ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

MÉMOIRE PRÉSENTÉ À
L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

COMME EXIGENCE PARTIELLE
À L'OBTENTION DE LA
MAÎTRISE EN TECHNOLOGIES DE L'INFORMATION
M.Ing.

PAR
Chahid OUALI

INDEXATION D'ANNOTATIONS VOCALES DANS UN CONTEXTE DE GESTION DOCUMENTAIRE

MONTRÉAL, LE 08 JUILLET 2010

©Tous droits réservés, Chahid Ouali, 2010

PRÉSENTATION DU JURY

MÉMOIRE A ÉTÉ EVALUÉ

PAR UN JURY COMPOSÉ DE :

M. Pierre Dumouchel, directeur de mémoire
Département de génie logiciel et des technologies de l'information à l'École de technologie supérieure

Mme Sylvie Ratté, président du jury
Département de génie logiciel et des technologies de l'information à l'École de technologie supérieure

M. Martin Choquette, programmeur sénior
Irosoft

IL A FAIT L'OBJET D'UNE SOUTENANCE DEVANT JURY ET PUBLIC

LE 23 JUIN 2010

A L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

REMERCIEMENTS

Une tâche difficile m'incombe ici afin de remercier en seulement quelques lignes, et surtout comme il se devrait, toutes les personnes qui ont, de près ou de loin, contribué à la réalisation de ce travail.

Je tiens à remercier en premier lieu mon directeur de recherche M. Pierre Dumouchel, pour m'avoir témoigné une confiance indéfectible au cours de ce mémoire. Je le remercie pour sa disponibilité, sa simplicité, sa grande sagesse et le souci intéressant pour le bien et la satisfaction de ses étudiants. Je le remercie également pour la qualité distinguée de son encadrement, pour ses idées innovatrices, pour son aide et ses précieux conseils ce qui m'a aidé à l'accomplissement de ce travail.

Je tiens aussi à adresser mes plus sincères remerciements envers M. Martin Sévigny pour son suivi de mon travail, la clarté de ses idées et la patience dont il a fait à mon égard et sa disponibilité continue. Qu'il trouve ici l'expression de mon profond respect.

Je tiens également à remercier et à témoigner toute ma reconnaissance au personnel de l'entreprise Irosoft pour l'expérience enrichissante et pleine d'intérêt qu'ils m'ont fait vivre.

Enfin, je voudrais exprimer mes plus profonds remerciements à ma famille, et particulièrement mes chers parents, pour l'amour et l'affection dont ils m'ont toujours entouré, leurs encouragements et leurs sacrifices illimités.

INDEXATION D'ANNOTATIONS VOCALES DANS UN CONTEXTE DE GESTION DOCUMENTAIRE

Chahid OUALI

RÉSUMÉ

L'indexation des documents multimédia suscite actuellement un grand intérêt tant sur le plan expérimental que théorique. En particulier, la détection de mots clés dans des fichiers sonores est un secteur en pleine croissance. Cependant, malgré les progrès réalisés dans le domaine de l'indexation vocale, il reste beaucoup à faire notamment pour la recherche de mots clés dans la parole spontanée.

Le travail qu'on présente dans ce manuscrit s'inscrit dans le cadre de l'indexation d'annotations vocales dans un contexte de gestion documentaire. Tout d'abord, on présentera quelques systèmes de reconnaissance automatique de la parole. En se basant sur des critères précis, on a identifié deux moteurs de reconnaissance automatique de la parole qui ont fait l'objet de nos expérimentations.

Ensuite, on proposera un système de détection de mots clés dans les annotations vocales. Ce dernier sera basé sur les deux moteurs de reconnaissance automatique de la parole qu'on a choisi, à savoir le moteur de Dragon NaturallySpeaking et celui de Microsoft.

Pour tester les performances des deux systèmes, on a construit un corpus d'annotations vocales. L'évaluation des performances de transcription a été réalisée en se basant sur le taux de mot correct et le taux de précision. D'autre part, l'évaluation des performances d'indexation a été réalisée en se basant sur les courbes ROC et les taux de rappel et de précision.

Les meilleurs résultats ont été observés avec le moteur de reconnaissance de Microsoft pour le profil sans apprentissage. Alors que pour le profil entraîné, le moteur de Dragon présente les meilleures performances. Afin d'améliorer les performances, on propose d'entraîner le modèle de langage avec un grand corpus de texte d'annotations écrites.

Mots-clés: reconnaissance automatique de la parole, indexation vocale, détection de mots clés, annotation vocale, parole spontanée.

VOICE ANNOTATIONS INDEXATION IN A CONTEXT OF DOCUMENTARY MANAGEMENT

Chahid OUALI

ABSTRACT

The induction of multi-media documents arouses currently a great interest on the experimental as well as theoretical level. Particularly, the detection of key words in sound files is a sector in full progress. However, despite the progress realized in the field of voice indexation, much to be done remains and in particular for the search of key words in the spontaneous speech.

Our work presented in this manuscript registers within the framework of the voice annotations indexation in a context of documentary management. First of all we will present some automatic speech recognition systems. Based on selection criterion, we have identified two speech recognition engines. They are the subject of our experiments.

Then, we will propose a keyword detection system in the voice annotations. This latter will be based on the two automatic speed recognition engines which we have chosen; namely the engines of Dragon Naturally Speaking and the one of Microsoft. In order to test the performance of the two systems, we have built a corpus of voice annotations. The evaluation of the transcription performances was realized through being based upon the percentage of correct words and that of precision. On the other hand, the evaluation of the indexation performances was realized through being based on ROC curves and the recall and precision rates.

The best results were observed with the Microsoft recognition engine for the profile without training. While for the trained profile, the Dragon engine presents the best performances. So in order to improve the performances, we propose to involve the model language with a great corpus of written annotations text.

Keywords: speech recognition, voice indexation, keyword detection, voice annotations, spontaneous speech.

TABLE DES MATIÈRES

Page

LISTE DES TABLEAUX

Page

LISTE DES FIGURES

Page

LISTE DES ABRÉVIATIONS, SIGLES ET ACRONYMES

BSD	Berkeley Software Distribution
DAP	Décodage Acoustico-Phonétique
DTW	Dynamic Time Warping
MDI	Multiple Document Interface
FOM	Figure-Of-Merit
HMM	Hidden Markov Models
HTK	Hidden markov model toolkit
IVR	Interactive Voice Response
JSAPI	Java Speech Application Programming Interface
JSGF	Java Speech Grammar Format
LPS	Lecture de la Parole Spontanée
LVR	Large Vocabulary Recognition
MFCC	Mel-Frequency Cepstral Coefficients
OOV	Out-Of-Vocabulary
PA	Percent Accuracy
PLS	Phone Lattice Scanning
PS	Parole Spontanée
PSPL	Position Specific Posterio Lattices
RAP	Reconnaissance Automatique de la Parole
ROC	ROC Receiver Operating Characteristics
SAPI	Speech Application Programming Interface
SDK	Software Developer kit

SRGS	Speech Recognition Grammar Specification
SVM	Support Vector Machine
TE	Taux d'Erreur
TMC	Taux de Mots clés Correct
TP	Taux de Précision
WCR	Word Correct Rate
WER	Word Error Rate
WS	Word Spotting

INTRODUCTION

La communication humaine se base sur divers moyens de communication comme les gestes, l'écriture, les images et la voix. Toutefois, la parole reste la façon la plus efficace pour exprimer les désirs et transmettre la pensée. Ainsi, il n'est pas surprenant qu'on cherche depuis des années à interagir avec les machines par la voix. En effet, les activités dans le domaine de la reconnaissance automatique de la parole ont débuté depuis les années 1950 avec l'apparition du premier système de reconnaissance des chiffres. L'idée de communication entre l'homme et la machine a été introduite au grand public par des œuvres de science-fiction et des films comme « *2001 : l'odyssée de l'espace* » lancé en 1968. Toutefois, les premiers systèmes commerciaux ont vu le jour dans les années 1970, notamment avec les travaux de DARPA sur la reconnaissance automatique de la parole continue. Depuis, les recherches ne cessent de progresser pour produire des systèmes capables de décoder la parole.

Les progrès réalisés dans le domaine de la reconnaissance automatique de la parole ont permis l'apparition de nouveaux domaines connexes comme l'indexation vocale. En effet, la reconnaissance automatique de la parole est une technologie-clé pour la recherche de mots clés dans les documents audio. Le besoin d'accéder par le contenu aux documents sonores a augmenté avec l'explosion des nouvelles technologies de l'information qui ont permis la production d'une grande masse d'information. Ceci a donné lieu à plusieurs projets et réalisations comme l'indexation et la classification des documents parlée, l'extraction de l'information, la surveillance des appels téléphoniques, etc.

Actuellement, l'indexation vocale touche différents domaines et commence à prendre une place dans des systèmes de gestion telle que les progiciels de gestion documentaire. En effet, l'entreprise Irosoft a développé le progiciel Docuthèque qui permet, entre autres, d'ajouter des annotations repositionnables à tout type de document. Cette fonctionnalité peut être rénovée en remplaçant l'annotation textuelle par l'annotation vocale. Ainsi, on étudiera la possibilité de réaliser la recherche de mots clés dans ces annotations en se basant sur les

techniques de l'indexation vocale. En effet, l'indexation peut se réaliser en se basant sur le résultat de la transcription automatique des annotations en utilisant un moteur de reconnaissance automatique de la parole continue.

L'indexation vocale pose un réel défi lorsque l'on traite la parole spontanée qui se caractérise par les disfluences[1] et agrammaticalité. En effet par rapport à la dictée vocale, les annotations vocales risquent d'être plus spontanées, moins formelles, plus courtes, ou peut-être de style télégraphique, selon le locuteur.

Dans ce cadre, ce mémoire s'intéresse à l'indexation des annotations vocales dans un contexte de gestion documentaire. Notre objectif est de concevoir et de développer un prototype de système d'indexation d'annotations vocales. Ce dernier sera basé sur un moteur de reconnaissance automatique de la parole continue. Tout d'abord, on a mené une étude sur différents systèmes de reconnaissance vocale présents sur le marché. Cette étude a permis, en se basant sur quelques critères de sélection, d'identifier deux moteurs de reconnaissance automatique de la parole. Pour l'évaluation des performances de transcription et d'indexation, on a construit un corpus d'annotations vocales. Ainsi, la performance de transcription sera mesurée par le taux de mot correct et le taux de précision. D'autre part, l'évaluation des performances d'indexation a été réalisée en se basant sur les courbes ROC et les taux de rappel et de précision.

Ce mémoire comporte quatre chapitres. Tout d'abord, le premier chapitre exposera notre problématique, les objectifs à atteindre ainsi que les défis. Le deuxième volet de ce chapitre concernera les divers problèmes de la reconnaissance vocale. Cette partie sera suivie par la description du processus de reconnaissance automatique de la parole. Une revue sur les différentes méthodes d'indexation vocale fera l'objet de la dernière partie de ce chapitre.

---

[1] Morphèmes spécifiques comme les répétitions les faux départs, mauvaises élocutions, etc.

Le deuxième chapitre couvrira en grande partie la méthodologie et les outils utilisés pour la réalisation de notre étude. Dans un premier lieu, on présentera différents systèmes de reconnaissance automatique de la parole ainsi que les critères de sélection utilisés afin d'identifier les deux moteurs qui ont fait l'objet de nos expérimentations. Dans un deuxième lieu, on présentera les différentes métriques utilisées pour l'évaluation des performances de transcription et d'indexation. Dans la troisième partie de ce chapitre, on détaillera le processus expérimental ainsi que le corpus d'annotation vocale qu'on a utilisé dans notre étude.

Au chapitre 3, on s'intéressera aux différents éléments qu'on a utilisés pour la mise en place d'un système d'indexation d'annotations vocales. De même, on présentera les différentes fonctionnalités et outils fournis par ce système.

Quant au dernier chapitre, il sera entièrement consacré à la présentation et l'interprétation des résultats de nos expériences. En effet, les performances de transcription et d'indexation seront analysées selon différents axes d'analyse. Ceci permettra d'étudier, entre autres, les différentes sources d'erreurs et d'établir la relation entre le taux d'erreur et la performance d'indexation. On s'intéressera dans la dernière partie de ce chapitre à l'amélioration des performances de transcription par l'entraînement du modèle de langage.

CHAPITRE 1

REVUE DE LA LITTÉRATURE

Ce chapitre est une introduction au domaine de l'indexation vocale comme étant une composante de la reconnaissance automatique de la parole. Dans un premier temps, le sujet de notre étude est abordé : indexation d'annotations vocales dans un contexte de gestion documentaire. Plus particulièrement, on présentera la problématique, les objectifs à atteindre ainsi que les défis. On exposera dans la deuxième partie les divers problèmes de la reconnaissance automatique de la parole. La méthode d'indexation qu'on a utilisée se base sur la transcription fournie par un système de reconnaissance automatique de la parole. Ainsi, les problèmes rencontrés par un tel système forment un grand défi pour l'indexation vocale, notamment pour la recherche de mots clés dans des annotations de la parole spontanée. Ensuite, on présentera le processus de la reconnaissance automatique de la parole. Finalement, on décrira les principales méthodes publiées et utilisées dans le domaine de l'indexation vocale avec les références appropriées.

1.1 Cadre général

1.1.1 Problématique

L'entreprise Irosoft a développé et installé chez de nombreux clients le progiciel Docuthèque, une application de gestion documentaire dédiée à la gestion des documents et des archives électroniques et papier, orientée moyennes et grandes organisations publiques et privées. Parmi ses fonctionnalités de gestion, Docuthèque offre des annotations **écrites** repositionnables qui peuvent être ajoutées par l'usager aux textes, dossiers, pièces justificatives, ou tout autre document. Dans le cadre de ce projet, de recherche et de développement, on vise à rendre Docuthèque plus convivial en intégrant de nouvelles fonctionnalités issues des technologies de la langue.

