MINISTÈRE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITÉ FERHAT ABBAS – SÉTIF 1–
UFAS (ALGERIE)

MEMOIRE

Présenté à la Faculté de Technologie

Département d'ÉLECTRONIQUE

Pour obtention du Diplôme de

MAGISTER

Option : Communication

Par

Mr. **DOUIB OUALID**

THEME

RECONNAISSANCE AUTOMATIQUE DE LA PAROLE ARABE PAR CMU SPHINX 4

Soutenu le 18/12/2013 devant La commission d'examen :

Mr. A. ZEGADI	Prof à l'université de Sétif	Président
Mr. H. CHEMALI	MCA à l'université de Sétif	Examinateur
Mr. A. BARTIL	MCA à l'université de Sétif	Examinateur
Mr. T. MOHAMADI	Prof à l'université de Sétif	Rapporteur

Remerciements

Ce modeste travail est la résultante de la contribution de plusieurs personnes dont je tiens à remercier vivement :

Toute ma reconnaissance et remerciement vont à Mr le professeur T. Mohamadi qui a accepté de m'encadrer, merci infiniment de vous être montré patient, de m'avoir orienté, dirigé, conseillé et encouragé. Veuillez trouver ici l'expression de ma profonde gratitude.

Mr le professeur A. Zegadi, pour m'avoir fait le grand honneur d'accepter de présider le jury de soutenance. Veuillez trouver ici l'expression de ma sincère gratitude.

Mr le professeur A. Bartil, pour avoir accepté de juger ce travail. Sincères remerciements.

Mr le professeur H. Chemali, pour avoir accepté de juger cette modeste contribution et d'avoir consacré du temps à sa lecture. Veuillez trouver ici l'expression de toute mon estime et de ma plus vive reconnaissance.

Le personnel du département d'électronique, de la faculté de technologie à l'université de Ferhat Abbas-Sétif 1, pour les facilités qu'ils nous ont accordé.

Je remercie aussi tous ceux qui ont aidé de prés ou de loin à la réalisation de ce travail. Je n'oublie pas de remercier aussi tous mes amis de ma promotion.

Je tiens enfin, à remercier les membres de ma famille pour leur incessant soutien et plus particulièrement mes parents qui m'ont guidé sur le chemin des études.

Sommaire

Chapitre II : Les différents outils de reconnaissance de la parole

Chapitre III : Application de CMU Sphinx à la reconnaissance de l'arabe

Chapitre IV : Application et résultats

INTRODUCTION

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) sont 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,....

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 RAP. Elle consiste à réaliser un Système de Reconnaissance Automatique de la Parole (SRAP) basé sur le CMU Sphinx4 qui est basé sur les Modèles de Markov Cachés (MMC) pour la reconnaissance des 10 premiers chiffres arabes.

Ce mémoire s'articule autour de quatre Chapitres :

- Le premier est une introduction générale au domaine de la reconnaissance de la parole, ses problématiques et ses difficultés, les principales méthodes utilisées en reconnaissance, le problème acoustique-phonétique ainsi que les modes de reconnaissance.

- Le second chapitre illustre les différentes applications de reconnaissance de la parole ainsi que leur classification selon plusieurs critères sous forme de comparaissant

- Le troisième chapitre présente les MMC et le logiciel CMU SPHINX 4 ainsi que la reconnaissance de la langue arabe avec ce dernier.

- Le dernier chapitre présente l'implantation de système réalisé.

CHAPITRE I

LA RECONNAISSANCE AUTOMATIQUE DE LA PAROLE

I.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) ?

La reconnaissance automatique de la parole est l'un des deux domaines du traitement automatique de la parole, l'autre étant la synthèse vocale. La reconnaissance automatique de la parole permet à la machine de comprendre et de traiter des informations fournies oralement par un utilisateur humain. Elle consiste à employer des techniques d'appariement afin de comparer une onde sonore à un ensemble d'échantillons, composés généralement de mots mais aussi, plus récemment, de phonèmes (unité sonore minimale : voir plus loin).

Dans ce chapitre, on va essayer de donner une idée générale sur la RAP, de discuter les problèmes de cette dernière, ainsi que les méthodes utilisées pour résoudre ces problèmes.

I.2 Système de reconnaissance de la parole

La reconnaissance automatique de la parole pose de nombreux problèmes d'un point de vue théorique. Leur complexité fait que seuls des sous-problèmes ont pu être à ce jour résolus. Ces solutions partielles correspondent à des contraintes plus ou moins fortes, et les systèmes existants supposent une coopération plus ou moins grande des utilisateurs.

Pour classer les systèmes de reconnaissance automatique, on a généralement recours aux critères suivants :

- Le mode d'élocution : des syllabes ou mots isolés aux mots connectés, jusqu'à une parole dite, "continue" c'est-à-dire sans pauses artificielles.
- Taille du vocabulaire et difficulté de la grammaire (la complexité du langage autorisé).
- La dépendance plus ou moins grande vis-à-vis du locuteur.
- L'environnement protégé ou non (la robustesse aux conditions d'enregistrement).

En outre, il n'est pas inintéressant de les différencier selon deux points qui ont aussi leur importance :

- La compréhension est-elle requise ou non ? (Un système de compréhension cherche à accéder à la signification de l'énoncé parlé).
- Le discours est-il naturel, ou la syntaxe des phrases doit-elle être contraignante?

Les systèmes réalisés de la RAP, sont conçus pour des applications spécifiques. Cela conduit à une restriction de l'univers du dialogue homme-machine. L'objectif essentiel des études sur la reconnaissance et la compréhension automatique de la parole est « de permettre, à terme, un dialogue le plus naturel possible entre l'homme et la machine, dans le cadre d'une application spécifique» [1].

Dans le domaine de la reconnaissance automatique de la parole, on distingue trois grands types de systèmes :

- Les systèmes de commandes vocales ;
- Les systèmes de dictée automatique ;
- Les systèmes de compréhension.

I.2.1 Vue générale d'un système de reconnaissance automatique de la parole

Uniquement les systèmes de reconnaissance automatique de la parole (SRAP) probabilistes à grand vocabulaire seront décrits. Bien qu'il existe d'autres méthodes, comme les réseaux de neurones par exemple, l'approche basée sur l'utilisation de méthodes statistiques [2], dominent depuis plus de 20 ans.

Un SRAP a pour objectif de transcrire sous forme textuelle un signal. A partir d'une séquence d'observations acoustiques X, l'objectif est de trouver la séquence de mots prononcés par les locuteurs. Il s'agit donc de déterminer la suite de mots $\widehat{W}$ la plus probable parmi l'ensemble des séquences possibles W. Cela se traduit par la recherche de $\widehat{W}$ maximisant la probabilité d'émission de W sachant X correspondant à l'équation suivante :

$$\widehat{W} = arg \max_{W} P(W \backslash X) \tag{1.1}$$

avec $P(W \backslash X)$ la probabilité d'émission de W sachant X. Après application du théorème de Bayes, cette équation devient :

$$\widehat{W} = arg \max_{W} \frac{P(X \backslash W)\, P(W)}{P(X)} \tag{1.2}$$

où $P(X)$ représente la probabilité d'observation de la séquence acoustique X, qui peut être considérée comme une valeur constante et donc être retirée de l'équation (1.2). Nous avons donc :

$$\widehat{W} = arg \max_{W} P(X \backslash W)\, P(W) \tag{1.3}$$

La probabilité $P(W)$ est fournie par le modèle de langage, et la probabilité $P(X\backslash W)$ correspond à la probabilité attribuée par les modèles acoustiques. Schématiquement, dans le cas d'un modèle acoustique dont l'unité de représentation est le phonème (le phonème est l'unité minimale du langage parlé), ce dernier va permettre de rechercher les suites de phonèmes les plus probables à partir des observations acoustiques. Les suites de phonèmes sont contraintes par le dictionnaire de phonétisation qui permet d'associer chacun des mots du vocabulaire avec sa (ou ses) représentation(s) phonétique(s). Le modèle de langage va permettre de sélectionner, parmi toutes les séquences de mots possibles, celle qui a la plus grande probabilité d'apparition.

La figure I.1 présente une vue d'ensemble d'un SRAP probabiliste, avec les trois ressources essentielles à son fonctionnement (modèles acoustiques, dictionnaire de phonétisation et modèle de langage).

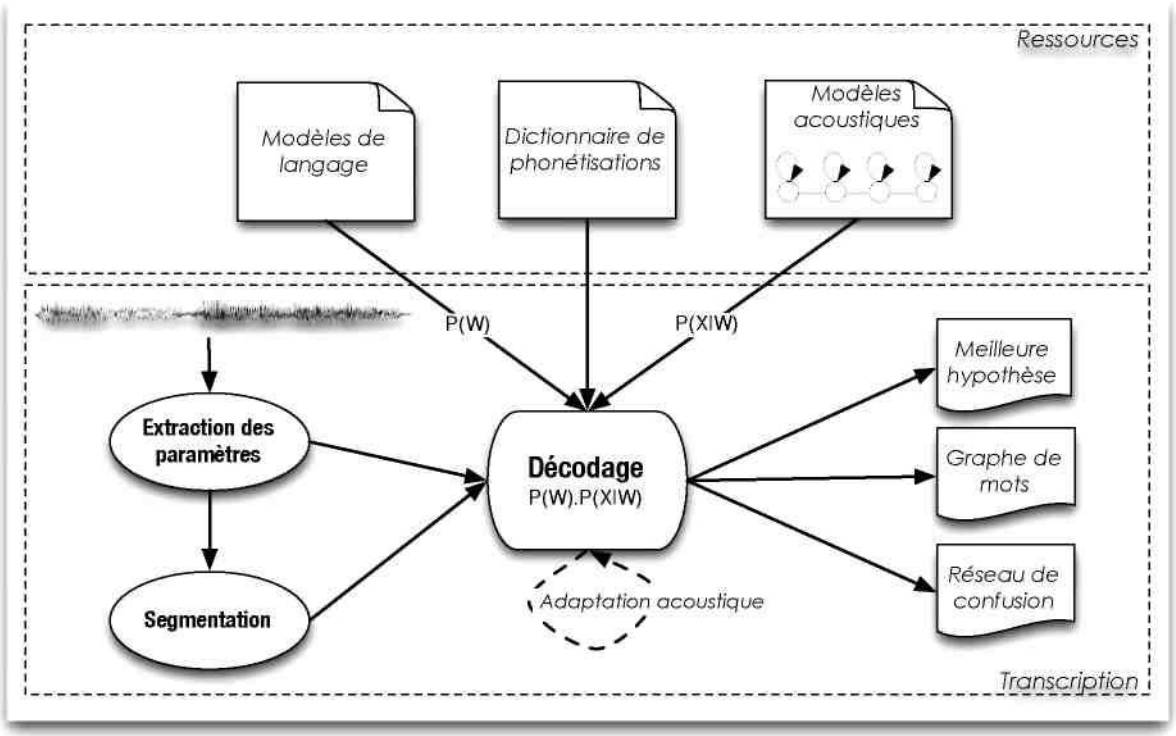

Figure I.1 : Vue d'ensemble d'un système de SRAP

I.2.2 Les avantages de la reconnaissance automatique de la parole

Les avantages de la reconnaissance de la parole sont nombreux. Elle libère complètement l'usage de la vue et des mains, et laisse l'utilisateur libre de ses mouvements. La vitesse de transmission des informations est supérieure, dans la RAP à celle que permet l'usage du clavier. Enfin tout le monde ou presque sait parler, alors que peu de gens sont à l'abri des fautes de frappe et d'orthographe, etc.

Ces avantages sont tellement importants que l'on trouve déjà sur le marché des dispositifs d'utilisation limitée, mais néanmoins efficaces. Citons certaines applications qui ont déjà vu le jour :

- Saisie vocale de données ;
- Donne des ordres tout en pilotant une automobile ou un avion ;
- Aide aux handicapés ;
- Chambre d'hôpital avec possibilités de commandes vocales pour le malade ;
- Commande vocale de machines ou de robots ;
- Commande vocale d'une montre portable, etc.

I.2.3 Complexité du problème

Pour appréhender le problème de reconnaissance automatique de la parole, il est bon d'en comprendre les différents niveaux de complexités et les différents facteurs qui en font un problème difficile.

Le signal parole est des plus complexes, et sujet à beaucoup de variabilités. Ayant été produit par un système phonatoire humain complexe, il n'est pas facile de le caractériser à partir d'une simple représentation bidimensionnelle de propagation de sons. On peut distinguer certaines de ses caractéristiques comme les sons élémentaires ou phonèmes, la hauteur, le timbre, l'intensité, la vitesse... Mais en réalité, la voix est beaucoup plus complexe qu'on ne peut percevoir par l'oreille. L'onde sonore varie non seulement avec les sons prononcés, mais également avec les locuteurs. La très grande variabilité que peut présenter un même discours selon la façon de parler du locuteur

qui peut chanter, créer, murmurer, être enroué ou enrhumé, et aussi selon le locuteur lui-même (homme, femme, enfant, voix nasillarde, différences de timbre), sans parler des accents régionaux, rend très délicate la définition d'invariants. Il faut pouvoir séparer ce qui caractérise les phonèmes, qui devrait être une constante quel que soit le locuteur et sa prononciation, de l'aspect particulier à chaque locuteur. Qu'est ce qui permet à notre cerveau de distinguer un mot d'un autre, indépendamment de celui qui nous parle ?

De plus, la mesure du signal de parole est fortement influencée par la fonction de transfert du système de reconnaissance (les appareils d'acquisition et de transmission), ainsi que par le milieu ambiant.

Ainsi l'obstacle majeur d'avoir une grande précision de la reconnaissance, est la grande variabilité des caractéristiques d'un signal vocal. Cette complexité du signal de parole provient de la combinaison de plusieurs facteurs, la redondance du signal acoustique, la grande variabilité inter et intralocuteur, les effets de la coarticulation en parole continue, et les conditions d'enregistrement.

- ***Problèmes indépendants de la langue :***

Le signal de parole n'est pas un signal ordinaire : il s'inscrit dans le cadre de la communication parlée, un phénomène des plus complexes. Afin de souligner les difficultés du problème, nous ferons ressortir essentiellement quelques caractéristiques notoires de ce signal :

- un débit intense
- une extrême redondance
- une grande variabilité
- un lieu d'interférences

D'un point de vue mathématique, il est ardu de modéliser le signal de parole, car ses propriétés statistiques évoluent au cours du temps.

I.2.3.1 Redondance du signal de parole

Quiconque a vu une représentation graphique de l'onde sonore a certainement été frappée par le caractère répétitif du signal de parole. Un grossissement à la loupe d'une brève émission de parole donne à voir une succession de figures sonores semblant se répéter à l'excès. Un peu de recul laisse apparaître des zones moins stables qu'il convient, de qualifier de transitoires. Ce qui semblerait de prime abord superflu, s'avère en réalité fort utile. Les répétitions confèrent à ce signal une robustesse. La redondance le rend résistant au bruit. Dans une certaine mesure, elle fonctionne comme un code correcteur d'erreur, puisqu'un interlocuteur humain sait décrypter un message même s'il est entaché de bruits dus à de possibles interférences.

I.2.3.2 Une grande variabilité

À contenu phonétique égal, le signal vocal est très variable pour un même locuteur (variabilité intralocuteur) ou pour des locuteurs différents (variabilité interlocuteur).

a- Variabilité intra-locuteur

Une même personne ne prononce jamais un mot deux fois de façon identique. La vitesse d'élocution en détermine la durée. Toute affection de l'appareil phonatoire peut altérer la qualité de la production. Un rhume teinte les voyelles nasales; une simple fatigue et l'intensité de l'onde sonore fléchit, l'articulation perd de sa clarté. La diction évolue dans le temps: l'enfance, l'adolescence, l'âge mûr, puis la vieillesse, autant d'âges qui marquent la voix de leurs sceaux.

b- Variabilité interlocuteur

Est encore plus flagrante. Les différences physiologiques entre locuteurs, qu'il s'agisse de la longueur du conduit vocal ou du volume des cavités résonnantes, modifient la production acoustique. En plus, il y a la hauteur de la voix, l'intonation et l'accent différent selon le sexe, l'origine sociale, régionale ou nationale.

