour le Système de reconnaissance des chiffres arabes utilisant un corpus arabe accentué de téléphonie saoudienne. Le système utilisait des outils Cambridge HTK basés sur des HMM et signalait un taux de reconnaissance des chiffres correct de 93,67%. En outre, le Saint Coran a également été pris en compte pour la reconnaissance de la parole en arabe dans [9], qui utilise un moteur Sphinx-IV basé sur des HMM et obtient un taux de reconnaissance des mots de 70,81% et un taux d'erreur de mot] de 40,18% pour un corpus de 18,35 heures.

D'autre part, un système de reconnaissance de la parole en arabe utilisant un corpus de nouvelles radiodiffusées a été développé dans [8]. Le système a été formé en utilisant environ 7 heures de parole à l'aide des outils Sphinx 3 basés sur des HMM et testé en utilisant 400 énoncés, soit environ une demi-heure de conversation. Le système a obtenu un taux de reconnaissance de mots correct de 90,78% et un WER de 10,87% avec des signes diacritiques complets, alors qu'il obtenait un taux de reconnaissance de mots correct de 93,04% et un WER de 8,61% sans signes diacritiques. D'autres systèmes de reconnaissance automatique de la parole en arabe ont été développés pour différentes tâches, comme dans [39, 9, 40]. Un système de commandement et de contrôle couvrant environ 30 mots a été développé dans [9] avec le moteur Sphinx-IV basé sur HMM et a obtenu un taux de reconnaissance des mots de 98,18%, alors qu'un système arabe ASR reconnaissant 16 phrases de proverbes égyptiens a été développé [39]. basé sur HMM et a obtenu des taux de reconnaissance de mots de 56,8%, 66,65% et 81,79% pour la reconnaissance basée sur un même téléphone, Tri-téléphone et Syllabe, respectivement.

Le rapport technique dans [41] est l'un des tout premiers travaux sur la production de données de formation écrites en arabe sur la base de phrases tristement riches et équilibrées. Ce rapport technique a été soumis à la Cité des sciences et de la technologie du roi Abdulaziz (KACST) en Arabie saoudite comme dernier produit livrable du projet Base de données de sons arabes: phrases. Ces données de formation écrites ont été créées par des experts de KACST et consistent en 367 phrases écrites en utilisant 663 mots riches en prononciation. Les données de formation écrite (texte) KACST ont été utilisées comme base pour la création de notre corpus de parole phonétiquement riche et équilibré. Dix autres phrases écrites ont été créées à des fins de test, extraites de nouvelles en arabe

1.5 La contribution

Dans cette thèse, nous construirons un système de reconnaissance de la parole en arabe basé sur un ensemble de données de parole en arabe riche et équilibré, la couverture de données utilisée couvrant tous les regroupements de phonèmes en arabe avec un minimum de répétition des mots et une structure de phrases simples. La première chose à faire est d'obtenir les prononciations phonétiques de tous les mots du vocabulaire du domaine, ainsi que les prononciations variantes des mots si leur, toutes ces prononciations phonétiques possibles seront considérées comme un dictionnaire.

La deuxième étape consiste à saisir les propriétés d'une séquence de mots au moyen d'une distribution de probabilité et à prédire le mot suivant dans une séquence vocale en générant l'uni-gramme, le bi-gramme et le tri-gramme des mots, en conservant les dans le modèle de langage. Ensuite, nous allons générer les unités HMM correspondantes des mots utilisés, en créant des représentations statistiques des sons qui les composent. En utilisant les trois modules ensemble, nous pouvons construire le système de reconnaissance de la langue arabe pour un grand vocabulaire et reconnaître la parole arabe continue.

La troisième étape consiste à tester le système. Les tests comprennent l'enregistrement d'un son vocal provenant de chaînes d'information et sa reconnaissance via le système, ainsi que l'utilisation du microphone pour entrer directement le son dans le système à reconnaître.

1.6 Technologie de la parole

Tous les développeurs ne sont pas familiarisés avec la technologie vocale, en tant que technologie émergente. Il existe des capacités subtiles et puissantes avec la fonction de base des deux synthèses de la parole et reconnaissance de la parole fournie par la parole informatisée, que les développeurs voudront comprendre et utiliser Synthèse de la parole et de la parole Les technologies de reconnaissance ont encore d'importantes limites en dépit d'investissement dans la recherche en technologie de la parole au cours des dernières années. Plus important encore, la technologie vocale ne répond pas toujours aux attentes élevées des utilisateurs familiarisés avec la communication vocale naturelle entre humains. C'est important pour une utilisation efficace de l'entrée et la sortie vocales dans une interface utilisateur pour comprendre les limites, ainsi que les points forts Une compréhension des capacités, des forces et des limites de la parole la technologie est également importante pour les développeurs dans la prise de décision quant à savoir si une application particulière bénéficiera de l'utilisation de la parole entrée et sortie ou non.

1.6.1 Synthèse vocale (TTS)

La tâche d'un synthétiseur de parole est de convertir un texte écrit donné en une langue parlée La synthèse vocale est également appelée conversion de texte en parole (TTS). La production d'un discours à partir d'un texte comporte de nombreuses étapes. Les principales étapes de la production d'un discours à partir d'un texte sont les suivantes:

Analyse de structure et Prétraitement du texte.

Nous allons illustrer les étapes de mise en œuvre de l'algorithme de manière plus détaillée:

- Analyse de la structure: cette étape permet de déterminer le début et la fin des paragraphes, des phrases et des autres structures après le traitement du texte saisi. Les données de ponctuation et de formatage sont utilisées à cette étape pour la plupart des langues.

- Prétraitement du texte: lors de cette étape, le synthétiseur de parole analyse le texte d'entrée donné pour rechercher des constructions spéciales du langage.

1.6.2 Reconnaissance de la parole

La reconnaissance vocale peut être définie comme le processus de conversion de la langue parlée en texte écrit ou sous une forme similaire [44].

1.7 Principes de base de la reconnaissance vocale

Dans cette section, nous définirons quelques termes importants qui seront beaucoup utilisés dans les sections suivantes. Ce sont les bases nécessaires à la compréhension de la technologie de reconnaissance vocale.

1.7.1 Énonciation

Les énoncés peuvent être un seul mot, quelques mots, une phrase ou même plusieurs phrases, il s'agit de la vocalisation (parlant) d'un mot (mots) ou de phrases qui représentent un seul sens pour l'ordinateur.

1.7.2 Dépendance de locuteur

Il y a deux types de dépendance du locuteur :

Le premier sont les systèmes indépendants du locuteur conçus pour une variété locuteur. Les systèmes adaptatifs commencent généralement par des systèmes indépendants des haut-parleurs et utilisent des techniques de formation pour s'adapter à la haut-parleur pour augmenter leur précision de reconnaissance.

Le deuxième type sont les systèmes dépendant de haut-parleur qui a conçu autour d'un haut-parleur spécifique. Ils supposent que locuteur parlera d'une voix et d'un tempo cohérents. Ainsi, ils sont généralement plus précis pour le bon locuteur, mais beaucoup moins précis pour les autres locuteurs.

1.7.3 Vocabulaires

Les vocabulaires (ou dictionnaires) peuvent être définis comme des listes de mots ou d'énoncés pouvant être reconnus par le système. Il y a une différence entre les mots grand et petit en reconnaissance informatique. Par exemple, pour expliquer plus de vocabulaires plus petits sont plus faciles à reconnaître pour un ordinateur, tandis que les vocabulaires plus volumineux sont plus difficiles. Contrairement aux dictionnaires normaux, chaque entrée ne doit pas obligatoirement être un mot. Ils peuvent être aussi longs qu'une phrase ou deux. Les vocabulaires plus petits peuvent avoir aussi peu que 1 ou 2 énoncés reconnus (par exemple, "Réveillez-vous"), alors que les très grands vocabulaires peuvent avoir cent mille ou plus!

1.7.4 **Précision**

Nous pouvons examiner la capacité d'un dispositif de reconnaissance en mesurant sa précision - ou comment bien que la reconnaissance reconnaît les énoncés. Cela inclut non seulement l'identification correcte un énoncé mais également identifier si l'énoncé parler n'est pas dans son vocabulaire. La précision dépend vraiment de l'application.

1.7.5 **Entraînement**

Parfois, le système dispose d'un identificateur capable de s'adapter à un locuteur. Et lorsque le système dispose de cette capacité, il peut permettre une formation. La formation d'un identificateur améliore généralement sa précision. La formation peut être utilisée par des locuteurs ayant des difficultés à parler ou à prononcer certains mots. Tant que le locuteur peut Répétez systématiquement un énoncé, les systèmes de reconnaissance vocale formés doivent pouvoir s'adapter. Un système de reconnaissance vocale est formé en demandant au locuteur de répéter des phrases standard ou courantes et en ajustant ses algorithmes de comparaison pour correspondre à ce locuteur particulier.

1.8 Types de reconnaissance vocale

Les systèmes de reconnaissance vocale peuvent être séparés en plusieurs classes différentes en décrivant les types d'énoncés qu'ils sont capables de reconnaître. Ces classes sont basées sur le fait que l'une des difficultés de la reconnaissance vocale est la capacité de déterminer quand un parleur commence et termine une dictassions. La plupart des packages peuvent s'inscrire dans plusieurs classes, selon le mode utilisé.

1.8.1 **Classe de mots isolés**

Les identificateurs de mots isolés exigent généralement que chaque énoncé soit silencieux (absence de signal audio) des deux côtés de l'échantillon. Cela ne signifie pas qu'il accepte des mots simples, mais nécessite une seule énonciation à la fois. Souvent, ces systèmes ont des états "Listen / Not-Listen", dans lesquels ils obligent le locuteur à attendre entre les énoncés (en général, il effectue le traitement pendant les pauses). L'expression isolée pourrait être un meilleur nom pour ce type.

1.8.2 **Classe de mots connectés**

Les systèmes de mots connectés (ou plus exactement les "énoncés connectés") sont similaires aux mots isolés, mais permettent aux énoncés séparés d'être "combinés" avec une pause minimale entre eux.

1.8.3 **Classe de discours continu**

La prochaine étape est la reconnaissance continue. Il est extrêmement difficile de créer des systèmes de reconnaissance dotés de capacités de parole continue, car ils doivent utiliser des méthodes spéciales pour déterminer les limites de l'énoncé. Les reconnaisseurs de parole en continu permettent aux utilisateurs parler presque naturellement, alors que l'ordinateur détermine le contenu. Fondamentalement, il s'agit de dictée informatique.

1.8.4 Classe de discours spontané

La parole spontanée a un grand nombre de définitions. À la base, cela peut être considéré comme un discours qui sonne naturellement et qui n'est pas répété. Le locuteur commence un énoncé et atteint un point où il ne peut pas trouver le mot juste ou pense mieux à un mot, et a besoin de temps pour trouver une alternative appropriée, ils répètent un mot comme une sorte de "course" à la deuxième tentative. Un système ASR avec une capacité de parole spontanée devrait être capable de gérer une variété de caractéristiques de parole grammaticale non standard telles que des mots courir ensemble, "امممم" et "ايييه", et même de légers bégaiements.

1.9 Classification d'un ASR

Classification automatique du système de reconnaissance vocale:

L'arborescence suivante met l'accent sur les applications de traitement de la parole Selon le critère choisi, les systèmes de reconnaissance automatique de la parole peuvent être classés comme indiqué à la figure 2.1 [45] pour de nombreuses classes, Le mode de parole définissant le type d'énoncé qui sera utilisé comme isolement ou énoncé continu.

Le mode de conversation comprend trois types de mode de locuteur utilisés dans les systèmes: dépendants, indépendants et adaptatifs. Chaque système de reconnaissance vocale doit gérer un corpus d'entrée, la taille du corpus pouvant être petite, moyenne ou grande.

La dernière classification est le style de conversation qui détermine si le système sera considéré comme un système de dictée ou spontané. Dans notre système, nous couvrons la parole continue et le locuteur indépendant en utilisant un vocabulaire volumineux, le style de conversation étant la dictée.

Dans notre cas on choisit un système qui traite une dictée de la parole continue indépendant du locuteur en se basant sur un grand vocabulaire

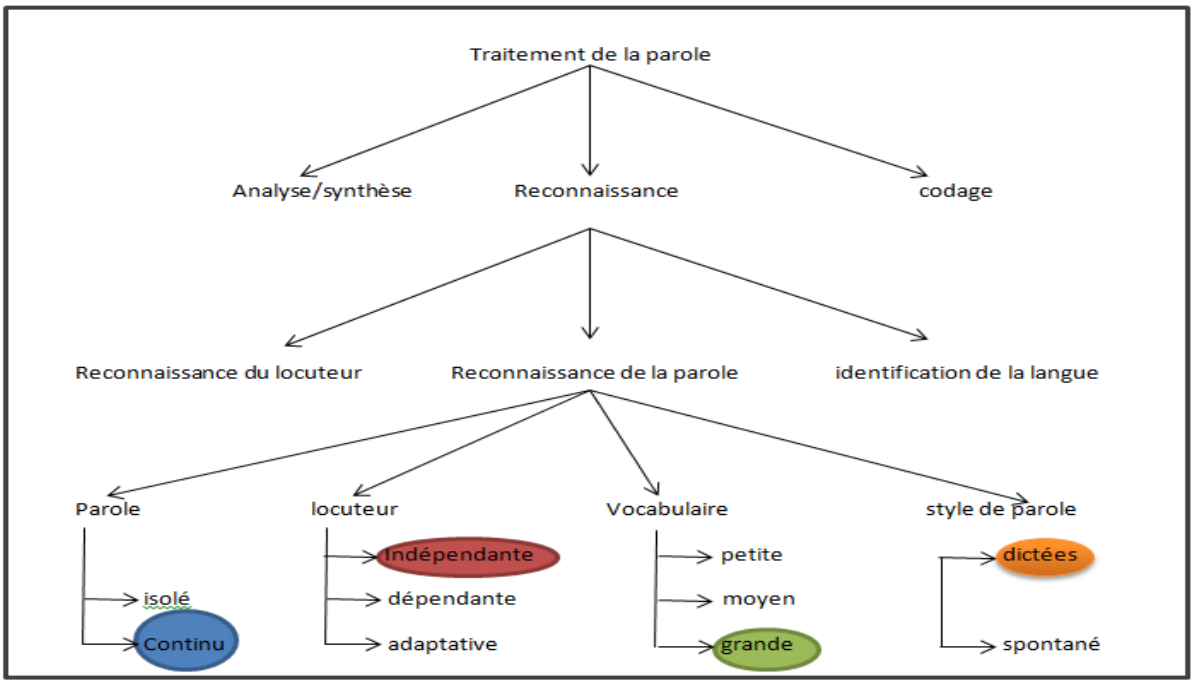


Figure 1.1 : la classification de traitement de la parole

1.10 Utilisations et applications

Bien que toute tâche impliquant une interface avec un ordinateur puisse potentiellement utiliser la récupération automatique du système, les applications suivantes sont les plus courantes actuellement.

1.10.1 La dictée

Aujourd'hui, la dictée est l'utilisation la plus courante des systèmes ASR. les transcriptions , la dictée juridique et commerciale, ainsi que le traitement de texte général sont parmi les exemples les plus importants d'utilisations de la dictée. Dans certains cas, des vocabulaires spéciaux sont utilisés pour augmenter la précision du système.

1.10.2 Le contrôle et commandes

Les systèmes ASR conçus pour exécuter des fonctions et des actions sur le système sont définis comme des systèmes de commandement et de contrôle. Des énoncés comme "ouvrir chrome" feront exactement cela.

1.10.3 Téléphonie

Certains systèmes vocaux permettent aux appelants de prononcer des commandes au lieu d'appuyer sur des boutons pour envoyer des tonalités spécifiques.

1.10.4 Médical / handicap

De nombreuses personnes ont du mal à taper à cause de contraintes physiques telles que des microtraumatismes répétés, la dystrophie musculaire et bien d'autres. Par exemple, les personnes malentendantes pourraient utiliser un système connecté à leur téléphone pour convertir le discours de l'appelant en texte.

1.10.5 Applications embarquées

Certains téléphones cellulaires plus récents incluent la reconnaissance vocale qui permet des énoncés tels que "Appeler maison". Cela pourrait être un facteur majeur dans l'avenir d'ASR et de Linux [43].

1.11 Matériel (Hardware)

1.11.1 Cartes son

Le son doit être activé dans votre noyau et les pilotes appropriés doivent être installés. Étant donné que la parole nécessite une bande passante relativement faible, pratiquement toute carte son 16 bits de qualité moyenne pourra faire l'affaire. La qualité de la carte son entame souvent une discussion animée sur son impact sur la précision et le bruit.

Les cartes son avec les conversions A / N (analogiques-numériques) les plus « propres » sont recommandé, mais le plus souvent, la clarté de l'échantillon numérique dépend davantage de la qualité du microphone et encore plus du bruit de l'environnement. Les "bruits" électriques provenant des moniteurs, des logements PCI, des disques durs, etc. ne sont généralement rien comparés au bruit audible des ventilateurs d'ordinateur, des chaises qui grincent des dents ou une respiration difficile.

On doit peser les avantages et les coûts si on envisage des packages nécessitant un matériel spécifique pour fonctionner correctement. Certains logiciels ASR peuvent nécessiter une carte son spécifique. En règle générale, il faut évite les configurations matérielles spécifiques, car cela limite bon nombre de nos futures options et décisions.

1.11.2 Microphones

La qualité du microphone est très importante. C'est la clé lors de l'utilisation de l'ASR. Un microphone de bureau ne fera tout simplement pas le travail dans la plupart des cas. Parce qu'ils ont tendance à capter davantage de bruit ambiant, ce qui rend les programmes ASR difficiles.

Les microphones à main ne sont pas non plus le meilleur choix car ils peuvent être encombrants à prendre tout le temps. Bien qu'ils limitent la quantité de bruit ambiant, ils sont particulièrement utiles dans les applications nécessitant de changer souvent de haut-parleur ou lorsque la conversation avec le dispositif de reconnaissance n'est pas effectuée fréquemment (lorsque le port d'un casque n'est pas une option).

Le style de casque est le meilleur choix et de loin, il est le plus commun. Cela minimise le bruit ambiant, tout en vous permettant d'avoir le microphone au bout de la langue tout le temps. Les casques sont disponibles sans écouteurs et avec des écouteurs (mono ou stéréo). Nous recommandons les écouteurs stéréo, mais c'est juste une question de goût personnel [43].

1.11.3 Ordinateurs / Processeurs

La vitesse de traitement est très importante dans toutes les applications en général. ASR les applications dépendent fortement de la vitesse de traitement. Et c'est parce qu'une grande quantité de filtrage numérique et de traitement du signal peut avoir lieu en ASR. C'est le logiciel le plus rapide et le plus performant. Le plus de mémoire le mieux aussi. La plupart des progiciels répertorient leurs exigences minimales, en raison du traitement requis Pour un traitement rapide (dictionnaires volumineux, schémas de reconnaissance complexes ou taux d'échantillonnage élevés), vous devez utiliser une mémoire vive minimale de 400 MHz et de 128M de RAM.

1.12 A l'intérieure de la reconnaissance vocale

1.12.1 Comment fonctionnent les Reconnaissance

Les systèmes de reconnaissance peuvent être divisés en deux types principaux. Les systèmes de reconnaissance de formes comparent les formes à des formes connues / formées pour déterminer une correspondance. Les systèmes phonétiques acoustiques utilisent les connaissances du corps humain (production de la parole, et audition) pour comparer les caractéristiques de la parole (phonétique telle que les voyelles). La plupart des systèmes modernes se concentrent sur l'approche de reconnaissance des formes car elle s'associe parfaitement aux techniques informatiques actuelles et tend à avoir une précision supérieure.