Le travail qu'on présente dans ce manuscrit s'inscrit dans le cadre de l'indexation d'annotations **vocales** dans un contexte de gestion documentaire. Étant donné que les moteurs de recherche fonctionnent qu'avec du texte, notre travail porte sur l'intérêt et la faisabilité de la détection de mots clés dans des annotations vocales par l'utilisation de la reconnaissance vocale. La reconnaissance vocale est utilisée dans un contexte original et distinct des applications de dictée vocale, d'IVR[2] ou d'indexation de documents audio par les aspects suivants : le style du contenu, l'indexation en temps réel, le traitement des requêtes en temps constant et l'adaptation non-supervisée au contenu.

Bien que l'indexation puisse s'accommoder d'un taux de reconnaissance nettement inférieur à ce qui serait nécessaire pour de la transcription en texte, il sera difficile, dans le cadre de ce projet, d'atteindre un taux de reconnaissance suffisant puisqu'on traite la parole spontanée. En effet, la reconnaissance vocale fonctionne relativement bien lorsque le domaine est restreint et qu'un corpus de textes pertinents de grande taille est disponible pour l'entraînement du modèle de langage. De ce point de vue, le projet présente les défis techniques suivants :

- L'absence de corpus : Il s'agit d'une contrainte nouvelle, qui permettra aux usagers d'ajouter des annotations vocales à des documents en utilisant simplement la parole naturelle. Ainsi, on ne dispose d'aucun corpus préalable permettant d'entraîner des modèles de reconnaissance performants ni d'estimer la performance de reconnaissance à laquelle on pourrait s'attendre.
- Style d'annotation : Par rapport à la dictée vocale, les annotations vocales risquent d'être plus spontanées, moins formelles, plus courtes, ou de style télégraphique, selon le locuteur. On n'obtiendra pas un modèle de langage aussi prédictif que dans une application de dictée vocale. Même si on disposait de suffisamment d'annotations écrites

[2] Serveur vocal interactif (Interactive Voice Response) : système de dialogue entre un utilisateur et un serveur téléphonique.

pour estimer un modèle de langage, elles auraient aussi un style trop différent des annotations vocales pour servir de corpus d'entraînement.

- La simplicité d'utilisation : Elle serait compromise si on exigeait de l'usager qu'il entraîne le système ou corrige les annotations. Du point de vue acoustique, il doit s'agir soit d'un système indépendant du locuteur, sans adaptation, ou d'un système avec adaptation non-supervisée et transparente pour l'utilisateur.
- Couverture du vocabulaire : Pour assurer une bonne performance lors de la recherche d'information, il est crucial d'avoir une bonne couverture du vocabulaire spécialisé, particulier au contexte d'utilisation, et qui évolue constamment.

1.1.2 Objectifs à atteindre

Notre objectif principal est de concevoir et développer un prototype de système d'indexation vocale basé sur un moteur de reconnaissance automatique de la parole. Le système permettra de détecter des mots clés prononcés dans des séquences sonores. Ainsi, le système comprendra un outil de transcription des annotations vocales en temps différés. Toutefois, la recherche peut se réaliser également sur le résultat de la reconnaissance vocale en temps réel.

En outre, le prototype permettra l'évaluation de la performance de transcription des annotations vocales et celle de l'indexation en se basant sur des métriques bien connues. Ainsi, on se basera sur le taux de mots correct et le taux de précision pour l'évaluation de la performance de transcription. D'autre part, on utilisera les courbe ROC et les taux de rappel et de précision pour l'évaluation de la performance d'indexation. En plus, le système permettra de présenter les résultats de l'évaluation afin de les analyser. En outre, le moteur de reconnaissance automatique de la parole doit être choisi de sorte que son intégration au progiciel Docuthèque soit possible.

1.2 Le problème de la reconnaissance vocale

La reconnaissance vocale est une technique qui permet d'analyser un signal acoustique, capté au moyen d'un microphone, afin de l'exploiter par une machine. Ceci permet notamment de réaliser des interfaces homme-machine garantissant une communication vocale entre un utilisateur et une machine. Cette interaction facilite davantage l'utilisation des machines comme les ordinateurs, les PDAs, et les téléphones intelligents qui sont devenus indispensable dans la vie quotidienne. En effet, la communication vocale avec ces machines est très utile lorsque les mains ou les yeux sont occupés, ou lorsque la personne utilisant la machine souffre d'un handicap. En outre, la parole est le mode de communication naturel entre les humains, et c'est évident qu'il reste le moyen préféré d'interaction avec les machines. Les avantages liés à l'utilisation de la reconnaissance vocale ont incité l'homme à faire des recherches plus approfondies. Les travaux réalisés pendant des siècles ont donné naissance à plusieurs applications pratiques, souvent spectaculaires, bien qu'encore très limités. En réalité, la reconnaissance automatique de la parole est une tâche très compliquée qui soulève plusieurs difficultés. Ces derniers peuvent être liées aux niveaux linguistiques, variabilité intra-locuteur et interlocuteur, matériel, environnement et phénomène de confusion.

1.2.1 Niveaux linguistiques

Bien qu'elle semble naturelle et aisée, la compréhension de la parole nécessite diverses sources d'information. Ces dernières forment le « lexique mental »[3] qui est une sorte de dictionnaire mental contenant toutes les connaissances qu'un individu possède sur les mots d'une langue (Grataloup 2007). Chaque source d'information est une discipline de la linguistique et forme un niveau d'analyse. La combinaison entre celles-ci permet d'attribuer une signification au message verbal. On peut distinguer le niveau phonétique, phonologique, lexical, syntaxique, sémantique et pragmatique. Le décodage acoustico-phonétique (DAP)

[3] Concept introduit par Anne Treisman (Treisman 1960).

assure la transformation du signal acoustique continu en une sorte d'unités telles que phonèmes, syllabes, etc. Cette opération est en général décrite comme une succession d'étapes depuis le niveau acoustique jusqu'au niveau sémantique (Lea 1980; Haton, Bonneau et al. 1990). Pour une bonne reconnaissance vocale, on doit prendre en compte les différents niveaux linguistiques. Cependant, il existe des architectures qui ne permettent pas d'introduire de façon explicite ces connaissances. En effet, ces niveaux présentent beaucoup d'informations qui sont parfois difficiles à modéliser. Pour mieux comprendre ces derniers, on va les étudier plus en détails.

1.2.1.1 Niveau phonétique

Un trait phonétique est souvent une caractéristique phonétique d'un phonème. Toutefois, ce terme peut s'appliquer à tout son de quelque taille qu'il soit. En effet, la phonétique analyse l'articulation, la transmission, et la perception des sons dans la communication verbale. On parle de phonétique acoustique, articulatoire et auditive :

- **Phonétique acoustique** : étudie les propriétés physiques de l'onde au cours de sa transmission de son émetteur à son récepteur.
- **Phonétique articulatoire** : étudie l'émission des sons, selon l'emplacement dans l'appareil phonatoire de l'émetteur et selon l'obstacle rencontré.
- **Phonétique auditive** : étudie la perception des sons et la façon dont ils sont décodés par le récepteur.

Il existe plusieurs types de traits phonétiques pour décrire les consonnes et les voyelles notamment le point d'articulation, le mode d'articulation, la nasalité, etc. Pour plus de détails, on peut consulter les articles suivants (Lausanne 2010; Wikipédia 2010).

1.2.1.2 Niveau phonologique

La phonologie étudie le rôle de chaque son, d'une langue donnée, du point de vue de leur contribution au sens. En d'autres termes, elle étudie l'organisation des sons d'une langue afin de former un énoncé. Contrairement à la phonétique, la phonologie nécessite la compréhension de la langue concernée pour l'étudier. Ceci permet notamment d'identifier les différences de prononciation qui correspondent à des différences de sens (oppositions distinctives).

1.2.1.3 Niveau lexical

La lexicologie étudie la nature et l'étymologie des mots ainsi que les relations sémantiques entre elles. Le lexique d'une langue constitue l'ensemble de ses mots, ou plus couramment le vocabulaire. En traitement automatique de la parole, les questions lexicales n'ont pas été abordées avant la réalisation du projet ARPA (Reddy 1976; Klatt 1977). La mise en place d'un véritable lexique n'était pas nécessaire avec des systèmes orientés vers de petits vocabulaires. L'information lexicale permet notamment d'identifier les homophones[4] qui désignent le rapport entre deux mots différents au niveau de la graphie mais possédant la même prononciation. Pour résoudre ce phénomène, les moteurs de reconnaissance de la parole, se basent sur le contexte de la phrase. La reconnaissance sera donc plus pertinente avec de longues phrases qu'avec de petits groupes de mots. Voici quelques exemples d'homophone :

- sa/ça/çà;
- différent/différend/diffèrent;
- prie/prit/pris/pries/prix;
- quand/quant/qu'en/camp/khan.

---

[4] Voir aussi l'homographie qui désigne le rapport entre deux mots différents possédant la même orthographe.

Il existe des cas plus complexes, par exemple, l'équipe de LIMSI (Gauvain, Lamel et al. 1994) a dénombré 32000 homophones possible pour la phrase « j'ai mal aux pieds », dont :

- Geai mâle au pied;
- Geais ma lot pieds;
- J'aime allo pillé;
- Jet malles hop y est.

1.2.1.4 Niveau syntaxique

La syntaxe étudie la façon dont les mots se combinent pour former les phrases. Elle regroupe les principes et les règles de construction des phrases dans un langage naturel. Chaque langue impose ses contraintes syntaxiques. Voici un exemple de bonne et mauvaise syntaxe :

- Bonne syntaxe : un moteur de reconnaissance vocale performant.
- Mauvaise syntaxe : vocal moteur un reconnaissance performant de

1.2.1.5 Niveau sémantique

La sémantique présente des outils et des techniques pour l'étude de signification des mots. Ce niveau d'analyse met en jeu plusieurs niveaux de données allant du sens des mots, à celui des phrases, aux relations sémantiques entre phrases dans le discours.

1.2.1.6 Niveau pragmatique

La pragmatique s'intéresse à l'usage du langage dans la communication, ou de l'usage du langage en contexte. L'information pragmatique permet de comprendre la signification des mots qui ne peuvent être compris qu'en connaissant le contexte. Par exemple, « la Bleue » au Québec désigne une bière alors qu'en France c'est la carte de crédit Visa.

1.2.1.7 Discussion

Dans la pratique, un système de reconnaissance automatique de la parole modélise peu de niveaux linguistiques. Ainsi, le niveau phonétique est exploité dans le modèle acoustique, notamment pour la modélisation des chaînes de Markov cachée. En outre, les niveaux syntaxique, sémantique et pragmatique sont utilisés dans le modèle de langage. Par contre, ils utilisent que des modèles statistiques qui n'ont rien à voir avec la grammaire que l'on connait. Il est à noter que certains niveaux ne sont modélisés que pour certains types de systèmes de reconnaissance automatique de la parole bien spécifiques. Par exemple, le niveau sémantique est, en général, limité aux applications qui nécessitent l'extraction du sens de la phrase, comme les applications de réponse vocale interactif. De même, le niveau pragmatique est utilisé pour des systèmes spécifiques à un domaine donné et ne s'adapte pas automatiquement comme l'humain peut arriver à faire lorsque l'on change de sujet de discussion. D'autre part, la construction d'un dictionnaire nécessite l'utilisation du niveau lexical et le niveau phonologique. La modélisation de ces derniers, est très importante dans les systèmes de reconnaissance de la parole continue à grand vocabulaire.

1.2.2 Variabilité intra-locuteur et interlocuteur

La variabilité interlocuteur représente les caractéristiques propres à chaque personne. La variabilité entre les locuteurs est due principalement à deux facteurs. D'une part, elle provient des différences physiologiques (dimension du conduit vocal, fréquence d'oscillation des cordes vocales), et d'autre part à la différence de style de prononciation (héritage linguistique, milieu socioculturel de l'individu) (Ezzaidi 2002). De nombreux efforts ont été investis dans le but de contrôler certains aspects de cette variabilité. En effet, l'étude de la variabilité interlocuteur peut améliorer les performances de la reconnaissance vocale jusqu'à 16% (Tubach, Chollet et al. 1990).

La variabilité intra-locuteur est la différence de prononciation du même mot par le même locuteur. En effet, la voix humaine peut varier au cours du temps ou selon les conditions

psychologiques et physiologiques du locuteur. En d'autres termes, cette variabilité dépend de l'état émotionnel, le débit de locution, le stress, le sommeil, etc (Ezzaidi 2002). La prise en considération de ces variations est nécessaire surtout avec les systèmes de vérification automatique du locuteur. En effet, cela permettra à ces systèmes de diminuer le nombre de faux-rejets sans augmenter la quantité de fausses-acceptations (T. Bänziger, G. Klasmeyer et al. 2000).

1.2.3 Matériel

Une étape très importante dans la reconnaissance vocale est la conversion du signal acoustique en un signal électrique. Donc, il est impératif d'utiliser un microphone de qualité, sinon on aura une mauvaise reconnaissance. En outre, la performance de reconnaissance est aussi influencée par la technologie microélectronique. En effet, le traitement automatique de la parole nécessite une grande capacité de calcul et de mémoire. Ces exigences, liées à la bonne reconnaissance vocale, peuvent s'absenter dans certaines situations. Par exemple, les appareils mobiles, comme les téléphones intelligents, fonctionnent avec des ressources limitées. En plus, ils ne sont pas équipés d'un microphone de bonne qualité. La performance des systèmes de reconnaissance de la parole reste donc dépendante de ces facteurs.