Enfin toute parole s'inscrit dans un processus de communication où entrent en jeu de nombreux éléments comme le lieu, l'émotion, l'intention, la relation qui s'établit

entre les interlocuteurs. Chacun de ces facteurs détermine la situation de communication, et influe à sa manière sur la forme et le contenu du message.

I.2.3.3 La continuité

La production d'un son est fortement influencée par les sons qui le précédent et le suivent en raison de l'anticipation du geste articulatoire. L'identification correcte d'un segment de parole isolé de son contexte est parfois impossible. Évidement il est plus simple de reconnaître des mots isolés bien séparés par des périodes de silence que de reconnaître la séquence de mots constituant une phrase. En effet, dans ce dernier cas, non seulement la frontière entre mots n'est plus connue mais, de plus, les mots deviennent fortement articulés.

I.2.3.4 Le système est-il robuste ?

Autrement dit, le système est-il capable de fonctionner proprement dans des conditions difficiles? En effet, de nombreuses variables pouvant affecter significativement les performances des systèmes de reconnaissance ont été identifiées :

- Bruits d'environnement (dans une rue, un bistrot etc…) ;
- Déformation de la voix par l'environnement (réverbérations, échos, etc…) ;
- Qualité du matériel utilisé (micro, carte son etc…) ;
- Bande passante fréquentielle limitée (fréquence limitée d'une ligne téléphonique) ;
- Elocution inhabituelle ou altérée (stress, émotions, fatigue, etc…).

Certains systèmes peuvent être plus robustes que d'autres par rapport à l'une de ces perturbations, mais en règle générale, les systèmes de reconnaissance de la parole sont encore sensibles à ces perturbations.

I.2.3.5 Les effets de co-articulation

La production "parfaite" de chaque son suppose théoriquement un positionnement précis des organes phonatoires. Or, lorsque le débit de parole s'accélère, le déplacement de ces organes est limité par une certaine inertie mécanique.

Les sons émis dans une même chaîne acoustique subissent l'influence de ceux qui les suivent ou les précèdent, ces effets de co-articulation sont des interférences. Ils entraînent l'altération des formes sonores en fonction des contextes droits ou gauches, selon des règles étudiées par les acousticiens d'un point de vue articulatoire ou perceptif.

I.3 Le décodage acoustico-phonétique

Un décodage acoustico-phonétique (DAP) est défini généralement comme la transformation de l'onde vocale, en unités phonétiques telles que (phonèmes, diphones, syllabes, etc.), une sorte de transcodage qui fait passer d'un code acoustique à un code phonétique ou plus exactement comme la mise en correspondance du signal et d'unités phonétiques prédéfinies dans lequel le niveau de représentation passe du continu au discret [3].

Donc le décodage acousto-phonetique, consiste à décrire le continuum acoustique de parole en termes d'unités linguistiques discrètes.

Ce module est composé d'une première partie consistant à extraire les paramètres choisis pour représenter le signal, et d'une seconde partie qui, à partir de ces jeux de paramètres, apprend des modèles d'unités acoustiques où décode le signal d'entrée, selon que l'on veuille apprendre ou reconnaître.

I.4 Approche de la reconnaissance [4]

Il existe deux approches permettant d'aborder la reconnaissance de la parole : l'approche globale et l'approche analytique [5]. Elles se distinguent essentiellement par la nature et par la taille des unités abstraites qu'elles s'efforcent de mettre en correspondance avec le signal de parole.

I.4.1 Approche globale

Dans l'approche globale, l'unité de base sera le plus souvent le mot considéré comme une entité globale, c'est à dire non décomposée. L'idée de cette méthode est de donner au système une image acoustique de chacun des mots qu'il devra identifier par la suite. Cette opération est faite lors de la phase d'apprentissage, où chacun des mots est prononcé une ou plusieurs fois. Cette méthode a pour avantage d'éviter les effets de coarticulation, c'est à dire l'influence réciproque des sons à l'intérieur des mots. Elle est cependant limitée aux petits vocabulaires prononcés par un nombre restreint de locuteurs (les mots peuvent être prononcés de manière différente suivant le locuteur).

I.4.2 Approche analytique

L'approche analytique, qui tire parti de la structure linguistique des mots, tente de détecter et d'identifier les composantes élémentaires (phonèmes, syllabes, …). Celles-ci sont les unités de base à reconnaître. Cette approche a un caractère plus général que la précédente : pour reconnaître de grands vocabulaires, il suffit d'enregistrer dans la mémoire de la machine les principales caractéristiques des unités de base.

Dans l'approche globale, l'unité de base est le mot : le mot est considéré comme une entité indivisible. Une petite phrase, de très courte durée, peut aussi être considérée comme un mot.

Dans l'approche analytique, on tente de détecter et d'identifier les composantes élémentaires de la parole que sont les phonèmes.

Pour la reconnaissance de mots isolés à grand vocabulaire, la méthode globale ne convient plus car la machine nécessiterait une mémoire et une puissance considérable pour respectivement stocker les images acoustiques de tous les mots du vocabulaire et comparer un mot inconnu à l'ensemble des mots du dictionnaire. Il est de plus impensable de faire dicter à l'utilisateur l'ensemble des mots que l'ordinateur a en mémoire. C'est donc la méthode analytique qui est utilisée : les mots ne sont pas mémorisés dans leur intégralité, mais traités en tant que suite de phonèmes. Mais la

méthode analytique a un grand inconvénient : l'extrême variabilité du phonème en fonction du contexte (effets de la coarticulation).

I.4.3 Principe général de la méthode globale et analytique

Le principe est le même que ce soit pour l'approche analytique ou l'approche global, ce qui différencie ces deux méthodes est l'entité à reconnaître : pour la première il s'agit du phonème, pour l'autre du mot.

On distingue deux phases:

I.4.3.1 La phase d'apprentissage

Un locuteur prononce l'ensemble du vocabulaire, souvent plusieurs fois, pour créer en machine le dictionnaire de références acoustiques. Pour l'approche analytique, l'ordinateur demande à l'utilisateur d'énoncer des phrases souvent dépourvues de toute signification, mais qui présentent l'intérêt de comporter des successions de phonèmes bien particuliers.

I.4.3.2 La phase de reconnaissance

Un locuteur prononce un mot du vocabulaire. Ensuite la reconnaissance du mot est un problème typique de reconnaissance de formes. Tout système de reconnaissance des formes comporte toujours les trois parties suivantes:

- Un capteur permettant d'appréhender le phénomène physique considéré (dans notre cas un microphone) ;
- Un étage de paramétrisation des formes (par exemple un analyseur spectral) ;
- Un étage de décision chargé de classer une forme inconnue dans l'une des catégories possibles.

I.5 Les modes de fonctionnement d'un système de reconnaissance

Un système de reconnaissance peut être utilisé sous plusieurs modes :

I.5.1 Dépendant du locuteur (monolocuteur)

Dans ce cas particulier, le système de reconnaissance est configuré pour un locuteur spécifique. C'est le cas de la plupart des systèmes de reconnaissance de parole disponibles sur le marché. Les principaux systèmes de dictée vocale actuels possèdent une phase d'apprentissage recommandée avant toute utilisation (voire même une adaptation continue des paramètres au cours de l'utilisation du logiciel) afin d'effectuer une adaptation des paramètres à la voix de l'utilisateur.

I.5.2 multi-locuteur

Le système de reconnaissance est élaboré pour un groupe restreint de personnes. Le passage d'un locuteur à un autre du même groupe se fait sans adaptation.

I.5.3 Indépendant du locuteur

Tout locuteur peut utiliser le système de reconnaissance.

I.6 La reconnaissance selon le mode d'élocution

Le mode d'élocution caractérise la façon dont on peut parler au système. Il existe quatre modes d'élocution distincts :

I.6.1 Reconnaissance de mots isolés

La segmentation d'un message parlé en ses constituants élémentaires est un sujet difficile. Pour l'éviter, de nombreux projets de la RAP se sont intéressés à la reconnaissance de mots prononcés isolement. La reconnaissance des mots isolés ou tous les mots prononcés sont supposés être séparés par des silences de durée supérieure à quelques dixièmes de secondes, se fait essentiellement par l'approche globale. La structure d'un tel système est représentée par la figure I. 2.

Dans l'utilisation de ces systèmes, on distingue deux phases :

I.6.1.1 La phase d'apprentissage

Pour constituer le dictionnaire de référence acoustique (R_1,……, R_n), chaque mot du vocabulaire est représenté par une ou plusieurs références.

I.6.1.2 La phase de reconnaissance

Dans la quelle le système va chercher la référence R_m qui est la plus proche de l'image acoustique du mot a identifié T, au sens d'un indice de ressemblance D.

$$m = \arg\max_{1 \le r \le n}[D(T, R_r)] \qquad (1.4)$$

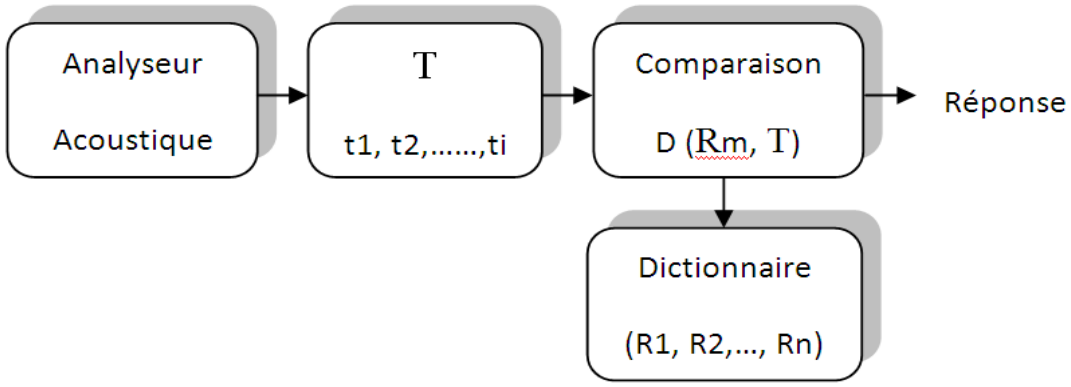

Figure I.2 : Reconnaissance de mots isolés

Les deux principaux problèmes inhérents à ce type de reconnaissance sont d'une part, l'extraction des paramètres pertinents du signal vocal afin de construire les images acoustiques des mots et d'autre part, la définition de l'indice de ressemblance p, qui doit prendre en compte les différences entre les échelles temporelles des images acoustiques à comparer.

I.6.2 Reconnaissance de mots enchaînés

Que se passe-t-il lorsqu'on essaye de passer de la reconnaissance de mots isolés à la reconnaissance de parole continue, sans silence obligatoire entre les mots? Évidement, la prononciation de chaque mot peut être fortement altérée par le phénomène de co-articulation entre les mots. En principe, il faudrait comparer la séquence d'entrée à toutes les références possibles correspondant à toutes les phrases

possibles. Il est clair qu'en général le nombre de ces combinaisons est beaucoup trop élevé, résultant en un besoin de place mémoire et en temps de calcul prohibitif. Cependant, pour résoudre le problème des mots enchaînés, il faut trouver des solutions aux problèmes suivants :

- La reconnaissance du nombre de mots dans la séquence ;
- La connaissance de fin et du début de chaque mot dans la séquence ;
- La coarticulation entre mots adjacents ;
- L'accroissement de l'espace mémoire et la quantité de calcul requise par les algorithmes de comparaison.

Comment peut-on déterminer la séquence optimale des mots enchaînés correspondant à la séquence prononcée ?

Une première méthode consiste à définir des règles de segmentation, pour un vocabulaire donné, en se basant sur des critères acoustiques. Cette méthode ne peut s'appliquer facilement à des vocabulaires plus importants comprenant des mots de plus d'une ou deux syllabes. De plus, les erreurs inévitables commises à ce niveau viennent limiter les performances du système avant même de considérer l'étage de reconnaissance.

La méthode préconisée est basée sur une comparaison globale de la suite à reconnaître avec une suite de références. Une suite de références est constituée par la concaténation de mots qui ont été prononcés par un locuteur d'une façon isolée. On ne tient donc aucun compte des effets de co-articulation : chaque mot est suppose indépendant de son entourage.

I.6.3 Reconnaissance de la parole continue [6]

Bien que les méthodes les plus adaptées à la reconnaissance de la parole continue sont les méthodes analytiques, plusieurs tentatives ont été faites pour la généralisation des méthodes de reconnaissance globales. Ces systèmes dont l'étape de décodage acoustico-phonélique est fondamentale s'articulent le plus souvent sur le niveau lexical.

Le schéma de principe (voir la figure I.3) d'un système de reconnaissance de la parole continue fait apparaître une succession de modules réalisant les différentes étapes du processus de reconnaissance Tout d'abord, un module acoustique extrait les caractéristiques physiques du signal, destinées au module phonétique, qui reconnaît les phonèmes prononcés, c'est à dire les sons élémentaires de la langue, en faisant appel à un dictionnaire de phonèmes. Ensuite le module lexical et le module phonologique souvent confondus, reconnaissent les mots, et utilisant pour cela un lexique des mots autorisés par l'application considérée, ainsi que des règles phonologiques décrivant les assemblages possibles de phonèmes dans la langue. Les modules syntaxiques et sémantiques, qui n'en font parfois qu'un, reconnaissent la phrase, en faisant appel à une description des règles de grammaire, et du sens des mots autorisés pour l'application considérée. Un niveau supplémentaire, appelé prosodique, portant sur la mélodie, le rythme, l'intensité du discours orale, intervient en parallèle avec les autres modules et fournit les informations qu'il extrait du niveau acoustique (hauteur, intensité, rythme) à tous les autres niveaux.

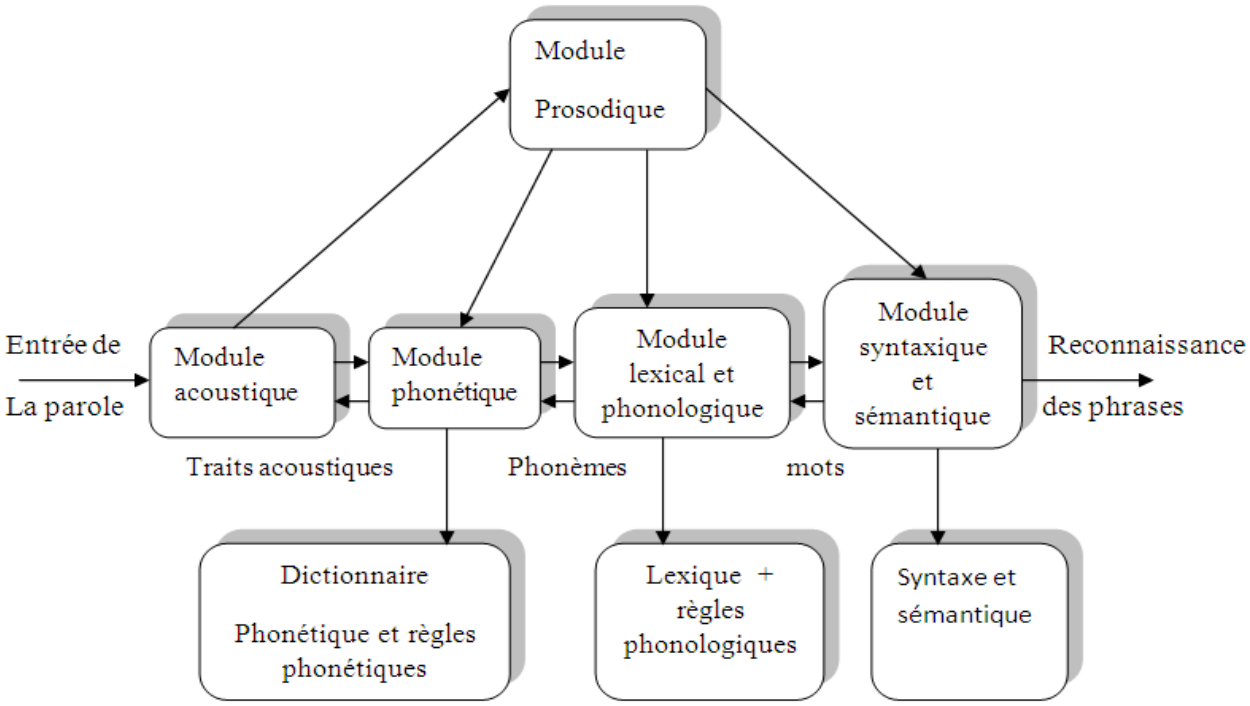

Figure I.3 : Schéma de principe d'un système de reconnaissance de la parole continue

I.7 Applications [7]

La technologie de reconnaissance n'est pas encore parfaite. Elle souffre encore des nombreux facteurs de variabilité relatifs aux conditions réelles d'utilisation. On peut distinguer plusieurs domaines importants d'application dont notamment :

I.7.1 Télécommunication

De nombreuses applications pourraient bénéficier de la reconnaissance de la parole afin de faciliter l'accès à des données ou des services sur lignes téléphoniques. Parmi celles-ci, on peut citer :

- L'automatisation des services de renseignements téléphonique ;

- Composition vocale du numéro d'appel (utilisation " mains libres", par exemple un téléphone de voiture) ;
- Commande et contrôle de services d'accès aux bases de données ;
- Services de réservation ou d'achat par téléphone.