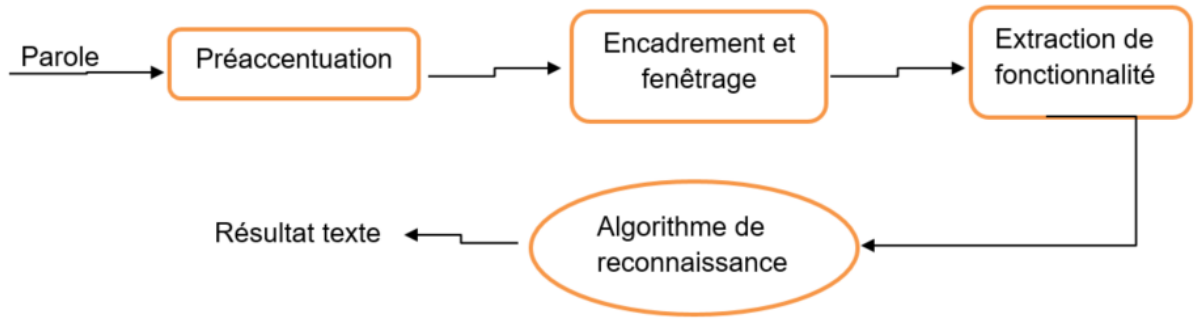


Figure 1.2 Processus général de reconnaissance

La plupart des dispositifs de reconnaissance peuvent être décomposés en une étape suivante, illustrée à la figure 2.2:

1. Enregistrement audio et détection de prononciation

2. Pré-filtrage (préaccentuation, normalisation, bandes, etc.)

3. Cadrage et fenêtrage (découpage des données dans un format utilisable)

4. Filtrage (filtrage supplémentaire de chaque fenêtre / cadre / bande de fréquences)

5. Comparaison et correspondance (reconnaître l'énoncé)

6. Action (fonction associée au motif reconnu)

Bien que chaque étape semble simple, chacune peut impliquer une multitude de techniques différentes (et parfois totalement opposées).

(1) Enregistrement audio / énonciation: peut être réalisé de différentes manières. Les points de départ peuvent être trouvés en comparant les niveaux audio ambiants (énergie acoustique dans certains cas) avec l'échantillon qui vient d'être enregistré. La détection des points finaux est plus difficile, car les locuteurs ont tendance à laisser des « artefacts », notamment la respiration / les soupirs, le bavardage des dents et les échos.

(2) Le pré filtrage: s'effectue de différentes manières, en fonction d'autres caractéristiques du système de reconnaissance. Les méthodes les plus courantes sont la méthode "Bank-of-Filtres" qui utilise une série de filtres audio pour préparer l'échantillon et la méthode de codage prédictif linéaire qui utilise une fonction de prédiction pour calculer les différences (erreurs). Différentes formes d'analyse spectrale sont également utilisées.

(3) Le cadrage / fenêtrage implique la séparation des données de l'échantillon en tailles spécifiques. Ceci est souvent intégré aux étapes 2 ou 4. Cette étape implique également la préparation des limites de l'échantillon pour l'analyse (suppression des clics de bord, etc.).

(4) Le filtrage supplémentaire n'est pas toujours présent. C'est la préparation finale de chaque fenêtre avant la comparaison et l'appariement. Cela consiste souvent en un alignement temporel et une normalisation.

(5) Il existe un grand nombre de techniques disponibles pour la comparaison et l'appariement. La plupart impliquent la comparaison de la fenêtre en cours avec des échantillons connus. Certaines méthodes utilisent les modèles de Markov cachés (HMM), l'analyse de

fréquence, l'analyse différentielle, les techniques / raccourcis d'algèbre linéaire, la distorsion spectrale et la distorsion temporelle. Toutes ces méthodes sont utilisées pour générer une correspondance de probabilité et de précision.

(6) Les actions peuvent être à peu près tout ce que le développeur souhaite

1.12.2 Principes de base de l'audio numérique

Le son est intrinsèquement un phénomène analogique. L'enregistrement d'un échantillon numérique s'effectue en convertissant le signal analogique du microphone en un signal numérique via le convertisseur A / N de la carte son. Lorsqu'un microphone est en marche, des ondes sonores font vibrer l'élément magnétique dans le microphone, provoquant un courant électrique sur la carte son. (Pensez à un haut-parleur travaillant en sens inverse). Fondamentalement, le convertisseur A / N enregistre la valeur de la tension électrique à des intervalles spécifiques. Il y a deux facteurs importants au cours de ce processus. Le premier est le "taux d'échantillonnage" ou la fréquence d'enregistrement des valeurs de tension. Deuxièmement, il y a les "bits par échantillon" ou la précision avec laquelle la valeur est enregistrée. Un troisième élément est le nombre de canaux (mono ou stéréo), mais pour la plupart des applications ASR, le mode mono suffit. La plupart des applications utilisent des valeurs prédéfinies pour ces paramètres et l'utilisateur ne doit les modifier que si la documentation le suggère.

Les développeurs doivent expérimenter différentes valeurs pour déterminer ce qui fonctionne le mieux avec leurs algorithmes.

Alors, quel est un bon taux d'échantillonnage pour ASR? Comme la voix est une bande passante relativement faible (généralement entre 100 Hz et 8 kHz), 8 000 échantillons / seconde (8 kHz) suffisent pour la plupart des ASR de base. Mais certaines personnes préfèrent 16 000 échantillons / sec (16 kHz), car elles fournissent des informations haute fréquence plus précises. Si vous avez la puissance de traitement, utilisez 16kHz. Pour la plupart des applications ASR, des taux d'échantillonnage supérieurs à 22 kHz environ sont une perte. Et quelle est la bonne valeur pour "bits par échantillon"? 8 bits par échantillon enregistreront les valeurs comprises entre 0 et 255, ce qui signifie que la position de l'élément de microphone est dans l'une des 256 positions. 16 bits par échantillon divisent la position de l'élément en 65536 valeurs possibles. Semblable à la fréquence d'échantillonnage, si vous avez assez de puissance de calcul et de mémoire, choisissez 16 bits par échantillon. À des fins de comparaison, un disque compact audio est codé avec 16 bits par échantillon à environ 44 kHz.

1.13 Langue arabe

L'arabe est une langue sémitique, c'est l'une des plus anciennes langues du monde. Actuellement, c'est la deuxième langue la plus parlée en termes de nombre de locuteurs [47]. L'arabe est la première langue du monde arabe, c'est-à-dire 25 pays. Les alphabets arabes sont utilisés dans d'autres langues que l'arabe, telles que le persan et l'ourdou [24]. Le nombre estimé d'arabophones est d'environ 300 millions. Cependant, un plus grand nombre de locuteurs ont une connaissance passive de l'arabe, langue d'enseignement de l'islam. Les approches récentes en matière de traitement de la langue et de la parole catégorisent la langue arabe en tant qu'arabe moderne

standard (MSA) et en arabe parlé moderne (MCA). L'arabe moderne standard est la forme de l'arabe utilisée dans l'éducation, les médias et les discussions formelles. L'arabe familier est ce qui est parlé dans les conversations quotidiennes et varie considérablement non seulement d'un pays à l'autre, mais également au sein d'un même pays. Il présente de nombreuses différences par rapport aux langues indo-européennes. Certaines des différences incluent des phonèmes uniques et des caractéristiques phonétiques, ainsi qu'une structure de mot morphologique complexe [49]. L'arabe est comparativement beaucoup moins recherché que d'autres langues telles que l'anglais et le japonais. La plupart des études rapportées à ce jour portent sur le traitement numérique de la langue arabe et de la parole en général.

Un nombre limité d'études ont été menées sur les versions MSA, classiques et coraniques de l'arabe. L'arabe standard moderne (MSA) compte 36 phonèmes, dont six voyelles et deux diphtongues et 28 consonnes. En plus des deux diphtongues, les six voyelles sont / a, i, u, a:, i :, u: / où les trois premières sont des voyelles courtes et les trois dernières sont leurs versions plus longues correspondantes (c'est-à-dire, les trois les voyelles courtes sont / a, i, u / et leurs trois équivalents longs sont / a:, i :, u: /) [48]. La langue arabe a moins de voyelles que la langue anglaise, l'anglais américain ayant douze voyelles. Un phonème est le plus petit élément d'unités de parole qui fait la différence dans la signification d'un mot ou d'une phrase. Les phonèmes arabes contiennent deux classes distinctes, appelées phonèmes pharyngé et emphatique. Ces deux classes ne peuvent être trouvées que dans d'autres langues sémitiques telles que l'hébreu [34]. Les syllabes autorisées dans la langue arabe sont: CV, CVC et CVCC où V indique une voyelle (longue ou courte) tandis que C indique une consonne.

Tableau 1.1: Modèle de syllabes arabes

	ouvrir	Exemple	ferme	exemple
court	CV	با		
long			CVC	جيم
			CVCC	عين

Toutes les syllabes arabes doivent contenir une voyelle. De plus, les voyelles arabes ne peuvent pas être des mots initiaux et doivent apparaître entre deux consonnes ou à la position finale des mots. Les syllabes arabes peuvent être classées comme courtes ou longues. Le type de syllabe CV est court alors que tous les autres sont longs. Les syllabes peuvent aussi être classées comme ouvertes ou fermées. Une syllabe ouverte se termine par une voyelle alors qu'une syllabe fermée se termine par une consonne. Pour l'arabe, une voyelle forme toujours un noyau syllabique, et il y a autant de syllabes dans un mot que de voyelles [50]. À de très rares exceptions près, la conversion alphabet-son en arabe permet généralement une simple correspondance un-à-un entre graphèmes et phonèmes pour un texte correctement dicritisé correct [51].

1.14 Ensemble de phonèmes arabes

Afin de produire un système de reconnaissance vocale automatique, arabe, continu, robuste et indépendant du locuteur, il est nécessaire de disposer d’un ensemble d’enregistrements vocaux

riches et équilibrés. La caractéristique riche est en ce sens qu'elle doit contenir tous les phonèmes de la langue arabe. Il faut également équilibrer la préservation de la distribution phonétique de la langue arabe. Cet ensemble d'enregistrements de discours doit être basé sur un ensemble écrit de phrases et de phrases créées par des experts. Par conséquent, il est crucial de créer un ensemble écrit (texte) de haute qualité de phrases et de phrases avant de les enregistrer.

Pour créer un corpus de texte phonétiquement riche et équilibré, vous devez sélectionner un ensemble de mots riches en phonétique, qui sont combinés pour produire des phrases et des phrases. Ces phrases et expressions sont vérifiées et vérifiées pour une distribution phonétique équilibrée. [64]. En 1997, KACST a créé une base de données pour les sons en langue arabe. Le but de ce travail était de créer le moins de mots arabes riches en phonétique. En conséquence, une liste de 663 mots riches en phonogrammes contenant tous les phonèmes arabes, qui sont soumis à toutes les règles de phono tactique arabe, a été produite. Ce travail est l'épine dorsale de la création de phrases individuelles, utilisables pour les applications de synthèse ASR en arabe et de synthèse de texte à parole. La liste de 663 mots phonétiquement riches a été créée sur la base des caractéristiques et directives suivantes:

1. Couverture de tous les phonèmes arabes qui doivent être équilibrés de manière à être aussi proches que possible en fréquence.
2. Couverture de tous les groupes de phonèmes arabes.
3. La présence du plus petit nombre possible de mots pour que la liste ne contienne pas un seul mot dont le but d'existence est atteint par un autre mot de la même liste.

En 2003, KACST a établi un rapport technique sur le projet, intitulé Base de données des sons arabes: phrases., Des phrases indépendantes en arabe ont été écrites à l'aide des 663 mots riches en prononciation. La base de données comprend 367 phrases; 2 à 9 mots par phrase. Par conséquent, nous avons des phrases et des phrases en arabe riches et équilibrées sur la base de la liste précédemment créée de 663 mots riches en richesse, qui ont été mises en phrases et en tenant compte des objectifs suivants [41]:

- Avoir le moins possible de répétitions de mots.
- Avoir des phrases structurellement simples afin de faciliter la lecture et la prononciation.
- Avoir le nombre minimum de phrases.

Une moyenne de 2 mots riches en transcription et de 5 autres mots ont été utilisés dans chaque phrase. L'analyse statistique montre que 1333 mots ont été répétés une seule fois et 99 mots ont été répétés plus d'une fois sur 367 phrases au total, alors que 17 mots ont été répétés 5 fois et plus. Les phrases KACST 367 phonétiquement riches et équilibrées sont utilisées à des fins de formation dans notre système. Le tableau 1.2 présente des détails plus techniques de notre corpus de parole [42].

Tableau 1.2: Critères de la base de données

Critères	Données
Nombre de phrases	367 phrases
Nombre de mots	1435 mots

Nombre moyen de mots / phrase	5 mots
Min. et Max. Nombre de mots / phrase Min	de 2 et max. de 9
Nombre de locuteurs	40 locuteurs
Haut-parleurs de	18 à 66 ans
Genre	20 hommes et 20 femmes
Moyenne no. de fichiers son / phrase	100 fichiers son / phrase
Taux d'échantillonnage (Hz)	44,1 KHz
Nombre de bits	16 bits

1.15 Architecture de Système la reconnaissance automatique de la parole

La parole étant l'un des moyens de communication les plus importants entre les humains, le traitement de la parole est devenu un sujet intéressant pour les chercheurs et les ingénieurs [75]. La reconnaissance vocale est une technologie utilisée pour convertir le signal vocal en mots compréhensibles ou en séquences de mots compréhensibles. ASR est le domaine d'étude des signaux de parole et des moyens de traiter ces signaux. Pour obtenir ce signal, les humains utilisent leurs cordes vocales pour produire une quantité de son ou de parole, puis ils enregistrent ce son à l'aide d'un microphone de haute qualité. Après cela, la parole passe par un système de reconnaissance vocale pour reconnaître et convertir ce signal en une série de mots (texte). Le schéma fonctionnel de la figure 1.3 illustre le mécanisme général de traitement de la parole.

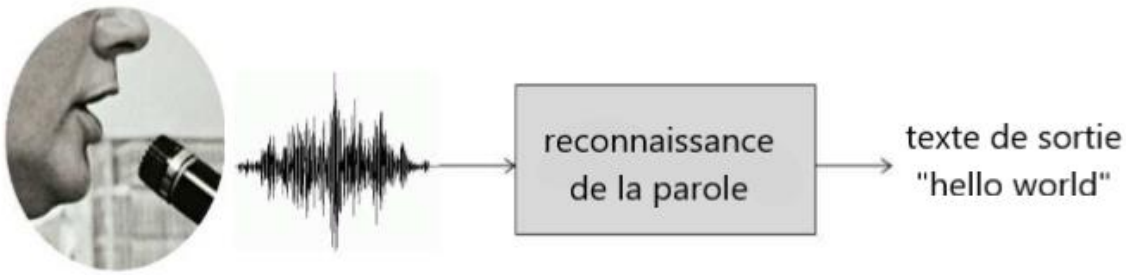


Figure 1.3 : Mécanisme du système de reconnaissance vocale

Une fois que les humains ont converti leur son en signal vocal, le système de reconnaissance vocale commence à traiter ce signal à travers différentes étapes pour produire le texte le plus probable. La figure 1.4 montre les étapes générales du traitement du système de reconnaissance vocale

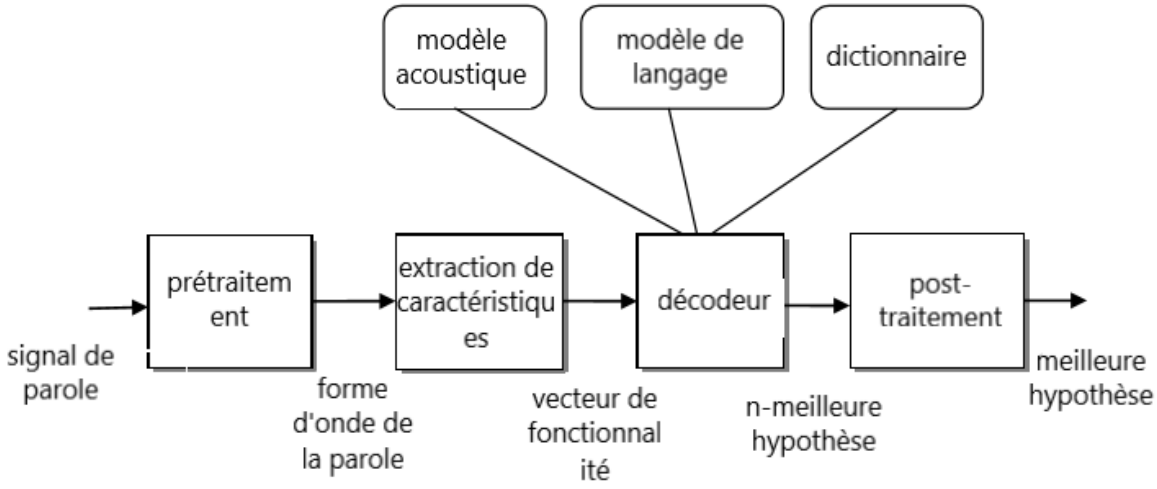


Figure 1.4 : Architecture d'un système de reconnaissance vocale typique

Au cours de la phase de prétraitement, le signal vocal subit des opérations de prétraitement pour l'améliorer, comme l'application de filtres de préaccentuation et la suppression ou la réduction du bruit. Après cela, le signal vocal passe par une étape d'extraction de caractéristiques pour obtenir des caractéristiques parcimonieuses utiles pour les étapes suivantes. La phase de décodage vise à trouver la correspondance la plus probable entre toute séquence de mots et le signal de vecteurs de caractéristiques en appliquant les modèles de langage et d'acoustique considérés comme le cœur des systèmes de reconnaissance de la parole [76,77]. L'objectif de la dernière étape, l'étape de post-traitement, est d'obtenir l'hypothèse parfaite parmi l'hypothèse n-meilleure [76].

1.15.1 **Prétraitement**

L'objectif principal de la phase de prétraitement est de minimiser les effets négatifs de l'environnement sur le signal de parole et de travailler à son amélioration pour obtenir une meilleure reconnaissance de ce signal lors des prochaines étapes. Il existe plusieurs techniques et les opérations pourraient être effectuées à ce stade [79]. La conversion du signal vocal analogique en numérique est considérée comme l'une des étapes les plus importantes de cette étape. Cela peut être fait en échantillonnant le signal en utilisant une fréquence d'échantillonnage appropriée, telle que 16 kHz ou 8 kHz, et en appliquant le processus de quantification [79]. En outre, l'une des premières améliorations pouvant être apportées au signal à cette phase est le traitement de préaccentuation. Le filtre de préaccentuation a pour objectif d'augmenter l'amplitude des signaux à hautes fréquences et réduire l'amplitude des signaux de basses fréquences (par exemple 100Hz) [76,77].

En général, le système ASR normal s'attache à attribuer une probabilité quelconque à chaque son de la parole [76]. Ainsi, tout type de bruits pouvant survenir pendant l'enregistrement pourrait conduire à l'insertion d'un ou plusieurs mots dans les hypothèses de sortie. La phase de prétraitement tente d'éviter cela en appliquant l'opération de segmentation parole / non-parole. Cette opération supprime une partie de l'enregistrement, telle que la partie située entre le début de

l’enregistrement et le moment où la parole commence, et la fin de l’enregistrement à la fin de la parole [76]. Cette méthode de segmentation est connue par la détection de point final.