1.2.4 Environnement

L'environnement est une variante qui peut influencer significativement la performance des systèmes de reconnaissance vocale. Des expériences ont été réalisées pour étudier l'influence du bruit sur la performance de la reconnaissance vocale. Le résultat montre que les performances de reconnaissance vocale se dégradent dans les milieux bruités (Pearce and Hirsch septembre 2000). Pour résoudre ce problème, certains chercheurs ont travaillé au niveau du spectre afin d'estimer le bruit dans le signal (Hirsch and Ehrlicher 1995). D'autres travaux ont été réalisés pour améliorer les coefficients spectraux MFCC (Mel-Frequency Cepstral Coefficients) (Ezzaidi 2002).

1.2.5 Phénomène de confusion

Le phénomène de confusion pose un problème pour certaines architectures de systèmes de reconnaissance vocale. Afin de mieux comprendre ce phénomène, on prendra l'exemple analysé par le chercheur Frederick Jelinek dans son livre « Statistical Methods for Speech Recognition » (Jelinek 1997). La Figure 1.1 représente l'expression « *Bishop moves to king knight five* » alignée avec son signal temporel. Ce signal a été enregistré à l'Université Stanford pour un projet de reconnaissance vocale (Reddy 1966).

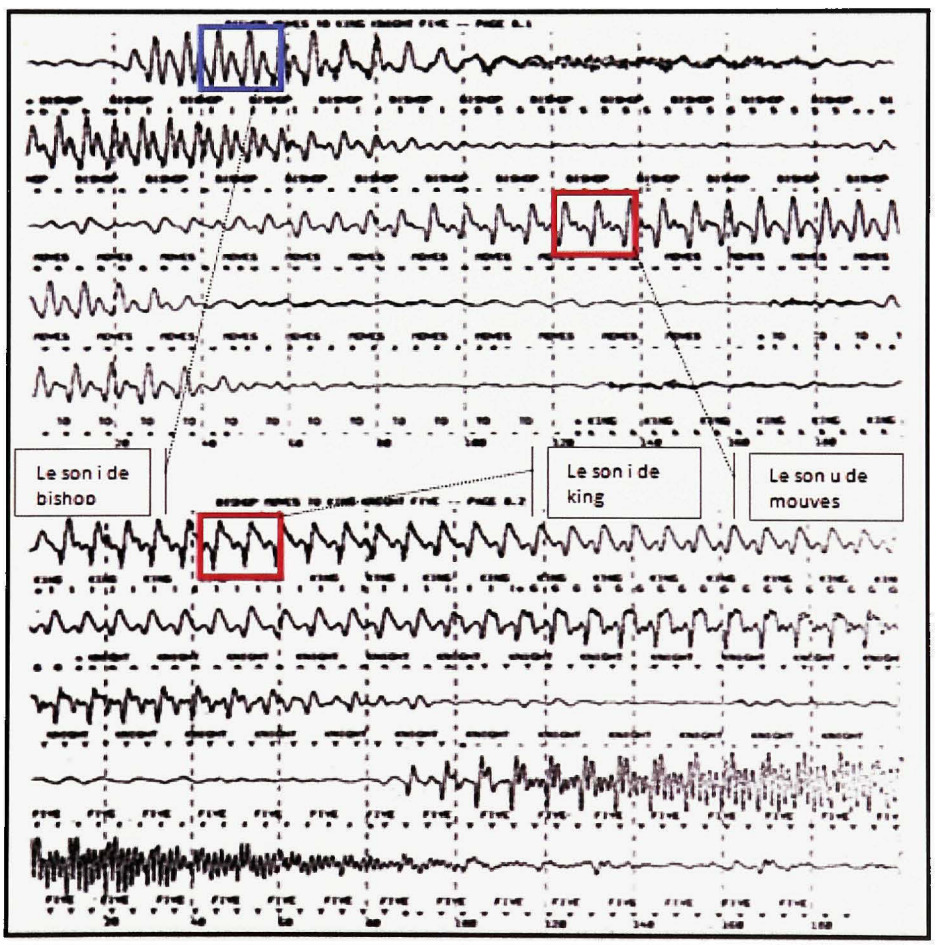

Figure 1.1 La phrase « Bishop moves to king knight five » alignée avec son signal.
Tirée de Jelinek (1997, p. 2)

Jelinek remarque qu'après avoir aligné la phrase avec son signal les extrémités de chaque phonème sont fusionnées avec celui qui le précède et qui le suit. Par conséquent, la prédiction du phonème à cet endroit sera difficile voire impossible. En outre, le son *i* de KING ressemble beaucoup plus au *u* de MOVES qu'au *i* de BISHOP. Selon l'auteur, ceci est dû à la nasalité du *ng* qui suit le *i* dans KING et l'influence du *m* qui précède le *u* dans MOVES.

1.2.6 Discussion

La performance de la reconnaissance automatique de la parole dépend énormément des différents problèmes cités ci-haut. En effet, ces problèmes expliquent grandement les performances de la reconnaissance automatique de la parole relativement faible. Toutefois, les travaux de recherche réalisés jusqu'à maintenant ont permis de surmonter certains problèmes. Par exemple, la modélisation statistique du modèle acoustique permet de résoudre le phénomène de confusion (Jelinek 1997). En outre, l'évolution technologique a permi d'augmenter la capacité de calcul et de fournir un matériel plus sophistiqué et par la suite améliorer les performances de reconnaissance vocale. Cependant, les autres problèmes posent encore un sérieux défi pour les chercheurs. En effet, les recherches se poursuivent pour contrôler les différents aspects de la variabilité intra-locuteur et interlocuteur. De même, plusieurs travaux ont vu le jour pour faire face au problème des environnements bruités (Hirsch and Ehrlicher 1995; Ezzaidi 2002). En outre, la difficulté de modéliser tous les niveaux linguistiques rend la tâche de mettre des systèmes de reconnaissance automatique de la parole plus compliquée.

1.3 Processus de reconnaissance automatique de la parole continue

Le processus de perception de la parole représente la façon dont les humains sont capables d'interpréter les sons pour comprendre la parole. Ce processus fait l'objet de plusieurs études dans différents domaines comme la linguistique, la psychologie cognitive, la perception en psychologie, etc. Les chercheurs dans ces domaines essaient de comprendre, entre autres,

comment l'être humain analyse l'information contenue dans le signal acoustique pour comprendre le langage parlé. Dans le même volet, les chercheurs dans le domaine de traitement automatique de la parole essaient d'exploiter l'information acoustique pour permettre à des machines de reconnaître la parole. En effet, le signal acoustique de la parole est riche en informations linguistiques relatives au message que le locuteur veut faire parvenir. Cependant, le signal vocal doit subir quelques opérations avant qu'il soit prêt à être exploité par une machine. La Figure 1.2 montre les étapes principales d'un système de reconnaissance automatique de la parole depuis la production de la parole par un locuteur passant par l'obtention des trames acoustiques jusqu'à la reconnaissance de la parole. Les étapes à partir de la production de la parole jusqu'à l'extraction des paramètres sont détaillées à la fin du document (Voir annexe I, p. 96).

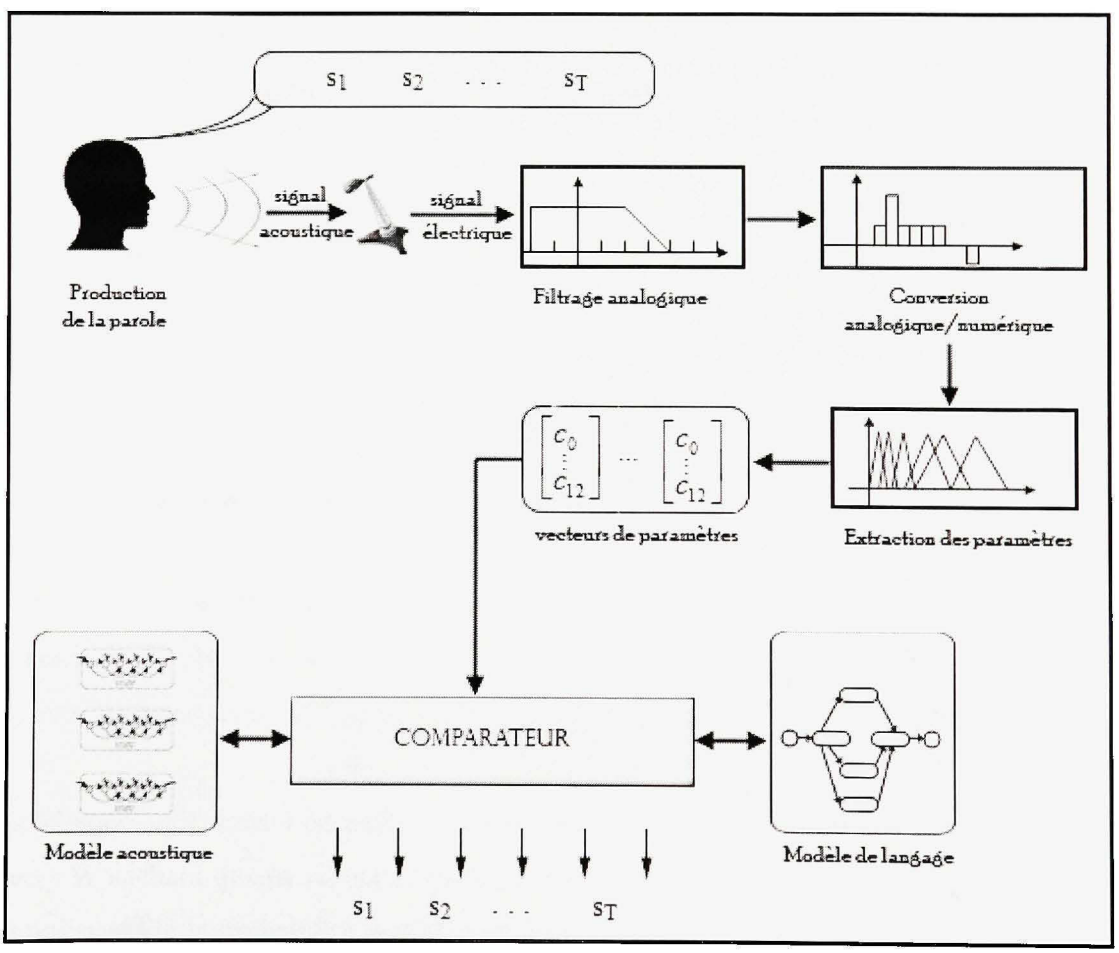

Figure 1.2 Encodage de la parole.

1.3.1 Modèle acoustique

Les premiers systèmes de reconnaissance automatique de la parole utilisaient la méthode de comparaison dynamique (Dynamic Time Warping, DTW) pour comparer une forme inconnue à une forme de référence. Cette approche a connu un succès pour la reconnaissance de mots isolés. Cependant, elle est peu efficace pour la reconnaissance de la parole continue et reste limitée à l'utilisation d'un petit vocabulaire. En effet, l'utilisation des formes acoustiques par un dictionnaire statique reste encore sensible à l'ensemble des variabilités acoustiques. Néanmoins, cette technique est plus adaptée à un contexte d'utilisation mono-locuteur en environnement peu bruité (Linares 2003).

Ainsi, cette méthode a été détrônée par les méthodes de modélisation statistique comme les modèles de Markov cachés (Hidden Markov Models, HMM). Pour plus de détail sur les méthodes de modélisation (Voir annexe II, p. 100). Ces méthodes utilisent des modèles acoustiques afin de représenter mathématiquement les formes acoustiques. Cette représentation sert, en se basant sur certains algorithmes, à identifier le mot le plus probable qui a été prononcé. En effet, la reconnaissance revient à trouver la probabilité de vraisemblance pour un système composé de :

- **Entrée** : signal acoustique, A avec $A = x_1, x_2, \ldots, x_T$; $x_i \in A$
- **Sortie** : séquence de mots, W avec $W = w_1, w_2, \ldots, w_n$; $w_i \in W$

A est la séquence de paramètres qui définit le signal acoustique en amant du système composé de T observations, et W représente un ensemble de n mots appartenant à un vocabulaire bien connu.

Pour chaque mot dans l'ensemble du vocabulaire le système doit calculer la probabilité d'avoir W sachant que la séquence de mots A est prononcée. La décision sera en faveur du mot qui possède la probabilité la plus élevée qu'on note $\hat{W}$. Par la suite, la règle de décision sera :

$$\hat{W} = \arg\max_{W} P(W|A) = \arg\max_{W} \frac{P(W)P(A \mid W)}{P(A)} \quad (1.1)$$

Puisque l'objectif est de déterminer le mot $\hat{W}$ (ou plus précisément la séquence de mots) qui maximise le produit P(W) P(A|W), on peut ne pas calculer P(A) et réduire par la suite le temps de calcul. De toute façon la valeur de P(A) n'influe pas le résultat final puisqu'elle garde la même valeur quelque soit la séquence de mots W. Donc pour une probabilité apriori P(W) le mot le plus probable dépend seulement de P(A|W). Par la suite, $\hat{W}$ sera calculé comme suit:

$$\hat{W} = \arg\max_{W} P(W)\, P(A \mid W) \quad (1.2)$$

1.3.2 Modèle de langage

Le texte transcrit par un système de reconnaissance automatique de la parole comporte toujours des erreurs dues à plusieurs causes (voir section 1.2). En général, ces erreurs ne causent pas de problèmes pour les systèmes qui nécessitent un petit vocabulaire. Dans ce cas, des méthodes simples, comme la comparaison dynamique, suffisent pour identifier les mots dans le flot de paroles. Cependant, les systèmes fondés sur un grand vocabulaire nécessitent un modèle de langage pour aider le décodeur à choisir le mot qui s'adapte le mieux à la forme acoustique prononcée. De point de vue théorique, on parle de la modélisation de la probabilité apriori P(W).