Toutes ces applications, ainsi que beaucoup d'autres, sont généralement reprises sous un terme de "téléphone informatique" (computer telephony, en anglais) et leur développement est en pleine exploitation.

I.7.2 Application bureau/PC

Donner la possibilité de la reconnaissance de la parole par applications, et aux environnements des stations de travail, le contrôle et l'interaction homme- PC, dictée vocale.

I.7.3 Applications médicales et légales

Utilisant la reconnaissance automatique pour la rédaction de rapports ou remplissage de formulaires (rapports de radiologie, lettres légales, etc…)

I.7.4 Application militaires

Telles que le contrôle vocal de certaines des fonctions des appareils militaires (avions de chasse).

I.7.5 Production

Où la reconnaissance de la parole peut être utilisée pour la commande de contrôle vocal de processus de fabrication (par exemple, pour l'accès aux systèmes de contrôle de qualité) et apporter une aide au tri et envoi de paquets.

I.7.6 Autres applications

Telles que l'aide aux handicapés (contrôle par voix, machines à parler vocale) et l'utilisation de la reconnaissance de la parole dans les jeux électroniques.

Un système de reconnaissance de la parole peut être très précieux pour une catégorie de personne et même devenir indispensable pour d'autres. Mais, il vaut mieux réfléchir aux besoins réels d'un tel système avant d'investir du temps et l'argent. En générale « le choix d'une application doit faire l'objet d'une étude attentive, fondée sur l'ensemble de critères objectifs, en particulier, il est important

d'examiner si la voix apporte probablement un croisement des performances ou un meilleur confort d'utilisation. Par ailleurs, il ne faut pas trop attendre de la commande vocale mais la considérer en tout état de cause, comme un moyen complémentaire parmi d'autres moyens d'interaction homme/machine plus traditionnels » [3].

Cependant Plusieurs systèmes de reconnaissance de la parole sont disponibles sur le marché, ces systèmes se présentent généralement sous forme d'un ensemble de logiciels et de cartes d'acquisition et traitement de la parole. Ces produits sont mis en oeuvre sur des micro-ordinateurs, sur des stations de travail ou bien ils sont totalement autonomes.

I.8 Conclusion

La reconnaissance de la parole est l'une des tâches pionnières de l'intelligence artificielle consistant à reproduire la capacité d'un être humain à extraire des informations de la parole produite par un autre être humain. Cette tâche, trop complexe pour être reproduite par un système informatique unique, a été subdivisée en plusieurs sous-problèmes en fonction du type d'informations à extraire et à reconnaître. Les problématiques les plus étudiées sont la reconnaissance du locuteur, de son état émotionnel, de la langue employée et du langage parlé.

Le problème de la RAP est généralement abordé selon deux approches, analytique et globale. L'approche analytique permet d'aborder le problème de la reconnaissance de la parole continue, quant à l'approche globale inspirée par les méthodes de reconnaissance des formes, elle privilégie l'aspect information acoustique sur l'aspect linguistique. Cette approche est une manière de contourner les difficultés de l'analyse linguistique. La décision du système s'effectue après évaluation d'un taux de ressemblance entre la forme à reconnaître et une série de formes préalablement mémorisées. Chacune de ces approches a ses avantages et ses inconvénients, mais les approches analytiques s'avèrent supérieures aux approches globales quant à la qualité

des résultats, lorsque le nombre des énoncés potentiels est élevé et/ou lorsque la redondance acoustique est faible.

Que la reconnaissance soit globale ou analytique, le processus de reconnaissance de la parole commence par un prétraitement acoustique du signal vocal dont le but de réduire le flux d'information et d'éliminer les redondances présentées dans celui-ci.

Malgré ces problèmes et difficultés, les systèmes de reconnaissance automatique de la parole deviennent chaque année de plus en plus performants

CHAPITRE II

LES DIFFÉRENTS OUTILS DE RECONNAISSANCE DE LA PAROLE

II.1 Introduction

La Reconnaissance Automatique de la Parole (RAP) est une technologie informatique permettant à un logiciel d'interpréter une langue naturelle humaine. Elle permet à une machine d'extraire le message oral contenu dans un signal de parole. Cette technologie utilise des méthodes informatiques des domaines du traitement du signal et de l'intelligence artificielle [8]. Les applications qu'ont peut imaginer sont nombreuse : aider les personnes handicapées, contrôle vocal des machines, réservation des vols, apprentissage d'autres langues, etc. [9].

Vue l'importance de la RAP, plusieurs systèmes ont été développés pour la reconnaissance vocale, parmi les plus connus: Dragon Naturally Speaking, IBM Via voice, Microsoft SAPI et d'autre. Aussi, il y a des Open Sources comme HTK (Hidden Markov Model Toolkit) [10], ISIP [11], AVCSR [12] et CMU Sphinx [13-15]. Nous nous sommes intéressés à ce dernier qui est un système basé sur les Modèles de Markov Cachés (MMC), en anglais Hiden Markov Models (HMM) [16].

II.2 Les différentes applications

Dans le domaine de la reconnaissance automatique de la parole, on distingue trois grands types d'applications:

- les systèmes de commandes vocales ;
- les machines à dicter ;
- les systèmes de compréhension.

II.2.1 Les commandes vocales

On trouve aujourd'hui, un grand nombre de produits fiables sur le marché qui permettent de contrôler l'environnement (informatique ou non) au moyen d'une entrée vocale. La presse spécialisée, et dans une large mesure la grande presse, rendent compte de l'ensemble des produits aujourd'hui disponibles. Les applications multiples et variées vont du jouet gadget à l'outil de travail sophistiqué. Il ne nous est pas

possible de dresser une liste exhaustive de ces produits. Nous nous contenterons de citer quelques exemples d'applications, puis nous décrirons leurs traits communs.

- . Voiture (projet R 25) ;
- Jeux vidéo ;
- SNCF (noms de gares) ;
- Aide aux handicapés ;
- Reconnaissance de chiffres.

Pour des raisons plus ergonomiques que purement techniques, la taille du vocabulaire, dont disposent au moment du décodage les systèmes de commande vocale, est de façon générale, limitée à quelques centaines de mots.

Avec un vocabulaire dépassant le millier de commandes, les utilisateurs se heurteraient alors à deux problèmes pratiques :

- S'il est donné à tout individu moyen de mémoriser le contenu d'une liste de cent mots, il ne lui est pas du tout évident d'en mémoriser mille avec la même précision. Si un conducteur doit hésiter entre klaxon et avertisseur, feux de position, feux de croisement, feux de routes et lanternes, codes ou phares ... il peut se faire tard avant que ses lumières ne s'allument ou s'éteignent,
- Deuxièmement, dans le cas des systèmes mono-locuteurs, les utilisateurs trouveraient fastidieux un apprentissage qui consiste à prononcer plusieurs fois chacun des mots de la liste.

Bien qu'une grande souplesse soit donnée à l'utilisateur quant au choix du vocabulaire, il est recommandé de choisir des mots contrastés pour réduire les risques d'ambiguïté. Ces systèmes sont d'autant plus performants (reconnaissance fiable à 99 %) que les mots sont bien différenciables par la longueur, ou leur transcription phonétique. Certains de ces systèmes sont multi-locuteurs, et ne nécessitent donc pas d'apprentissage préalable. Dans ce cas, le taux de succès avoisine les 95 %. La méthode privilégiée était pendant longtemps essentiellement la comparaison de références par programmation dynamique [17]. Depuis le début des années 80, on utilise de plus en plus, des modèles markoviens [18, 19], auxquels certains chercheurs préfèrent aujourd'hui les réseaux neuronaux.

Des systèmes d'un autre type s'intéressent à la recherche de quelques mots clés ("word-spotting") en ignorant délibérément le reste des mots qui apparaissent dans le flot d'un discours non contraint. S'ils fonctionnent de façon satisfaisante, ils détectent la présence dans l'énoncé des mots du dictionnaire. Trois cas de dysfonctionnement de surface sont possibles : rester passifs face à un mot connu, en détecter un autre à la place, prendre à tort un mot inconnu pour un mot connu. Plus profondément, et même en cas de fonctionnement parfait, il est difficile d'exploiter les détections. En effet, un mot extrait d'un contexte négatif petit entraîne l'effet contraire à celui souhaité.

II.2.2 Les systèmes de compréhension

Bien que de nature profondément différente, les systèmes de compréhension se caractérisent aussi par un vocabulaire limité à quelques centaines de mots, et donc un domaine sémantique fermé. Plus gênant, la syntaxe est aussi contrainte. Pour faciliter la gestion du dialogue par le système, les phrases acceptables pour le système doivent se conformer à des schémas grammaticaux simplifiés. La fermeture du domaine sémantique peut être tolérable si l'application est bien ciblée.

En attestent les applications généralement choisies comme l'interrogation d'une base de données, les standards téléphoniques automatisés qui donnent des renseignements météo, ou même permettent de réserver des places.

Ces systèmes sont connectés à des modules d'interprétation du message reconnu, dont le but est de réagir soit par l'émission d'une réponse vocale soit par une action mécanique sur l'environnement, après prise de décision. De ce fait, la performance de ces systèmes doit être jugée sur la base du nombre de phrases reconnues. Le critère pourrait sembler sévère si l'on devait compter pour fausse toute phrase dont le moindre mot a été mal reconnu. Comme en pratique les paraphrases sont acceptées, la difficulté s'en trouve atténuée, mais l'évaluation ici n'est pas chose aisée et inclut, de fait, une part de subjectivité.

Beaucoup de grands systèmes ont été conçus autour du projet ARPA (Agence des projets de recherche avancée, en anglais "Advanced Research Project Agency ") lancé en 1977 aux USA. Aujourd'hui, plusieurs laboratoires travaillent sur le projet

DARPA (Agence pour les projets de recherche avancée de défense) (CMU, MIT, BBN, SRI, Bell labs, ...). Par exemple, le système SPHINX développé à CMU (l'université Carnegie Mellon), suppose l'emploi d'un petit vocabulaire (les mille mots du corpus « Ressource Management Data Base »). Il a par contre le mérite de permettre aux locuteurs une parole continue et ne nécessite pas d'apprentissage préalable. Le taux de reconnaissance sur les mots est excellent [20], puisqu'il est légèrement supérieur à 96%. En France, a été aussi explorée l'idée que les imperfections du décodage acoustico-phonétique n'empêchaient pas d'accéder au niveau supérieur de compréhension, mais qu'au contraire les faiblesses des niveaux "inférieurs" pouvaient être épaulées par le recours à un des niveaux plus "élevés". C'est dans cette optique qu'ont été conçus les systèmes : KEAL (système de Compréhension de la parole continue développé au CENT (Centre National d'Etudes des Télécommunication)) [17], Esope au LIMSI (Laboratoire d'Informatique pour la Mécanique et les Sciences de l'Ingénieur) [17], Myrtille au CRIN (Centre de Recherche en Informatique de Nancy) [21], Arial au laboratoire CERFIA de Toulouse [22].

II.2.3 Les systèmes de dictée automatique

Les machines à dicter ont pour but de retranscrire un texte dicté par un locuteur devant un microphone aussi bien qu'une secrétaire pourrait le faire, c'est-à-dire, en respectant au mieux les règles d'usage et d'accord orthographiques propres à la langue utilisée. La compréhension des phrases n'est nullement requise. Aussi, la plupart des systèmes de dictée automatique ne savent pas discriminer les différents sens d'un mot donné. S'ils le pouvaient, leurs performances seraient certainement meilleures.

On peut remarquer que ce domaine occupe un lieu charnière, à la frontière de l'oral et de l'écrit. De fait, les registres de langue traités ne sont pas ceux du langage parlé, mais plutôt ceux de l'écrit. En fonction de l'application envisagée, seront dictés des rapports, des articles de journaux, des lettres administratives, etc. Il en résulte que la complexité est moindre que s'il fallait retranscrire des dialogues à l'état brut. Dans le

vif d'une conversation, les phrases agrammaticales se mêlent aux phrases incomplètes, tandis que fourmillent hésitations, reprises, retours en arrières, ou autres répétitions. Cependant, l'exercice même de la dictée sous-entend l'utilisation de plusieurs dizaines voire plusieurs centaines de milliers de formes fléchies. Les mots sont pris en contexte, et régis par une syntaxe aussi libre que la grammaire de la langue naturelle le permet. De plus, s'il nous faut comparer la séance de dictée à une scène connue, il s'agit de reproduire plutôt le scénario du médecin dictant une lettre à sa secrétaire, que celle de l'institutrice vérifiant les connaissances de ses élèves. Cette précision est importante dans la mesure où l'utilisateur n'a pas forcément comme l'enseignant, devant les yeux un texte déjà écrit. Il improvise, et donc doit pouvoir se tromper, revenir en arrière afin de corriger un mot ou remanier une tournure.

Actuellement, l'état de l'art en reconnaissance de parole ne permet pas l'affranchissement conjoint de l'ensemble des quatre contraintes majeures décrites en tête de chapitre I (section I.2). Les systèmes qui gèrent des vocabulaires de grande taille (plusieurs milliers de mots), ne sont pas indépendants du locuteur. Ils nécessitent un apprentissage préalable, et ne peuvent encore aujourd'hui supporter un mode d'élocution non-contraint.

Historiquement, l'équipe de recherche IBM dirigée par F. Jelinek, est la première à avoir développé un système grand vocabulaire (Tangora 5 000 mots en 1985, 20 000 en 1987) pour la dictée vocale. Par la suite, l'ensemble des grands systèmes développés ici et là se sont inspirés peu ou prou du système Tangora. Avec un taux de réussite supérieur à 95% pour un vocabulaire de 20 000 mots, Tangora tend à devenir un système multi-lingue existant pour l'anglais [23], l'italien [24, 25], le français [26], et l'allemand [27], bientôt l'espagnol.

Le système Dragon [28] est l'un des rares produits présents aujourd'hui sur le marché. Il fonctionne en mots isolés avec un vocabulaire de base de 16 000 entrées extensible à 30 000. Un de ses points forts est sa capacité d'adaptation (par validation au clavier) en cours de décodage. Bien qu'un peu fastidieuse (il faut attendre le décodage d'un mot avant de prononcer le mot suivant), l'adaptation doit être faite convenablement sinon, revers de la médaille, le système diverge. Dans de telles

conditions, il est difficile d'interpréter le taux de reconnaissance (estimé aux alentours de 90 %) si sa variance n'est pas connue. Dragon a été conçu pour la langue anglaise telle qu'on la parle outre-atlantique, mais une transposition du système dans les langues européennes est en voie d'être confiée [29] à la société belge Lernout et Hauspie.