1.15.2 Extraction de fonctionnalités

L'étape d'extraction de caractéristiques est la plus importante de tout le processus, car elle est chargée d'extraire les informations pertinentes des trames de parole, en tant que paramètres de caractéristiques ou vecteurs. Les paramètres couramment utilisés dans la reconnaissance vocale sont les coefficients de codage prédictif linéaire (LPC) et les coefficients de fréquence de Mel (MFCC). Ces paramètres ont été largement utilisés dans les systèmes de reconnaissance, en partie pour les raisons suivantes:

-Le calcul de ces paramètres conduit à une séparation source-filtre.

-Les paramètres ont un modèle analytiquement traitable.

-L'expérience prouve que ces paramètres fonctionnent bien dans les applications de reconnaissance.

En raison de leur importance, ils seront décrits dans deux sous-sections différentes.

1.15.2.1 Préaccentuation

Afin d’aplatir le spectre de la parole, un filtre de préaccentuation est utilisé avant l’analyse spectrale. Son objectif est de compenser la partie haute fréquence du signal de parole qui a été supprimée pendant le mécanisme de production de son humain. Le filtre le plus utilisé est un filtre passe-haut

1.15.2.2 Blocage de cadre et fenêtrage

Le signal de parole est divisé en une séquence de trames où chaque trame peut être analysée indépendamment et représentée par un seul vecteur de caractéristiques. Étant donné que chaque trame est supposée avoir un comportement stationnaire, un compromis pour utiliser le verrouillage de trame consiste à utiliser une fenêtre de 20-25 ms appliquée à des intervalles de 10 ms (cadence de 100 trames / s) et chevauchement de fenêtres. On peut le voir à la figure 1.5.

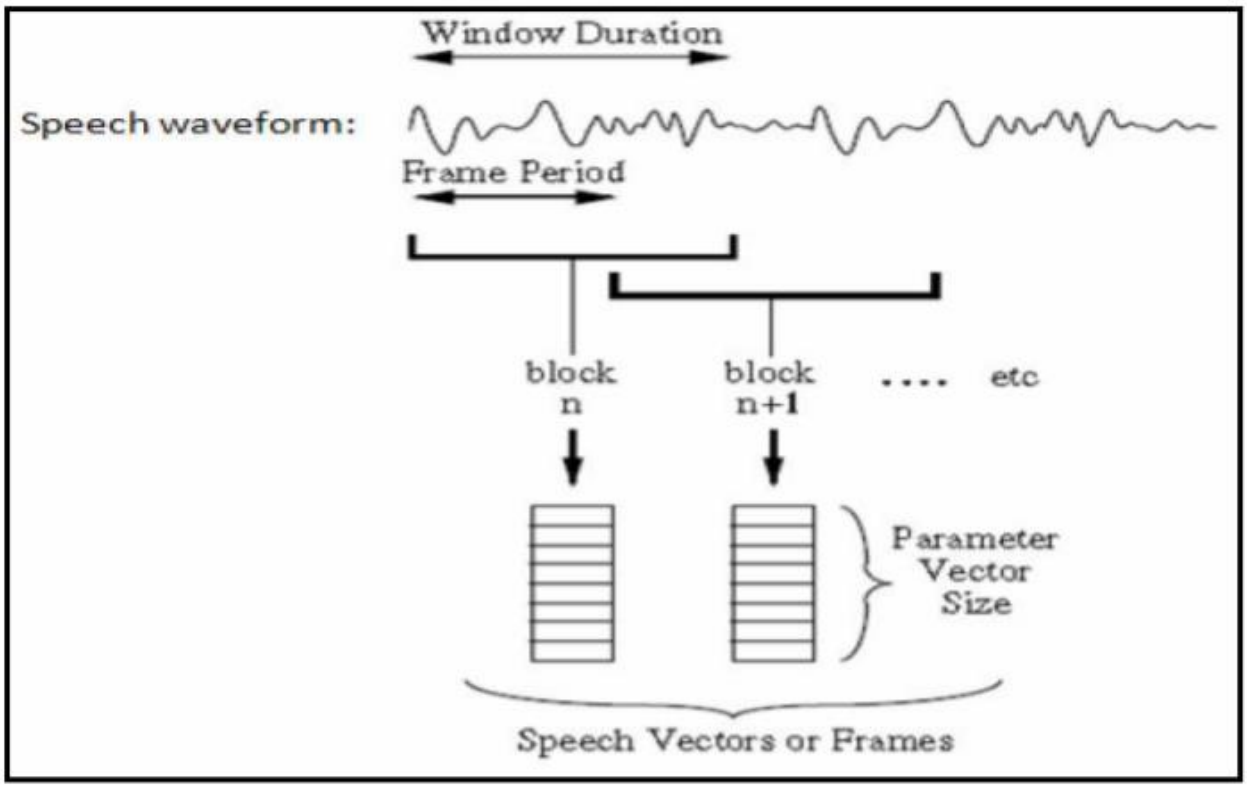


Figure 1.5 : Blocage du cadre

1.15.2.3 Mel- Cepstrum

La représentation des coefficients Mel Frequency Cepstrum6 (MFCC) en tant qu'approche bénéfique pour la reconnaissance de la parole. La fonction MFCC est une représentation du signal de parole défini comme étant le cepstre réel d'un signal à fenêtre à court terme dérivé de la transformée de Fourier rapide (FFT) de ce signal qui est soumis pour la première fois à une transformation de l'axe de fréquence (m -échelle de fréquence), puis décorrélée à l'aide d'une transformée en cosinus discrète modifiée (DCT-II). La figure 2.8 illustre le processus complet d'extraction des vecteurs MFFC du signal de parole. Il convient de souligner que le processus d'extraction MFCC est appliqué indépendamment à chaque trame de signal de parole.

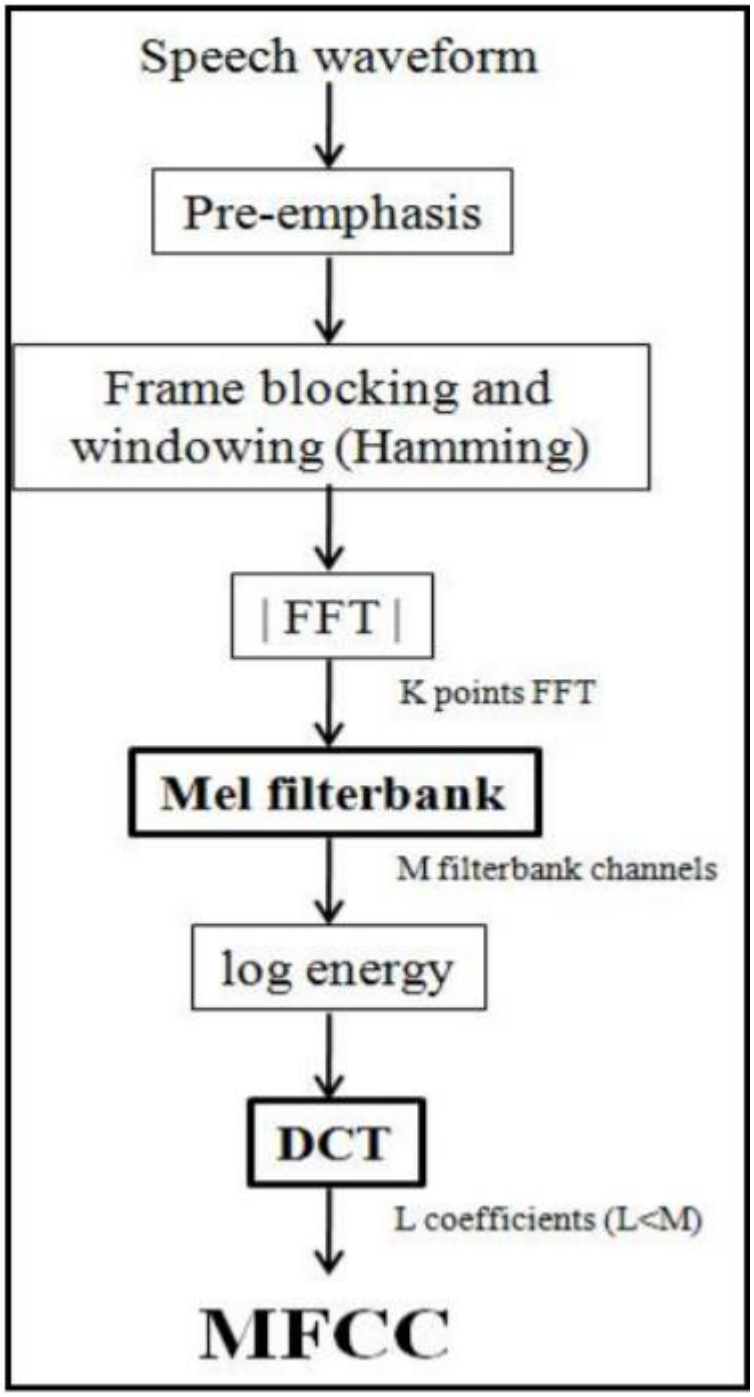


Figure 1.5 : Le processus d'extraction MFCC

+

Après la phase de préaccentuation et de blocage de trame et de fenêtrage, les vecteurs MFCC seront obtenus à partir de chaque trame de parole. Le processus d'extraction MFFC sera décrit ci-dessous en considérant à tout moment que toutes les étapes sont appliquées sur des trames de parole. La première étape du processus d'extraction MFCC consiste à calculer la transformée de Fourier rapide (FFT) de chaque trame et à obtenir sa magnitude.

La FFT est un algorithme efficace en calcul de la transformée de Fourier discrète (DFT). Si la longueur de la FFT est égale à une puissance de deux (K = 2 ^ n), un algorithme plus rapide peut être utilisé, ainsi un zéro-padding à la puissance la plus proche de deux dans la longueur de la trame vocale est effectué. L'étape suivante consistera à adapter la résolution de fréquence à une échelle de fréquence perceptuelle qui satisfait les propriétés de l'oreille humaine, telle qu'une échelle de fréquence mel perceptuelle. Ce problème correspond au stade de la banque de filtres Mel. L'analyse de la banque de filtres consiste en un ensemble de filtres passe-bande dont les largeurs de bande et les espacements sont à peu près égaux à ceux des bandes critiques et dont la plage des fréquences centrales couvre les fréquences les plus importantes pour la perception de la parole [52].

1.15.3 **Décodage**

La phase de décodage est considérée comme l'une des parties les plus importantes du système ASR. L'objectif principal de cette étape est de trouver les meilleures séquences de mots () pouvant correspondre au signal acoustique représenté par les composants des caractéristiques observées. L'opération de décodage utilise deux types de modélisation, à savoir les modèles acoustiques et langagiers, pour trouver la correspondance optimale. Trois types d'informations essentielles doivent être disponibles pour le processus de décodage:

• Un modèle acoustique et le modèle de Markov caché (HMM) pour chaque énoncé

• Un modèle de langage et des probabilités de séquences de mots.

• Un dictionnaire de prononciation contenant une liste de mots et leurs mots apparentés.

Phonèmes.

Le processus de décodage cherche à trouver la liste des mots pouvant être dits. Ces mots pourraient être trouvés dans la partie dictionnaire, qui est une liste de tous les mots possibles plus leurs phonèmes. Une autre partie importante est le modèle acoustique qui a une fonction de probabilité, à savoir le mélange gaussien et la vraisemblance de chaque composante observée P (O | W). Cependant, la modélisation du langage n'est pas très importante par rapport à la modélisation acoustique, mais elle est utilisée car elle permet d'élever la précision des mots et de

Corrigez la grammaire des mots.

Pour une meilleure compréhension, selon l'équation 5, le mécanisme de décodage tente de trouver la séquence de mots () la plus susceptible de correspondre au vecteur d'observation (O). De plus, la valeur du P (W) est obtenue par la modèle de langage, et la valeur de P (O | W) peut être calculée à l'aide du mot phonèmes disponible dans le

Dictionnaire de prononciation.

Il n'est pas possible de calculer les probabilités de tout le chemin disponible à travers le réseau d'états si l'espace des états est très grand. Par conséquent, certains algorithmes sont utilisés pour trouver les états cachés, tels que l'algorithme de recherche de Viterbi qui dépend de

La méthode de programmation dynamique [72].

1.15.4 Le modèle acoustique

Le modèle acoustique est l'un des processus les plus essentiels du système ASR et peut être considéré comme le cœur de l'opération de reconnaissance. L'objectif principal du modèle acoustique est d'améliorer la précision de la reconnaissance vocale en spécifiant la modélisation.

Unités et calculer la probabilité que les caractéristiques des caractéristiques acoustiques des unités phonétiques soient reconnues [78]. Il est très important de sélectionner les unités linguistiques, qui peuvent être des téléphones, des mots ou des sous-parties de téléphones, car l'utilisation de plusieurs unités peut affecter les performances du système ASR.

L'opération de modélisation acoustique peut être définie comme l'opération consistant à calculer la représentation statistique des vecteurs de caractéristiques observées du signal de parole. Selon [78], le modèle acoustique est comme un fichier qui collecte toutes les représentations statistiques de chaque phonème distinct du discours. De plus, ces représentations statistiques doivent être attribuées à une étiquette spéciale nommée par le « modèle acoustique des phonèmes » et obtenue à l'aide du corpus de parole et d'algorithmes d'apprentissage spéciaux.

Plusieurs méthodes sont utilisées pour construire le modèle acoustique, telles que le modèle de Markov caché (HMM), les champs aléatoires conditionnels, les modèles segmentaires et les modèles à entropie maximale. Cependant, le modèle de Markov caché HMM (expliqué dans ce chapitre) est considéré comme l'un des modèles statistiques les plus puissants qui soit utilisé de manière intensive. Dans le domaine de la reconnaissance vocale.

Dans les systèmes ASR, la plupart des étapes du modèle acoustique utilisent le modèle de Markov caché avec le modèle de mélange gaussien (GMM) pour extraire les vecteurs de caractéristiques acoustiques [92]. Le GMM calcule les probabilités du vecteur d'observation P (o | q) pour chaque état HMM (q), en fonction du téléphone (o). Pour plus d'informations sur le modèle de mélange gaussien (GMM), veuillez consulter [92].

1.15.5 Modélisation linguistique

Selon la langue américaine, il existe de nombreuses séquences de mots ayant des phonèmes très similaires, tels que « Épave une belle plage » et « Reconnaître la parole ». Ces deux phrases ont presque la même prononciation, mais avec des significations différentes. En général, les humains n'ont aucun problème à reconnaître ces mots car ils savent déjà quels mots sont utilisables dans le contexte. Cependant, l'ordinateur n'aura pas la capacité humaine de reconnaître les sons correspondants, et le modèle de langage statistique est utilisé à cette fin pour le système ASR.

Le modèle de langage consiste à estimer la distribution des probabilités sur une séquence de mots. En d'autres termes, s'il existe une séquence de mots, w = (w1, w2…, wK), le modèle de langage utilise la distribution de probabilités du P (w) sur cette séquence. Selon to [80], le but principal de la modélisation du langage est de minimiser l'hypothèse du modèle acoustique et de la hiérarchiser en même temps. Ces résultats de probabilité sont regroupés avec les résultats de probabilité du modèle acoustique pour calculer la probabilité finale de la totalité transcription.

Il existe différents types de modélisation de langage utilisés dans le domaine ASR, mais le n-gramme est considéré comme l'une des méthodes les plus courantes. Le n-gramme utilise les mots précédents (n-1) pour estimer le mot suivant.

Ce processus d'estimation s'appelle un Markov, d'après le mathématicien Andrey Markov, qui a inventé ce processus qui indique que la probabilité de chaque mot s'appuie sur le mot précédent.

Le bi gramme et le trigramme sont des types courants de la modélisation en langage n-gramme utilisables dans la reconnaissance vocale. Le bi gramme peut être obtenu lorsque le n du n-gramme est égal à deux (n = 2) et le trigramme peut être réalisé lorsque le n du n-gramme est égal à trois (N = 3). Par conséquent, le n-gramme peut être obtenu en regardant un mot (n-1) dans le passé. L'équation suivante peut être utilisée pour calculer le n-gramme du système de reconnaissance de grands vocabulaires, Où n est compris entre 2-4.

1.15.6**Post-traitement**

Selon [76], la plupart des algorithmes de décodage, tels que l'algorithme de recherche de Viterbi, produisent un ensemble de toutes les séquences de mots possibles (cinq à dix hypothèses limitées), classées par leur score total. Cette liste de sortie s'appelle la liste n-best car elle contient les meilleures hypothèses parmi d'autres. Le but de l'étape de post-traitement est d'arranger cette liste et de choisir les meilleures hypothèses parmi les n meilleures hypothèses. Donc, les mots avec un score plus élevé seront la sortie de reconnaissance.

Les méthodes utilisées pour remarquer la liste utilisent une source d'informations différente, telle qu'un modèle de langage d'ordre supérieur.

1.16 La mesure d'évaluation des systèmes ASR

Les performances des systèmes ASR peuvent être évaluées selon différentes mesures, telles que l'erreur, la précision et la correction [76,78]. En général, les systèmes ASR ont trois types d'erreur communs: substitution, insertion et suppression. La substitution se produit lorsque

Le système reconnaît un mot différent du mot prononcé. L'insertion se produit lorsque les hypothèses de sortie contiennent un mot qui n'est pas prononcé par le locuteur. La suppression survient lorsque les résultats reconnus ont manqué des mots prononcés [76,77]. Habituellement, le système ASR utilisait le taux d'erreur de mot (WER) comme métrique de précision, défini comme suit:

$$\text{WER} = \frac{\text{I+S+D}}{\text{N}}$$

Où :

- N est le nombre de mots de référence,
- S est le nombre de substitutions (mots incorrectement reconnus),
- D est le nombre de suppressions (mots omis),
- I est le nombre d'insertions (mots ajoutés),

En outre, le terme de précision pourrait être mesuré avec la métrique WRR (Word Recognition Rate) et le taux d'erreur de phrase (SER), mais la plupart des systèmes de reconnaissance vocale utilisent généralement le WER comme métrique principale [78,77]. Le WRR peut être représenté comme:

$$\text{WRR=1-WER} = \frac{\text{N}-\text{S}-\text{D}-1}{\text{N}}$$

De plus, la métrique de correction peut être calculée comme suit:

$$\text{CORR} = \frac{N - D - S}{N}$$

2 Chapitre 2 : Modélisation

2.1 Introduction :

Modéliser un système avant sa réalisation permet de mieux comprendre le fonctionnement du système. C'est également un bon moyen de maîtriser sa complexité et d'assurer sa cohérence. Il existe plusieurs méthodes de modélisation : Merise, UML…
On a choisi UML, plus précisément le Diagramme de classes et les diagrammes de séquence et le diagramme de cas d'utilisation.

2.2 LE DIAGRAMME DE CAS D'UTILISATION :

Le diagramme de cas d'utilisation a pour but de donner une vision globale sur les interface de futur application. C'est le premier diagramme **UML** constitué d'un ensemble d'acteurs qui agit sur des cas d'utilisation et qui décrit sous la forme d'actions et des réactions, le comportement d'un système de point de vue utilisateur.
Acteur : un acteur est un utilisateur qui communiquer et interagir avec les cas d'utilisation du system. C'est une entité ayant un comportement comme une personne, system

System : cet élément fixe les limites du système en relation avec les acteurs qui l'utilisent (en dehors de system) et les fonctions qu'il doit fournir (a l'intérieur du system)

2.3 Les Diagramme de séquences

Les diagrammes de séquences permettent de décrire COMMENT les éléments du système interagissent entre eux et avec les acteurs :

- Les objets au cœur d'un système interagissent en s'échangent des messages.
- Les acteurs interagissent avec le système au moyen d'IHM (Interfaces Homme-Machine).