L'objectif principal du modèle de langage est d'identifier à tout moment du processus de reconnaissance la séquence de mots la plus adéquate. Toutefois, il peut servir aussi pour choisir la structure qui correspond mieux à la phrase prononcée. Pour montrer l'importance de ce modèle, prenons l'exemple présenter par Haton (Haton 2006) :

La phrase « un portable maudit » s'écrit phonétiquement : [œ̃pɔrtablmodi]

La segmentation syllabique de cette phrase donne différents mots correspondant à la prononciation de chaque syllabe (voir Tableau 1-1). Par conséquent, la phrase prononcée peut être l'une des phrases résultantes du produit cartésien des mots obtenus par la segmentation :

- Un port tables maux dis
- Un porc table maux Di
- Uns porc tables maux dit
- Hein pores tables maux dis
- Etc

Tableau 1-1 Mots possible extraits d'une chaîne phonétique
Tiré de Haton, Cerisara et al (2006, p. 158)

œ̃	un	uns	Hein			
por	porc	porcs	port	Ports	pore	pores
tabl	table	tables				
mo	maux					
di	dis	dit	Di			

Malgré le fait que ces phrases sont conformes à la forme phonétique, le résultat peut être très éloigné de ce qu'on a prononcé, d'où l'importance d'un modèle de langage.

La conception d'un modèle de langage revient à modéliser la probabilité P(W), où **W** = w_1, w_1, …, w_n est une suite de mots appartenant à un vocabulaire bien connu. L'utilisation de la théorie de Bayes permet de formuler la probabilité P(W) comme suit :

$$P(w) = \prod_{i=1}^{n} P(w_i \mid w_1, \ldots, w_{i-1}) \tag{1.3}$$

Donc la probabilité de prononcé le mot w_i dépend de l'historique des mots $h_i = w_1,\ldots,w_{i-1}$ qui sont déjà prononcés. Plusieurs méthodes peuvent être utilisées afin de modéliser l'historique, comme par exemple le modèle de langage trigramme qui est une méthode basique mais performante. Ce dernier prend en compte seulement les deux derniers mots qui précèdent le mot courant. Pour plus de détails sur le modèle trigramme et l'évaluation d'un modèle de langage (Voir annexe III, p. 123). Une autre approche peut-être utilisée, à savoir la classification des mots par équivalence qui accorde à chaque mot une fonction grammaticale. Cependant, le même mot peut parfois désigner un verbe ou un nom, ce qui complique la tâche et nécessite l'analyse du mot dans son contexte.

Le modèle de langage modélise les informations lexicale, syntaxique et sémantique sous forme statistique avec un historique temporel de trois mots. Cependant, ceci est très limitatif pour une grammaire comme celle du français ou de l'anglais.

1.3.3 Recherche de la séquence la plus probable

Pour qu'un système de reconnaissance automatique de la parole puisse transcrire efficacement un signal acoustique, il doit utiliser des techniques pour rechercher la séquence de mots la plus probable. En effet, on distingue plusieurs algorithmes qui peuvent accomplir cette tâche. Dans ce qui suit on va présenter grossièrement la recherche par l'algorithme de Viterbi et la recherche par la méthode de recherche N-meilleurs. Pour plus de détails, se référer à (Jelinek 1997; Haton 2006).

- L'algorithme de Viterbi est capable de trouver le chemin le plus probable en se basant sur les modèles HMMs. Pour chaque mot, la probabilité de l'ensemble des chemins possibles aboutissant à ce mot doit être trouvée. Par la suite, il faut identifier le mot dont l'ensemble des chemins correspondant est le plus élevé. Tous les chemins possibles peuvent être représentés par un graphe correspondant à un grand ensemble de HMMs.
- La méthode de recherche N-meilleurs effectue la recherche de la même façon que l'algorithme de Viterbi avec une seule différence : plutôt que de retenir un chemin unique

dans chaque treillis, la méthode N-meilleurs découpe chaque état de treillis en N états, un pour les N meilleurs chemins les plus probables. Cette modification de l'algorithme de Viterbi permet d'obtenir les N meilleurs séquences de mots en déterminant les N plus probables états du dernier niveau du treillis et par l'identification des chemins conduisant à eux.

La modélisation des deux modèles, acoustique et de langage, ne suffit pas pour avoir une bonne reconnaissance de la parole. En effet, les machines ont besoin d'une phase d'apprentissage pour reconnaître les nouvelles formes. Cette phase est très importante pour tout système de reconnaissance automatique de la parole. Dans le cas du modèle acoustique, l'apprentissage se base sur un corpus de parole pour ajuster les paramètres du modèle. Une bonne estimation de ces paramètres maximisera la vraisemblance de génération d'une séquence d'observations. Le modèle de langage peut être entraîné en se basant sur un corpus de textes distincts. Ce dernier sert à apprendre le maximum possible de construction langagière. Dans les deux cas, la taille du corpus a une grande influence sur la fiabilité des modèles concernés. Pour l'apprentissage du modèle de langage par exemple, la taille du corpus peut atteindre quelques milliards de mots.

1.4 Revue sur les méthodes d'indexation vocale

L'évolution de la reconnaissance automatique de la parole a conduit à l'apparition des systèmes d'indexation des documents audio. En outre, cette évolution a permis de passer de l'indexation des documentaires ou des émissions de radio (Renals, Abberley et al. 2000), qui sont généralement bien structurés, à l'indexation des fichiers de paroles spontanée. Parmi les systèmes d'indexation de fichiers de parole spontanée on trouve les systèmes d'indexation des e-mails vocaux (Jones, Foote et al. 1994), les conférences universitaires (Park, Hazen et al. 2005), les annotations vocales des images (Mills, Pye et al. 2000), etc. Ces systèmes ont introduit de nouveaux problèmes comme les hésitations, les faux départs, la mauvaise structuration des phrases, etc. Ceci a incité les chercheurs à mettre en place de nouvelles techniques pour faire face à ces problèmes. Toutefois, les différentes techniques dérivent de

trois approches principales d'indexation vocale. On distingue la méthode de détection de mots clés, la détection de mots clés sur flux phonétique et l'indexation à base de la reconnaissance à grand vocabulaire.

1.4.1 Détection de mots clés

La détection de mots clés (Word Spotting, WS) est une technique d'indexation vocale qui consiste à détecter dans un flux de parole des mots clés, généralement prédéfinis, et de rejeter le reste de la parole. En d'autres termes, elle cherche à détecter seulement les mots qui ont une importance pour l'interprétation sémantique de la phrase prononcée et qui sont déjà modélisés dans le vocabulaire des mots clés. La nécessité d'une telle méthode a été mise en évidence durant un projet pour la reconnaissance de mots isolés appliquée à la téléphonie (Wilpon, Rabiner et al. 1990). Ce projet, ainsi que d'autres travaux, comme ceux de Rose, ont fourni une solution acceptable pour la reconnaissance de la parole spontanée (Rose 1995).

Toutefois, le problème de la détection de mots clés dans un flux de parole a été traité depuis les années 70. En effet, Christiansen et Rushforth ont proposé une méthode basée sur la comparaison entre des séquences de références (où chaque séquence représente un mot clé) et le flux de parole continue (Christiansen and Rushforth 1977). Pour cela, ils ont utilisé une fenêtre de taille fixe qui glisse tout au long du flux de parole. Chaque mot de référence est alors comparé avec cette fenêtre par un algorithme d'alignement temporel. Un mot clé est détecté lorsque la distance entre la fenêtre glissante et le mot de référence est suffisamment faible pour plusieurs fenêtres successives.

La méthode proposée par Christiansen et Rushforth a été améliorée par la modélisation des mots poubelles introduite par Higgins et Wohlford (Higgins and Wohlford 1985). En effet, ces modèles ont pour rôle la modélisation des mots hors vocabulaire (le vocabulaire des mots clés), le bruit, les faux départs, etc. Les modèles poubelles peuvent être utilisés pour

modéliser seulement une partie des mots hors vocabulaire, comme les mots de liaison par exemple, ou pour modéliser tous les mots possibles (Rose 1992).

Plusieurs méthodes de modélisation statistiques ont été utilisées pour la détection des mots clés. En effet, les modèles de Markov cachés ont été introduits pour la détection des mots clés émis de façon quasi isolée lors d'appels téléphoniques (Wilpon, Lee et al. 1989). Ils ont été ensuite utilisés pour des tâches plus complexes telles que la détection des mots clés dans la parole continue (Rose and Paul 1990), la catégorisation des documents multimédia (Wilcox and Bush 1992), etc. D'autres chercheurs ont utilisé les réseaux de neurones comme une technique de validation (Morgan, Scofield et al. 1991). Dans ce cas, le réseau de neurones ne fait pas la détection, mais il valide ou non la présence d'un mot clé détecté par un système standard. Dans d'autres cas, le problème de détection de mots clés a été présenté comme un problème de classification (mot clé *correcte* ou *incorrecte*). Ainsi, la classification est réalisée en utilisant les machines à vecteur support (Support Vector Machine, SVM) (Ben Ayed 2004).

La détection des mots clés a été utilisée dans différentes applications d'indexation vocale, on peut citer par exemple le projet *Video Mail Retrieval* (VMR) réalisé dans les laboratoires de l'Université de Cambridge (Jones, Foote et al. 1994). Ce dernier a pour objectif de retrouver les fichiers audiovisuels en se basant sur des méthodes de recherche documentaire et des techniques d'indexation vocale. Le système réalisé se base sur la méthode de détection de mot clés avec 35 mots clés prédéfinis. Chaque mot clé est modélisé en concaténant la séquence de phonèmes appropriée obtenus à partir d'un dictionnaire phonétique. En outre, un modèle bigramme est utilisé pour la modélisation du début et la fin de chaque mot clé, alors que la structure interne est modélisée par un modèle trigramme. Les mots qui ne font pas partie de l'ensemble des mots clés sont modélisés par un réseau monogramme sans contraintes (*filler model*). Deux systèmes ont été réalisés : le premier est dépendant du locuteur (Jones, Foote et al. 1994; Jones, Jones et al. 1996) et l'autre est indépendant du locuteur (Foote, Jones et al. 1995). L'évaluation de ces deux systèmes a montré une

amélioration de la valeur de mérite[5] (Figure-Of-Merit, FOM) pour le système dépendant du locuteur. La moyenne des valeurs de mérite associée à ce dernier fut de 81.14% contre 69.89% pour le système indépendant du locuteur.

Les recherches se poursuivent pour l'amélioration de la méthode de détection des mots clés et pour trouver des solutions pour ses limites (notamment pour le vocabulaire de mots clés prédéfinis). Parmi les nouvelles méthodes on peut citer les travaux de Gelin dans le cadre de sa thèse (Gelin 1997). Ce dernier a proposé trois méthodes différentes pour traiter les contraintes de l'indépendance du locuteur et du vocabulaire ainsi que la rapidité de recherche sur n'importe quel mot clé. Ainsi, il a montré, entre autres, que les améliorations apportées par ses méthodes peuvent réduire le temps de gestion des documents multimédias de 80%.

1.4.2 Détection de mots clés sur flux phonétique

La détection de mots clés sur flux phonétique (Phone Lattice Scanning, PLS) est une méthode d'indexation des fichiers audio à partir des transcriptions phonétiques issues d'un Décodage Acoustico-Phonétique (DAP). Cette méthode ne nécessite pas la définition apriori d'un vocabulaire de mots clés comme dans la méthode de détection de mots clés. Plus encore, elle permet même de surmonter le problème des mots hors vocabulaire, comme les noms propres, présents dans la méthode LVR (Large Vocabulary Recognition). En effet, cette méthode est capable de détecter n'importe quel terme en se basant sur sa représentation phonétique.

En 1994, James et Young ont proposé une nouvelle méthode rapide pour l'indexation des fichiers vocaux sans limitation de vocabulaire (James and Young 1994). Cette approche consiste à générer en temps différé un treillis de phonèmes pour chaque fichier audio en utilisant une version modifiée de l'algorithme Viterbi. Chaque treillis est composé de

---

[5] Moyenne des probabilités de détection pour un taux de fausses alarmes par mot clé et par heure variant entre 0 et 10.

plusieurs hypothèses de séquences phonétiques sur ce qui a été prononcé. La détection des mots clés est réalisée par comparaison dynamique entre le mot recherché et les séquences de phonèmes dans le treillis. Cette conception permet une recherche rapide puisque les treillis sont générés et sauvegardés avant la phase de recherche. En effet, la recherche de 20 mots clés par cette méthode est 60 fois plus rapide que la recherche en temps réel.

Chelba et Acero ont proposé une nouvelle représentation du treillis (Position Specific Posterio Lattices, PSPL) permettant de surmonter les limites d'une méthode de recherche 1-best (Chelba and Acero 2005). En effet, la méthode PSPL retient seulement les informations sur la position de chaque mot dans la phrase prononcée. Ceci permet d'augmenter de 20% la performance de détection de mots clés dans les documents. Cependant, la performance de cette méthode, par rapport à la recherche dans des données textes, reste encore médiocre.

La dernière version du projet VMR, mentionné dans la section 1.4.1, a apporté plusieurs avantages par rapport aux versions précédentes (Young, Brown et al. 1997). En effet, le nouveau système, basé sur la méthode PLS, est indépendant du locuteur et il permet de faire l'indexation des fichiers audio sans se limiter à un vocabulaire de mots clés prédéfinis. Le système génère un treillis de phonème pour chaque message vocal, la recherche de mots clés se fait par la suite dans le treillis. La Figure 1.3 représente le treillis de phonème pour le mot «*manage*», le chemin correspondant est représenté en gris. L'évaluation du système a montré que la moyenne des précisions est de 0.5 lorsque le système est dépendant du locuteur et 0.47 lorsqu'il est indépendant du locuteur. Cependant, ces valeurs se dégradent dans le cas où les mots clés sont limités (35 mots clés dans ce cas). Pour améliorer les performances, Jones, Foote et al. ont proposé de combiner la méthode PLS avec la transcription fournie par un moteur de reconnaissance de la parole grande vocabulaire (Jones, Foote et al. 1996).