Aux USA, le second grand système vendu est la machine de Kurzweil (1 000 à 10 000 mots). Le KVT (Kurzweil Voice Terminal) ne s'est pas vraiment détaché de la commande vocale. Ses concepteurs affirment qu'il offre la possibilité de dicter en mots isolés un texte, propre à un domaine spécialisé (radiologie), tout en permettant de contrôler des machines, véhicules et autres robots. Cependant, la presse [30] ne lui confère pas le statut de système de dictée, les critères de définition n'étant pas atteints.

En France, la machine à dicter développée au LIMSI (5 000 à 10 000 mots) autour du circuit μPCD (micro-wave detected photo conductance decay) a abouti au produit DATAVOX (5 000 mots) commercialisé par la société VECSYS. Le taux de reconnaissance publié s'élève à 95% [31] pour un locuteur masculin. Le système Hamlet [32] est une maquette développée parallèlement. Ce prototype peut traiter un vocabulaire de 7 000 mots en supposant une élocution en mots isolés.

Par ailleurs, de nombreux laboratoires de recherche ont mis ou mettent encore au point les prototypes de produits futurs. Nous décrirons par la suite le système Parsyfal développé depuis 1985 au Centre Scientifique IBM France de Paris. Disons qu'une de ses caractéristiques fortes est de pouvoir traiter un dictionnaire de très grande taille (quelques centaines de milliers de formes en entrée). La liste des systèmes qui peuvent concurrencer Parsyfal quant à la taille du dictionnaire est réduite : le système développé l'INRS (Bell Northern) par M. Lennig fonctionne en anglais avec une capacité de 86 000 mots. Ensuite, on trouve les systèmes dédiés aux langues asiatiques comme celui réalisé pour le mandarin [33], et pouvant traiter 60 000 mots.

Les thèmes majeurs de recherche portent sur:

- un mode d'élocution naturel ;
- l'ouverture maximale du domaine de l'énoncé ;
- les méthodes connexionnistes concurrentes des modèles markoviens ;

- les approches de type intelligence artificielle ;
- l'adaptation rapide au locuteur, voire l'indépendance vis à vis du locuteur.

Ainsi, des recherches sur le mode de parole dit continu sont menées au laboratoire IBM de Yorktown. Des expériences [34] portantes sur un vocabulaire limité à 5 000 mots, ont donné un taux d'erreur sur les mots de 11%.

II.3 Classification des SRAP [34]

Les systèmes de reconnaissance automatique de la parole, ou RAP, sont classés selon plusieurs critères. Le premier est le nombre de locuteurs qui utilisent le système: dépendant du locuteur pour un seul utilisateur, multilocuteur pour quelques utilisateurs ou indépendant du locuteur pour un nombre illimité de personne. En règle général, plus le nombre d'utilisateur augmente, plus la performance diminue, et vice versa.

Le deuxième critère déterminant est le rythme de la parole ou l'élocution, comme des mots isolés pour un mot à la fois, des mots connectés pour des mots enchainés, la parole continue et la conversation. Plus le nombre de mots qui peuvent être dit par le locuteur augmente plus la performance diminue. Il existe aussi des systèmes de RAP par mots clés. La taille et complexité du vocabulaire joue aussi un rôle, car le vocabulaire peut aller d'une dizaine de mots à des dizaines de milliers de mots. La contrainte grammaticale peut aussi être importante pour classifier ces systèmes, car il est facile d'imaginer des grammaires de quelques lignes et d'autres de quelques 250MB compressées.

La qualité de transmission du signal acoustique et sa qualité d'acquisition sont aussi prises en compte pour la classification. La figure II.1 montre la classification des systèmes suivant la qualité de signal acoustique et les performances demandées pour la reconnaissance.

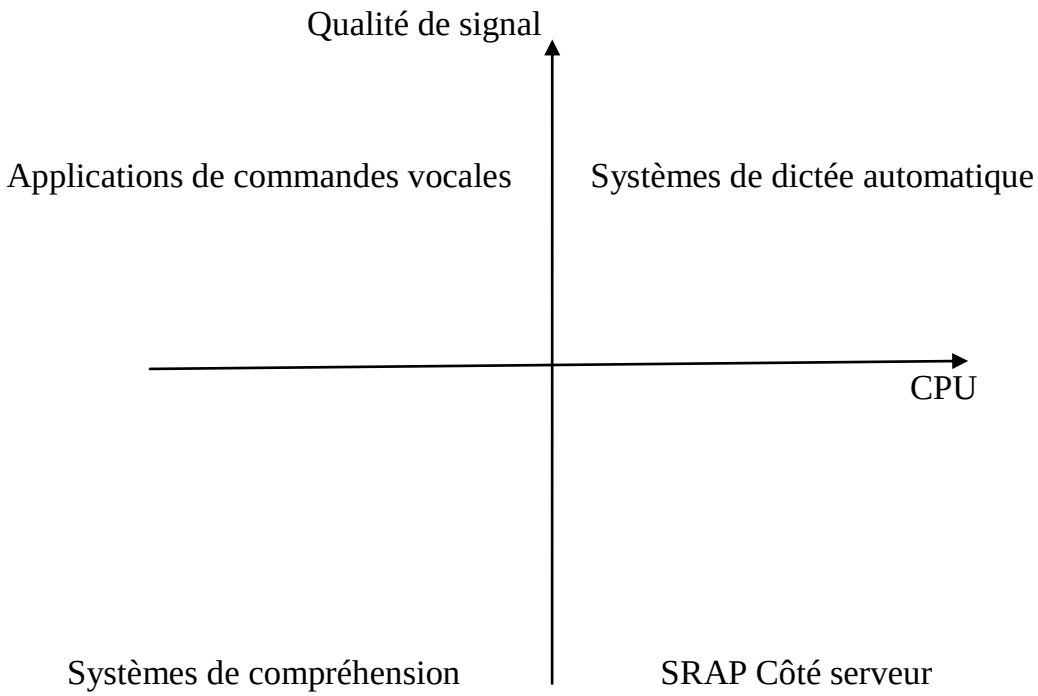

Figure II.1: Classification des systèmes RAP selon la qualité de signal et le CPU

Les systèmes utilisant la reconnaissance de la parole sont très exigeants pour le CPU et la RAM. Généralement la diminution de la qualité du signal s'accompagne par la réduction de la puissance CPU nécessaire. C'est le cas par exemple dans des systèmes de reconnaissance par mots clés (une douzaine de mots en général) qui peuvent fonctionner dans des téléphones portables pauvres en ressource.

Dans les applications de contrôle comme la navigation Internet, le système même s'il utilise peu de CPU, a besoin d'une bonne qualité de signal. En revanche pour des systèmes côté serveur, même s'il n'y a pas vraiment besoin d'une bonne qualité, le système utilise beaucoup de CPU et de RAM, à cause du grand nombre d'utilisateurs qui peuvent utiliser le système en même temps.

Pour des applications de dictée, ou aux buts pédagogiques pour apprendre une nouvelle langue, il faut beaucoup de ressources tant du CPU que de RAM. En général, une bonne qualité de signal s'accompagne d'un grand besoin de ressources.

II.4 Les applications de reconnaissance de la parole

II.4.1 Progression des applications de RAP

Ces technologies sont en effet porteuses d'un énorme potentiel, mais le marché professionnel n'en est encore qu'à ses débuts face à ce qui l'attend. Certes, il reste difficile à cerner, et même s'il concerne encore peu d'acteurs côté fournisseurs, les clients et utilisateurs sont déjà très nombreux. Pourtant, ce domaine semble encore appartenir à celui de la recherche. Ce n'est cependant pas tout à fait l'avis de Jean-Paul Haton, professeur au LORIA INRIA, spécialiste de l'intelligence artificielle et de la reconnaissance vocale, qui a inauguré le cycle des conférences. « Le traitement de la parole arrive à maturité, mais nous continuons de faire évoluer les techniques de reconnaissance vocale ». Le taux d'erreurs est décroissant au fur et à mesure que les technologies évoluent. Nous sommes aujourd'hui proches d'un taux d'erreurs de 10 % ».

Une méthodologie qui pourtant n'a pas changée. La reconnaissance vocale reste basée sur un modèle statistique. Entre le message vocal et le message reconnu viennent se placer les algorithmes de reconnaissance, qui fonctionnent selon un modèle stochastique (la parole dans le temps). Ces algorithmes reposent sur trois modèles : le modèle acoustique basé sur la technique « Markov caché » qui automatise la reconnaissance stochastique des états ; le modèle des mots autour d'un lexique ; le modèle linguistique basé sur la technique « n-grammes » [36], liée aux probabilités de rencontrer des suites de mots. C'est l'ensemble de ces modèles qui forme un outil de reconnaissance vocale. Mais comme le reconnait le professeur Haton, « il y a un problème majeur : il faut disposer d'énormes bases de données pour atteindre un taux de performance pertinent ». Alors, qu'en est-t-il exactement ? En réalité, il faut discerner les outils individuels, comme la dictée vocale, qui ont fait leur preuve mais

sous réserve de les utiliser dans un environnement favorable, et les outils collectifs, dont les performances se dégradent rapidement si on élargit le cercle de leur usage.

Par exemple, la dictée vocale fonctionne aujourd'hui avec un fort taux de satisfaction, sous réserve que ce soit dans un environnent isolé. La même application dans un environnement sonore identique au niveau de la voix de l'utilisateur, donnera un résultat nul. De même, un système même simple de commandes en anglais affichera un taux d'erreurs de l'ordre de 3,6 % sur des natifs anglophones, mais de 34,9 % sur les non natifs. Le français qui s'exprime sur une machine à commande vocale programmée pour une population américaine rencontrera inévitablement des difficultés.

II.4.2 Les meilleurs logiciels de reconnaissance de la parole

II.4.2.1 Dragon Naturally Speaking [28]

Ce produit révolutionnaire et facile à utiliser vous donne tout ce dont vous avez besoin pour être plus efficace avec votre PC. Transformez votre voix en texte trois fois plus vite que la plupart des gens de type avec une précision allant jusqu'à 99%. Il est si facile, vous pouvez l'utiliser dès la sortie de la boîte. Il apprend à reconnaître votre voix instantanément et améliore sans cesse plus que vous l'utilisez.

Dragon Naturally Speaking fonctionne avec les applications bureautiques les plus couramment utilisés, y compris Microsoft Word, Internet Explorer, AOL et plus. Juste au sujet de ce que vous faites maintenant en tapant peut être fait plus rapidement en utilisant votre voix. Créer et modifier des documents ou des courriers. Ouvrir et fermer des applications. Contrôlez votre souris et ordinateur de bureau entier.

II.4.2.2 MacSpeech Dictate [37]

MacSpeech Dictate vous étonnera par sa précision. Vous parlez simplement et laissez la reconnaissance au MacSpeech Dictate. MacSpeech Dictate reconnaît et comprend 13 variations de langue anglaise, neuf avec l'orthographe des États-Unis et quatre avec l'orthographe britannique.

II.4.2.3 E-Speaking [37]

Un système de compréhension et un programme de reconnaissance de la parole pour la commande et le contrôle de votre ordinateur et dictée en utilisant votre voix. Réduire ou éliminer les clics de souris ou du clavier. Ouvrez les sites Web, des documents, ou des programmes en utilisant votre voix. Effectuer des fonctions de navigation et d'édition, de dicter les lettres, notes de service, et les messages électroniques. Il permet aussi d'entendre et voir votre ordinateur, lire des emails, des documents et des notes de service.

II.4.2.4 Talking desktop [37]

Ce logiciel vous permet de donner des ordres à votre ordinateur avec un casque et d'écouter les réponses de votre ordinateur dans ses haut-parleurs. Le projet a commencé avec l'objectif général de développer un logiciel parlant et en écoutant avec une intelligence artificielle à l'arrière de l'esprit de chacun. Dans cette société l'objectif voulu offrir un environnement de bureau mains-libres combinées avec un produit de technologie vocale interactive pour la technologie d'aide et les marchés informatiques à domicile.

II.4.2.5 Tazti Logiciel de reconnaissance vocale par Voice Tech Group [37]

L'application Tazti a été conçue pour être un logiciel de recherche de reconnaissance de la parole. Les utilisateurs peuvent se connecter à internet, naviguer et rechercher les sites. Il permet à un utilisateur d'effectuer des recherches sur des moteurs de recherche populaires tels que : Google, Yahoo, MSN, Wikipedia, ASK, pour les vidéos, photos, musiques et autres éléments sur internet en parlant à leur PC, de contrôler quelques logiciels installés sur leur PC.

II.4.2.6 Reconnaissance vocale de Windows [37]

Reconnaissance vocale de Windows dans Windows Vista permet aux utilisateurs d'interagir avec leurs ordinateurs par la voix. Il a été conçu pour les personnes qui

veulent limiter considérablement leur utilisation de la souris et le clavier tout en maintenant ou en augmentant leur productivité globale. Vous pouvez dicter des documents et des courriers dans les applications grand public, utiliser les commandes vocales pour lancer et basculer entre les applications, contrôler le système d'exploitation, et même remplir des formulaires sur le Web.

II.4.2.7 Julius [37]

Il est d'une haute performance, basé sur la technique « n-grammes » et dépendant du contexte HMM, il peut effectuer pratiquement en temps réel le décodage sur la plupart des PC actuels dans la tâche de dictée 60000 mots. Des techniques de recherche principales sont entièrement intégrées, telles que lexique arbre, la manipulation des mots croisés de contexte dépendant, la sélection gaussienne [38, 39], etc. En outre de l'efficacité de recherche, il est également modulé soigneusement pour être indépendant de structures de modèles, et des différents types de HMM qui sont pris en charge. Des formats standard sont adoptées pour faire face à d'autres outils de modélisation gratuits tels que HTK, CMU-Cam SLM toolkit (CMU-Cambridge Statistical Language Modeling toolkit), etc.

II.4.2.8 Finger Voix [37]

Finger voix est un outil logiciel qui vous permet de contrôler votre souris et le clavier en utilisant seulement votre voix, de la manière la plus rapide possible.

II.4.2.9 IBM ViaVoice [37]

Grâce à la technologie ViaVoice intégré dans les petits périphériques mobiles actuels et systèmes télématiques automobiles, les développeurs peuvent facilement apporter aux utilisateurs un accès vocal à l'information, fournir l'intégration complète de la reconnaissance automatique de la parole et la lecture des textes pour les petits appareils mobiles, y compris les systèmes de télématique automobile et les téléphones à mains libres, aide à minimiser les compétences et le temps nécessaire pour

développer des applications vocales avancées pour les appareils et les systèmes à distance. Comprend une forme libre, aide au commandement pour les phrases de commande naturelles et intuitives qui n'ont pas besoin d'être connu à l'avance ou mémorisé pour une utilisation future.

II.4.2.10 VoxForge [37]

VoxForge a été mis en place pour recueillir la parole transcrite pour une utilisation dans les moteurs de reconnaissance de la parole Open Source (Speech Recognition Engines "SRE") tels que ISIP, HTK, Julius et le Sphinx.

II.4.2.11 Hidden Markov Model Toolkit (HTK) [40]

Le Hidden Markov Model Toolkit (HTK) est une boîte à outils portable pour créer et manipuler des modèles de Markov cachés. HTK est principalement utilisée pour recherche sur la reconnaissance vocale mais il a été utilisé pour de nombreuses autres applications, y compris la recherche en synthèse vocale, reconnaissance de caractères et séquençage de l'ADN. HTK est utilisé dans des centaines de sites à travers le monde.

HTK consiste en un ensemble de modules de bibliothèques et des outils disponibles sous langage source C. Les outils fournissent des équipements sophistiqués pour l'analyse du discours, HMM analyse de la formation, les tests et les résultats. La logiciel prend en charge HMM en utilisant à la fois continu gaussiennes mélange de densité et de distributions discrètes et peut être utilisé pour créer des systèmes HMM complexes. Le logiciel HTK contient une documentation complète et des exemples. Il est à noter, que ce qui a contribué au succès de HTK, est qu'il est accompagné d'une assez bonne documentation. Cette dernière manquant cruellement à d'autres systèmes de RAP tel que Sphinx 4 (voir section suivante).