2.4 Les Diagramme de class :

Les diagrammes de classes permettent de spécifier la structure et les liens entre les objets dont le système est composé : ils spécifient QUI sera à l'œuvre dans le système pour réaliser les fonctionnalités décrites par les diagrammes de cas d'utilisation.

2.5 Identification les acteurs et leur rôle :

Un acteur est une personne qui a un rôle bien déterminé dans notre application :

Acteur	Rôle
utilisateur	Parler dans un microphone Chargé un fichier Vérifie le résultat(play wave file)

2.6 Diagramme de cas d'utilisation :

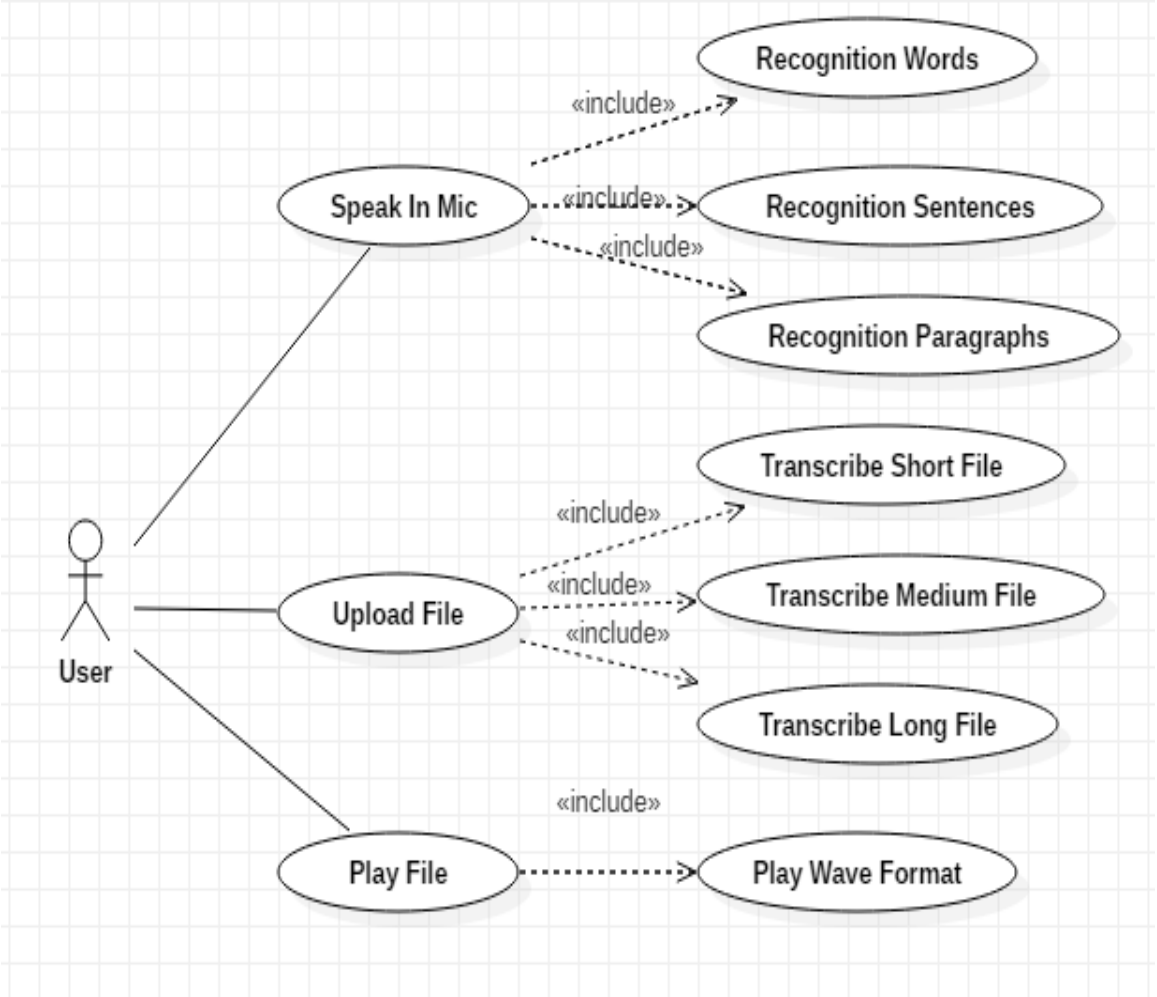


Figure 2.1 diagramme de cas d'utilisation ASR

2.7 Diagrammes de séquences :

2.7.1 **Reconnaissance de la parole par le microphone (speech recognition):**

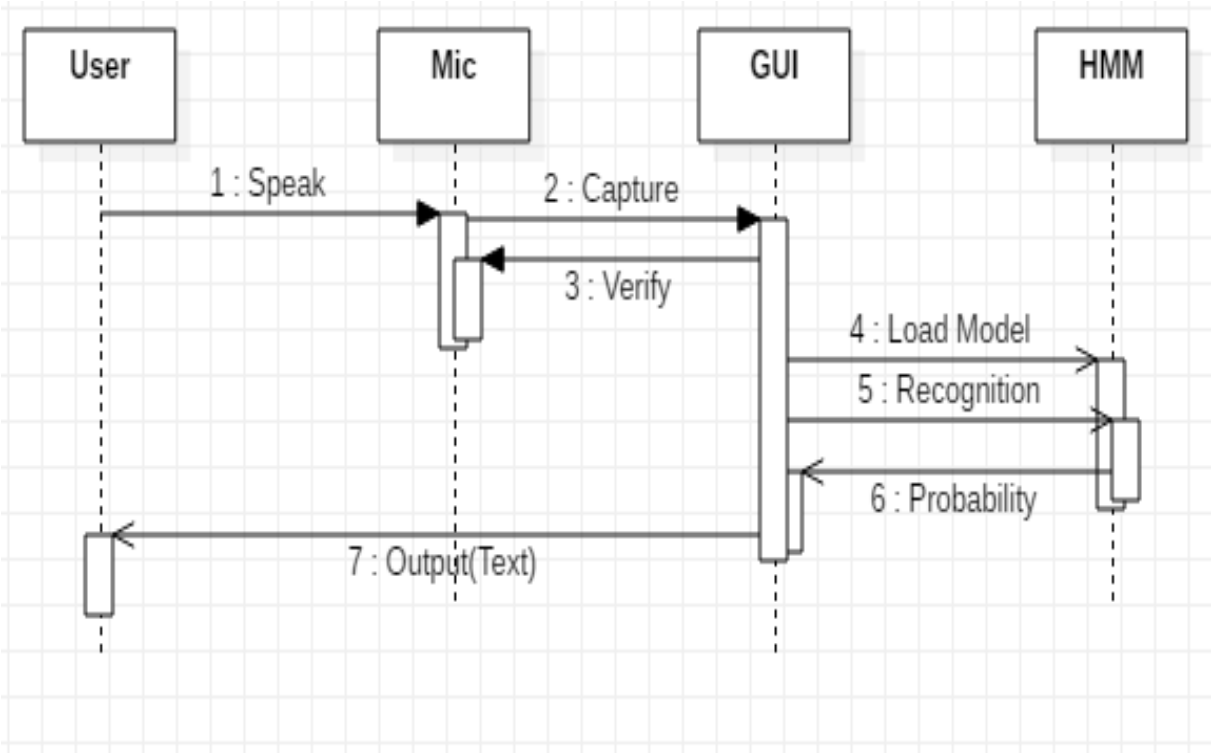


Figure 2.2 diagramme de séquence speech recognition

2.7.2 **Reconnaissance de la parole de fichier (audio recognition)**

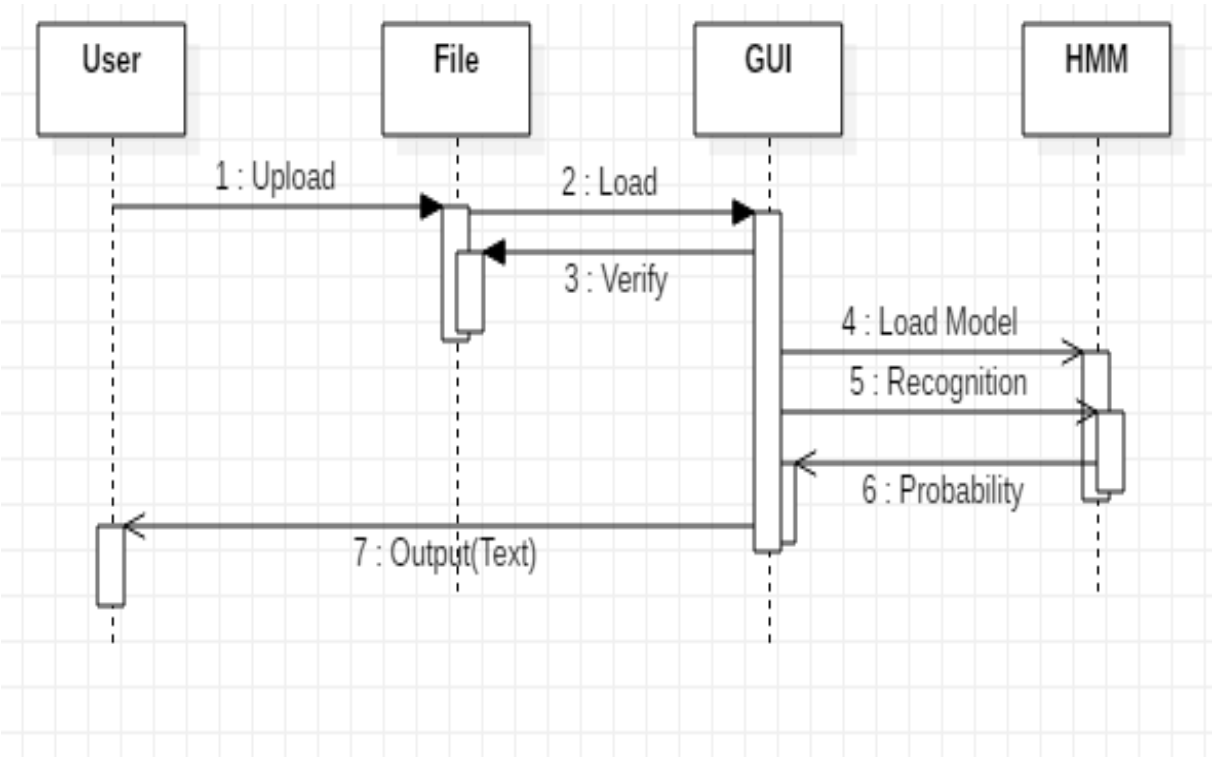


Figure 2.3 diagramme de séquence audio recognition

2.7.3 **Sélectionner et lire un fichier :**

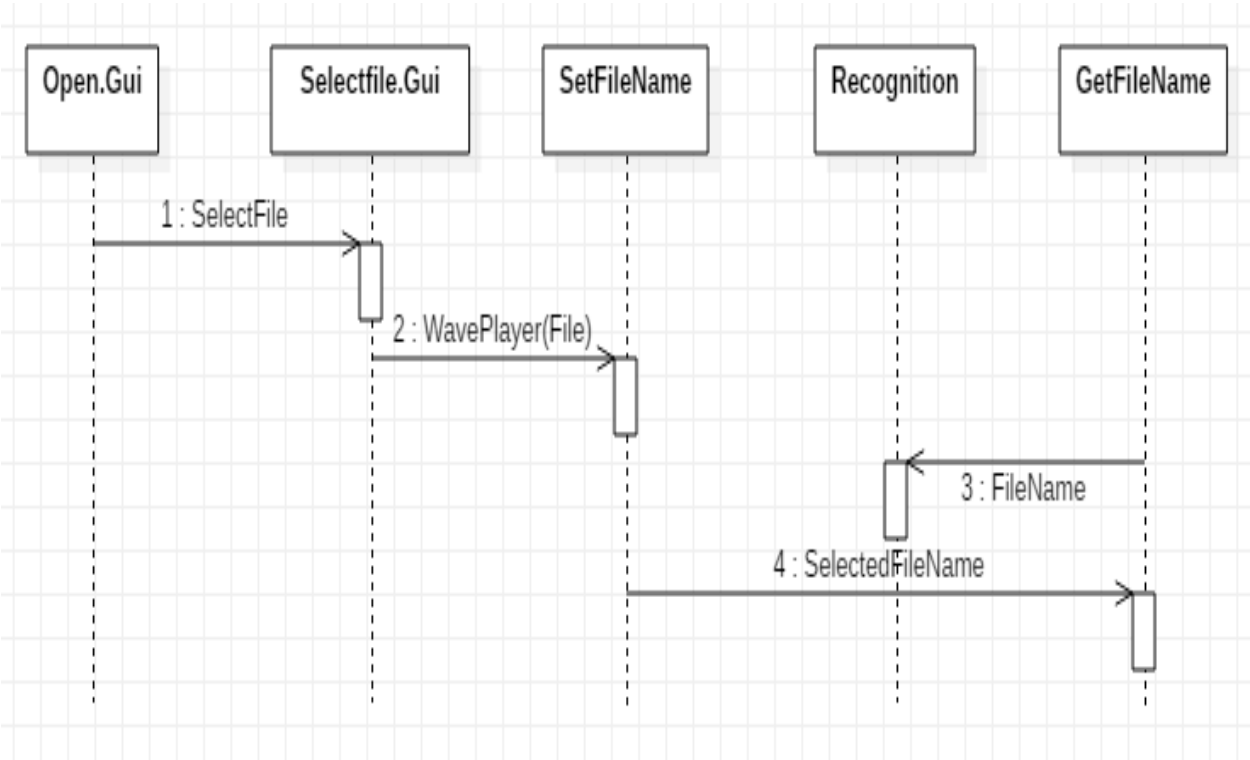


Figure 2.4 diagramme séquence select and Play file

2.8 Diagrame de class

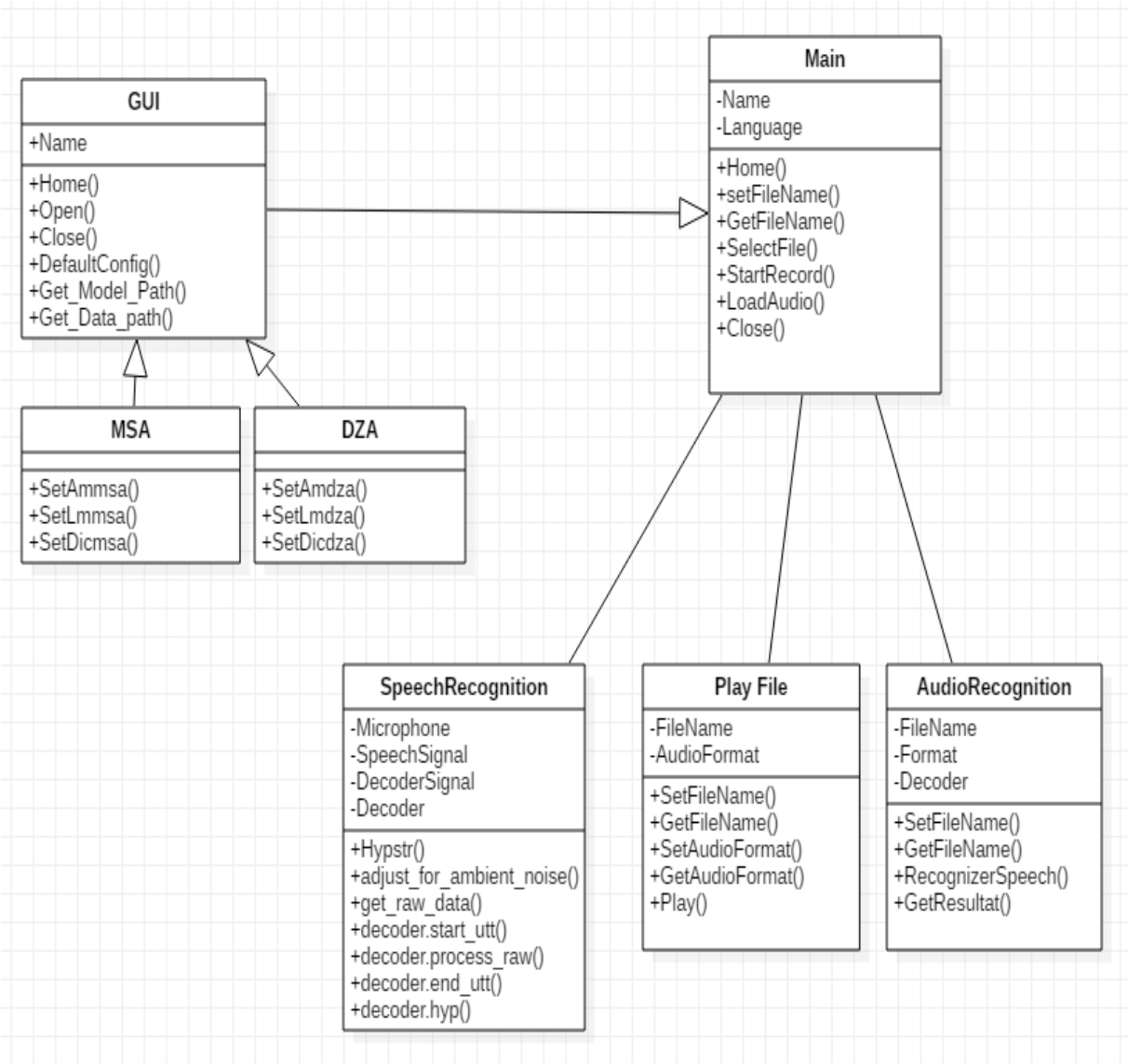


Figure 2.5 Diagramme de class ASR

- Ce diagramme contient 7 classes :
    1. La class principale de l'application
    2. La class GUI représente l'interface graphique de l'application.
    3. La classe MSA représente l'interface graphique hérité de la classe GUI qui est responsable de la reconnaissance de l'Arabe standard
    4. La classe MSA représente l'interface graphique hérité de la classe GUI qui est responsable de la reconnaissance de l'Arabe dialectale(algérien)
    5. La class speech recognition représente le mode de reconnaissance par le microphone.
    6. La class audio recognition représente la reconnaissance d'un fichier Wave (fichier audio).
    7. La class Play file représente la vérification de fichier qui vous avez transcrire et hypothèse de reconnaissance.

2.9 Conclusion :

Ce chapitre avait pour but d'expliquer les objectifs ainsi que la méthodologie suivie pour mener à bien ce projet de recherche. En effet, nous avons présenté, les trois diagrammes utilisés en s'appuyant sur la démonstration des acteurs et leur rôle et les déférentes entités.

3 Chapitre 3 : Réalisation et expérimentation

3.1 Comparaison toolkit

Nous avons comparé plusieurs toolkit pour savoir le quel est le mieux adapter à notre cas, on fait un résume dans ce tableau :

Tableau 3.1 comparaison des toolkit

TOOLKIT	LANGUAGE DE PROGRAMMATION	ACTIVITE DE DEVLOPEMENT	TUTORIELS ET EXEMPLES	COMMUNAUTE
CMU Sphinx	Java,C,Python,Autre	+++	+++	+++
Kaldi	C++,Python	+++	++	+++
HTK	C,Python	++	+++	++
Julius	C,Python	++	++	+
ISIP	C++	++	++	+

D'après les résultats on a conclus que CMU Sphinx est le toolkit le plus adapte à nos besoins.