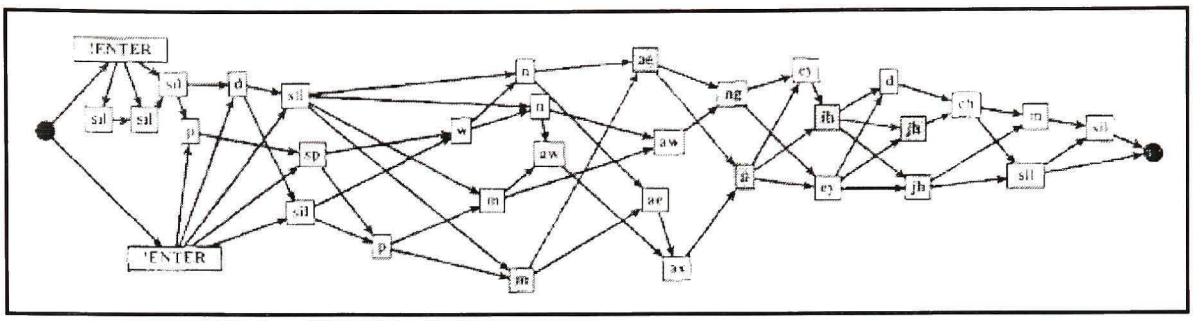

Figure 1.3 Treillis phonétique du mot "manage".
Tirée de Young, Brown et al. (1997, p.200)

1.4.3 Reconnaissance de la parole à grand vocabulaire

La méthode Large Vocabulary Recognition permet l'indexation des fichiers audio en se basant sur la transcription fournie par un système de reconnaissance automatique de la parole continue. Contrairement à la méthode de détection de mots clés, la méthode LVR utilise des systèmes à grand vocabulaire pour transcrire tous les mots de l'expression prononcée. Le résultat de la transcription sert par la suite pour rechercher n'importe quel mot contenu dans le vocabulaire.

La société « Dragon System » a utilisé cette méthode pour identifier des sujets prédéterminés en se basant sur la transcription des phrases prononcées par un système de reconnaissance de la parole contenue sur large vocabulaire (Gillick, Baker et al. 1993). L'objectif était d'identifier quelle catégorie (parmi dix catégories) correspond mieux à une discussion téléphonique de 4,5 minutes. En se basant sur la transcription fournie par un système de reconnaissance de la parole continue à grand vocabulaire, 72.5% des discussions ont été bien classées. En utilisant la méthode de détection de mots clés, pour les mêmes données, avec 4600 mots clés prédéfinis, 74% des discussions ont été bien classées. Pour la même méthode, mais avec seulement 260 mots clés prédéfinis, 70% des discussions ont été bien identifiées.

La précision de transcription automatique des textes lus peut dépasser 90% en utilisant un système de reconnaissance de la parole continue à grand vocabulaire. Cependant, pour la

parole spontanée la performance se dégrade considérablement. Par exemple, une expérience réalisée au cours du projet VMR pour la reconnaissance de la parole spontanée a donné une précision de 53% (Jones, Foote et al. 1996). La valeur de précision relativement faible n'a pas beaucoup d'importance dans le cas d'indexation vocale. En effet, l'évaluation des systèmes de recherche documentaire audio a montré que malgré la mauvaise transcription on peut obtenir des résultats d'indexation proche de ceux obtenus par des systèmes basés sur des transcriptions textuelles (Garofolo, Auzanne et al. 2000). En outre, les travaux de Park, Hazen et al. sur l'indexation des conférences universitaires confirment que la précision peut être satisfaisante même avec un taux d'erreur élevé (Park, Hazen et al. 2005). Les expériences qu'ils ont menées montrent que la précision atteint 92.9% pour un taux d'erreur de 52.6%. De même, la précision augmente jusqu'à 95% avec une adaptation du modèle acoustique. Cependant, l'adaptation du modèle de langage n'a pas influencé la valeur de la précision.

Malgré la grande taille du vocabulaire utilisée par les systèmes de reconnaissance automatique de la parole continue, la méthode LVR souffre du problème de mots hors vocabulaire (Out-Of-Vocabulary, OOV). Dans le cas des conférences universitaires par exemple, les présentations contiennent beaucoup de mots techniques qui ne sont pas inclus dans le vocabulaire du système. En étudiant l'impact de la taille du vocabulaire sur la précision, Park, Hazen et al. ont montré que l'augmentation de la taille du vocabulaire diminue le taux OOV, et par la suite augmente la précision, mais diminue aussi le rappel (Park, Hazen et al. 2005).

La méthode LVR donne des résultats satisfaisants pour l'indexation des fichiers audio. Toutefois, le résultat peut être amélioré en augmentant le taux de reconnaissance des mots (Park, Hazen et al. 2005). Ceci peut se réaliser, entre autres, par l'intégration de modèle acoustique trigramme, l'adaptation en temps réel et l'amélioration de la modélisation de la parole spontanée.

1.5 Conclusion

Dans ce chapitre on a présenté le cadre général de notre sujet d'étude. Ainsi, on a décrit la problématique, les objectifs à atteindre ainsi que les défis. Afin de mieux comprendre la difficulté de l'indexation vocale, on a exposé les principaux problèmes de la reconnaissance automatique de la parole. On a décrit ensuite la structure générale d'un système de reconnaissance automatique de la parole. On a terminé ce chapitre par une revue sur les différentes approches d'indexation vocale. Pour chaque approche, on a donné les différentes techniques utilisées souvent en citant leurs avantages et leurs faiblesses. Il est judicieux de mentionner qu'aucune des approches présentées n'est totalement satisfaisante. Pour cette raison les recherches commencent à s'orienter vers des approches hybrides pour combiner entre plusieurs d'entre elles. Toutefois, même les approches hybrides présentent des limitations, ce qui incite les chercheurs à explorer différentes directions.

CHAPITRE 2

METHODOLOGIE

Ce chapitre couvre en grande partie la méthodologie et les outils utilisés pour la réalisation de notre étude. On consacrera la première partie pour décrire quelques systèmes de reconnaissance vocale présents sur le marché. On sélectionnera, en se basant sur quelques critères, deux moteurs de reconnaissance automatique de la parole qui feront l'objet de nos expérimentations. Parmi ces critères on peut citer la taille du vocabulaire, la possibilité d'intégration du moteur de reconnaissance à une application, l'adaptation par rapport au contexte de l'annotation vocale, etc.

Dans la deuxième partie, on présentera les différentes métriques utilisées pour l'évaluation et la comparaison des performances des deux moteurs de reconnaissance qu'on a choisis. On introduira tout d'abord le Taux de Mots clés Correct (TMC) et le taux d'erreur (TE) qui serviront pour l'évaluation et la comparaison de la performance de reconnaissance des deux moteurs. Ensuite, on présentera différentes métriques dont le rappel, la précision, la F-mesure et les courbe ROC (Receiver Operating Characteristics) qui seront utilisés pour évaluer et comparer la capacité de détection de mots clés par les deux moteurs.

Dans la dernière partie de ce chapitre, on détaillera le processus expérimental qui représente les différentes étapes de notre étude. Toutefois, l'aspect pratique et les détails techniques ne feront pas l'objet de ce chapitre. Ils seront présentés dans le chapitre 3 qui sera consacré entièrement à ce sujet.

2.1 Choix de la méthode d'indexation

Chacune des trois méthodes d'indexation présentées dans la section 1.4 présente des inconvénients. En effet, la méthode de détection de mots clé se base sur un vocabulaire de mots clé prédéfinis. Par conséquent, chaque nouveau mot clé doit être modélisé à partir d'un dictionnaire phonétique avant de réaliser la recherche. La détection de mots clés sur flux

phonétique est une solution à ce problème. En se basant sur la représentation phonétique du mot clé recherché, cette technique permet la détection de n'importe quel terme dans le treillis de phonèmes. Cependant, la performance d'indexation avec cette méthode se dégrade en utilisant un nombre limités de mots clés. La méthode de reconnaissance de la parole à grand vocabulaire permet l'indexation vocale en se basant sur le résultat de transcription obtenu par un moteur de reconnaissance automatique de la parole. Ainsi, elle permet la recherche de n'importe quel mot clé contenu dans le vocabulaire du système. Cette technique est utilisée dans plusieurs systèmes d'indexation vocale.

Dans notre étude, on a utilisé la méthode de reconnaissance de la parole à grand vocabulaire pour réaliser nos expérimentations. À part la possibilité de détecter n'importe quel mot clé, cette méthode permet de générer la transcription des fichiers audio contrairement aux deux autres méthodes. Cependant, l'utilisation de cette méthode nécessite une bonne gestion des erreurs de transcription afin d'avoir une performance acceptable.

2.2 Critères de sélection

Les systèmes de reconnaissance automatique de la parole présentent plusieurs caractéristiques et il est souvent difficile de faire une comparaison entre deux systèmes. Toutefois, il est possible de les classer suivant des critères de sélection permettant de donner une vision plus représentative. Dans notre étude, on a besoin d'un système capable de réaliser la transcription de la parole continue. Un tel système doit se baser sur un vocabulaire de grande taille. En outre, la reconnaissance de la parole est utilisée dans un contexte d'annotation vocale, ainsi, le système doit être capable de s'adapter à un tel contexte. D'autre part, le moteur de reconnaissance automatique de la parole doit être accessible par un langage de programmation pour faciliter son intégration à Docuthèque. De plus, il doit supporter au moins le français comme langue de reconnaissance. Ainsi, la sélection du système sera basée sur les critères suivants : langue, technologie d'intégration, modèle économique, taille du vocabulaire et adaptabilité au contexte de l'annotation.

2.3 Sélection de systèmes de reconnaissance vocale

2.3.1 Revue sur les systèmes de reconnaissance vocale

2.3.1.1 IASRT

ISIP (Institute for Signal and Information Processing) (ISIP 2010) est une organisation fondée en 1994, elle a travaillé sur plusieurs projets en reconnaissance vocale. Ainsi, elle a réalisé le projet Internet-Accessible Speech Recognition Technology (IASRT 2010) dans le but de mettre en place un système de reconnaissance de la parole open source. Ce système est développé en C++ permettant l'ajout et la modification de ses composantes. Ainsi, il est possible de l'exploiter pour bâtir une application de reconnaissance vocale en C++ ou même en langage Java pour l'intégrer à une page web en tant que applet. En outre, plusieurs outils complémentaires sont fournis avec le moteur de reconnaissance vocale. Toutefois, aucun dictionnaire global n'est fourni, il est nécessaire d'en construire un pour faire l'apprentissage des modèles acoustiques. De même, l'adaptation par rapport au contexte de l'annotation vocale nécessite la construction d'un dictionnaire incluant les termes et expressions spécifiques au contexte de l'annotation.

2.3.1.2 LumenVox

LumenVox est un système de reconnaissance vocale composé de trois modules :

- *Speech Engine* : c'est un moteur de reconnaissance vocale multilingue, il permet la reconnaissance automatique de la parole de n'importe quelle source audio.
- *Speech Tuner* : ce module permet d'effectuer des tests et d'analyser les différentes étapes de la reconnaissance vocale.
- *Speech Platform* : c'est une plateforme de développement de solutions de reconnaissance vocale, fourni gratuitement avec *Speech Engine*. Cette plateforme inclu une bibliothèque qui peut être enrichie par d'autres fonctions crées par l'utilisateur. Ainsi, des DLL peuvent êtres développées avec le langage C/C++ ou des ActivesX avec Visual Basic.

La solution est basée sur une architecture client/serveur permettant la reconnaissance en temps réel et indépendamment du locuteur. Le vocabulaire est fourni au moteur de reconnaissance sous forme de liste de mots, prononciations ou selon le standard SRGS (Speech Recognition Grammar Specification). Toutefois, le système est orienté vers des applications IVR, il n'est pas conçu comme un système de dictée vocale. Il est basé sur un vocabulaire de petite taille (entre 500 et 12000 mots) qui ne permet de faire ni la dictée ni l'AudioMining.

2.3.1.3 ViaVoice

ViaVoice (IBM 2010) est une solution de reconnaissance vocale développée par IBM et offerte au grand public en 1993. En 2003, IBM donne les droits de distribution exclusifs à ScanSoft (avant sa fusion avec Nuance). ViaVoice est compatible avec le standard JSAPI, donc il est facile d'utiliser son moteur de reconnaissance pour développer des applications vocales. En outre, le kit de développement *Embedded Viavoice Multiapplication SDK* permet à plusieurs applications d'accéder simultanément aux ressources de la reconnaissance vocale et à la synthèse vocale à partir de périphériques embarqués. D'autres versions, comme la version *Pro USB*, permettent le développement des macros exécutables à partir d'autres applications. De point de vue adaptabilité, le système est fourni avec un dictionnaire de 300 000 mots avec des vocabulaires spécialisés pour le domaine juridique et médical. En plus, le système s'adapte au fur et à mesure de son utilisation, il est donc possible qu'il s'adapte plus au contexte de l'annotation.

2.3.1.4 VoiceObject

Voxeo (Voxeo 2010) est une compagnie qui fournit des solutions pour développer des applications de reconnaissance vocale. On trouve parmi ses produits VoiceObject qui regroupe plusieurs outils permettant le développement et le test d'applications vocales. En outre, VoiceObject peut être intégré dans l'IDE Netbeans pour développer des applications vocales par le langage Java. Cependant, son moteur de reconnaissance vocale n'est pas basé

sur un grand vocabulaire et ne peut pas être adapté à un contexte d'annotation vocale. En plus, il n'est pas conçu comme un système de reconnaissance vocale proprement dit, mais il utilise cette technologie afin de fournir des solutions et des plateformes de développement de solutions pour les centres d'appels, agences de voyages, réservations de billets, etc.