Malgré tout, Sphinx 4 est mieux que le HTK et cela pour plusieurs raisons :

- De nombreuses applications utilisent Sphinx, en particulier JVoiceXML ;

- Sphinx bénéficie d'une plus grande communauté et par conséquent il est plus facile d'obtenir de l'aide ;
- Les droits de HTK sont détenus par Microsoft qui propose une licence un peu plus restrictive que Sphinx ;
- Une étude indienne a cherché à comparer HTK et Sphinx. Il s'est avéré que ces deux systèmes sont approximativement équivalents en terme d'erreur de reconnaissance, mais Sphinx a l'avantage d'être légèrement plus rapide.

II.4.2.12 CMU Sphinx 4 [41]

CMU Sphinx est l'un des systèmes open source les plus populaires de reconnaissance de la parole. Il est actuellement utilisé par les chercheurs et les développeurs dans de nombreux endroits dans le monde entier, y compris les universités, les instituts de recherche et de l'industrie. Le termes licence libérale de CMU Sphinx a fait un membre important de la communauté open source et a fourni un moyen à faible coût pour les compagnies pour bâtir des entreprises autour de la reconnaissance de la parole. Il permet aussi dans le cadre de la recherche principalement d'obtenir la transcription écrite de données orales. Avec un langage de programmation assez simple, basé sur des phonèmes, il permet d'obtenir des résultats prometteurs pour le développement d'applications libres.

Nous sommes intéressés à ce logiciel et nous allons l'étudier plus en détail dans le chapitre suivant.

II.5 Conclusion

Les applications de la reconnaissance de la parole existent là où ils peuvent remplacer/compléter une interaction déjà existante. Parfois, c'est le seul moyen de communication qui peut exister, par exemple pour des applications mains-libres (pour téléphones portables, smartphones, etc.). On a vu qu'en peut classer les systèmes de reconnaissance automatique de la parole selon son rôle ou objectif et suivant la qualité de signal acoustique et les performances demandées pour la reconnaissance.

Dans ce chapitre quelques applications de reconnaissance de la parole sont brièvement décrites. Il y a parmi eux des applications dites open source, le plus célèbre d'entre eux est le CMU Sphinx 4. La reconnaissance de ce dernier supporte notamment les mots isolés et les phrases (utilisation de grammaires), son architecture est modulable pour permettre de nouvelles recherches et pour tester de nouveaux algorithmes.

CHAPITRE III

APPLICATION DE CMU SPHINX À LA RECONNAISSANCE DE L'ARABE

III.1 Introduction

Le développement d'un système de reconnaissance automatique de la parole (SRAP) arabe a récemment suscité l'intérêt d'un certain nombre de chercheurs. Plusieurs tentatives de construction d'un SRAP arabe (SRAPA) ont été notées, et elles ont abouti à des résultats encourageants. [42, 45]. Cependant la plupart de ces tentatives se sont limités à un vocabulaire réduit.

Notre travail s'inscrit dans le cadre général de la RAP, il traite la langue arabe utilisant l'Open Source CMU Sphinx. Nous présentons dans ce travail, les bases de construction d'un système de reconnaissance automatique de l'arabe classique basée sur le CMU Sphinx4. Nous avons abouti à développer un SRAP multi-locuteur (speaker dépendant System).

Dans la première section de ce chapitre, nous allons commencer à présenter les Modèles de Markov Cachés pour expliquer les deux composants d'un SRAP à savoir le modèle acoustique et le modèle de langage. Dans la deuxième section, nous présenterons le logiciel SPHINX suivi de la reconnaissance de la langue arabe avec ce dernier, et pour finir nous présenterons le système réalisé, avant de conclure.

III.2 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:

- 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].
- 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, P(W|S) peut s'écrire :

$$\mathbf{P(W|S)} = P(W).P(S|W)/P(S) \qquad \text{(III.1)}$$

avec:

* P(W) est la probabilité à priori de la suite de mots W.

* P(S|W) est la probabilité du signal acoustique S, étant donné la suite de mots W.

* P(S) est la probabilité du signal acoustique (que nous supposons, par simplification, ne pas dépendre de W). P(S|W) est nommé Modèle Acoustique, et P(W) est nommé Modèle de Langage. Ces deux modèles peuvent être représentés comme un modèle Markovien.

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

- Une matrice A qui indique les probabilités de transition d'un état q_i vers un autre état (ou vers lui-même), soit $p(q_j|q_i)= a_{ij}$.
- 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 multigaussienne, définie par les vecteurs moyens, les matrices de covariance et des poids associés à chaque gaussienne.

- Une matrice $\prod$ 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 q_i. Cet état est particulier puisqu'il ne peut émettre d'observations.

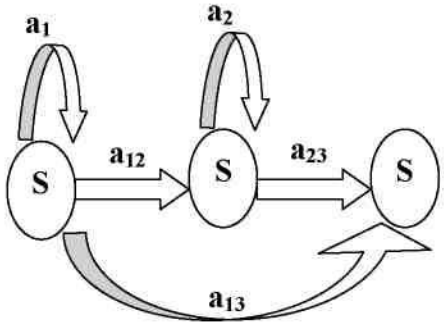

Figure III.1 : Modèle HMM dit gauche-droit d'ordre 1 à 3 états

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 a_i sur la figure précédente). Alors que pour les articulations rapides, le modèle admet le saut à l'état suivant (transition a_{13}).

Parmi les produits réussis dans le domaine de la reconnaissance de la parole, figure l'outil CMU Sphinx 4. Ses points forts résident, entre autres, dans son code ouvert qui offre une grande souplesse d'utilisation, et son appui sur le modèle HMM.

III.3 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.

- **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

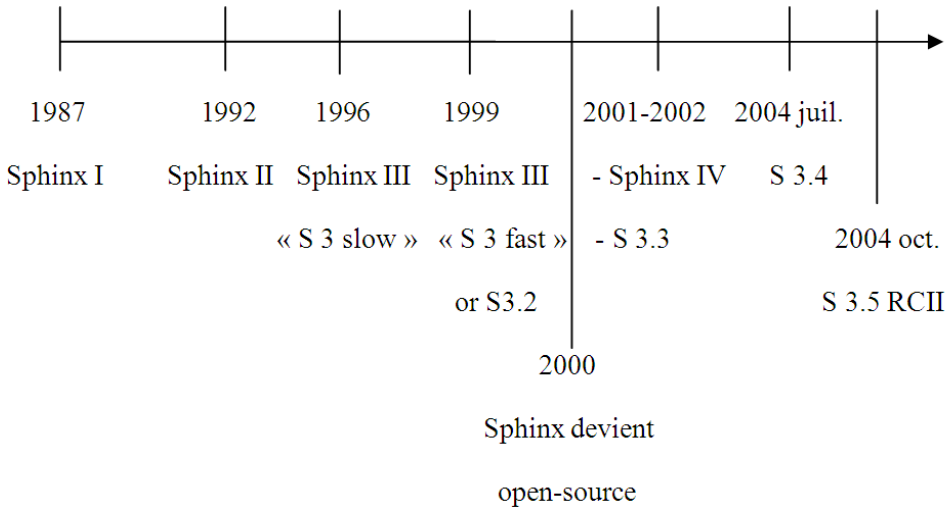

Figure III.2 : L'évolution des versions de SPHINX [51]

La disponibilité de code source Sphinx4 rend possible sa flexibilité et encourage la recherche dans les universités et les laboratoires spécialisés (HP, Sun...).

Le logiciel Sphinx4 utilise à la fois le modèle acoustique et le modèle de langage pour décoder un signal acoustique.

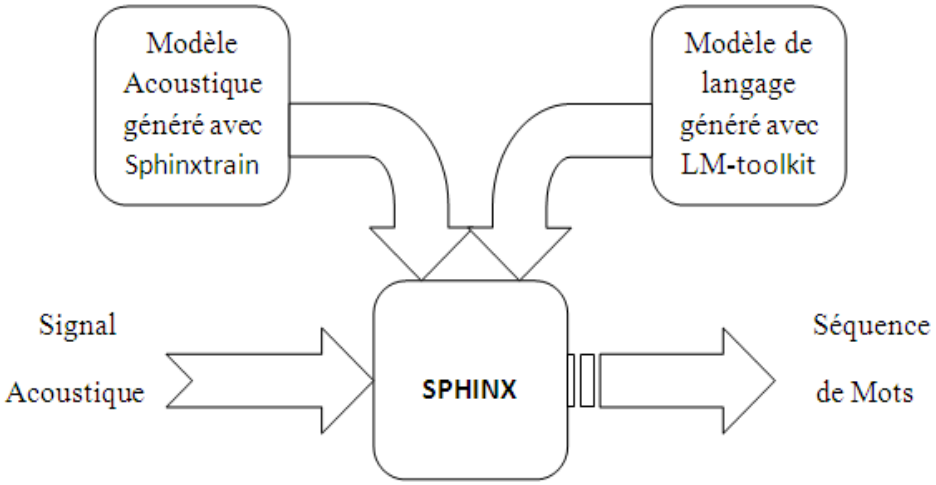

Figure III. 3 : Schéma simplifiant le traitement en utilisant le logiciel Sphinx 4

III.3.1 Sphinx-4

Sphinx est un projet lancé par l'université Carnegie Mellon (CMU) dans le but de concevoir un environnement pour la recherche dans le domaine de la reconnaissance automatique de la parole. CMU Sphinx 4 est une librairie de classes et d'outils disponible en langage de programmation Java. Cette librairie est gratuite à télécharger ; elle vise principalement à faciliter la construction des systèmes de reconnaissance vocale.

CMU Sphinx-4 est un système de RAP basé sur les Models 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].

Sphinx-4 utilise des HMM continus et fournit une grande flexibilité, exactitude et vitesse. Sphinx-4 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 comme le absoluteBeamWidth, relativebeamWidth, absolutWordBeamWidth, languageWeight, acousticLookahead, etc.

III.3.2 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.

III.3.3 Architecture

Sphinx-4 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.

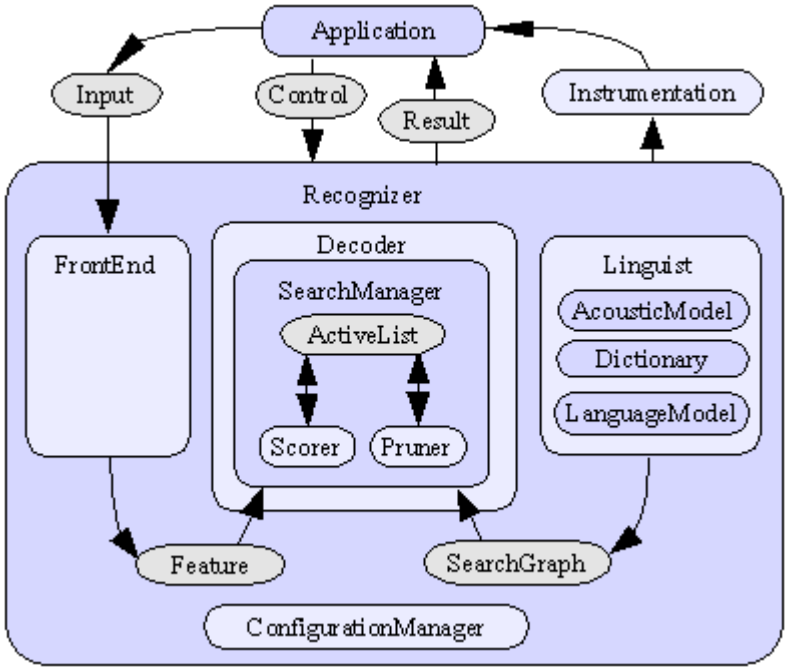

Figure III.4: Architecture du CMU Sphinx-4.

• **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.

• **Decoder :** ou Décodeur qui est le cœur de Sphinx-4 ; 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.

Sphinx-4 fournit différentes implémentations dans son système sous le répertoire demo/sphinx:

Hellodigits : utilisant une grammaire au format "Java Speech Grammar", et des modèles acoustiques "TIDIGITS" pour reconnaître des "connected digits". Où le "TIDIGITS" est un modèle acoustique pour l'anglais entraîné pour 11 mots (chiffres de 0 à 9 et le mot oh), utilisé dans les démos "hellodigits" et "zipcity", "wavfile" et "transcriber" avec différentes grammaires à état fini.

helloworld : reconnaît des phrases simples pour dire "hello will", etc. et utilise un JSGF grammaire (Java Speech Grammar Format), et des modèles acoustiques "WSJ". Où :

- JSGF est une écriture de grammaire standardisée adopte le style et la convention des développeurs Java. On peut donc importer d'autres grammaires JSGF, commenter le code de la grammaire, mettre des modificateurs de

visibilité. Peut être utilisé pour définir les phrases valides qui peuvent être reconnus.

- WSJ est un modèle acoustique qui a été entraîné en utilisant la base de données (Wall Street Journal).

hellongram : utilise un modèle de langage SimpleNGram et le même modèle acoustique "WSJ" que le helloworld.

Zipcity : pour reconnaître des codes postaux, zipcity est intégré dans une application Java Web Start pour montrer sur une carte la ville qui est associée aux codes postaux prononcés par l'utilisateur. C'est un JSGF grammaire avec le modèle acoustique TIDIGITS qui est utilisé.

wavfile, transcriber : montre comment la reconnaissance de nombres connectés depuis un fichier audio peut se faire. Ils utilisent un JSGF grammaire et le modèle acoustique TIDIGITS.

confidence : est un exemple d'implémentation pour montrer comment obtenir des "confidencescores" en utilisant Sphinx-4. Avec le modèle de langage 3-gram de 170 mots et le même modèle acoustique que le hellongram.

III.3.4 Création du 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 son 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.

Les données d'entrées sont composées, entre autre :

- D'un ensemble de fichiers acoustiques (corpus) ;
- D'un fichier de transcription qui contient l'ensemble de mots prononcés pour chaque enregistrement (fichier acoustique) ;
- D'un fichier dictionnaire qui définie la décomposition phonétique pour chaque mot ;
- D'un fichier qui définie la liste des phonèmes utilisées.

Sphinxtrain est l'outil destiné à créer le modèle acoustique. Il est composé d'un ensemble de script PERL (C'est un langage interprété, polyvalent et adapté au traitement et à la manipulation de fichiers texte, notamment du fait de l'intégration des expressions régulières dans la syntaxe même du langage) en outre des fichiers de configuration.

Nous avons procédé à une notation latine pour représenter les mots ainsi que les phonèmes utilisés.

Notre fichier des phonèmes contient 19 phonèmes.

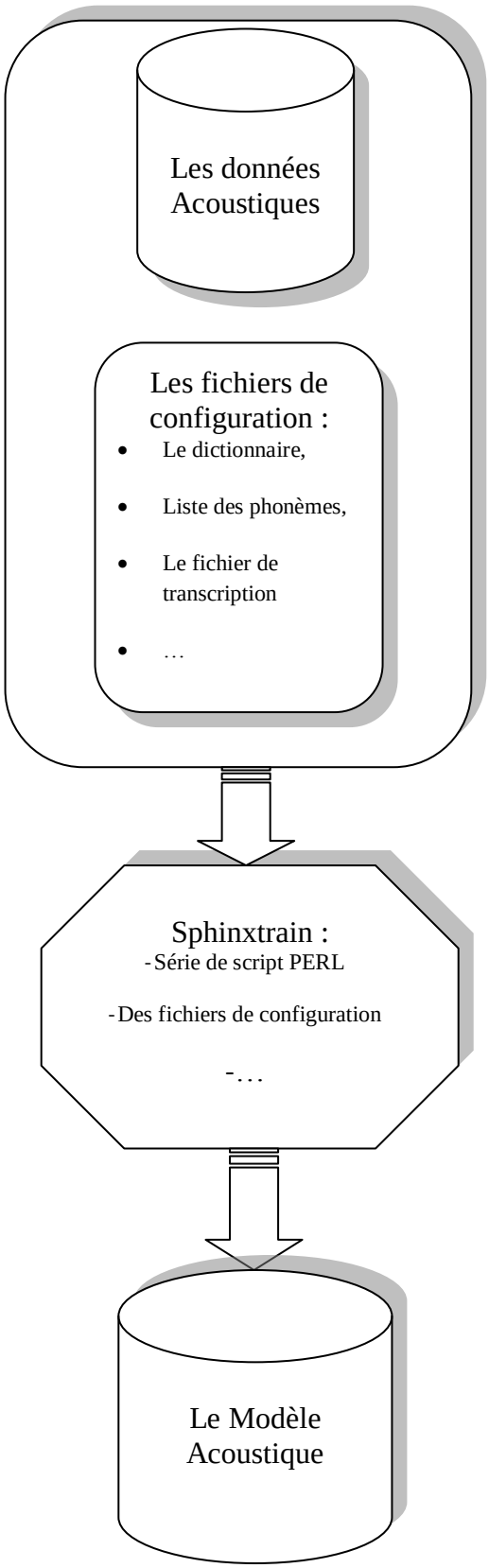

Figure III.5 : Schéma représentant la création d'un modèle acoustique avec Sphinxtrain

III.3.5 Installation

III.3.5.1 Sphinx-4

Sphinx-4 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 Sphinx-4 demande des logiciels supplémentaires qui sont :

• Java 2 SDK, Standard Edition 5.0 [59].