3.2 Le travail proposé

Cette section décrit comment créer et développer un système de reconnaissance de la parole arabe en utilisant le Framework open source Sphinx. Les processus de formation et de reconnaissance utilisent des caractères arabes. Un système de reconnaissance de la parole arabe complet basé sur le système CMUSphinx, qui est basé sur MMC. Le système est une reconnaissance continue et indépendante du locuteur Il est capable de gérer de grands vocabulaires.

3.2.1 Les modèles de Markov cachés

Dans l'état actuel des connaissances, les modèles des Markov cachés (HMM) sont les outils de modélisation les plus utilisés en matière de reconnaissance de la parole. Leur application débouche sur d'excellents résultats quand ils sont bien appliqués [46]. Généralement, les SRAP sont composés de deux éléments essentiels : le modèle acoustique et le modèle du langage. Nous pouvons définir les deux modèles comme suit:

1- Le modèle acoustique regroupe l'ensemble des informations concernant la représentation phonétique, la variabilité de l'environnement du locuteur, du sexe du locuteur etc. [47].
2- Le modèle de langage a pour objectif de répondre aux contraintes du langage naturel, surtout lorsqu'il s'agit de mots qui ont la même prononciation, et améliorant ainsi leur décodage [48]. Théoriquement, nous pouvons expliquer ces deux modèles de la façon suivante:

Dans une approche stochastique, si nous considérons un signal acoustique S, le principe de la reconnaissance peut être expliqué comme le calcul de la probabilité P(W|S) qu'une suite de mots (ou phrase) W correspond au signal acoustique S, et de déterminer la suite de mots qui peut maximiser cette probabilité. En utilisant la formule de Bayes,

Un modèle HMM est défini par l'ensemble de données suivantes:

1- Une matrice A qui indique les probabilités de transition d'un état qi vers un autre état (ou vers lui-même), soit p(qj|qi) = aij.
2- Une matrice B qui indique les probabilités d'émission des observations dans chaque état. Si nous considérons le cas de la parole continue, cette probabilité est de type multi gaussienne, définie par les vecteurs moyens, les matrices de covariance et des poids associés à chaque gaussienne.
3- Une matrice ∏ donne la distribution de départ des états, c'est-à-dire pour chaque état la probabilité d'être atteint à partir de l'état initial qi. Cet état est particulier puisqu'il ne peut émettre d'observations.

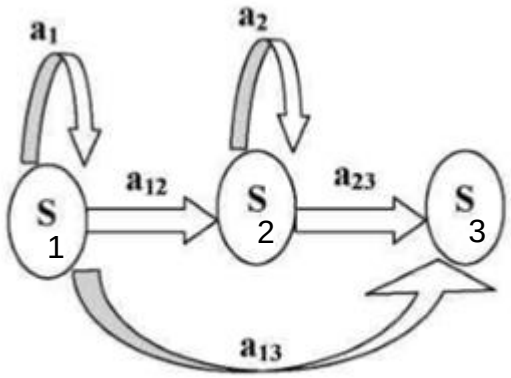


Figure 3.1 : Modèle HMM dit gauche-droit d'ordre 1 à 3 états

Cette structure rend possible la représentation des changements dus à la prononciation. En effet pour l'articulation lente, il y a répétition d'états. Il sera représenté dans un modèle HMM par une transition d'état sur lui-même (la transition ai sur la figure précédente). Alors que pour les articulations rapides, le modèle admet le saut à l'état suivant (transition a13).

Parmi les produits réussis dans le domaine de la reconnaissance de la parole, figure l'outil CMU Sphinx. Ses points forts résident, entre autres, dans son code ouvert qui offre une grande souplesse d'utilisation, et son appui sur le modèle HMM.

Notre approche pour la modélisation des sons arabes dans le système CMU Sphinx consiste à construire et à former les modèles acoustiques et de Language avec Données de la parole en arabe et générer le dictionnaire avec des caractères arabes La figure 4.1 présente une représentation mathématique du système de reconnaissance de la parole dans des équations simples contenant une unité frontale, une unité de modèle acoustique, une unité de modèle de langage et une unité de recherche.

L'approche standard de la reconnaissance vocale continue à vocabulaire étendu consiste à supposer un modèle probabiliste simple de production de la parole dans lequel une séquence de mots spécifiée, W, produit une séquence d'observation acoustique Y, avec une probabilité P (W, Y).

Le but est alors de décoder la chaîne de mots, en fonction de la séquence d'observation acoustique, afin que la chaîne décodée ait la probabilité maximale a posteriori (MAP).

3.2.2 **L'Outil CMU Sphinx**

Le système CMU Sphinx est une série d'outils servant à construire des applications de reconnaissance vocale. Au cours des dernières années, elle comprend également des ressources connexes, telles que des modèles acoustiques et des modèles de langages [49, 50].

Le CMU Sphinx comprend entre autres les outils suivants :

- Sphinx 2 : est un système de reconnaissance de la parole à grande vitesse. Il est habituellement employé dans des systèmes de dialogue et des systèmes d'étude de prononciation.

- Sphinx 3 : est un système de reconnaissance de la parole légèrement plus lent mais plus précis.

- Sphinx 4 : Une réécriture complète du Sphinx en Java. Il offre à la fois la précision et la rapidité.

-pocketSphinx : est une librairie permettant d'intégrer la reconnaissance vocale dans vos projets écrit en langage C à l'aide des fonctionnalités du projet open source

- SphinxTrain : Une suite d'outils qui permet de créer le modèle acoustique

- CMU-Cambridge Language Modeling Toolkit : Une suite d'outils qui permet de créer le modèle de langage

La figure III.2 montre l'évolution de l'outil CMU Sphinx

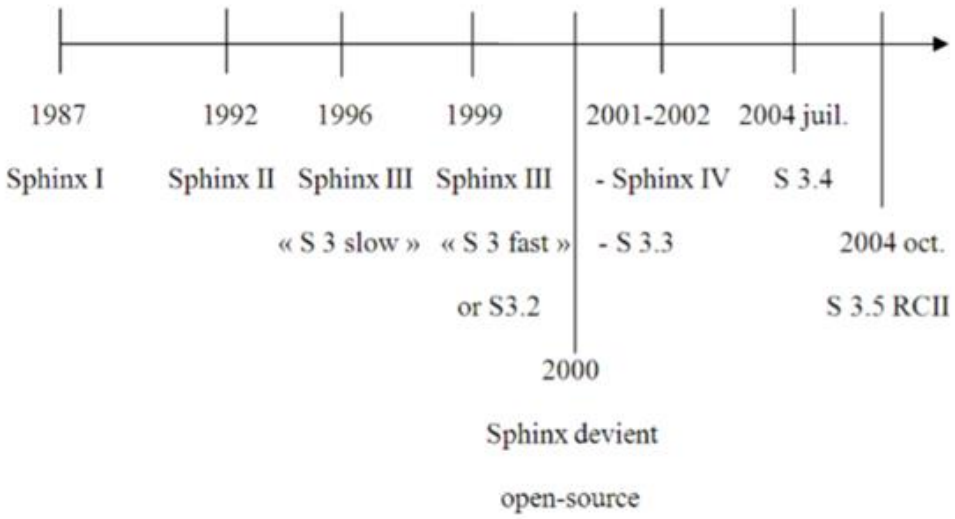


Figure 3.2 : L'évaluation des version de sphinx [51]

La disponibilité de code source pocketSphinx rend possible sa flexibilité et encourage la recherche dans les universités et les laboratoires spécialisés (HP, Sun...). Le logiciel pocketSphinx utilise à la fois le modèle acoustique et le modèle de langage pour décoder un signal acoustique.

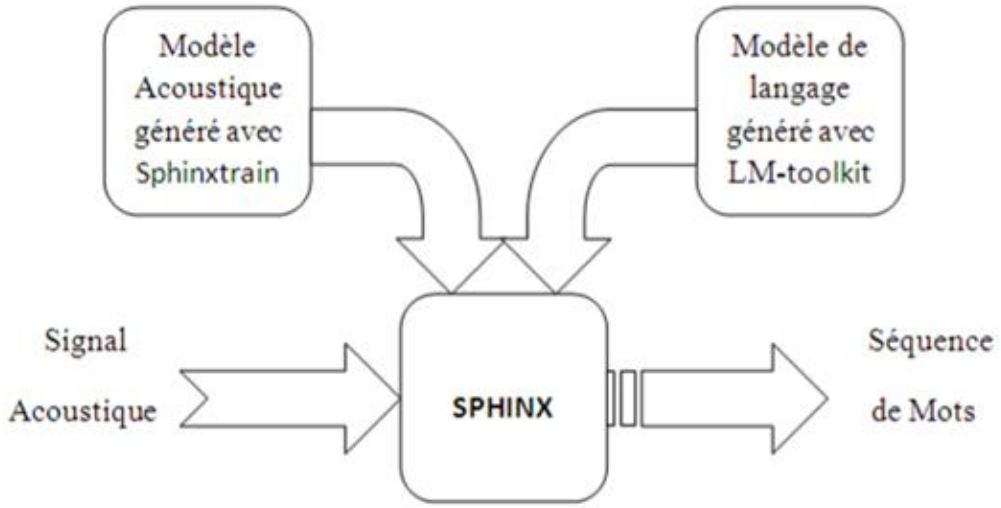


Figure 3.3 : Schéma simplifiant le traitement en utilisant le toolkit sphinx

3.2.2.1 Pocket Sphinx

CMU pocketSphinx est une librairie de classes et d'outils disponible en langage de programmation c. Cette librairie est gratuite à télécharger ; elle vise principalement à faciliter la construction des systèmes de reconnaissance vocale.

CMU pocketSphinx est un système de RAP basé sur les Modèles de Markov Cachés (HMM). Il a été créé conjointement par le groupe Sphinx à l'université CMU, les laboratoires Sun Microsystems et Hewlett-Packard company [52-54]. PocketSphinx utilise des HMM continus et fournit une grande flexibilité, exactitude et vitesse. PocketSphinx est modulaire, flexible, accepte différentes grammaires et langues. Il faut néanmoins trouver un équilibre entre l'exactitude et la vitesse en jouant sur les paramètres du fichier de configuration.

3.2.2.2 SphinxTrain

Est l'outil crée par CMU pour le développement des modèles acoustiques. C'est un ensemble de programmes et documentations pour réaliser et construire des modèles acoustiques pour n'importe quelle langue.

3.2.2.3 Architecture

PocketSphinx présente un ensemble d'outils de reconnaissance vocale (voir figure III.4) flexibles modulaires et extensibles formant un véritable banc d'essais et un puissant environnement de recherche pour les technologies de reconnaissance automatique de la parole

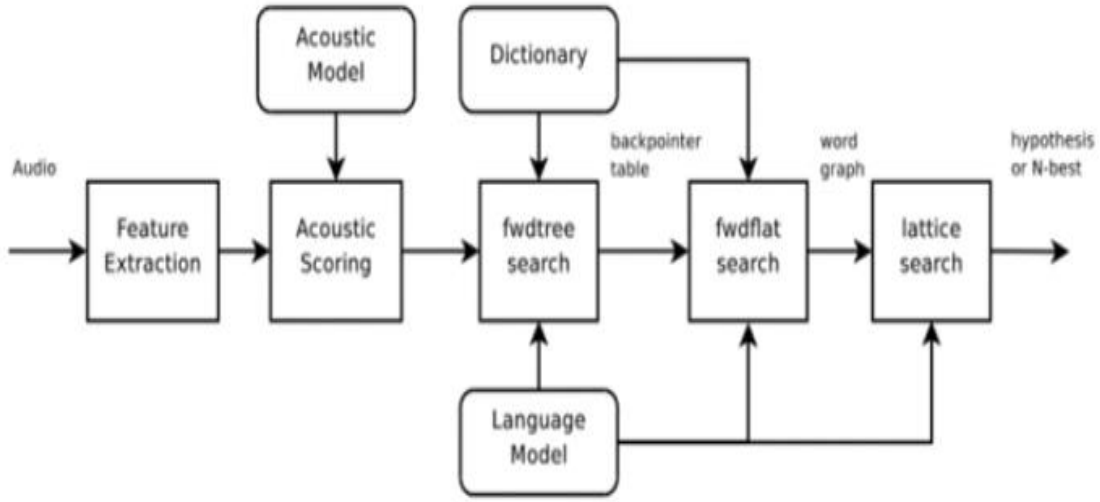


Figure 3.4 : architecture pocketSphinx

• FrontEnd : découpe la voix enregistrée en différentes parties et les prépare pour le décodeur. Il est responsable de la génération des vecteurs caractéristiques représentant les caractéristiques du signal vocal.

• Features : est utilisé pour estimer les paramètres du modèle acoustique.

• Linguist : ou base de connaissances qui est l'information qu'utilise le décodeur pour déterminer les mots et les phrases prononcées, elle est composée de :

– Dictionary : dictionnaire, définit les prononciations des mots trouvés dans le LanguageModel en utilisant l'AcousticModel.

– AcousticModel : modèle acoustique, un modèle statistique décrivant la distribution des données de phonèmes.

– LanguageModel : un modèle de langage, il donne la probabilité d'apparition d'un mot donné, basée sur des connaissances tirées du dictionnaire. Il permet de :

– décrire ce qui peut être dit dans un contexte bien spécial ;

– aider à rétrécir l'espace de recherche.

Il y a trois sortes de modèle de langage : le plus simple est utilisé pour les mots isolés, le deuxième pour les applications basées sur des commandes et des contrôles et le dernier pour le langage courant.

• SearchGraph : contient toutes les séquences de phonèmes possibles basées sur le LanguageModel.

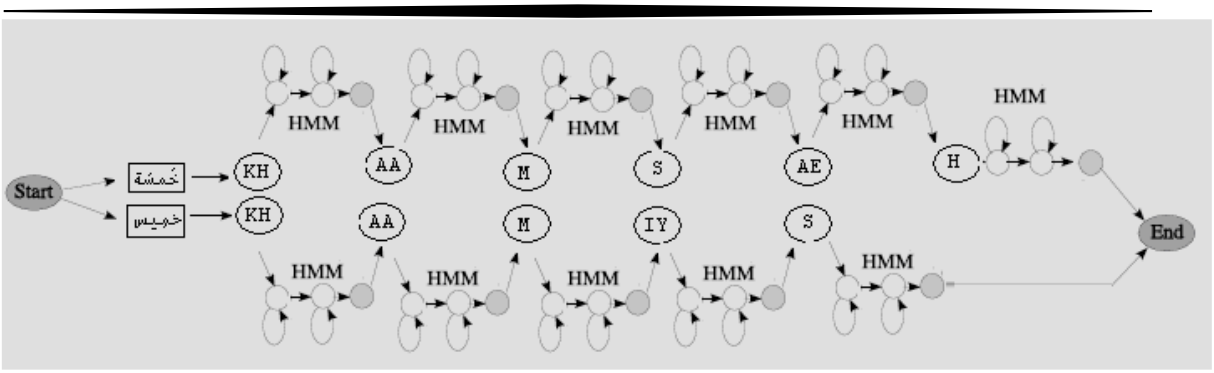


Figure 3.5 exemple de SearchGraph

• Decoder : ou Décodeur qui est le cœur de Sphinx ; c'est lui qui traite les informations reçues depuis le FrontEnd, il les analyse et les compare avec la base de connaissances pour donner un résultat à l'application.

• Configuration Manager : sert à configurer différents paramètres. La configuration peut se faire de manière dynamique pendant l'exécution de l'application.

3.2.2.4 Installation

3.2.2.5 Pocket Sphinx

PocketSphinx peut être téléchargé de l'internet soit sous forme binaire soit sous forme source code [58]. Il a été compilé et testé sur plusieurs versions de Linux et sur Windows. L'exécution de pocketSphinx demande des logiciels supplémentaires qui sont :

- Python.
- Les différentes librairies qui composent pocketSphinx
- Microsoft Visual Studio : Pour compiler les sources en C afin de produire les exécutables.

3.2.2.6 Sphinxtrain

SphinxTrain téléchargeable dont le lien se trouve dans Tools du site de CMU Sphinx [52]. Les différentes librairies qui composent SphinxTrain :

• Active Perl : L'outil pour éditer des scripts pour SphinxTrain et permet de travailler dans un Unix-like environnement pour Windows plateforme [61].

• Microsoft Visual Studio : Pour compiler les sources en C afin de produire les exécutables.

3.2.2.7 Implantation

La mise en œuvre de pocketSphinx se compose de:

1- Créer un nouveau model contient (hmm, lm, dictionnaire) ;
2- Insérer les composant de model dans le projet ;
3- La dernière étape consiste à écrire un code python dans le but de gérer et de déterminer les composantes à être utilisé dans le système.

3.3 Corpus

Dans notre projet on a utilisé deux corpus distinct :

3.3.1 Le corpus de l'arabe standard :

Le corpus est constitué des 620 phrase (2863 mots) de l'arabe standard 6 locuteurs, 3 males et 3 femelles, sont invités à prononcer les 620 phrase. Le corpus comprend six répétitions par chaque locuteur de la même phrase. Ainsi, le corpus est constitué de 3720 tokens (620 phrase. 6 locuteurs). Pendant l'enregistrement, chaque répétition a été rejouée pour s'assurer que le mot a été inclus dans le signal enregistré. Dans le tableau III.3 sont donnés certains paramètres d'enregistrement du corpus.

Tableau 3.2 : paramétré d'enregistrements utilisé pour la préparation du corpus arabe standard

Paramètre	Valeur
Echantillonnage	16 kHz , 16 bits
Wave format Corpus	Mono, Wave 620 phrase de l'arabe standard
Locuteur	6 (3 male + 3 femelle)

3.3.2 Le corpus de l'arabe dialectale :

On a Télécharger plusieurs fichier Vidéo open source qu'on a trouvée sur internet après les avoir convertie en fichier WAVE, on les a traitées manuellement en suppriment les bruits parasites et les musique et on les a segmentées en plusieurs phrase utilisable pour notre programme. Ce corpus et constitue de 388 phrases (3528 mots), il ne comprend pas des répétitions de la même phrase (chaque locuteur a des phrases différentes).

Tableau 3.3 paramétré d'enregistrements utilisé pour la préparation du corpus arabe dialectale

Paramètre	Valeur
Echantillonnage	16 kHz , 16 bits
Wave format Corpus	Mono, Wave 388 phrase de l'arabe dialectale

3.4 Les éléments nécessaires :

3.4.1 Le dictionnaire

Dans cette étape, nous avons mappé chaque mot du vocabulaire à une séquence d'unités sonores représentant la prononciation; qu'il contient tous les mots avec toutes les variantes possibles de leur prononciation.

Tenir compte de la variabilité de la prononciation, causée par diverses manières de parler et la spécificité de l'arabe. Une préparation minutieuse du dictionnaire phonétique empêche l'association incorrecte à un phonème avec des paramètres audio, ce qui aurait pour effet de diminuer la précision du modèle [65].

Le tableau 4.3 montre la liste des jeux de phonèmes utilisés lors de la phase d'apprentissage et les symboles correspondants. Le tableau montre également des exemples illustratifs de l'utilisation de voyelles.