2.3.1.5 Winscribe

Winscribe est une solution de dictée vocale à grand vocabulaire qui permet la reconnaissance en temps réel ou en temps différé. La compagnie travaille en partenariat avec Nuance, mais il n'est pas indiqué si c'est le moteur de reconnaissance de Nuance qui est utilisé pour Winscribe. Ce dernier s'intègre à plusieurs types de périphériques tels que les téléphones fixes et portables, BlackBerry, PDAs, appareils de dictée portables, microphones de bureaux, ordinateurs portables et VoIP. La reconnaissance vocale est réalisée en premier plan (client) et/ou en arrière-plan (serveur). En outre, des fonctionnalités d'accès par Internet permettent aux utilisateurs de faire leurs dictées (grâce à leurs PC ou appareils mobiles) puis les envoyées pour la transcription. Le système utilise un dictionnaire de 300 000 mots avec un certain nombre de vocabulaires spécialisés : professionnels de santés, radiologie, compagnies financières et d'assurance, professionnels du droit, etc. En plus, un dictionnaire individuel pour chaque utilisateur est ajouté. Toutefois, il n'est pas possible d'utiliser un langage de programmation pour l'intégrer à une application ni d'ajouter des extensions.

2.3.1.6 Sphinx

Sphinx-4 (Walker, Lamere et al. 2004) est un système de reconnaissance vocale open source incorporant les nouvelles techniques de l'état d'art du domaine de la reconnaissance automatique de la parole. Il est le fruit des collaborations entre les chercheurs du groupe Sphinx du CMU (Carnegie Mellon University), Laboratoires Sun Microsystems, Mitsubishi Electric Research Labs (MERL), et Hewlett Packard (HP), avec la contribution de l'Université de Californie à Santa Cruz (UCSC) et l'Institut de Technologie du Massachusetts (MIT) (CMU 2008). Plusieurs autres versions ont précédé celle-ci notamment

Sphinx-3 (Vojtko, Korosi et al. 2008), il diffère principalement de ces antécédents par le fait qu'il est écrit entièrement dans le langage de programmation Java. Le premier objectif du développement de Sphinx-4 est d'avoir un système de reconnaissance de la parole hautement flexible, modulaire et paramétrable. En effet, ces caractéristiques permettent de faire de nouvelles recherches et tester de nouveaux algorithmes. En outre, Sphinx-4 permet de bâtir trois types d'applications vocales : reconnaissance de mots isolés, commande vocale et dictée générale. Toutefois, la qualité de la reconnaissance dépend de la qualité des données vocales et de la performance de l'apprentissage. Ainsi, CMU a développé un module d'apprentissage capable d'entraîner des modèles acoustiques pour le système Sphinx-3 exploitable aussi par Sphinx-4. Pour la description de l'architecture de ce système (Voir annexe IV, p. 126).

2.3.1.7 HTK

HTK (Hidden Markov Model Toolkit) est une trousse à outils pour la conception et la manipulation des chaînes de Markov, développée au département d'ingénierie de l'Université de Cambridge. L'infrastructure de HTK est conçue dès le départ pour répondre à des tâches de reconnaissance vocale. Elle a évolué ces dernières années pour toucher la synthèse vocale, la reconnaissance des caractères et le séquençage de l'ADN. La première version de HTK, réalisée par Steve Young, a vu le jour en 1989 dans le groupe *Speech Vision and Robotics*. Elle représentait une bibliothèque de modules et d'outils développés en langage C et formant la base des versions ultérieures. Aujourd'hui, la version 3.4 fournit des outils sophistiqués permettant de traiter un signal acoustique et de l'analyser, entraîner des modèles HMMs, etc. Quelques extensions de HTK ont été développées par les chercheurs tels que HDecode, HTS, MASV, etc. Pour plus de détail sur l'architecture de ce système (Voir annexe V, p. 133).

2.3.1.8 Système de reconnaissance vocale de Microsoft

Microsoft a travaillé depuis 1997 sur l'intégration de la technologie de la reconnaissance vocale à son système d'exploitation. À l'époque, Bill Gates prévoyait dix ans pour que la reconnaissance vocale devienne le troisième moyen d'interaction avec l'ordinateur (avec la

souris et le clavier). En 2007, Microsoft a intégré la reconnaissance vocale à son nouveau système d'exploitation Vista pour qu'elle soit accessible à toute application installée sur le système. Durant ces dix ans, Microsoft a développé plusieurs applications basées sur la reconnaissance vocale telle que *Narrator*[6]. Avant ça, elle travaillait en collaboration avec Carnegie Mellon University (CMU) pour la mise en place d'un moteur de reconnaissance vocale. En effet, ils ont réussi à développer quatre versions de moteur de reconnaissance et leurs standards associés regroupés dans SAPI 4 (Speech Application Programming Interface). Ensuite Microsoft a travaillé sur le développement de son propre moteur de reconnaissance et elle a réussi à mettre en place la version SAPI 5.3 qui inclu le français. Pour plus de détails sur ce standard, on peut consulter (Microsoft 2010).

Le moteur de reconnaissance vocale de Microsoft est basé sur les modèles de Markov cachés. Il est composé de différents modules incluant plusieurs éléments de Sphinx et HTK. La Figure 2.1 représente ces composantes ainsi que les relations entre elles.

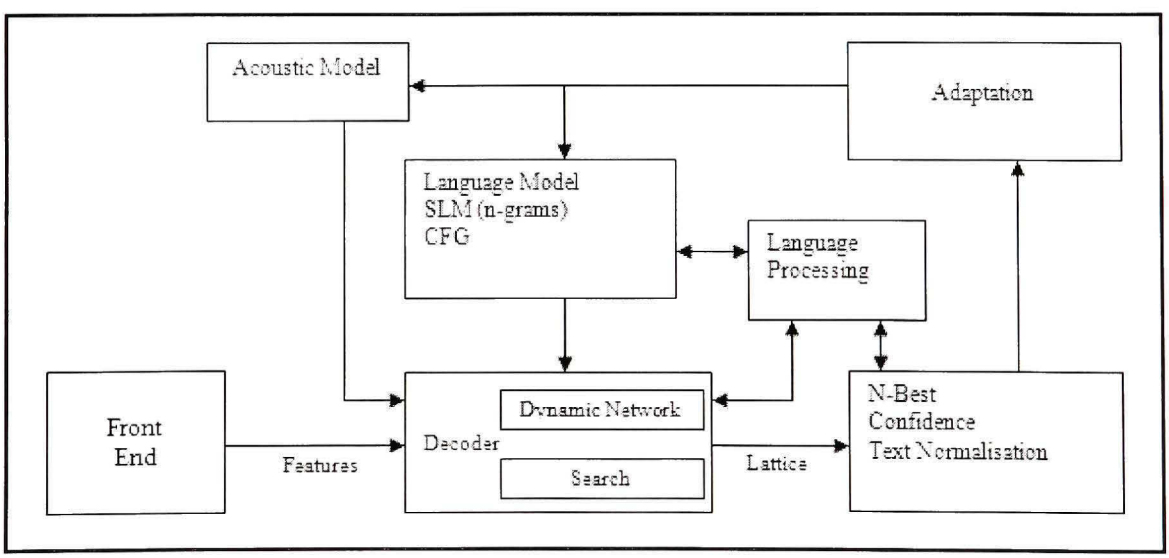

Figure 2.1 Architecture du moteur de reconnaissance vocale de Microsoft.
Tirée de Odell et Mukerjee (2007, p.1162)

[6] Application basique de synthèse vocale pour la lecture de documents texte.

Le système commence par le traitement du signal d'entrée dans le module *front-end* pour capter les caractéristiques spectral de ce signal. Ainsi, chaque 10 ms une fenêtre de 25 ms est appliquée au signal afin de générer des vecteurs MFCC. Ensuite, le module *Decoder* recherche le mot le plus proche des vecteurs obtenus en se basant sur le module *Acoustic model*. En se basant sur le module *Language Model*, le système met à jours l'expression générée en choisissant le mot le plus adapté au contexte de la phrase. Le module *Adaptation* permet une amélioration continuelle du modèle acoustique et le modèle de langage. Le résultat final de la reconnaissance est envoyé à l'SAPI qui le passe à son tour à l'application appropriée. Pour consulter l'architecture détaillée du système, on peut consulter (Odell and Mukerjee 2007).

2.3.1.9 Dragon NaturallySpeaking

Dragon NaturallySpeaking est un logiciel de reconnaissance vocale qui permet aux utilisateurs de créer des documents et des e-mails, remplir des formulaires électroniques et gérer leurs flux de travail par voix. Cette solution est commercialisée par la société Nuance (Nuance 2010) en offrant différentes éditions orientées métier. Dans notre étude, on a utilisé le moteur de reconnaissance vocale de Dragon NaturallySpeaking 10 Preferred. En outre, Nuance offre un kit de développement logiciel (Software Developer Kit, SDK) pour l'intégration des fonctionnalités vocales aux applications. Le produit SDK de Dragon inclut trois éditions :

- Dragon NaturallySpeaking SDK Client (DSC): pour le développement des applications de reconnaissance vocale en temps réel.
- Dragon NaturallySpeaking SDK Server (DSS) : pour le développement d'application de reconnaissance vocale en temps différé basé sur une architecture client/serveur.
- Dragon NaturallySpeaking SDK AudioMining : pour le développement d'application d'indexation vocale.

Dragon NaturallySpeaking présente plusieurs outils permettant l'adaptation du modèle acoustique et du modèle de langage :

- Outils pour l'amélioration acoustique : plusieurs outils sont fournis pour la vérification et le réglage de paramètres audio, l'optimisation acoustique et linguistique, l'apprentissage supplémentaire, etc.
- Outils pour l'amélioration du vocabulaire : ces outils permettent d'ajouter des mots uniques ou une liste de mots au vocabulaire. En plus, d'autres outils permettent de scanner des documents et analyser les messages pour améliorer le modèle de langage.

D'autres outils et utilitaires sont offerts pour faire la transcription des fichiers sonores, la synthèse vocale, la création de commandes, etc.

2.3.2 Grille de comparaison

Afin d'avoir une vue comparative des différents systèmes détaillés dans la section précédente, on a représenté les caractéristiques de ces systèmes par une grille de comparaison. En effet, le Tableau 2-1 représente les différents systèmes ainsi que les critères de sélection déjà définis (voir section 2.2).

Tableau 2-1 Grille de comparaison

Critère / Système	Langue	Technologie d'intégration	Modèle économique	Taille du vocabulaire	Adaptable au contexte de l'annotation
IASRT	Selon le dictionnaire construit	Langage C++ et langage Java	Licence publique	Un dictionnaire doit être construit	Oui
LumenVox	Anglais, français international, français québécois	Langage C, C++ et Visual Basic	Entre 245$ et 595$ US pour les versions standards sans SDK	Entre 500 et 12000 mots	Oui
ViaVoice	Anglais et français	JSAPI	49.99 et 189.99 US	300 000 mots	Oui
VoiceObjects	Anglais	Langage Java	VoiceObjects Developpers est gratuit	n/a	Non
WinScribe	Anglais et français	n/a	Selon les fonctionnalités	300 000 mots	Oui
Sphinx	Selon le dictionnaire construit	Langage Java	Licence BSD	Un dictionnaire doit être construit	Oui
HTK	Selon le dictionnaire construit	Langage C, C++,...	Licence non-exclusive pour les produits commerciaux	Un dictionnaire doit être construit	Oui
Dragon	Anglais et français	JSAPI et Dragon SDK	Entre 99$ et 800$ US	Grand vocabulaire	Oui
Microsoft	Anglais et français	JSAPI	gratuit	Grand vocabulaire	Oui

2.3.3 Choix du système

Pour la réalisation de nos expérimentations, on a besoin d'un moteur de reconnaissance vocale fourni parmi les différents systèmes présentés ci-dessus. On remarque que plusieurs

systèmes ne répondent pas parfaitement à nos critères de sélection. Le premier système ISIP ne fournit aucun dictionnaire avec son moteur de reconnaissance. Dans notre étude, on a besoin d'un système à grand vocabulaire, toutefois, il est très difficile de construire un tel vocabulaire dans le cadre de ce travail. Les deux systèmes LumenVox et VoiceObject sont plus orientés vers les applications IVR et il se base sur un vocabulaire de petite taille. Pour le logiciel Winscribe, l'entreprise qu'il l'a développé est un partenaire de Nuance, ainsi il est fort probable qu'il utilise les technologies de reconnaissance vocale de Nuance. En ce qui concerne les deux systèmes Sphinx et HTK, on trouve qu'ils sont des systèmes évolués et présentent beaucoup d'avantages. Toutefois, l'utilisation de ces deux systèmes nécessite aussi la construction d'un grand vocabulaire. Le système ViaVoice répondant aussi à nos critères de sélection mais n'étant plus supporté, on a préféré de choisir les deux systèmes restants, à savoir Dragon NaturallySpeaking et celui fourni par Microsoft. Ces deux systèmes répondent parfaitement à nos critères. En effet, ce sont des systèmes de reconnaissance automatique de la parole continue basés sur un grand vocabulaire. En plus, leurs moteurs de reconnaissance sont accessibles par le langage de programmation Java et ils sont tous les deux compatibles avec le standard JSAPI. En outre, ils permettent l'ajout de nouveaux termes et expressions à leurs dictionnaires.

2.4 Systèmes d'évaluation

Dans cette section, on présentera les différentes métriques utilisées dans notre étude. D'une part, ces métriques serviront à évaluer les deux moteurs de reconnaissance automatique de la parole qu'on a utilisés. D'autre part, elles serviront à comparer leur performance de reconnaissance et leurs capacités de détection de mots clés.

2.4.1 Taux de reconnaissance du système de RAP

Les systèmes de reconnaissance automatique de la parole peuvent être classifiés suivant différents critères. Néanmoins, le critère de comparaison le plus significatif est le taux de reconnaissance. Ce dernier vari selon les erreurs qu'un système de reconnaissance

automatique de la parole peut commettre. L'estimation de ces erreurs permet l'évaluation de la performance d'un système de reconnaissance vocale donné et de le comparer à d'autres systèmes. On distingue trois types d'erreurs pouvant être commises par un système de reconnaissance automatique de la parole continue :

- **Erreur d'omission** (D) : le mot prononcé n'est pas reconnu par le système.
- **Erreur d'insertion** (I) : un mot n'est pas prononcé, mais reconnu par le système.
- **Erreur de substitution** (S): le mot prononcé est reconnu comme un autre mot.