• Java Runtime Environement (JRE)

• Les différentes librairies qui composent Sphinx-4.

• Ant : L'outil pour faciliter la compilation en automatisant les taches répétitives [60].

III.3.5.2 Sphinxtrain

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

• ActivePerl : L'outil pour éditer des scriptes 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.

III.3.5.3 Implantation

La mise en œuvre de Sphinx-4 se compose de:

- Créer un nouveau projet à partir du menu Fichier ;
- Insérer les API Sphinx-4 dans le nouveau projet ;
- La dernière étape consiste à écrire un code Java dans le but de gérer et de déterminer les composantes à être utilisé dans le système.

III.4 Reconnaissance de la langue arabe

La langue arabe est une langue sémitique, elle est parmi les langues les plus anciennes dans le monde [62].

L'arabe classique standard a 34 phonèmes parmi lesquels 6 sont voyelles, trois voyelles brèves /a / /u/ /i/ et trois longues /aa/ /uu/ /ii/ qui s'opposent aux précédentes par une durée plus importante sur le plan temporel. Les 28 phonèmes restant sont des consonnes [63].

Les voyelles peuvent être arrangées en fonction du degré de l'étranglement au niveau du point d'articulation et en fonction de position de la langue (the tongue hump position) comme indiqué dans le tableau III.1.

Tableau III.1 : Classification des voyelles arabes

position de la langue / Degré de constriction	avant	central	arrière
Haut	i كسرة ii ياء ممدودة		u ضمة uu واو ممدودة
bas		a فتحة aa ألف ممدودة	

Les phonèmes arabes se distinguent par la présence de deux classes qui sont appelées pharyngales et emphatiques. Ces deux classes sont caractéristiques des langues sémitiques comme l'hébreu [63-65].

Les syllabes permises dans la langue arabe sont : CV, CVV, CVC, CVVC, CVCC et CVVCC. Où le V désigne voyelle courte ou longue et le C représente une consonne [63].

La langue arabe comporte six types de syllabes classées selon les trais ouvert/fermé et court/long. Une syllabe est dite ouverte (respectivement fermée) si elle se termine par une voyelle (respectivement une consonne). Le tableau III.2 montre leur classification.

Toutes les syllabes commencent par une consonne suivie d'une voyelle et elles comportent une seule voyelle. La syllabe CV peut se trouver au début, au milieu ou à la fin du mot [65-68].

Tableau III.2 : Classification des syllabes arabes

	ouverte	fermée
courte	CV	
longue	CVV	CVC, CVVC, CVCC CVVCC

III.4.1 Corpus

Le corpus est constitué des dix premiers chiffres de l'arabe classique de 0 à 9. 20 locuteurs, 12 males et 8 femelles, sont invités à prononcer les dix chiffres cinq fois. Le corpus comprend cinq répétitions par chaque locuteur du même chiffre. Ainsi, le corpus est constitué de 1000 tokens (10 chiffres. 5 répétitions. 20 locuteurs). Pendant l'enregistrement, chaque répétition a été rejouée pour s'assurer que le chiffre entier a été inclus dans le signal enregistré. Dans le tableau III.3 sont donnés certains paramètres d'enregistrement du corpus.

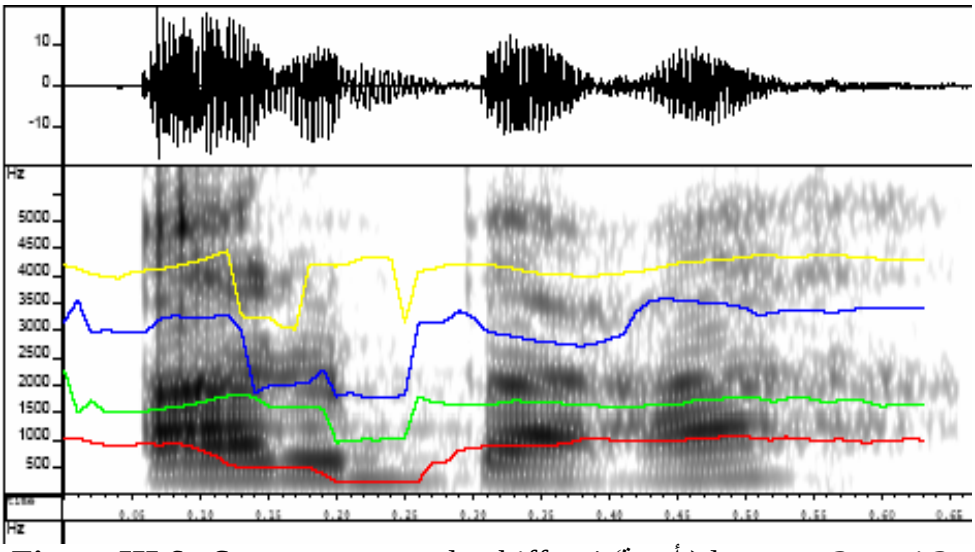

Figure III.6 : Spectrogramme du chiffre 4 (أربعة) locuteur 2 essai 2, généré par l'open source wavesurfer [68].

Tableau III.3: Paramètres d'enregistrement utilisés pour la préparation du corpus Arabic digits.

Paramètre	**Valeur**
Echantillonnage	16 kHz, 16 bits
Wave format	Mono, wav
Corpus	10 chiffres arabes
Locuteur	6 (3 maies + 3 femelles)

III.4.2 Modèle acoustique (Acoustic model)

Le modèle acoustique est une représentation statistique de l'image acoustique la plus significative possible pour le signal vocale. Durant la phase d'apprentissage (training), chaque unité acoustique ou phonème est représentée par un modèle statistique décrivant la distribution des données. Le signal parole est transformé en une série de vecteurs de caractéristiques (feature vectors) comprenant les coefficients MFCC (Mel-Frequency Cepstral Coefficients) [57]. L'apprentissage réalisé à l'aide de 1000 répétitions des données de parole collectées auprès des locuteurs.

Tableau III.4 : Symboles de phonèmes, utilisé pour Arabic digits.

Symbole	Alphabet	Translittération
AA	**أ**	Alef
B	**ب**	Ba'
T	**ت**	Ta'
TH	**ث**	Tha'
HH	**ح**	Ha'
KH	**خ**	Emphatique Kha'
D	**د**	Dal
R	**ر**	Ra'
AIN	**ع**	Ayn
S	**س**	Sin
SS	**ص**	Emphatique Sad
L	**ل**	Lam
M	**م**	Mim
H	**ه**	Ha'
W	**و**	Waw
Y	**ي**	Ya'
A	**ـَ**	Fatha
I	**ـِ**	Kasra
E	**ـ**	Entre kasra et fatha

Dans notre application, l'ensemble des symboles précédents (voir tableau III.4) a été utilisé pour l'apprentissage des états HMM correspondant au modèle acoustique de la démonstration Arabic_digits. 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 III.5.

Tableau III.5 : Extrait du fichier dictionnaire de l'application Arabic digits.

Mot arabe	Notation	Décomposition phonétique	Syllabes
صفر	0	S S E F R	CVCC
واحد	1	W AA HH I D	CV-CVC
اثنان	2	AA I TH N A A N I	CVC-CVCC
ثلاثة	3	TH A L A A TH A H	CV-CV-CVC
أربعة	4	AA A R B A AIN A H	CVC-CV-CVC
خمسة	5	KH A M S A H	CVC-CVC
ستة	6	S I T T A	CVC-CVC
سبعة	7	S A B B AIN A	CVC-CVC
ثمانية	8	TH A M A A N I Y Y A	CV-CV-CV-CVC
تسعة	9	T I S AIN A	CVC-CVC

III.4.3 Modèle de langue (Language model)

Modèle de langue (Language model ou grammar model) c'est un modèle qui définit l'usage des mots dans une application. Chaque mot dans le modèle de langue doit être dans le dictionnaire de prononciation. Le choix d'un modèle de langue dépend de l'application, dans certains cas il n'est pas facile, ce n'est pas le cas dans notre démonstration arabic digits (voir figure III.7**)**.

```
/**
* JSGF Digits Grammar for Hello Arabic Digits
example
*/
grammar arabicdigits;
public <arabicdigits> (0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9)* ;
```

Figure III.7 Extrait du fichier de grammaire de l'application Arabic digits.

III.4.4 Configuration

Un système de reconnaissance automatique de la parole comme Sphinx 4 utilise deux éléments dépendant de la langue : le modèle acoustique et le modèle de langue. Dans notre application nous avons procédé à la modification de ces deux modèles comme décrit précédemment.

Le sphinx 4 doit être configuré en utilisant un fichier xml. Ainsi le choix d'algorithmes, l'extraction et la comparaison de vecteurs de caractéristiques et d'autres aspects important pour la création d'un système de RAP peuvent être personnalisés au besoin et au choix de l'application considérée.

La propriété absoluteBeamWidth a pour tâche de donner la taille désirée pour la liste active, il faut lui mettre comme valeur −1 pour ne pas poser de limites de tailles à cette liste. Pour l'exemple de hellongram, la valeur optimale est de 500. En général, la meilleure valeur est obtenue avec 2000 [69]. La propriété relativeBeamWidth sert à placer relativement les scores minimaux par rapport aux scores maximaux dans la liste pour l'élagage. Selon l'Université de Gothenburg la valeur optimale est de 1E-80 [69].

La propriété languageWeight définit le poids du langage pour la recherche. La diminution de cette valeur réduit la contribution du modèle de langage au score final.

Les lignes suivantes définissent les propriétés fréquemment utilisés. Ils sont situés en haut du fichier de configuration afin qu'elles puissent être modifiées rapidement.

```
<property name="logLevel" value="WARNING"/>

<property name="absoluteBeamWidth" value="-1"/>
<property name="relativeBeamWidth" value="1E-80"/>
<property name="wordInsertionProbability" value="1E-36"/>
<property name="languageWeight" value="8"/>

<property name="frontend" value="epFrontEnd"/>
<property name="recognizer" value="recognizer"/>
<property name="showCreations" value="false"/>
```

III.5 Conclusion

Pour créer un système de reconnaissance de la parole avec sphinx 4, il faut tout d'abord créer une base de données audio à partir de laquelle, on construit un modèle acoustique passant par une importante étape qui est la phase d'apprentissage.

La construction d'un modèle acoustique pour sphinx 4 nécessite un logiciel spécifique qui est Sphinxtrain. Ce dernier est composé d'un ensemble de script PERL et des fichiers de configuration, qu'il faut modifier leur contenue selon les exigences de notre système et selon nos besoins.

CHAPITRE IV

APPLICATION ET RÉSULTATS

IV.1 Introduction

SphinxTrain est une application qui permet de créer des modèles acoustiques pour les différentes versions de Sphinx.

Créé à la base pour Sphinx2 et les différentes déclinaisons de Sphinx3, il est également utilisé pour la version Java à savoir Sphinx4. En effet, ce dernier utilise les mêmes modèles acoustiques que Sphinx3, cependant il faudra à la fin de la création procéder à quelques opérations supplémentaires, puisque Sphinx4 utilise des fichiers JAR (en informatique, un fichier JAR (Java ARchive) est un fichier compressés utilisé pour distribuer un ensemble de classes Java) pour la lecture de ces modèles (auxquels il faut fournir quelques informations).

En débutant avec Sphinx et plus généralement avec le domaine de la reconnaissance vocale, on regrette le fait qu'il y ait peu d'informations concernant la création d'un modèle acoustique.

C'est pourquoi, dans ce chapitre, nous expliquons en premier temps, étape par étape comment utiliser SphinxTrain pour créer un modèle acoustique permettant la reconnaissance des chiffres arabes allant de zéro à neuf et les points susceptibles de provoquer une erreur et sur laquelle on pourrait perdre beaucoup de temps à chercher la solution.

IV.2 Apprentissage du modèle acoustique

Une fois que l'on possède un corpus, on peut passer à l'étape de la création du modèle acoustique. Pour cela, il faut savoir quelle base sonore utiliser (phonèmes, syllabes, mots).

Par exemple, l'utilisation des phonèmes permet par la suite de pouvoir rajouter de nouveaux mots dans le dictionnaire (en spécifiant quels phonèmes interviennent pour les mots donnés) sans même avoir à recréer un nouveau modèle.

Il est à noter, que même si Sphinx est pourvu d'une communauté relativement importante, les étapes de la création d'un modèle acoustique sont difficilement trouvable (les informations sont assez dispersés sur l'Internet et on trouve plusieurs bribes dans les différents forums du CMU). Par ailleurs, une certaine expertise est nécessaire avant toute création, puisqu'il faut non seulement définir plusieurs paramètres qui influenceront la qualité de la reconnaissance, mais aussi apporter certaines modifications dans le code des scripts de SphinxTrain corrigeant quelques bugs.

L'apprentissage des modèles acoustiques passe par plusieurs étapes préalables. La durée de traitement étant assez variable. Pour avoir une idée, elle varie de quelques minutes, dans le cas d'un petit corpus, à plusieurs semaines de traitements si l'on souhaite avoir un modèle grand vocabulaire, indépendant du locuteur et à base de phonèmes dépendant du contexte.

IV.3 Création d'un modèle acoustique avec SphinxTrain

IV.3.1 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
    - file_1.wav - *Recording of speech utterance*
  - speaker_2
    - file_2.wav

IV.3.2 Compilation des packages nécessaires

Les packages suivants sont requis pour l'apprentissage :

- sphinxbase
- SphinxTrain

Les packages suivants sont externes et 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 chiffres dans notre cas avec le contenu suivant :

```
Chiffres\

  SphinxTrain
  SphinxTrain.tar.gz
  sphinxbase
  sphinxbase.tar.gz
```

IV.3.3 Mise en place des scripts d'apprentissage

Dans cette étape, on a testé deux scripts différents celui de sphinxtrain et celui de sphinxtrain snapshot.

Pour commencer l'apprentissage, on se place dans le dossier de base de données et on exécute les commandes suivantes :

pour sphinxtrain earlier than 1.0.7

../SphinxTrain/scripts_pl/setup_SphinxTrain.pl -task chiffres
../pocketsphinx/scripts/setup_sphinx.pl -task an4

pour sphinxtrain snapshot

sphinxtrain -t chiffres setup

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

IV.3.3.1 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’extention et le type des fichiers audio enregistrés, dans notre cas, on change ‘sph’ par’wav’ et’ nist’ par ‘mswav.

IV.3.3.2 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"
```

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 (chiffres).

IV.3.3.3 Configuration des paramètres caractéristiques du son

La valeur par défaut pour les fichiers audio utilisés dans Sphinx est un taux de 16000 échantillons par seconde (16 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 8 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 = 8000.0;
$CFG_NUM_FILT = 31; # For wideband speech it's 40, for telephone 8khz
reasonable value is 31
$CFG_LO_FILT = 200; # For telephone 8kHz speech value is 200
$CFG_HI_FILT = 3500; # For telephone 8kHz speech value is 3500
```

IV.3.3.4 Configuration des paramètres de décodage

Pour sphinxtrain 1.0.7 ouvrir le dossier `etc/sphinx_decode.cfg`.