Symbole	Alphabet	Translittération
AA	أ	Alef
B	ب	Ba'
T	ت	Ta'
TH	ث	Tha'
HH	ح	Ha'
JH	ج	jim
KH	خ	Emphatique Kha'
D	د	Dal
DH	ذ	thal
R	ر	Ra'
Z	ز	zay
S	س	sin
SH	ش	shin
SS	ص	sad
DD	ض	dad
TT	ط	ta
DH2	ظ	dha
AI	ع	ayn
GH	غ	ghain
F	ف	fa
Q	ق	gaf
K	ك	kaf
L	ل	lam
M	م	mim
N	ن	nun
H	ه	Ha’
W	و	Waw
Y	ي	Ya'
A	◌َ	Fatha
I	◌ِ	kasra
E	-	Entre kasra et fatha
U	◌ُ	Damma

Tableau 3.4 : Symboles de phonèmes, utilisé pour l'arabe standard.

Nous utilisons les fichiers .wav de formation pour construire le dictionnaire. Le résultat est un fichier ar.dict contenant tous les mots prononcés. Par exemple:

Dans notre application, l’ensemble des symboles précédents (voir tableau 3.4) a été utilisé pour l’apprentissage des états HMM correspondant au modèle acoustique de la démonstration Arabic standard. Le système doit savoir à quel HMM correspond chaque variable (phonème). Ces informations sont stockées dans un fichier appelé dictionnaire. Il permet de faire une représentation

symbolique pour chaque mot. Il permet ainsi d'alimenter l'application Sphinxtrain pour produire le modèle acoustique. L'apprentissage a été fait en utilisant le dictionnaire représenté dans le tableau 3.4.

Mot	Transcription phonétique
آيْنِشْتايْن	A: Y : N I SH : T A Y : N :
أَصْدِقاءْ	AE SS : D I Q A E :
أَخْضَرُ	AE A KH : DD A R U
أساسياتِ	AE A S A S Y A T I
الخَبَرُ	A L KH A B A R U
الأَرْبِعاءْ	A L AE A R : B AI I A E :
الْحاسوبُ	A L : HH A S W B U
الجُمُعَةِ	A L JH U M U AI A T I

Tableau 3.5 : la transcription phonétique des mot d'un dictionnaire

3.4.2 **Modèle de Language :**

Ce sous-système contient les détails décrivant le langage reconnu lui-même. Ce sous-système est le lieu où la plupart des ajustements sont effectués afin de prendre en charge la reconnaissance de la langue arabe. Il se compose de trois modules principaux:

Le modèle acoustique: Ce module fournit les HMM des triphones arabes à utiliser pour reconnaître la parole.

Le modèle de langage: Ce module fournit la grammaire utilisée par le système (généralement la grammaire d'un langage naturel ou d'un sous-ensemble de celui-ci).

Le dictionnaire: Ce module sert d'intermédiaire entre le modèle acoustique et le modèle de langage. Il contient les mots disponibles dans la langue et la prononciation de chacun en termes de phonèmes disponibles dans le modèle acoustique.

Il utilisera les connaissances de ces trois composants pour construire un graphe de recherche approprié à la reconnaissance d'une tâche.

Ainsi, nous pouvons construire le modèle acoustique, le dictionnaire et le modèle de langage comme suit:

3.4.3 **Le modèle acoustique :**

Le but de l'analyse acoustique consiste à représenter le signal de parole sous une forme qui est plus adaptée pour la reconnaissance. Le plus souvent les représentations suivantes sont utilisées: MFCC (Mel Frequency Cepstral Coefficients : domaine cepstral), LPCC (Linear Prédictive Cepstral Coeffcients : domaine temporel) [55], ou PLP (Perceptual Linear Prediction : domaine spectral) [56]. Dans ce travail, nous avons utilisé des paramètres acoustiques de type MFCC [57] et leurs dérivées première et seconde. Ces vecteurs sont normalisés par rapport à la moyenne et la variance sur une phrase. La normalisation par rapport à la variance permet de diminuer la variabilité par rapport au locuteur. L'extraction des paramètres est réalisée avec l'outil Wave2feat de SPHINX. La procédure pour créer le

modèle acoustique consiste à regrouper un ensemble de données d'entrées, appelé la base de données de traitement, et de les traiter avec l'outil SphinxTrain.

3.5 La création Modèle de langue

Il existe deux types de modèles décrivant le langage: les grammaires et les modèles statistiques de langage Les grammaires décrivent des types très simples de langage de commande et de contrôle Ils sont généralement écrits à la main ou générés automatiquement avec du code clair.

Le modèle de langue est une autre exigence importante pour tout système ASR. La création d'un modèle de langage consiste à calculer le nombre de mots d'uni-gramme, lesquels sont ensuite convertis en vocabulaire de tâches avec fréquences de mots, générant les bigrams.

et des trigrammes du texte d'apprentissage basé sur ce vocabulaire, et convertissant finalement les n-grammes en un modèle de langage au format binaire et en un format ARPA standard.

Il existe de nombreuses façons de construire les modèles de langage statistique. Lorsqu'un modèle est petit, vous pouvez utiliser un service Web rapide en ligne. Lorsque votre ensemble de données est volumineux, il est judicieux d'utiliser le kit de modélisation du langage CMU (toolkit CMU SLM), que nous avons utilisé ici.

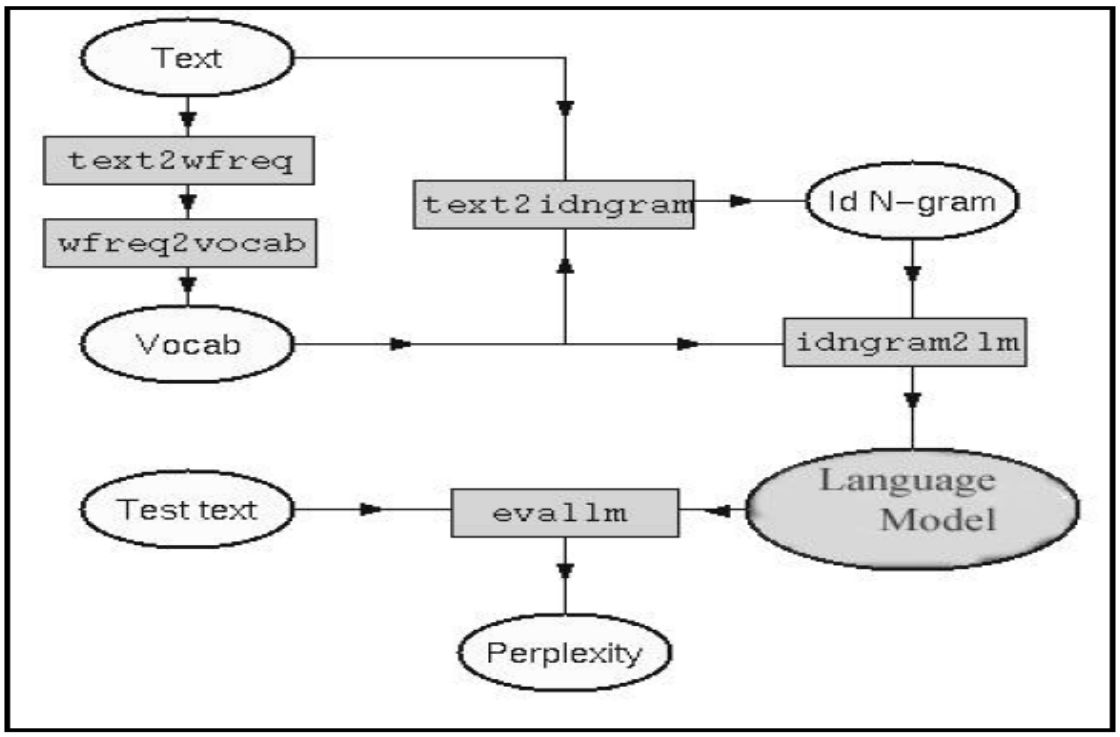


Figure 3.6 : Instructions de mappage de modèle de texte en langage

Les étapes de création et de test du modèle de langage sont illustrées à la figure 4.2.

La création d'un modèle de langage à partir d'un texte d'apprentissage comprend les étapes suivantes:

1) Préparez un texte de référence qui sera utilisé pour générer le modèle de langage. Le kit d'outils de modèles de langage s'attend à ce que ses entrées soient sous la forme de fichiers texte

normalisés, avec des énoncés délimités par des balises <s> et </ s>. Le fichier de sortie appelé a.txt.

2) Calculer le mot compte uni gramme

cat a.txt | text2wfreq > a.wfreq

3) Convertir les mots unigrammes en un vocabulaire composé des 20 000 mots les plus courants

Cat a.wfreq | wfreq2vocab -top 20000 > a.vocab

4) Générer un identifiant binaire de 3 grammes du texte d'apprentissage, basé sur ce vocabulaire

Cat a.text | text2idngram -vocab a.vocab > a.idngram

5) Convertir l'idngram en un modèle de langage au format binaire

Idngram2lm -idngram a.idngram -vocab a.vocab -binary a.binlm

Voir l'annexe A pour plus de détails.

Tester votre modèle de langue avec PocketSphinx

Dans PocketSphinx, nous avons un programme appelé pochesphinx_continuous qui peut être exécuté à partir de la ligne de commande pour reconnaître la parole. Nous essayons d'exécuter la commande suivante:

pocketsphinx_continuous -lm a.lm -dict dict.dic

3.5.1 **Conversion de modèle au format DMP**

Pour charger rapidement des modèles volumineux, PocketSphinx exige que vous soumettiez le modèle DMP au composant TrigramModel.

Le format DMP peut être converti mutuellement. Vous pouvez produire un autre fichier avec

sphinx_lm_convert commande de sphinxbase:

sphinx_lm_convert -i model.lm -o model.dmp

3.6 **Création du modelé acoustique :**

3.6.1 **La préparation des données**

La base de données contient des informations nécessaires pour extraire les statistiques de la parole sous la forme du modèle acoustique. Sphinxtrain Besoins de lui fournir les unités sonores dont vous voulez lui apprendre leurs paramètres, et au moins l'ordre dans lequel ils apparaissent dans chaque signal de parole dans notre base de données d’apprentissage.

Cette information est fournie à Sphinxtrain dans un fichier appelé fichier de transcription, dans lequel la séquence de mots et des sons non-vocaux sont écrits exactement comme ils se sont produits dans un signal de parole, suivi d'une étiquette qui peut être utilisé pour associer cette séquence avec le signal de parole correspondant.

Sphinxtrain cherche alors dans un dictionnaire qui fait correspondre chaque mot à une séquence d'unités de son, pour obtenir la séquence d'unités sonores associés à chaque signal.

Ainsi, en plus des signaux de parole, il faut que vous donnez également un ensemble de transcriptions pour la base de données (dans un seul fichier) et deux dictionnaires, l'un dans lequel les mots dans la langue légitimes sont cartographiés par des séquences d'unités sonores, et un autre dans lequel des sons non-vocaux sont cartographiés pour correspondre les non-paroles comme unités sonores. Nous ferons référence au premier comme le dictionnaire de la langue et le second comme le dictionnaire de remplissage « filler Dictionary ». La structure du fichier de la base de données est la suivante :

Etc

- Nom de projet.dic - *Phonetic dictionary*
- Nom de projet.phone - *Phoneset file*
- Nom de projet.lm.DMP - *Language model*
- Nom de projet.filler - *List of fillers*
- Nom de projet_train.fileids - *List of files for training*
- Nom de projet_train.transcription - *Transcription for training*
- Nom de projet_test.fileids - *List of files for testing*
- Nom de projet_test.transcription - *Transcription for testing*

Wav

- Speaker_1
- ile_1.wav - *Recording of speech utterance*
- Speaker_2
- file_2.wav

3.6.2 **Compilation des packages nécessaires**

Les packages suivants sont requis pour l'apprentissage :

- Sphinxbase
- SphinxTrain

Les languages suivants sont également requis:

- Perl
- Python

Fondamentalement, il faut tout mettre dans le dossier racine unique, après l'extraction des packages, on exécute configure et make et make install dans chaque dossier du package. On met le dossier de base de données dans ce dossier root. On aura donc un répertoire nommée msa dans notre cas avec le contenu suivant :

msa\

SphinxTrain SphinxTrain.tar.gz

Sphinxbase sphinxbase.tar.gz

3.6.3 Etapes d'apprentissage pour notre modèle acoustique

3.6.4 Installation :

Bien sûr, il est nécessaire de télécharger SphinxTrain. On peut le récupérer à l'adresse suivante *http://cmusphinx.sourceforge.net/html/download.php* L'installation se fait de manière classique avec les commandes suivantes (exécuter la dernière commande en tant que root)

tar xvzf SphinxTrain.tar.gz

cd SphinxTrain

./configure make

make install

3.6.5 Configuration de SphinxTrain :

La phase d'apprentissage d'un nouveau modèle est composée de différentes étapes. Il faut avant tout donner un nom au modèle en créant le répertoire correspondant qui en portera le nom. Dans notre cas, nous l'appellerons msa.

mkdir msa

cd msa

Il faut à présent créer la structure du répertoire que l'on vient de créer. Cela se fait automatiquement grâce à un script que nous avons programmé pour générer fileids et transcription.

Répertoire (msa/etc) =. / createfileids.py

Répertoire (msa/etc) =. / createtranscription.py

Il est impératif à ce niveau, de créer deux fichiers

touch etc/msa.fileids

touch etc/msa.transcription

Le premier fichier (msa.fileids) contient la liste des fichiers audio (sans leurs extensions) à utiliser pendant la phase d'apprentissage du modèle.

Fichiers qui devront par la suite être copiés dans le répertoire /wav créé automatiquement précédemment.

Le second fichier (msa.transcription) contient les transcriptions des phrases prononcées dans leurs fichiers audio respectifs (fichiers audio sans leurs extensions). Il faut ensuite créer le fichier de remplissage « filler » qui contient tous les sons contenus dans les fichiers audio mais qui ne

représentent pas de la parole à proprement parler. Par exemple on pourra y recenser les passages audio où y figurent les silences

touch etc/msa.filler

Notons que même si l'on ne souhaite pas reconnaître ce genre de sons, il faut malgré tout le créer.

De plus, ce fichier possède au minimum le contenu suivant :

etc/

msa.filler :

<s> SIL </s>

SIL <sil> SIL

Ne pas oublier d'inclure à la fin du fichier un retour à la ligne.

En effet, il est impératif et fait partie des erreurs presque invisibles.

Tout comme, on a créé le fichier filler, créons le dictionnaire et la liste des phonèmes. Rappelons qu'un phonème est la plus petite unité discrète permettant de distinguer des mots les uns des autres. C'est une entité abstraite qui peut correspondre à plusieurs sons.

touch msa.dic

touch msa.phone

Quelques précisions sur ces deux fichiers

- msa.dic va contenir la liste de tous les mots contenus dans le fichier de transcription (msa.transcription) avec pour chacun leur décomposition en phonèmes. Un mot peut être prononcé de différentes manières, c'est pourquoi Certains mots sont suivis d'une phrase entre parenthèse indiquant une variante de la prononciation.
- msa.phone va quant à lui contenir la liste de tous les phonèmes contenus dans le fichier msa.dic, en prenant soin de ne mettre qu'un phonème par ligne.

Tout comme dans le filler, penser à mettre une ligne vide à la fin de chaque fichier.

Enfin, il ne nous reste plus qu'à lancer l'apprentissage

3.6.6 **Mise en place des scripts d'apprentissage**

Dans cette étape, on a testé un script, celui de Sphinxtrain. Pour commencer l'apprentissage, on se place dans le dossier de base de données et on exécute les commandes suivantes :

Pour sphinxtrain

Sphinxtrain –t setup msa

Cela va copier tous les fichiers nécessaires dans notre dossier de base de données et préparer la base de données pour l'apprentissage, la structure sera :

- bin (seulement pour les premiers versions de sphinxtrain)
- bwaccumdir
- etc
- feat
- logdir
- model_parameters
- model_architecture
- python (seulement pour les premiers versions de sphinxtrain)
- scripts_pl (seulement pour les premiers versions de sphinxtrain)
- wav

Après cela, nous avons besoin d'éditer les fichiers de configuration dans le dossier etc... Il y a de nombreuses variables, mais pour commencer, nous avons besoin de changer que quelques-uns.

Tout d'abord, il faut trouver le fichier / etc sphinx_train.cfg

3.6.7 **Configuration du format audio de la base de données**

Dans le fichier de configuration, on cherche les lignes suivantes :

$CFG_WAVFILES_DIR = "$CFG_BASE_DIR/wav";

$CFG_WAVFILE_EXTENSION = 'sph';

$CFG_WAVFILE_TYPE = 'nist'; # one of nist, mswav, raw

Puis on remplace « sph » et « nist » par l'extension et le type des fichiers audio enregistrés, dans notre cas, on change « sph » par « wav » et « nist » par « mswav ».

```
# These are filled in at configuration time
$CFG_DB_NAME = "msa";
# Experiment name, will be used to name model files and log files
$CFG_EXPTNAME = "$CFG_DB_NAME";
```

3.6.8 **Configuration du chemin vers les fichiers**

Pour cela, on vérifie les lignes suivantes dans le fichier

etc /sphinx_train.cfg :

```
# Variables used in main training of models
$CFG_DICTIONARY     = "$CFG_LIST_DIR/$CFG_DB_NAME.dic";
$CFG_RAWPHONEFILE   = "$CFG_LIST_DIR/$CFG_DB_NAME.phone";
$CFG_FILLERDICT     = "$CFG_LIST_DIR/$CFG_DB_NAME.filler";
$CFG_LISTOFFILES    = "$CFG_LIST_DIR/${CFG_DB_NAME}_train.fileids";
$CFG_TRANSCRIPTFILE = "$CFG_LIST_DIR/${CFG_DB_NAME}_train.transcription";
$CFG_FEATPARAMS     = "$CFG_LIST_DIR/feat.params";
```

Ces valeurs seraient déjà comme ça si nous avons configuré la structure de fichiers comme décrit précédemment, mais il faut assurer que les fichiers sont vraiment appelés de cette façon.

La variable $ CFG_LIST_DIR est le répertoire / etc de notre projet, et la variable

$ CFG_DB_NAME est le nom de notre projet lui-même (msa).

3.6.9 **Configuration des paramètres caractéristiques du son**

La valeur par défaut pour les fichiers audio utilisés dans Sphinx est un taux de 64000 échantillons par seconde (64 KHz). Si c'est le cas, le fichier

etc/feat.params sera généré automatiquement avec les valeurs recommandées. Si vous utilisez des fichiers audio avec une fréquence d'échantillonnage de 16 kHz

(la téléphonie), vous devez changer certaines valeurs dans le fichier

etc / sphinx_train.cfg.

Un taux d'échantillonnage plus faible signifie aussi un changement dans les gammes de fréquences sonores utilisées et le nombre de filtres utilisés pour reconnaître la parole.

Les valeurs recommandées sont les suivantes :

```
# Feature extraction parameters
$CFG_WAVFILE_SRATE = 16000.0;
$CFG_NUM_FILT = 25; # For wideband speech it's 25, for telephone 8khz reasonable value is 15
$CFG_LO_FILT = 130; # For telephone 8kHz speech value is 200
$CFG_HI_FILT = 6800; # For telephone 8kHz speech value is 3500
$CFG_TRANSFORM = "dct"; # Previously legacy transform is used, but dct is more accurate
$CFG_LIFTER = "22"; # Cepstrum lifter is smoothing to improve recognition
$CFG_VECTOR_LENGTH = 13; # 13 is usually enough

$CFG_MIN_ITERATIONS = 1;  # BW Iterate at least this many times
$CFG_MAX_ITERATIONS = 10; # BW Don't iterate more than this, somethings likely wrong.
```

3.6.10 Configuration des paramètres de décodage

Ouvrir le dossier :

etc/sphinx_train.cfg.