Dans notre étude on a utilisé l'outil d'évaluation *HResults* fourni avec le système de reconnaissance automatique de la parole HTK (Young, Odell et al. 1995). Cet outil permet l'estimation de ces trois erreurs en se basant sur un algorithme de comparaison dynamique. Ainsi, le Taux de Mots Correct (TMC) (Word Correct Rate, WCR) est calculé comme suit :

$$TMC = 100\% \times \frac{N-D-S}{N} \tag{2.1}$$

où N est le nombre total de mots dans la transcription originale (transcription manuelle produite par un humain).

Cependant, la métrique TMC ne tient pas compte des erreurs d'insertion. On peut alors estimer le Taux de Précision (TP) (Percent Accuracy, PA) calculé comme suit :

$$TP = 100\% \times \frac{N-D-S-I}{N} \tag{2.2}$$

La prise en compte des erreurs d'insertion permet aussi d'estimer le Taux d'Erreur (TE) (Word Error Rate, WER) qui s'écrit sous la forme suivante :

$$TE = 1 - TP = 100\% \times \frac{D-S-I}{N} \tag{2.3}$$

On remarque que le taux d'erreur est plus sévère que le taux de mots correct puisqu'il tient compte des erreurs d'insertion. Toutefois, dans le cas d'un système de détection de mots clés, on peut obtenir un taux d'erreur supérieur à 1, ou encore un taux de mots corrects inférieur à 0. Ceci est observable dans des cas extrêmes où le nombre d'insertion est plus grand que le nombre de mots à reconnaître.

2.4.2 Les métriques de performance de l'indexation

La majorité des métriques utilisées pour représenter la performance d'un système sont dérivées de la matrice de confusion (voir Figure 2.2). Cette dernière, appelée aussi tableau de contingence, est une matrice où chaque colonne représente le nombre d'occurrences d'une classe réelle et chaque ligne représente le nombre d'occurrences d'une classe estimée. Le problème de détection de mots clés peut être considéré comme un problème de classification. Ainsi, on considère les deux classes Correcte et Incorrecte $\{C, I\}$, chaque instance de l'ensemble des données est alors libellé C ou I. L'évaluation de la performance d'un système de détection de mots clés peut être dégagée par l'analyse de la matrice de confusion. En effet, elle permet notamment de savoir combien d'instances Correcte sont faussement classées Incorrecte, et combien d'instances Incorrecte ne sont pas estimées comme telle.

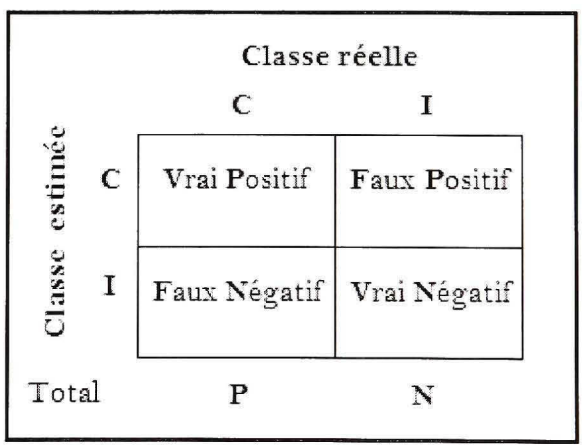

Figure 2.2 Matrice de confusion.

Comme représentée dans la Figure 2.2 il y a quatre types de sorties :

- **V**rai **P**ositif (True Positive) : si l'instance est Correcte et est classée Correcte.
- **F**aux **N**égatif (False Negative) : si l'instance est Correcte mais elle est classée Incorrecte.
- **V**rai **N**égatif (True Negative) : si l'instance est Incorrecte et est classée Incorrecte.
- **F**aux **P**ositif (False Positive) : si l'instance est Incorrecte mais elle est classée Correcte.

Dans notre étude, on a utilisé le taux d'erreur et le taux de mots corrects pour la comparaison de deux systèmes de reconnaissance vocale. Cependant, pour l'évaluation de la performance de détection de mots clés, on a utilisé d'autres métriques : le rappel, la précision et la F-mesure.

- Le rappel : le rappel est le nombre de mots clés correctement détectés au regard du nombre de mots clés à détecter.

$$Rappel = \frac{nombre\ de\ mots\ clés\ correctement\ détectés}{nombre\ de\ mots\ clés\ à\ détecter} \tag{2.4}$$

Le rappel peut être calculé à partir de la matrice de confusion :

$$Rappel = \frac{Vrai\ Positif}{Vrai\ Positif + Faux\ Négatif}$$

- La précision : la précision est le nombre de mots clés correctement détectés par rapport au nombre total de mots clés détectés.

$$Précision = \frac{nombre\ de\ mots\ clés\ correctement\ détectés}{nombre\ total\ de\ mots\ clés\ détectés} \tag{2.5}$$

La précision peut être calculée à partir de la matrice de confusion :

$$Précision = \frac{Vrai\ Positif}{Vrai\ Positif + Faux\ Positif}$$

- F-mesure : la F-mesure combine la valeur du rappel est celle de la précision en une mesure unique.

$$F\text{-}mesure = \frac{2}{\frac{1}{Rappel} + \frac{1}{Précision}} \tag{2.6}$$

2.4.3 Courbe Rappel/Précision

Pour tout système, les métriques de rappel et de précision changent continuellement, et d'une façon générale, l'augmentation de l'une des métriques entraîne la diminution de l'autre. Ainsi, on a tracé des courbes rappel/précision pour les deux moteurs et selon les deux profils afin d'étudier l'évolution de la précision en fonction du rappel. Un système qui aurait un rappel et une précision égaux à 1, signifie qu'il détecte tous les mots clés, et rien que les mots clés. Dans ce cas, on aura, comme courbe, une droite horizontale dont l'ordonné est égale à 1, toutefois, ce cas est quasiment impossible à atteindre. La Figure 2.3 illustre l'allure générale d'une courbe rappel/précision ainsi que la courbe optimale.

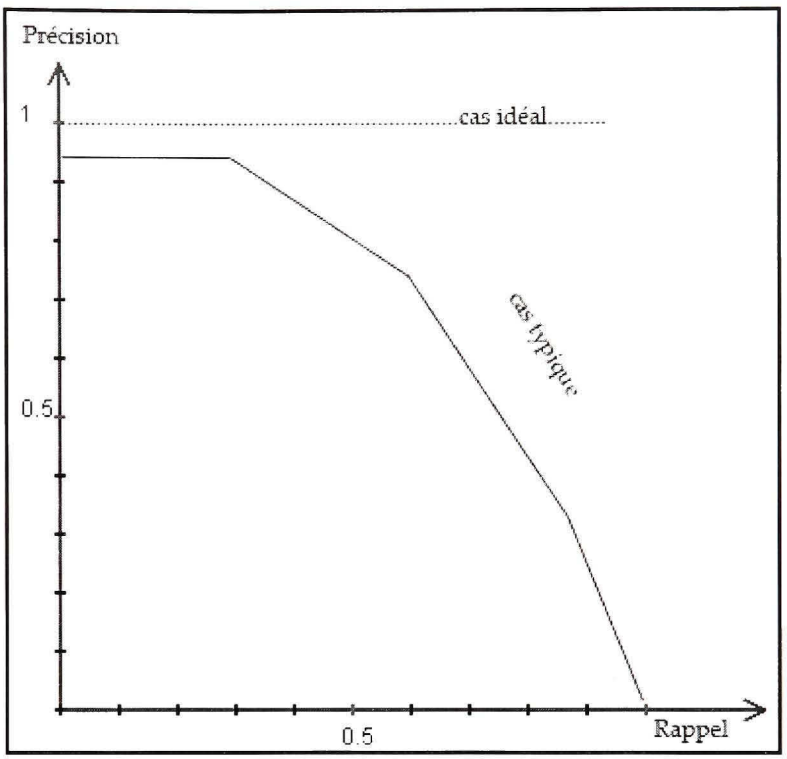

Figure 2.3 Cas idéal et cas typique d'une courbe rappel/précision.

Dans la littérature, on parle de la courbe rappel/précision interpolée où la précision est calculée pour des valeurs standardisées du rappel de 0 à 100% par pas de 10%. Pour notre étude, la courbe rappel/précision est obtenue en moyennant les valeurs des précisions pour chaque niveau i du rappel pour tous les mots. De même, cette courbe est tracée pour chaque mot clé séparément en se basant sur le même principe.

2.4.4 Les courbes ROC

Le graphe ROC (Receiver Operating Characteristics) est une technique permettant de visualiser et d'analyser les performances des modèles de classification afin de les comparer et les classifier selon leurs performances.

2.4.4.1 L'espace du graphe ROC

Dans cette section, on présentera l'espace des courbes ROC en se basant sur l'article de (Fawcett 2006). Comme représenté par la Figure 2.4, le graphe ROC est une courbe où l'abscisse représente le taux de Faux Positif et l'ordonnée représente le taux de Vrai Positif.

La ligne diagonale (y = x) désigne une stratégie de prédictions aléatoire. En effet, si la performance d'un modèle de classification est représentée sur cette ligne, cela signifie qu'il prédit le même nombre de Vrai Positif que de Faux Positif.

La région au-dessous de cette ligne représente le modèle de classification qui a une performance plus mauvaise que celle d'un modèle aléatoire puisqu'il produit plus de Faux Positif que de Vrai Positif. Cependant, des informations utiles peuvent êtres dégagées de cette situation en se basant sur le fait que la courbe ROC est symétrique par rapport à la diagonale (Flach and Wu 2003).

La performance d'un modèle de classification située dans la région de performance conservative montre que ce modèle génère peu de Faux Positif. Le point (0,0) représente la stratégie de ne jamais classer une instance comme positive mais toujours comme négative. Dans ce cas, le modèle ne classe aucun Vrai Positif et aucun Faux Positif. La stratégie inverse est représentée par le point (1,1) où le modèle ne classe aucune instance comme négative.

La performance d'un modèle de classification située dans la région de performance libérale signifie que ce dernier possède un bon taux de Vrai Positif mais il commet aussi quelques Faux Positif.

Finalement, la performance optimale est représentée par le point (0,1) qui signifie que le modèle classe correctement toutes les instances.

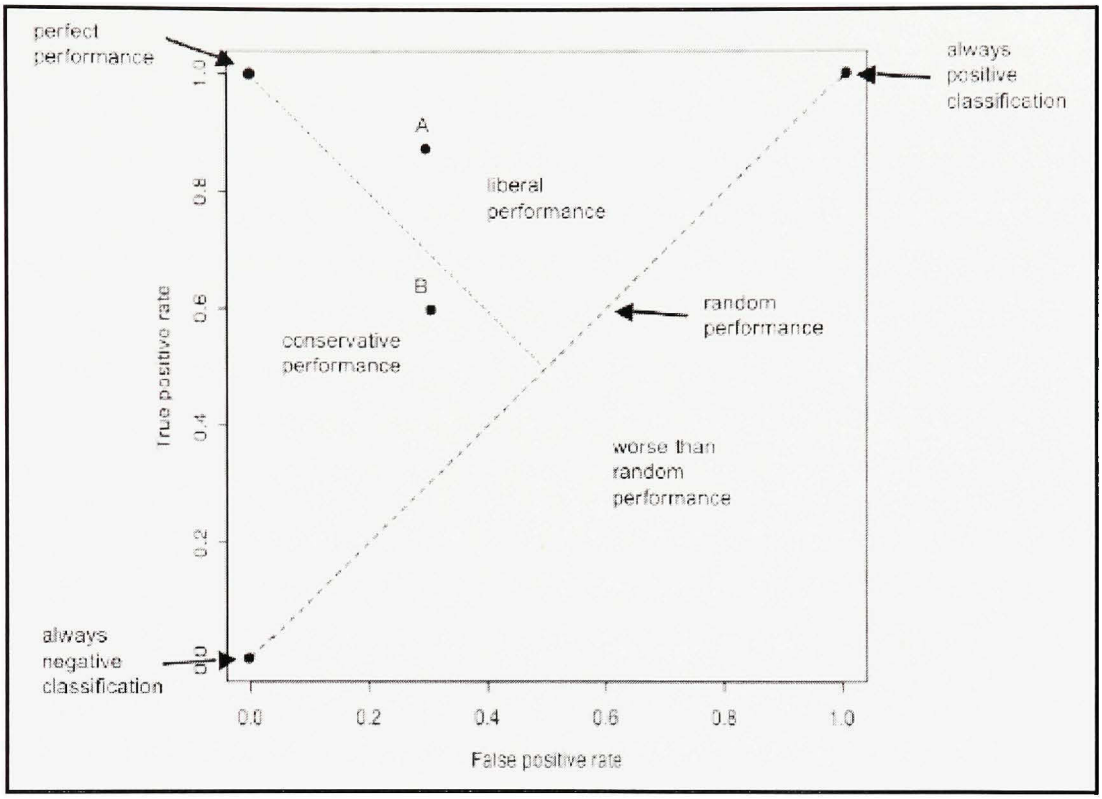

Figure 2.4 Le graphe ROC.
Tirée de Hamel (2008, p. 7)

2.4.4.2 Utilisation des courbes ROC

Comme mentionné dans le paragraphe précédent, la courbe ROC permet la représentation du taux de Faux Positif en fonction du taux de Vrai Positif. D'autres représentations peuvent être utiles pour l'évaluation d'un système de détection de mots clés. En effet, dans le cas d'un tel système on peut représenter le taux de Faux Négatif en fonction du taux de Faux positif.