Pour sphinxtrain snapshot ouvrir le dossier `etc/sphinx_train.cfg`.

Il faut assurer que les éléments suivants sont correctement configurés:

```
$DEC_CFG_DICTIONARY     =
"$DEC_CFG_BASE_DIR/etc/$DEC_CFG_DB_NAME.dic";
$DEC_CFG_FILLERDICT     =
"$DEC_CFG_BASE_DIR/etc/$DEC_CFG_DB_NAME.filler";
$DEC_CFG_LISTOFFILES    =
"$DEC_CFG_BASE_DIR/etc/${DEC_CFG_DB_NAME}_test.fileids";
$DEC_CFG_TRANSCRIPTFILE =
"$DEC_CFG_BASE_DIR/etc/${DEC_CFG_DB_NAME}_test.transcription";
$DEC_CFG_RESULT_DIR     = "$DEC_CFG_BASE_DIR/result";

# These variables, used by the decoder, have to be user defined, and
# may affect the decoder output

$DEC_CFG_LANGUAGEMODEL_DIR = "$DEC_CFG_BASE_DIR/etc";
$DEC_CFG_LANGUAGEMODEL  = "$DEC_CFG_LANGUAGEMODEL_DIR/nom de
projet.lm.DMP";
```

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

IV.3.4 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

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:

```
Baum welch starting for 2 Gaussian(s), iteration: 3 (1 of 1)
0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%
Normalization for iteration: 3
Current Overall Likelihood Per Frame = 30.6558644286942
Convergence Ratio = 0.633864444461992
Baum welch starting for 2 Gaussian(s), iteration: 4 (1 of 1)
0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%
Normalization for iteration: 4
```

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

IV.3.5 Déroulement interne de l'apprentissage

Cette section décrit ce qui se passe pendant l'apprentissage. Dans le répertoire des scripts (. / Scripts_pl), il ya plusieurs répertoires numérotés de manière séquentielle à partir de 00 * à 99 *. Chaque répertoire possède soit un répertoire nommé slave*.pl ou il a un seul fichier avec l'extension «. pl ». Le

script va successivement à travers les répertoires et exécute soit le slave*.pl. ou le seul fichier. Pl, comme ci-dessous.

```
perl scripts_pl/000.comp_feat/slave_feat.pl
perl scripts_pl/00.verify/verify_all.pl
perl scripts_pl/10.vector_quantize/slave.VQ.pl
perl scripts_pl/20.ci_hmm/slave_convg.pl
perl scripts_pl/30.cd_hmm_untied/slave_convg.pl
perl scripts_pl/40.buildtrees/slave.treebuilder.pl
perl scripts_pl/45.prunetree/slave-state-tying.pl
perl scripts_pl/50.cd_hmm_tied/slave_convg.pl
perl scripts_pl/90.deleted_interpolation/deleted_interpolation.pl
```

Plusieurs nouveaux répertoires ont été créés. Ces répertoires contiennent les fichiers qui sont générés dans le cadre de notre apprentissage. À ce stade, il n'est pas nécessaire de connaître le contenu de ces répertoires, même si certains des noms de répertoires peuvent être explicites et on peut les explorer.

L'un des fichiers qui apparait dans votre répertoire courant est un fichier. Html, nommé chiffres.html, selon la base de données que nous utilisons. Ce fichier contient un rapport sur l'état des travaux déjà exécutés.

Notons que dans le processus de passer par les scripts de 00 à 90 *, on aura généré plusieurs séries de modèles acoustiques, chacune d'entre elles pourraient être utilisées pour la reconnaissance. Notons également que certaines des étapes ne sont nécessaires que pour la création de modèles semi-continus. Si on exécute ces étapes tout en créant des modèles continus, les scripts seront avec bienveillance ne rien faire.

Sur l’étape 000.slave_feat les fonctionnalités ‘’Feles’’ sont extraites. Le système ne travaille pas directement avec des signaux acoustiques. Les signaux sont tout d'abord transformés en une séquence de vecteurs caractéristiques, qui sont utilisés à la place des signaux acoustiques actuels.

Le script "Make_feats.pl" va calculer, pour chaque énoncé d'entraînement, une séquence de 13-vecteurs de dimension (vecteurs

caractéristiques) constituées avec les Mel-frequency cepstral coefficients (MFCC).

Les MFCC seront placées automatiquement dans un répertoire appelé «feat». Notons que le type des vecteurs caractéristiques que nous calculons à partir des signaux de parole pour l'apprentissage et la reconnaissance, ne se limite pas à MFCC. Vous pouvez utiliser n'importe quelle technique de paramétrage raisonnable au lieu, et de calculer les caractéristiques autres que MFCC. CMUSphinx pouvez utiliser les fonctions de n'importe quel type ou dimension [70].

Une fois les travaux lancés à partir 20.ci_hmm et exécutés jusqu'à la fin, on aura entraîné un modèle acoustique Context-Independant (CI) pour les unités de sous-mots de notre dictionnaire.

IV.4 Etapes d'apprentissage pour notre modèle acoustique

IV.4.1 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
```

IV.4.2 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 chiffres.

```
mkdir chiffres
cd chiffres
```

Il faut à présent créer la structure du répertoire que l'on vient de créer. Cela se fait automatiquement grâce au script Perl fournit par SphinxTrain.

```
SPHINXTRAIN = /home/mon_nom/mon_rep/SphinxTrain/tutorial/SphinxTrain
$SPHINXTRAIN/scripts_pl/setup_SphinxTrain.pl –task chiffres
```

Il est impératif à ce niveau, de créer deux fichiers

```
touch etc/chiffres.fileids
touch etc/chiffres.transcription
```

Le premier fichier (chiffres.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 (chiffres.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 rires, les sonneries de téléphones ou tout simplement les « euh » « hum » « ah ».

```
touch etc/chiffres.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/chiffres.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.

Il faut maintenant convertir les fichiers audio du format raw, vers un format adapté à SphinxTrain et que l'on appellera fichiers de caractéristiques (features files). Nous utilisons pour cela un script.

```
cd chiffres
bin/make_feats –ctl etc/chiffres.fileids
```

Il existe dans ce script une erreur non corrigée qui provient de la mauvaise utilisation du format de fichiers. C'est pourquoi il faut éditer le fichier « bin/make_feats » pour y rechercher la ligne comportant

```
"bin/wave2feat –verbose yes –c \"$ctl\" –nist yes "
```

et la remplacer par la ligne suivante

```
"bin/wave2feat –verbose yes –c \"$ctl\" –raw yes "
```

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 chiffres.dic
touch chiffres.phone
```

Quelques précisions sur ces deux fichiers

- chiffres.dic va contenir la liste de tous les mots contenus dans le fichier de transcription (chiffres.transcription) avec pour chacun leur décomposition en phonèmes. Un mot peut être prononcé de différentes manières, c'est pourquoi

certains mots sont suivis d'un chiffre entre parenthèse indiquant une variante de la prononciation.

- chiffres.phone va quant à lui contenir la liste de tous les phonèmes contenus dans le fichier chiffres.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 le script de vérification qui nous permet de valider la configuration et de pouvoir procéder à la création du modèle acoustique.

```
./scripts_pl/00.verify/verify_all.pl
```

Malgré le soin pris dans la configuration de SphinxTrain, quelques erreurs lors de cette étape peuvent apparaître. Dans la plupart des cas cela résulte de la version de SphinxTrain utilisée et s'avère être bénigne.

Par exemple, il suffira simplement de renommer les fichiers chiffres.fileids et chiffres.transcription respectivement par chiffres_train.fileids et chiffres_train.transcription.

Aussi, si vous utilisez une ancienne version de SphinxTrain, il se peut que le fichier de configuration « etc/sphinxtrain.cfg » contienne la ligne suivante :

```
$CFG_FEATFILE_EXTENSION = 'feat'
```

Dans ce cas il est préférable de le remplacer par

```
$CFG_FEATFILE_EXTENSION = 'mfc'
```

On remarquera que, suite à l'exécution du script de vérification, un fichier HTML est créé. Celui-ci s'avérera fort utile pour le déboguage et l'évaluation de l'efficacité du modèle. En effet, durant toutes les étapes qui vont suivre, ce fichier sera mis à jour et contiendra toutes les erreurs et tous les avertissements (warnings), ainsi qu'une trace de l'exécution.

IV.4.3 Création du modèle acoustique

Fort heureusement, SphinxTrain contient une suite de scripts à exécuter. Ils possèdent un numéro qui correspond à l'ordre d'exécution.

Néanmoins, il ne suffit pas tout simplement de les exécuter un par un (bien que cela fonctionnerait également). En effet, certains scripts ne sont utiles que pour Sphinx2, d'autres inutiles dans le cas de la création d'un modèle acoustique Context-Independant (CI).

On entend par là, un modèle acoustique où les phonèmes ne sont pas modélisés de sorte à tenir compte du contexte dans lesquels ils apparaissent. En effet, un son ne va pas posséder les mêmes caractéristiques selon le son qui le précède et le son qui le suit. Dans le cas d'un modèle acoustique CI on ne tient pas compte de cette information, ce qui a pour conséquence de réduire le temps de création du modèle et de réduire en même temps la qualité de la reconnaissance (dans le cas d'un grand dictionnaire).

Les scripts de numéro 1, 8 et 9 sont exclusivement utilisés pour Sphinx2. Pour ceux qui seraient intéressés par la création d'un modèle pour cette version de Sphinx, prenez note qu'il existe une petite correction à apporter au script. En effet, l'exécution de la commande

```
./scripts_pl/01.vector_quantize/slave.VQ.pl
```

fait à son tour appel à deux autres scripts, dont « kmeans.pl » qu'il faudra éditer pour remplacer la ligne

```
$logfile = "$logfile/${CFG_EXPTNAME}.kmeans.log";
```

par

```
$logfile = "$logdir/${CFG_EXPTNAME}.kmeans.log";
```

On peut à présent utiliser le reste des scripts.

```
./scripts_pl/02.ci_schmm/slave_convg.pl
./scripts_pl/03.makeuntiedmdef/make_untied_mdef.pl
```

./scripts_pl/04.cd_schmm_untied/slave_convg.pl
./scripts_pl/05.buildtrees/make_questions.pl
./scripts_pl/05.buildtrees/slave.treebuilder.pl
./scripts_pl/06.prunetree/slave.state-tie-er.pl
./scripts_pl/07.cd-schmm/slave_convg.pl

Ne sont pas représentés ci-dessus les scripts 8 et 9 qui comme nous l'avons dit, ne sont utiles que pour Sphinx2.

Si l'on souhaite créer un modèle CI, il suffit de s'arrêter après l'exécution du script numéro 2. Par ailleurs, à chaque étape, il est tout à fait possible d'utiliser le modèle intermédiaire créé.

A la fin de l'exécution, il faut regarder dans le répertoire chiffres/model_parameters pour avoir le modèle acoustique.

Ou bien :

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

cd chiffres

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

Pour sphinxtrain 1.0.7 et avant

./scripts_pl/make_feats.pl -ctl etc/ chiffres_train.fileids

./scripts_pl/make_feats.pl -ctl etc/ chiffres_test.fileids

./scripts_pl/RunAll.pl

Pour sphinxtrain snapshot

sphinxtrain run

Pendant les étapes, l'étape la plus importante est la première qui vérifie que tout est configuré correctement et que votre entrée de données est conforme . Ne pas ignorer les erreurs signalées sur l'étape 00.verify_all premier.

IV.5 L'utilisation du modèle de SphinxTrain dans Sphinx4

Après l'apprentissage, le modèle acoustique est situé dans

model_parameters/<your_db_name>.cd_cont_<number_of senones>

Ou dans

model_parameters/<your_db_name>.cd_semi_<number_of senones>

Nous avons besoin seulement de ce dossier. Le modèle devrait avoir 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).

IV.5.1 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. Il faut Faire les

changements suivants dans le modèle et la configuration du dictionnaire, il suffit de pointer vers les fichiers:

```
    <component name="trigramModel"
type="edu.cmu.sphinx.linguist.language.ngram.large.LargeTrigramModel">
    <property name="unigramWeight" value="0.7"/>
    <property name="maxDepth" value="3"/>
    <property name="logMath" value="logMath"/>
    <property name="dictionary" value="dictionary"/>
    <Nom de la propriété = "emplacement"
            value = "le nom du fichier de modèle de langage
                par exemple <your_training_folder> / etc / <your_model_name>.
lm.DMP "/>
  </ Component>

  <component name="dictionary"
type="edu.cmu.sphinx.linguist.dictionary.FastDictionary">
    <Nom de la propriété = "dictionaryPath"
            value = "le nom du fichier de dictionnaire
                par exemple <your_training_folder> / etc / <your_model_name>. dic
"/>
    <Nom de la propriété = "fillerPath"
            value = "le nom du fichier de remplissage
                par exemple <your_training_folder> / etc / <your_model_name>. filler
"/>
    <property name="addSilEndingPronunciation" value="false"/>
    <property name="allowMissingWords" value="false"/>
    <property name="unitManager" value="unitManager"/>
  </ Component>
```

IV.5.2 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. Il y aura également des dossiers pour un nombre différent de gaussiennes comme _2 _4 _8, ils sont ceux intermédiaires et nous n'avons pas besoin d'eux.

Encore une fois, nous allons définir un modèle dans le fichier de configuration:

```
<Nom du composant = "sphinx3Loader"
            type = "edu.cmu.sphinx.linguist.acoustic.tiedstate.Sphinx3Loader">
        <property name="logMath" value="logMath"/>
        <property name="unitManager" value="unitManager"/>
  <Nom de la propriété = valeur de "localisation" = "le chemin vers le dossier de modèle
              par exemple <your_training_folder> / model_parameters / <your_model_name>. cd_cont_ <senones> "/>
 </ Component>