Il faut s'assurer que les éléments suivants sont correctement configurés:

```
$DEC_CFG_DICTIONARY     = "$CFG_BASE_DIR/etc/$CFG_DB_NAME.dic";
$DEC_CFG_FILLERDICT     = "$CFG_BASE_DIR/etc/$CFG_DB_NAME.filler";
$DEC_CFG_LISTOFFILES    = "$CFG_BASE_DIR/etc/${CFG_DB_NAME}_test.fileids";
$DEC_CFG_TRANSCRIPTFILE = "$CFG_BASE_DIR/etc/${CFG_DB_NAME}_test.transcription";
$DEC_CFG_RESULT_DIR     = "$CFG_BASE_DIR/result";
$DEC_CFG_PRESULT_DIR     = "$CFG_BASE_DIR/presult";
$DEC_CFG_LANGUAGEMODEL  = "$CFG_BASE_DIR/etc/${CFG_DB_NAME}.lm.DMP";
```

Si tout est correct, nous pouvons procéder à l'apprentissage.

3.7 L'apprentissage :

Tout d'abord, allez dans le répertoire de base de données :

Cd « nom de projet »

Pour s'entraîner, il suffit d'exécuter les commandes suivantes :

Pour sphinxtrain 1.0.7 et antérieures

./scripts_pl/make_feats.pl -ctl etc/ nom de projet_train.fileids ./scripts_pl/make_feats.pl -ctl etc/ nom de projet_test.fileids ./scripts_pl/RunAll.pl

Pour sphinxtrain snapshot

Sphinxtrain run

```
hakim@hakim-Inspiron-3543:~/Bureau/msa$ sphinxtrain run
Sphinxtrain path: /usr/local/lib/sphinxtrain
Sphinxtrain binaries path: /usr/local/libexec/sphinxtrain
Running the training
MODULE: 000 Computing feature from audio files
Extracting features from  segments starting at  (part 1 of 1)
Extracting features from  segments starting at  (part 1 of 1)
Feature extraction is done
```

Et il passera par toutes les étapes nécessaires. Cela prendra quelques minutes pour s'entraîner. Sur les grandes bases de données, l'apprentissage pourrait prendre un mois. La sortie typique lors du décodage ressemblera à ceci:

```
MODULE: 00 verify training files
    Phase 1: Checking to see if the dict and filler dict agrees with the phonelist file.
        Found 2866 words using 42 phones
    Phase 2: Checking to make sure there are not duplicate entries in the dictionary
    Phase 3: Check general format for the fileids file; utterance length (must be positive); files exist
    Phase 4: Checking number of lines in the transcript file should match lines in fileids file
    Phase 5: Determine amount of training data, see if n_tied_states seems reasonable.
        Estimated Total Hours Training: 3.22760833333333
        This is a small amount of data, no comment at this time
    Phase 6: Checking that all the words in the transcript are in the dictionary
        Words in dictionary: 2863
        Words in filler dictionary: 3
    Phase 7: Checking that all the phones in the transcript are in the phonelist, and all phones in the phonelist appear at least once
```

```
MODULE: 0000 train grapheme-to-phoneme model
Skipped (set $CFG_G2P_MODEL = 'yes' to enable)
MODULE: 01 Train LDA transformation
Skipped (set $CFG_LDA_MLLT = 'yes' to enable)
MODULE: 02 Train MLLT transformation
Skipped (set $CFG_LDA_MLLT = 'yes' to enable)
MODULE: 05 Vector Quantization
Skipped for continuous models
MODULE: 10 Training Context Independent models for forced alignment and VTLN
Skipped:  $ST::CFG_FORCEDALIGN set to 'no' in sphinx_train.cfg
Skipped:  $ST::CFG_VTLN set to 'no' in sphinx_train.cfg
MODULE: 11 Force-aligning transcripts
Skipped:  $ST::CFG_FORCEDALIGN set to 'no' in sphinx_train.cfg
MODULE: 12 Force-aligning data for VTLN
Skipped:  $ST::CFG_VTLN set to 'no' in sphinx_train.cfg
```

```
MODULE: 20 Training Context Independent models
    Phase 1: Cleaning up directories:
        accumulator...logs...qmanager...models...
    Phase 2: Flat initialize
    Phase 3: Forward-Backward
        Baum welch starting for 1 Gaussian(s), iteration: 1 (1 of 1)
        0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%
        Normalization for iteration: 1
        Current Overall Likelihood Per Frame = -159.962356027296
        Baum welch starting for 1 Gaussian(s), iteration: 2 (1 of 1)
        0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%
        Normalization for iteration: 2
        Current Overall Likelihood Per Frame = -158.357968877884
        Convergence Ratio = 1.60438714941171
        Baum welch starting for 1 Gaussian(s), iteration: 3 (1 of 1)
        0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%
```

```
MODULE: 60 Lattice Generation
Skipped:  $ST::CFG_MMIE set to 'no' in sphinx_train.cfg
MODULE: 61 Lattice Pruning
Skipped:  $ST::CFG_MMIE set to 'no' in sphinx_train.cfg
MODULE: 62 Lattice Format Conversion
Skipped:  $ST::CFG_MMIE set to 'no' in sphinx_train.cfg
MODULE: 65 MMIE Training
Skipped:  $ST::CFG_MMIE set to 'no' in sphinx_train.cfg
MODULE: 90 deleted interpolation
Skipped for continuous models
```

Ces scripts traitent toutes les démarches nécessaires pour l'apprentissage du modèle. À ce niveau-là, l'étape d'apprentissage sera terminée.

3.7.1 **Résultat de l'apprentissage du modèle acoustique**

Nous avons obtenu ce dossier. Qui contient les fichiers suivants:

- ✓ mdef
- ✓ feat.params
- ✓ mixture_weights
- ✓ means
- ✓ noisedict
- ✓ transition_matrices
- ✓ variances

L'utilisation de nouveaux modèles est facile, il nous suffit de configurer le système de reconnaissance correctement. Il comprend généralement trois étapes:

- La définition d'un dictionnaire et d'un modèle de langage ;
- Définition d'un modèle et d'un chargeur de modèle ;
- Configurer une interface (en option).

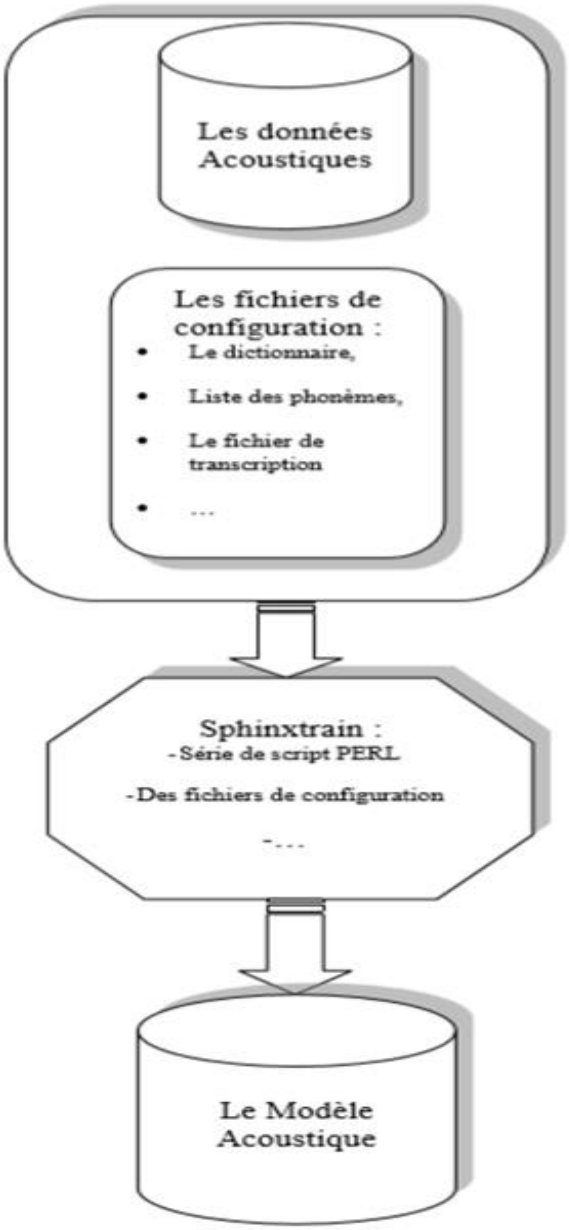


Figure 3.7 : Schéma représentant la création d'un modèle acoustique avec SphinxTrain

3.8 Définir un dictionnaire et un modèle de langage

On peut utiliser le même dictionnaire phonétique et le même modèle que celui utilisé pour l'apprentissage et le test initial. Ils sont situés dans le dossier <your_training_folder>/etc/ Et ont des noms comme <your_model_name>.dic Et <your_model_name>.lm.DMP . Sinon on peut le créer avec cmuclmtk et convertir plus tard au DPM forme avec sphinx_lm_convert du paquet sphinxbase.

3.9 Définir un modèle acoustique

Suivant le modèle acoustique, pendant l'apprentissage plusieurs modèles sont créés, nous avons besoin de l'un d'entre eux.

Pour un grand vocabulaire (modèle dépendant du contexte) le modèle se trouve dans :

<your_training_folder>/model_parameters/

<your_db_name>.cd_cont_<number of senones> .

Pour un petit vocabulaire (modèle indépendant de contexte), il suffit de prendre celui situé dans:

<your_training_folder>/model_parameters/

<your_db_name>.ci_cont .

Ce dossier doit comprendre plusieurs fichiers, comme les moyennes, variances, feat.params, MDEF.

3.10 Nos tests :

On a testé le system sur deux méthode

La première méthode à partir d'un microphone

La deuxième méthode a partie des fichiers enregistrés

On a fait le test sur 10 femmes et 10 hommes :

Locuteur	**#nbr de phrase**	**Duration (sec)**
1	10	30
2	10	29
3	30	86
4	40	120
5	40	110
6	75	242
7	100	327
8	50	171
9	40	123
10	40	114

Tableau 3.6 la duration et nombre des phrase parlé par des femmes

Locuteur	**#nbr de phrase**	**Duration (sec)**
1	10	30
2	10	29
3	30	86
4	40	120
5	40	110
6	75	242
7	100	327
8	50	171
9	40	123
10	40	114

Tableau 3.7 la duration et nombre des phrase parlé par des hommes

3.11 Résultat :

Microphone :

Ce tableau représente les Résultats de la reconnaissance par phrase parlé avec le microphone de notre application

La phrase correcte	hypothèse	accuracy
جوزيفْ إسم أصْلُهُ بِالعَرَبِيَّةِ يوسُفْ	جُزْءٌ ثالِثٌ أصْلُهُ بِالعَرَبِيَّةِ يوسُفْ	1-2/5=0,6=60%
دَخَلَ الْحاسوبُ كُلَّ مَجالاتِ الْحَياة	دَخَلَ الْحاسوبُ كُلَّ مَجالاتِ حُرّاً	1-1/5=0,8=80%
إنْتَشَرَ اسْتِخْدامُ الْهاتِفِ النَّقّالْ	إنْتَشَرَ اسْتِخْدامُ الْهاتِفِ مِنَ قَد	1-2/4=0,5=50%
مِنْ أساسياتِ العَمَلِ إتْقانُ فَنِّ التَّعامل مَعَ الزَّبائِنْ	مِنْ أساسياتِ العَمَلِ إتْقانُ فَمِنّا عُلوم مَعَ الزَّبائِنْ	1-2/8=0,75=75%
كُلّنا سَمِعَ عَنْ صُلْحِ الْحُدَيْبِيَة	كُلّنا سَمِعَ مُسْلِماً عديدة	1-3/5=0,4=40%

Tableau 3.8 Exemple

Ficher audio :

Voici les résultats pour les femmes :

Locuteur	Accuracy (%)	Wer (%)
1	78	22
2	57	43
3	73	27
4	80	20
5	64	36
6	61	39
7	68	32
8	64	36
9	72	28
10	72	28

Tableau 3.9 les taux de reconnaissance et erreur pour les femmes

Voici les résultats pour les hommes :

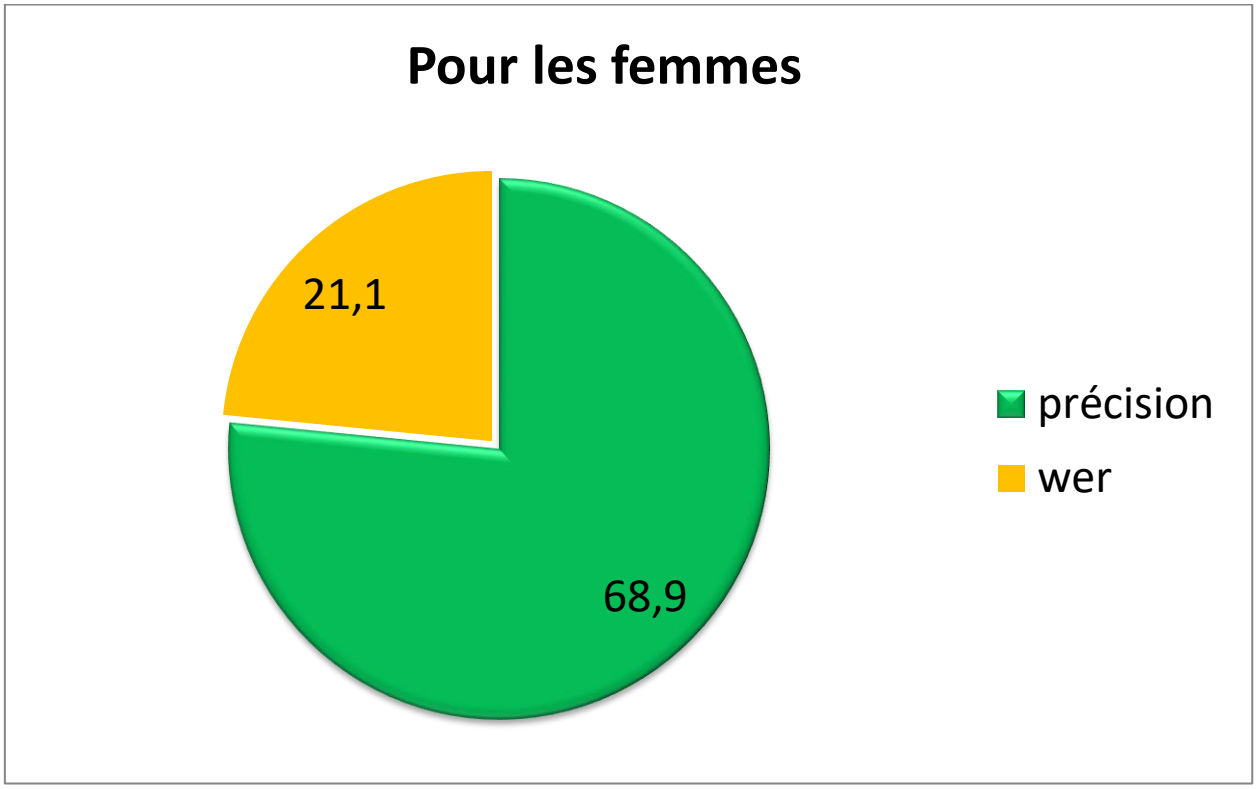
Locuteur	Accuracy (%)	Wer (%)
1	66	34
2	63	37
3	61	39
4	70	30
5	58	42
6	48	52
7	62	38
8	66	34
9	70	30
10	70	30

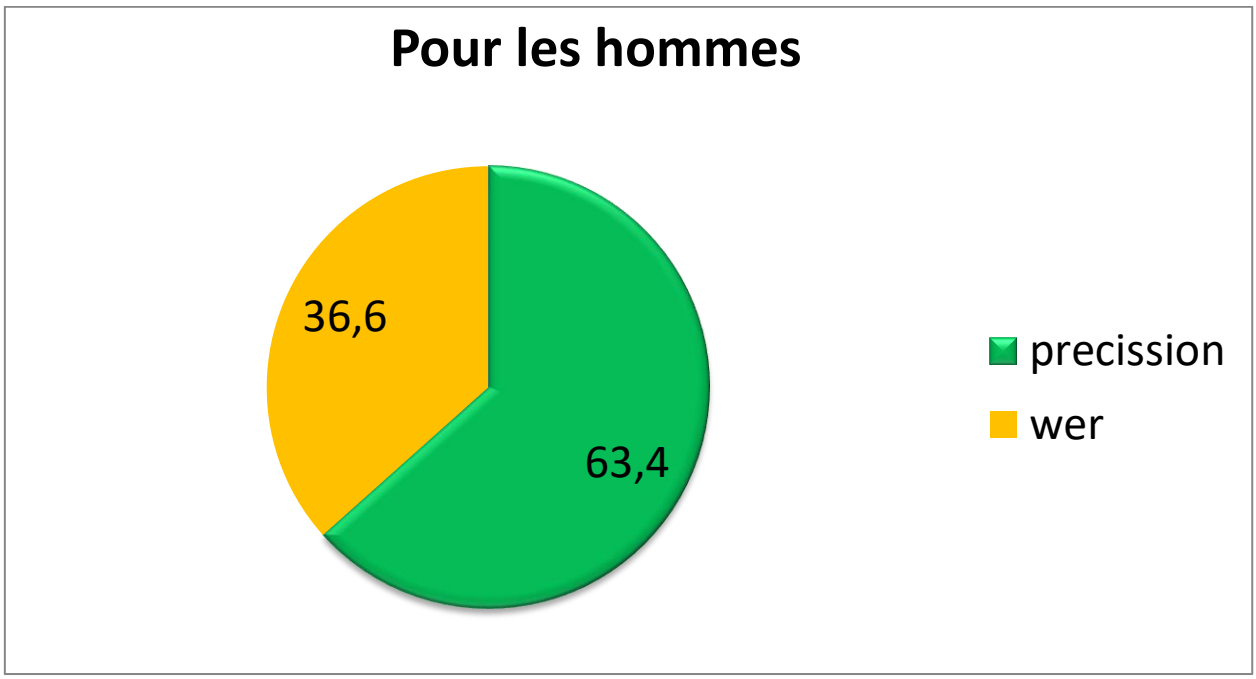
Tableau 3.10 les taux de reconnaissance et erreur pour les hommes

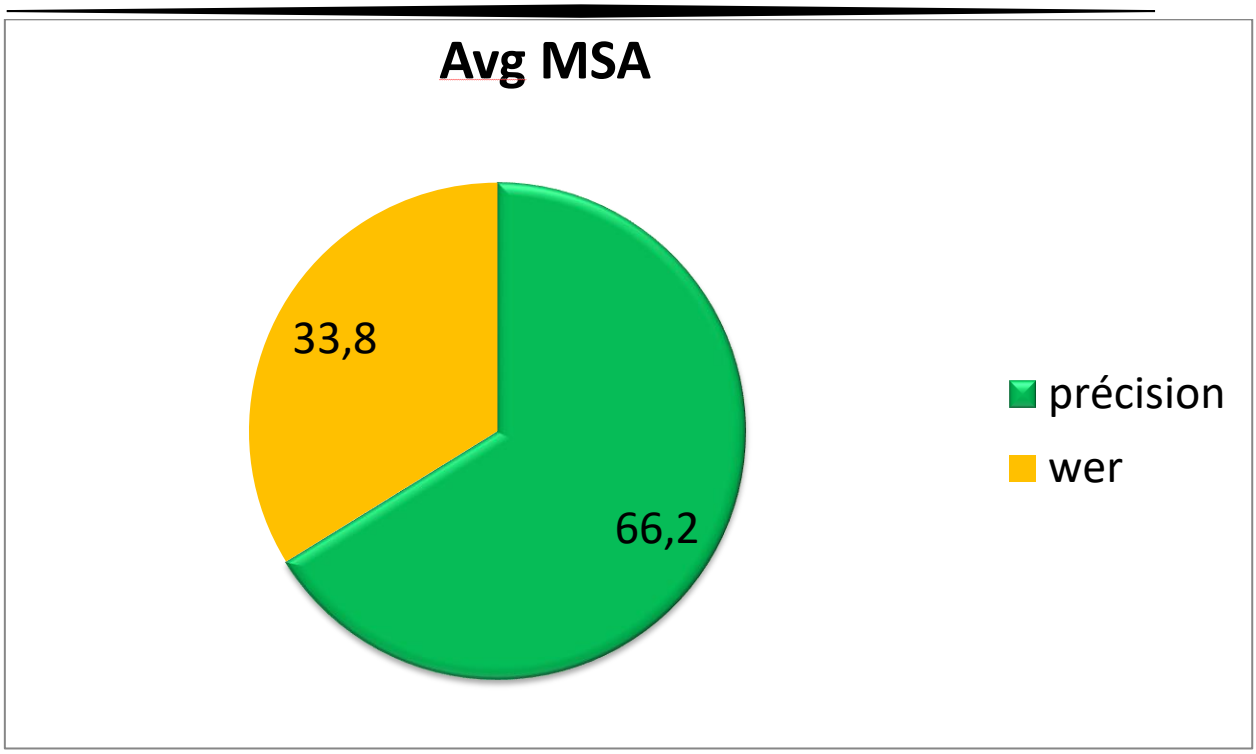
Voice Les moyennes des tests (résultat globale) :

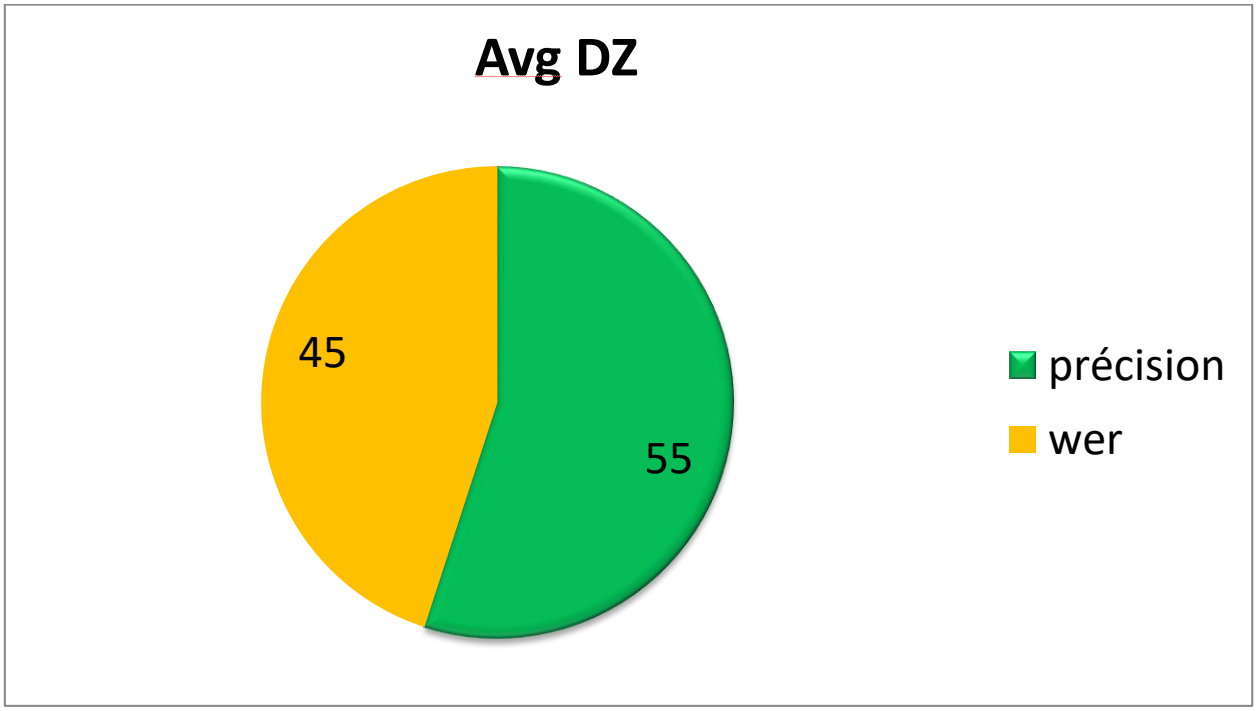
Locuteur	AVG Accuracy (%)	AVG Wer (%)
Femmes	68.9	31.1
Hommes	63.4	36.6

Tableau 3.11 résultat de reconnaissance pour les femmes et les hommes






Avg MSA
33,8
66,2
précision
wer


Avg DZ
45
55
précision
wer

3.1 Conclusion :

La création du modèle acoustique est une étape primordiale pour la reconnaissance de la parole elle passe par plusieurs étapes, ou l'étapes d'apprentissage a une grande importance. Grace à cette dernière on obtient notre modèle acoustique. Cependant on ne peut pas utiliser ce modèle directement pour pocketSphinx puisqu'il utilise des script python pour la lecture de ces modèles, donc il faut procéder à quelques opérations supplémentaires, après ça en peut tester notre système de reconnaissance en évaluant leur taux d'erreur automatiquement avec un script ou manuellement en testant la reconnaissance pour diffèrent locuteurs.

4 CONCLUSION GENERALE :

La recherche en reconnaissance vocale a été influencée par les progrès Technologiques. Ça a débuté avec les systèmes analogiques, et le développement rapide de l'informatique et de la microélectronique ont permis l'ouverture de nouveaux horizons pour ce domaine tant au niveau des techniques qu'au niveau des secteurs d'application.

La reconnaissance vocale recouvre tous les aspects liés à l'interprétation, par la machine, du langage humain. Les applications de cette technologie sont nombreuses : Navigation sur un serveur vocal au téléphone, Apprentissage d'une langue étrangère, commandes vocales dans les voitures, les téléphones ou bien encore dans les salles d'opérations chirurgicales, dictée vocale, identification vocale dans les zones sécurisées ou bien dans le cadre d'une enquête judiciaire, etc.

En conclusion, un système de reconnaissance de la langue arabe parlée a été conçu pour étudier le processus de reconnaissance automatique de la parole dans un environnement arabe. Les processus de formation et de reconnaissance utilisent des caractères arabes. Notre expérience démontre l'adaptabilité possible du CMU Sphinx à la langue arabe standard et la langue arabe dialectale. Le système se composait de trois composants de base: un dictionnaire phonétique arabe contenant tout le son phonétique des mots utilisés lors de la formation et du modèle linguistique statistique arabe qui nous donnait la probabilité de la séquence des mots. La dernière composante est le modèle acoustique qui a généré l'unité représentée par HMM pour chaque phonème. Nous avons utilisé un corpus de parole phonétiquement riche et équilibré pour former le programme de reconnaissance.

Le système développé offre une bonne précision avec la parole continue en arabe standard, naturel et indépendant du locuteur, avec un taux de précision de 84% avec les fichiers enregistrés. Les résultats de reconnaissance produits par notre système se sont révélés satisfaisants, en ce qui concerne la précision avec la parole continue en arabe dialectale elle est de 60% avec les fichiers enregistrés.

5 REFERANCES :

[1] Haton M., Cerisara C., Fohr D., Laprie Y., and Smaili K., *Reconnaissance* Automatique de la Parole du Signal a Son Interpretation, Monographies and Books, Oxford, 2006.

[2] Young S., .The HTK Hidden Markov Model Toolkit: Design and Philosophy,. Technical Report TR 152, Department of Engineering, Cambridge University, Cambridge, 1994

[3] Deshmukh N., Ganapathiraju A., Hamaker J., Picone J., and Ordowski M., .A Public Domain Speech to Text System,. *in Proceedings of 6th European* Conferences on Speech Communication and Technology, Hungary, pp. 2127-2130, 1999.

[4] Muhammad A., .Alaswaat Alaghawaiyah,. *in Proceedings of International* Conference on Signal Processing, Jordan, pp. 646-651, 1990.

[5] Li X., Zhao Y., Pi X., Liang H., and Nefian V., .Audio Visual Continuous Speech Recognition Using a Coupled Hidden Markov Model,. *in Proceedings of* 7th International Conferences on Spoken Language Processing, Denver, pp. 213-216, 2002.

[6] Huang X., Acero A., and Hon H., "Spoken Language Processing: A Guide to Theory", *Algorithm and System Design*, Prentice Hall, 2001.

[7] Gordon R., *Ethnologue: Languages of the World*, Texas: Dallas, SIL International, 2005.

[8] Alghamdi M., Elshafei M., and Al-Muhtaseb H., .Arabic Broadcast News Transcription System,. International Computer Journal of Speech Technology, vol. 10, no. 4, pp. 183-195, 2009

[9] Hyassat H. and Abu Zitar R., .Arabic Speech Recognition Using SPHINX Engine,. International Computer Journal of Speech Technology, vol. 9, no. 3-4, pp. 133-150, 2008.

[10] Kirchhoff K., Bilmes J., Das S., Duta N., Egan M., Ji G., He F., Henderson J., Liu D., Noamany M., Schone P., Schwartz R., and Vergyri D., .Novel Approaches to Arabic Speech Recognition: Report from the 2002 Johns- Hopkins Summer Workshop,. in Proceedings of IEEE International Conference on Acoustics, Speech, and Single processing, Hong Kong, vol. 1, pp. 344-347, 2003

[11] H., Harti M., and Chenfour N., .Arabic Speech Recognition System Based on CMUSphinx,. in Proceedings of IEEE International Symposium on Computational Intelligence and Intelligent Informatics, Morocco, pp. 31-35, 2007.

[12] Alsulaiti L. and Atwell E., .The Design of a Corpus of Contemporary Arabic,. International Computer Journal of Corpus Linguistics, John Benjamins Publishing Company, pp. 1-36, 2006.

[13] Alansary S., Nagi M., and Adly N., .Building an International Corpus of Arabic Progress of Compilation Stage,. in Proceedings of 8th International Conference *on Language Engineering*, Egypt, pp. 337-344, 2007.

[14] Parkinson D. and Farwaneh S., *Perspectives on Arabic Linguistics XV*, John Benjamins Publishing Company, Philadelphia, 2003.

[15] Black A. and Tokuda K., .The Blizzard Challenge Evaluating Corpus-Based Speech Synthesis on Common Datasets,. *in Proceeding of Interspeech*, Portugal, pp. 77-80, 2005.

[16] D.Arcy S. and Russell M., .Experiments with the ABI (Accents of the British Isles) Speech Corpus,. *in Proceeding of Interspeech 08*, Australia, pp. 293-296, 2008.

[17] Garofolo J., Lamel L., Fisher W., Fiscus J., Pallett D., Dahlgren N., and Zue V., .TIMIT Acoustic-Phonetic Continuous Speech Corpus,. *Technical Document*, Philadelphia, 1993.

[18] Chou F. and Tseng C., .The Design of Prosodically Oriented Mandarin Speech Database,. in Proceedings of International Congress of Phonetics Sciences, San Francisco, pp. 2375-2377, 1999.

[19] Sagisaka Y., Takeda K., Abel M., Katagiri S., Umeda T., and Kuwabara H., .A Large-Scale Japanese Speech Database,. *in Proceedings of International* Conference on Spoken Language Processing, 1990.

[20] Alotaibi Y., .Comparative Study of ANN and HMM to Arabic Digits Recognition Systems,. Journal of King Abdulaziz University: Engineering *Sciences*, vol. 19, no. 1, pp. 43-59, 2008.

[21] Alotaibi Y., Alghamdi M., and Alotaiby F., .Using a Telephony Saudi Accented Arabic Corpus in Automatic Recognition of Spoken Arabic Digits,. *in* Proceedings of 4th International Symposium on Image/Video Communications over Fixed and Mobile Networks, Spain, pp. 43-60, 2008.

[22] Ahmed Omar, .Study of Linguistic phonetics,. *Aalam Alkutob*, Eygpt 1991. (in Arabic)

[23] El-Imam, .An Unrestricted Vocabulary Arabic Speech Synthesis System., *IEEE* Transactions on Acoustic, Speech, and Signal Processing, Vol. 37,No. 12, Dec. 1989, pp.1829-45.

[24] Hassan S., Hussein H., Mostafa H. and Noureddine C., " Investigation Arabic Speech Recognition Using CMU Sphinx System", *The International Arab* Journal of Information Technology, Vol. 6, No. 2, April 2009

[25] Rabiner, and Juang, "Fundamentals of Speech Recognition." Prentice Hall, 1993.

[26] Rabiner, .A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition,. *Proceedings of the IEEE*, 77(2):257.286, February 1989.

[27] HTK speech recognition tool kit. http://htk.eng.cam.ac.uk/ (accessed July, 2012).

[28] Sphinx-4 Java-based Speech Recognition Engine, http://cmusphinx.sourceforge.net/sphinx4/ (accessed July, 2012).

[29] Huang, Alleva, Hon, Hwang, and Rosenfeld, .The SPHINX-II speech recognition system: an overview,. *Computer Speech and Language, vol*. 7, no. 2, pp. 137.148, 1993

[30] Lamere, Kwok, Walker, Gouvea, Singh, Raj, and Wolf, .Design of the CMU

Sphinx-4 decoder,. in Proceedings of the 8th European Conference on Speech *Communication and Technology*, Geneve, Switzerland, Sept. 2003, pp. 1181.
1184

[31] Algamdi M., *Arabic Phonetics*, Attaoobah, Riyadh, 2000.

[32] Algamdi M., "KACST Arabic Phonetics Database", *The Fifteenth International Congress of Phonetics Science*, Barcelona, 3109-3112, 2003.

[33] Elshafei M., Al-Muhtaseb H. and Alghamdi M., "Speech Units for Arabic Textto-speech", The Fourth Workshop on Computer and Inforamtion Sciences, 199-212, 2002.

[34] Elshafei M., Al-Muhtaseb H. and Alghamdi M., "Techniques for High Quality Text-to-speech", *Information Science*, 140 (3-4) 255-267, 2002
55

[35] Elshafei M., Al-Muhtaseb H. and Alghamdi M., .Statistical Methods for Automatic Diacritization of Arabic text., *Proceedings 18th National computer Conference NCC.18*, Riyadh, March 26-29, 2006.

[36] Al-Otaibi F., Speaker-Dependant Continuous Arabic Speech Recognition, M.Sc. Thesis, King Saud University, 2001.

[37] Billa, J.; Noamany, M.; Srivastava, A.; Liu, D.; Stone, R.; Xu, J.; Makhoul, J.; Kubala, F.,. Audio indexing of Arabic broadcast news., *Proceedings. (ICASSP '02). IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2002. Volume 1, 2002 Page(s):I- 5 - I-8 vol.1

[38] Vergyri D. and Kirchhoff K., Automatic Diacritization of Arabic for Acoustic Modelling in Speech Recognition, Editors, Coling, Geneva, 2004.

[39] Azmi M. and Tolba H., .Syllable-Based Automatic Arabic Speech Recognition in Different Conditions of Noise,. IEEE Proceedings of the 9th International Conference on Signal Processing, China, pp. 601-604, 2008.

[40] Nofal M., Abdel-Raheem E., El Henawy H., and Abdel Kader N., .Acoustic Training System for Speaker Independent Continuous Arabic Speech

Recognition System,. in Proceedings of the 4th IEEE International Symposium

on Signal Processing and Information Technology, Italy, pp. 200-203, 2004.

[41] Alghamdi M., Alhamid A., and Aldasuqi M., .Database of Arabic Sounds:

Sentences,. Technical Report, King Abdulaziz City of Science and Technology,

Saudi Arabia, 2003.

[42] Raja A., Roziati Z., Moustafa E., and Othman Kh., " Arabic Speaker-Independent

Continuous Automatic Speech Recognition Based on a Phonetically Rich and

Balanced Speech Corpus", International Arab Journal Information Technology,

vol. 9(1): 84-93 (2012)

[43] "The Speech Recognition Information Source", http://www.sayican.com,
(accessed July, 2012).

[45] M.A.Anusuya, S.K.Katti, "Speech Recognition by Machine:A Review",
(IJCSIS) International Journal of Computer Science and Information Security,
Vol. 6, No. 3, 2009

[46] Willie W.r, Paul L., Philip K., Bhiksha R.," Sphinx-4: A Flexible Open Source Framework for Speech Recognition", Sun Microsystems, Inc., November 2004

[47] Top Internet Languages - Internet World Stats,

http://www.internetworldstats.com/stats7.htm , 2010. (accessed July, 2012).

[48] Alotaibi1 Y., Hussain A.," Comparative Analysis of Arabic Vowels using

Formants and an Automatic Speech Recognition System", *International Journal*

of Signal Processing, Image Processing and Pattern Recognition Vol. 3, No. 2,

June, 2010

[49] Ghania Droua-Hamdani, " Algerian Arabic Speech Database: Corpus Design and

Automatic Speech Recognition Application", *The Arabian Journal for Science*

and Engineering, Volume 35, Number 2C 157, 2010

[50] El-Imam Y., .An Unrestricted Vocabulary Arabic Speech Synthesis System.,

IEEE Transactions on Acoustic, Speech, and Signal Processing, 37(12)(1989),

pp. 1829.1845.

[51] Youssef and Emam, .An Arabic TTS System on the IBM Trainable

Synthesizer., *Le Traitement Automatique de l.Arabe*, JEP-TALN 2004, Fes, 19.

21 avril 2004.

[52] Tsuhan Ch., "Audiovisual speech processing", *IEEE Signal Processing Magazine*, 18(1):9-21, 2001.

[64] Carnegie Mellon University. CMU pronouncing dictionary. [Online]. Available: http://www.speech.cs.cmu.edu/cgi-bin/cmudict (accessed July, 2012).

[72] Gruhn R., Minker W., & Nakamura S. (2011). *Statistical pronunciation modeling for non-native speech processing.* hardcover,isbn: 978-3-642-19585-3.

[75] X. Huang and L. Deng. (2010). An Overview of Modern Speech Recognition , in Handbook of Natural Language Processin, Second Edition, Chapter 15. *Chapman & Hall/CRC* , pp. 339-366.

[76] R.E. Gruhn et al. (2011). Statistical Pronunciation Modeling for Non-Native Speech Processing, Signals and Communication Technology, DOI: 10.1007/978-3-642-19586-0_2. Springer-Verlag Berlin Heidelberg .

[77] Marc Ferràs Font. (June 15, 2005). MULTI-MICROPHONE SIGNAL PROCESSING FOR AUTOMATIC SPEECH RECOGNITION IN MEETING ROOMS. Berkeley, California .

[78] Hamda M. M. Eljagmani. (May, 2017). Arabic Speech Recognition Systems. Florida Institute of Technology .

[79] A Review on Automatic Speech Recognition Architecture and Approaches. (2016). International Journal of Signal Processing, Image Processing *and Pattern Recognition* , Vol.9, No.4, pp.393-404.

[80] Ondřej Plátek. (2014). Automatic speech recognition using Kaldi. *Ústav* formální a aplikované lingvistiky .

[92] D. Yu and L. Deng. (2015). Automatic Speech Recognition: A Deep Learning Approach. *Springer-Verlag London* , XXVI, 321 p. 62 illus.