 <component name="acousticModel" type="edu.cmu.sphinx.linguist.acoustic.tiedstate.TiedStateAcousticModel">
   <property name="loader" value="sphinx3Loader"/>
   <property name="unitManager" value="unitManager"/>
 </ Component>
```

IV.5.3 Modèle en java (JAR)

En option, nous pouvons emballer les modèles dans un fichier JAR. L'avantage de les avoir dans un fichier JAR est que le fichier JAR peut simplement être inclus dans le chemin de classes (classpath) et référencée dans le fichier de configuration pour qu'il puisse être utilisé dans une application Sphinx4. Une fois que nous avons fait, il ne faut pas oublier d'inclure le JAR dans le "classpath". Pour configurer le chargement depuis les fichiers jars, Sphinx4 permet aux URls (Uniform Resource Locator, littéralement « localisateur uniforme de ressource ») de contient la ressource: < chemin d'acoustique ou langage modèle > ce qui permet aux fichiers de configuration XML de référencer facilement des modèles dans des fichiers JAR.

IV.6 Conclusion

La création du modèle acoustique est une étape primordiale pour la reconnaissance de la parole elle passe par plusieurs étapes, parmi ces étapes il y a l'étape d'apprentissage qui la plus importante. À la fin de cette dernière on obtient notre modèle acoustique.

Cependant on ne peut pas utiliser ce modèle directement pour Sphinx4 puisqu'il utilise des fichiers jar 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.

CONCLUSION GENERALE

Le travail que nous avons fait au long de cette étude, se base sur la conception d'un système de reconnaissance automatique de la parole arabe avec CMU Sphinx4 qui est basé sur les Modèles de Markov Cachés.

Tout en étant hautement configurable, la reconnaissance de la parole avec Sphinx 4 supporte notamment les mots isolés et les phrases (utilisation de grammaires). Son architecture est modulable pour permettre de nouvelles recherches et pour tester de nouveaux algorithmes. Ce dernier besoin de deux éléments essentiels : le modèle acoustique et le modèle du langage. Dans notre cas le modèle du langage est plus simple à réalisé que le modèle acoustique qui nécessite un outil complémentaire ou Shphinx4 pour le créer qui est le Sphinxtrain.

La création d'un nouveau modèle acoustique réclamant beaucoup de temps et de ressources. 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.

Une des tâches les plus longues et les plus fastidieuses concerne la collecte des données. En effet, nous devons fournir à SphinxTrain tout un ensemble de données sonores (corpus) sous un format spécifique et de données textuelles. Ces dernières étant les transcriptions des premières.

Il existe quelques corpus disponibles sur Internet mais qui sont assez coûteux (certains étant vendus quelques milliers d'euros). C'est pourquoi, il est dans certains cas préférable de réaliser sa propre base de données sonore. A l'heure actuelle, la constitution d'une base de données sonore avec transcription est difficilement réalisable automatiquement. La meilleure solution est alors d'en créer une à la main. Deux contraintes s'ajoutent alors :

- Pour que la reconnaissance soit indépendante du locuteur, il faut faire des enregistrements avec plusieurs personnes ;
- La qualité du modèle acoustique en fonction des données vocales fournies pour l'apprentissage.

La qualité de la reconnaissance dépend directement de la qualité des données vocales. Ces dernières étant les informations relatives à une voix propre. Ce sont par exemple les différents phonèmes, les différents mots (lexique), les différentes façons de prononciation. Plus ces informations seront importantes et connues par le système, meilleure sera sa réaction et ses choix à faire.

Afin d'augmenter le taux de reconnaissance, il faut augmenter La qualité du modèle acoustique en réalisant des enregistrements de plusieurs heurs avec différentes personnes et améliorer la qualité de ces enregistrements.

Pour conclure, un système de reconnaissance automatique de la parole a été conçu et adapté pour la reconnaissance des dix premiers chiffres de langue arabe de 0 à 9 pour démonter l'adaptabilité de CMU Sphinx4 pour la langue arabe. Nous considérons cependant, qu'il s'agit d'un travail préliminaire qui nous permettra par la suite l'accomplissement de l'objectif de base ; un système de reconnaissance indépendant de locuteur pour la langue arabe. En perspective, nous pouvons étendre l'application pour un large vocabulaire de la langue arabe.

BIBLIOGRAPHIE

[1] J.M. Pierrel, "Dialogue oral homme-machine ", Edition Hermes, Paris, 1987.

[2] L. R. Rabiner et B. H. Juang, "An introduction to hidden Markov models", IEEE Transactions on Acoustics, Speech, and Signal Processing, volume 3, pages 4–16, Janvier 1986.

[3] J. P. Haton, J. M. Pierrel, C. G. Perennou et J. L. Gauvain, "Reconnaissance automatique de la parole", Bordas, Paris, 1991.

[4] A. Mokeddem, "Reconnaissance multilocuteur de mots isolés pour les systèmes miniaturisés", thèse de doctorat, Université de Neuchâtel institut de microtechnique, 1985.

[5] J. P. Haton, "Reconnaissance automatique de la parole", 8-ième école d'été d'informatique de l'AFCET, Namur 1975.

[6] C. H. Lee et W. H. Juang, "A survey on automatic speech recognition with an illustration example on continues speech recognition of Mandarin, Computational linguistics and chinese language Processing", Vol. 1, pp. 1-36, August 1996.

[7] J. ALLEGRE, "Approche de la reconnaissance automatique de la parole "Rapport cycle probatoire, CNAM (Conservatoire National des Arts et Métiers), Année 2003.

[8] C. Barras, "Reconnassance de la parole continue : adaptation au locuteur et contrôle temporel dans les modèles de markov cachés ", Thèse doctorat, Université Paris VI, 1996.

[9] http://fr.wikipedia.org/wiki/Reconnaissance_vocale.

[10] S. Young, "The HTK hidden Markov model toolkit : Design and philosophy", Université Cambridge Département d'ingénieur, Sept. 1994.

[11] N. Deshmukh, A. Ganapathiraju, J. Hamaker, J. Picone, and M. Ordowski, "A public domain speech-to-text system", 6 éme Conférence européenne, communication et technologie de la parole, vol. 5, pp. 2127–2130, Budapest, Hungary, Sept. 1999.

[12] X.X. Li, Y. Zhao, X. Pi, L.H. Liang, et A.V. Nefian "Audio-visual continuous speech recognition using a coupled hidden Markov model", 7 éme International Conférence. Spoken Language Processing, Denver, CO, Sept. 2002.

[13] K.F. Lee, H.W. Hon, et R. Reddy, "An overview of the SPHINX speech recognition system ", IEEE Transactions on Acoustics, Speech and Signal Processing, vol. 38, no.1, pp. 35–45, Jan. 1990.

[14] X. Huang, F. Alleva, H. W. Hon, M.Y. Hwang, et R. Rosenfeld, " The SPHINX-II speech recognition system : an overview ", Computer Speech and Language, vol. 7, no. 2, pp. 137–148, 1993.

[15] M.K. Ravishankar, " Efficient algorithms for speech recognition ", Carnegie Mellon University, Pittsburgh, 1996.

[16] X.D. Huang, Y. Ariki et M.A. Jack, " Hidden Markov models for speech recognition ", Université d'Edimbourg, 1990.

[17] CALLIOPE CNET-ENST eds. "La parole et son traitement automatique". Masson, Sept. 1989.

[18] C. Gagnoulct, D. Jouvet, " Développements récents en reconnaissance de la parole ", L'écho des recherches, CNET-ENST, N° 135, pp. 27-36, 1989.

[19] S. Levinson, L. Rabiner, M. Sondhi "Speaker-Independant Isolated Digit Recognition Using Hidden Markov Models", ICASSP, pp 1049-1052, Boston, 1993.

[20] K.F Lee, H.W. Hon, R. Reddy, "An Overview of the SPHINX Speech Recognition System", IEEE Trans on ASSP, 38 N° 1, pp. 35-45, Jan. 1990.

[21] J.M. Pierrel, "Utilisation des contraintes linguistiques en compréhension de parole continue le système Myrtille II ". TSI, Vol 1, N° 5, 1982, pp. 403-421.

[22] G. Pérennou, " The ARIAL II Speech Recognition System In : J.P Haton ed (Automatic Speech Analysis and Recognition)", pp. 269-275, 1982.

[23] A. Averbuch et al. " Experiments with the TANGORA 20,000 word speech recognizer ", ICASSP, pp. 701-704, Dallas, 1987.

[24] P. Alto, M. Brandetti, M. Ferretti, G. Maltese et S. Scarci, " Experimenting Natural-Language Dictation with 20,000- Word Speech Recognizer ", Proceedings of IEEE, CompEuro, Section 2 pp. 78-91, Hambourg, 1989.

[25] P. D'Orta, M. Ferretti, A. Martelli, S. Melecrinis, S. Scarci et G. Volpi "A Speech Recognition System for the Italian Language ", ICASSP, pp 941-943, Dallas, 1987.

[26] H. Cerf-Danon, P. de La Noue, L. Diringer, M. El-Bèze et JC. Marcadet "A 20,000 words, automatic speech recognizer. Adaptation to French of the US TANGORA system", Nato, 1990.
[27] K. Wothke, U. Bandara, J. Kempf, E. Keppel, K. Mohr et G. Walch "SPRING Speech Recognition System for German", Eurospeech, Vol 2, pp. 9-12, Paris, Sept. 1989.
[28] J. Baker "DRAGON DICTATE (TM) -30K Natural Language Speech Recognition statistical methods". Eurospecch 89 Vol 2, pp. 161-163, Septembre, 1989.
[29] F. Fanchettc "Speech recognition .- Talk about progress ". Language Technology, N° 19, p. 4-5.
[30] J.L. Le Breton "Le premier traitement de texte vocal". L'ordinateur individuel N° 98, p. 108-110, décembre 1997.

[31] J.P. Tubach, C. Gagnoulet, J.L. Gauvain "Advances in speech recognition products from France", Conférence Speech Tech, 1989.
[32] J. Mariani "Hamlet un prototype de machine à écrire à entrée vocale", Journal d'acoustique, pp. 79-83, Paris, 2 mars 1989.
[33] L.S. Lee, C.Y. Tseng, H.Y. Gu, F.H. Liu, C.H. Chang, S.H. Hsich et C.H. Chen "A Real- Time Mandarin Dictation Machine for Chinese Language with Unlimited Texts and Very Large Vocabulary". ICASSP, pp 65-68, Albuquerque, 1990.
[34] L. Bahl, R. Bakis, J. Bellagarda, P. Brown, D. Burshtein, S. Das, P. de Souza, P. Gopalakrishnan, F. Jelinck, D. Kanevsky, R. Mercer, A. Nadas, D. Nahamoo et M. Picheny " Large Vocabulary Natural Language Continuous Speech Recognition". ICASSP Vol SI, pp 465-467, Glasgow, 1989.

[35] B. Ulucinar "Web WriteIt!", rapport de master Département d'Informatique, Université de Fribourg, avril 2007.
[36] B. Lecouteux "Reconnaissance automatique de la parole guidée par des transcriptions a priori",thèse de Doctorat, l'Université d'Avignon et des Pays de Vaucluse, 5 décembre 2008.

[37] CEOWORLD Magazine Graduate Business School Rankings for Executives and Entrepreneurs,http://ceoworld.biz/ceo/2010/01/25/top-best-voice-or-speech-recognition-software

[38]http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.16.5153&rep=rep1&type=pdf

[39] www.math.univ-toulouse.fr/~besse/Wikistat/pdf/st-m-app-select.pdf

[40] C. Aouad, "Rapport de stage Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier (LIRMM) ", Université Montpellier II, Année 2005 – 2006.

[41] P. Lamere, P. Kwok, E.B. Gouvêa, B. Raj, R. Singh, W. Walker et P. Wolf " The CMU Sphinx-4 Speech Recognition System ", USA.

[42] A.M. Alimi et M. Ben Jemaa, "Beta Fuzzy Neural Network Application in Récognition of Spoken Isolated Arabie Words", International Journal of Control and Intelligent Systems, Spécial Issue on Speech Processing Techniques and Applications, Vol. 30, No.2, 2002

[43] M. Alghamdi, M. Elshafei et H. Al-Muhtaseb, "Arabie broadeast news transcription System", University of Petroleum and Minerais, article publié sur le journal Springerlink, 1 April 2009.

[44] P. Price, W.M. Fisher, J. Bernstein et D.S. Pallett "The DARPA 1000-word resource management database for continuons speech récognition", IEEE, in Proceedings of the International Conférence on Acoustics, Speech and Signal Processing, vol. 1, pp. 651-654, 1988.

[45] H. Tabbal, W. Al-Falou, B. Monla "Analysis and Implementation of an Automated Délimiter ofQuranic Verses in Audio Files using Speech Récognition Techniques", Lebanese University, article publié sur "Robust Speech Récognition and Understanding, Book edited by: Michael Grimm and Kristian Kroschel", pp.460,1-Tech, Vienna, Austria, June 2007.

[46] X. Huang, A. Acero et H. Hon, "Spoken language processing a guide to theory, algorithm and System design", Prentice Hall, 2001.

[47] J. Mariani "Reconnaissance automatique de la parole : progrès et tendances", LIMSI-CNRS, volume 7 - ri 4, pp. 239-261, 1990.

[48] A. Chan, E. Gouvea, R. Singh, M. Ravishankar, R. Rosenfeld, Y. Sun, D. Huggins-Daines et M. Seltzer, "Building Speech Applications UsingCMU Sphinx and Related Resources", 2007.

[49] http://cmusphinx.sourceforge.net

[50] A. Chan, E. Gouvea, Y. Sun, D. Huggins-Daines et J. Sherwani, document disponible sur le lien:www.cs.cmu.edu/~archan/presentation/SphinxDev20040624.ppt, 2007.

[51] G.Z. Hong "Speech Recognition Techniques for Digital Video Library" University of Hong Kong, 2002.

[52] Carnegie Mellon University. Sphinx-4. Available: http://cmusphinx.sourceforge.net.

[53] X.D. Huang "The SPHINX-II Speech Recognition System: An Overview Computer Speech and Language", Vol. 2, 1993.

[54] J. D. MARKEL et A. H. Gray " Linear prediction of speech In communication and Cybernetics", Berlin Heidelberg New York, 1976.

[55] H. HERMANSKY et L.A. COX "Perceptual linear predictive (plp) analysis resynthesis technique ". IEEE, ASSP Workshop on Applications of Signal Processing to Audio and Acoustics Final Program and Paper Summaries, pages 037–038, 1991.

[56] http://cmusphinx.sourceforge.net/sphinx4.

[57] A. Varela, H. Cuayáhuitl et J.A. Nolazco-Flores "Creating a Mexican Spanish Version of the CMU Sphinx-III Speech Recognition System", Springer, Vol. 2905, 2003.

[58] Sun Microsystems. Available: http://java.sun.com.

[59] http ://ant.apache.org.

[60] http ://www.activestate.com.

[61] M. Al-Zabibi "An Acoustic–Phonetic Approach in Automatic Arabic Speech Recognition", La British Library en association avec UMI, 1990.

[62] A. Muhammad "Alaswaat Alaghawaiyah", Daar Alfalah, Jordan, 1990.

[63] J. Deller, J. Proakis et J.H. Hansen "Discrete-Time Processing of Speech Signal", Macmillan, New York, 1993.

[64] M. Elshafei "Toward an arabic text-to-speech system", The Arabian Science and Engineering vol. 4B no. 16, pp. 565–583, 1991.

[65] Y.A. El-Imam, "An unrestricted vocabulary arabic speech synthesis system", IEEE, Transactions on Acoustic, Speech, and Signal Processing vol. 37, no. 12, pp.1829–1845, 1989.

[66] Y. Ajami Alotaibi "Investigating spoken Arabic digits in speech recognition setting", Inf. and Comp. Sc. pp.173, 115, 2005.

[67] Baloul S. "Développement d'un système automatique de synthèse de la parole à partir du texte arabe standard voyellé", Thèse de Doctorat, Université de Maine, Le Mans, 2003.

[68] http://www.speech.kth.se/wavesurfer

[69] A. Wählby et C. Cassia "An oaa agent for sphinx-4", University of Gothenburg, Sweden, 2004.

[70] http://cmusphinx.sourceforge.net/wiki/mfcformat

ملخص

التعرف التلقائي على الصوت هو أحد تكنولوجيا الإعلام الآلي الذي يسمح بتفسير لغة الإنسان الطبيعية بواسطة برنامج خاص. في هذا الموضوع نهتم بلستخدام نظام التعرف على الصوت (CMU Sphinx4). هذا الأخير هو مشروع مفتوح المصدر خاص بجامعة كارنيجي ميلون ويستند على نماذج ماركوف الخفية (MMC) ، وسيتم تطبيق هذا النظام للتعرف على الأرقام العشرة الأولى من العربية الفصحى إعتمادا على عينات صوتية لعشرات من الأشخاص من كلا الجنسين .

كلمات البحث :التعرف التلقائي على الصوت ، نظام التعرف على الصوت (CMU Sphinx4)، نماذج ماركوف الخفية، العربية الفصحى.

Résumé

La reconnaissance automatique de la parole (RAP) est une technologie informatique permettant à un logiciel d'interpréter une langue naturelle humaine. Dans ce sujet nous nous intéressons à utiliser le système de reconnaissance de la parole CMU Sphinx4. Ce dernier est un projet Open Source de l'Université Carnegie Mellon et qui est basé sur les modèles de Marcov Cachés (MMC). Ce système sera appliqué à la reconnaissance des dix premiers chiffres de l'arabe classique prononcés par une dizaine de locuteurs des deux sexes.

Mots clés : Reconnaissance automatique de la parole, RAP, CMU Sphinx4, MMC, l'arabe classique.

Abstract

The automatic speech recognition (ASR) is a computer technology that allows software to interpret a natural human language. In our subject we are interested to using the system of CMU Sphinx4 for speech recognition. This is an Open Source project at Carnegie Mellon University and is based on Hiden Markov Models (HMMs), the system will be applied to the recognition of the first ten digits of the classical Arabic spoken by ten speakers of both sexes.

Keywords : Automatic speech recognition, ASR, CMU Sphinx4, HMMs, classical Arabic.