En outre, la courbe ROC sert aussi à représenter le taux de Faux Positif en fonction du taux de Vrai négatif. Comme on peut le remarquer, ces courbes permettent d'analyser le comportement d'un système de détection de mots clés suivant différents axes.

2.5 Corpus de données

Le contexte de la reconnaissance automatique de la parole spontanée est un contexte très particulier dans ce domaine. Ainsi, on ne dispose d'aucun corpus préalable permettant d'estimer la performance de reconnaissance à laquelle on pourrait s'attendre. On a décidé alors de construire notre propre corpus d'annotations vocales. Pour réaliser cette tâche, on a fait appel à 15 locuteurs (3 femmes et 12 hommes), dont 12 participants parlent le français comme langue maternelle. Parmi l'ensemble des participants, 10 personnes parlent le français québécois et 3 personnes parlent le français international[7].

Le corpus de données textes employé pour l'annotation est le corpus des lois du Canada. Ce dernier est fourni par Irosoft pour une utilisation restreinte pour la recherche.

Avant de procéder à cette expérience, un formulaire de consentement est remis à chaque participant. Ce formulaire a pour but de donner une idée générale de la nature de la recherche et de ce qu'entraine la participation des locuteurs. Une copie de ce formulaire est jointe à ce document (Voir annexe VI, p. 140). L'expérience a eu lieu dans un environnement de bureau calme, et le signal est acquis par un microphone à réduction de bruit (livré avec Dragon NaturallySpeaking). Tous les fichiers sont enregistrés avec une fréquence d'échantillonnage de 44,1 kHz, et les amplitudes sont codées sur 16 bits avec un seul canal d'enregistrement. Chaque séance d'enregistrement est d'une durée d'une heure. Durant cette séance, le locuteur doit choisir un document, à partir du corpus texte, et le lire. Durant la lecture, le locuteur peut choisir n'importe quelles zones de texte et l'annoter vocalement d'une façon naturelle (parole continue et spontanée). L'annotation vocale ainsi que le texte annoté, sont enregistrés et ajoutés à la base de données. Le locuteur répète ces opérations en choisissant d'autres documents jusqu'à la fin de la séance.

---

[7] Langue française excluant les régionalismes lexicaux et syntaxiques.

Comme résultat, on a obtenu 653 annotations vocales, pour une durée totale de 3 heures. La durée moyenne d'une annotation est de 16 secondes. La phase de l'annotation vocale est suivie par une étape de raffinement qui consiste à éliminer les fichiers corrompus. Par conséquent, le nombre final des annotations est passé à 617 annotations. Il est à noter qu'on a obtenu peu de donnée mais ça correspond à la réalité applicative.

2.6 Processus expérimental

L'indexation d'annotations vocales est basée, dans notre étude, sur le résultat de la transcription obtenue à partir d'un moteur de reconnaissance automatique de la parole continue basée sur un grand vocabulaire. Ainsi, on a fixé les objectifs suivants :

- Évaluer et comparer la performance de la reconnaissance vocale de deux moteurs de reconnaissance automatique de la parole continue. Les deux moteurs qu'on a choisis sont celui de Dragon NaturallySpeaking et celui de Microsoft livré avec le système d'exploitation Vista. Pour accomplir cette tâche, on s'est basé sur le taux de reconnaissance du système de RAP.
- Analyser les résultats de la transcription des deux moteurs afin de déterminer les sources d'erreurs et les causes possibles de la mauvaise transcription.
- Comparer et analyser la performance de détection de mots clés par les deux moteurs. Pour atteindre cet objectif, on a utilisé le rappel, la précision et la F-mesure comme métriques d'évaluation.
- Étudier la relation entre la bonne détection des mots clés et les fausses alarmes. Dans ce cas, on a utilisé les courbes ROC.
- Étudier l'influence de la performance de la reconnaissance vocale sur celle de l'indexation vocale.

Afin d'atteindre ces objectifs, il est nécessaire de passer par plusieurs étapes comme la transcription, l'indexation, le calcul des métriques, etc. La Figure 2.5 est une schématisation des étapes depuis l'annotation vocale par un locuteur jusqu'à l'analyse des données.

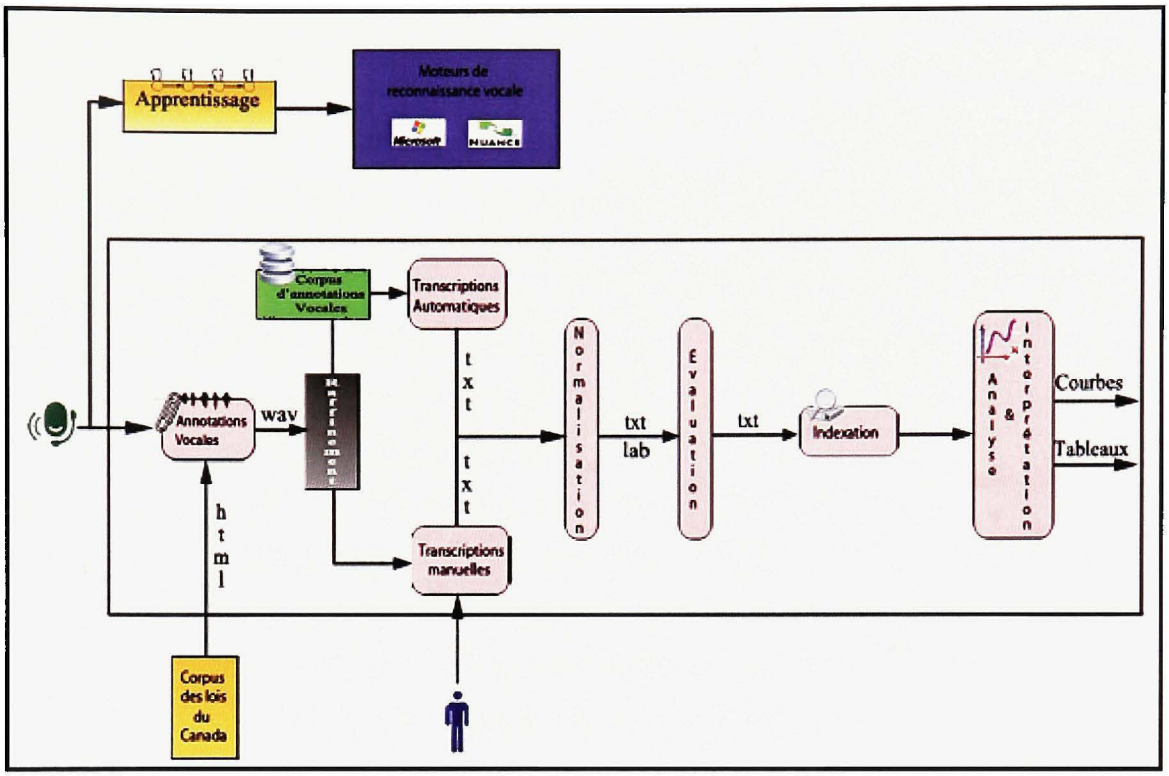

Figure 2.5 Processus expérimental.

2.6.1 Annotation vocale

L'annotation vocale consiste à lire un document et de l'annoter vocalement comme expliqué dans la section 2.5. Cette étape a comme premier objectif la construction d'un corpus d'annotations vocales pour tester les performances d'un système de reconnaissance automatique de la parole. Toutefois, il est possible d'annoter un document pour réaliser simplement la transcription des annotations obtenues, et par la suite l'indexation de ces dernières.

2.6.2 Apprentissage

Avant de commencer la séance de l'enregistrement, chaque participant doit passer par une phase d'apprentissage d'une durée de 20 minutes. Cette étape concerne la création d'un profil pour le locuteur, puis l'entraînement des deux moteurs de reconnaissance de la parole.

L'apprentissage se fait en lisant à voix haute quelques paragraphes afin que le moteur adapte ses modèles à la voix du locuteur. Les modèles acoustiques entraînés vont être utilisés par la suite dans la phase de transcription automatique des annotations.

2.6.3 Transcription

On distingue deux types de transcription :

- La transcription manuelle : chaque annotation vocale est transcrite manuellement par un humain. Ceci consiste à écouter l'annotation vocale et écrire, dans un fichier texte, ce qui a été dit. On a essayé de transcrire le plus fidèlement possible les annotations vocales. Toutefois, le contexte est parfois un peu flou (annotation très courte) ou l'expression n'est pas trop claire, dans ce cas il est difficile de déterminer le mot exact. En outre, on a remarqué que parfois les locuteurs utilisaient des termes anglophones ou des termes québécois qui n'appartiennent pas à la langue française. Ces termes sont considérés comme des mots hors vocabulaire.
- La transcription automatique : chaque annotation vocale est transcrite automatiquement par les deux moteurs de reconnaissance vocale. Chaque moteur transcrit l'annotation deux fois : la première fois avec un profil sans apprentissage et la deuxième fois avec le profil de chaque locuteur. Par la suite, on pourra évaluer et comparer la performance des deux moteurs selon le profil sans apprentissage et celui avec apprentissage.

Par conséquent, on a obtenu pour chaque annotation vocale un fichier contenant sa transcription manuelle et quatre fichiers résultants de la transcription automatique des deux moteurs (voir Figure 2.6).

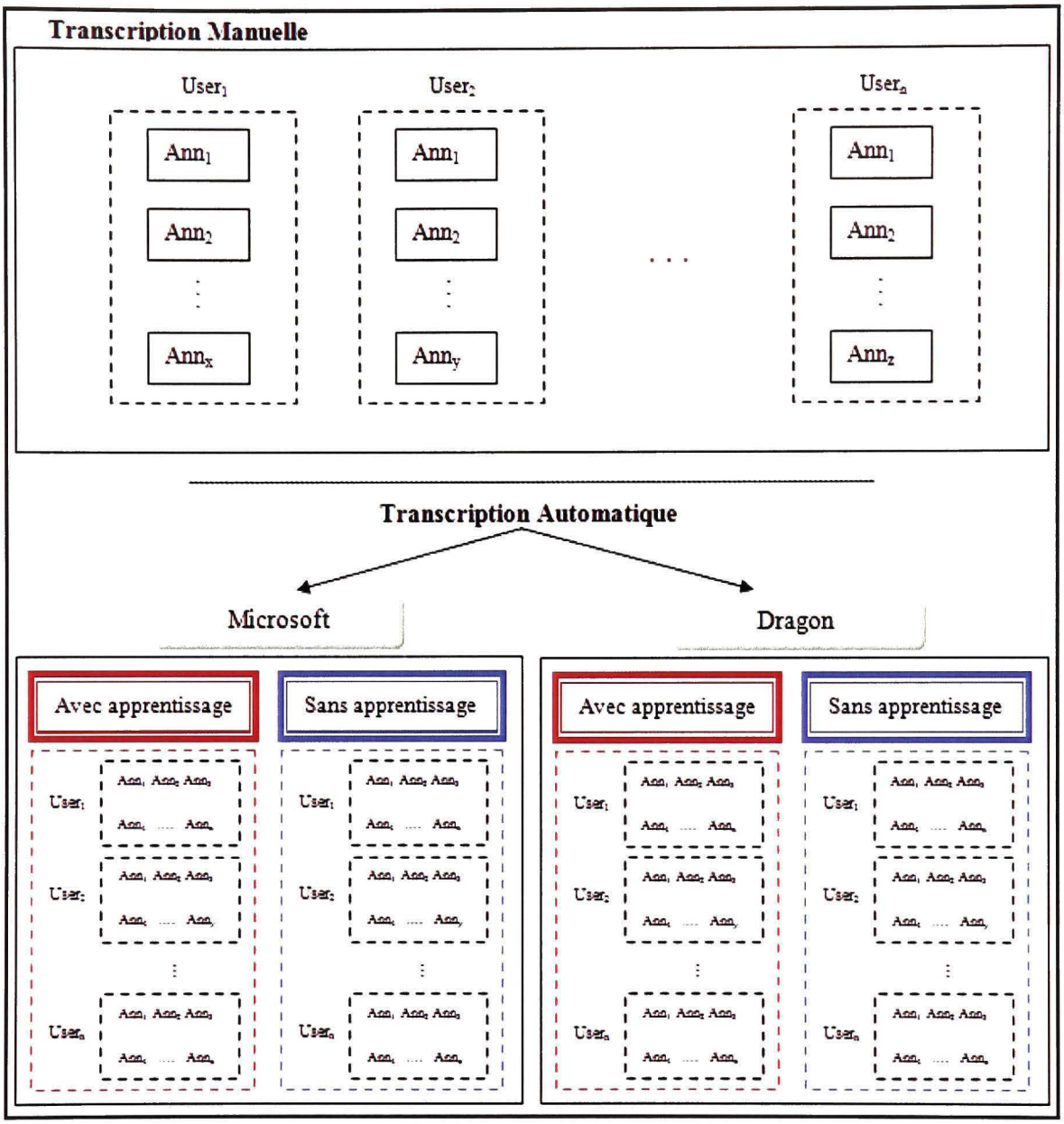

Figure 2.6 Organisation des fichiers résultants de la transcription.

2.6.4 Normalisation

La phase de normalisation consiste à préparer tous les fichiers obtenus, par les transcriptions manuelles et automatiques, à la phase de l'évaluation. En effet, chaque fichier doit subir les opérations suivantes :

- Convertir tous les nombres en lettres (exemple 2000 devient : deux milles).
- Éliminer les caractères spéciaux et les ponctuations.
- Éliminer le résultat de la prononciation des commandes vocales (cas de Dragon NaturallySpeaking).
- Convertir tous les caractères en minuscules.
- Mettre chaque mot du fichier en une ligne.

2.6.5 Évaluation

La phase d'évaluation consiste à comparer, en se basant sur l'algorithme de comparaison dynamique, les fichiers transcrits par un humain et les fichiers transcrits automatiquement. Ceci revient à calculer le taux d'erreur en utilisant l'outil *HResults* fourni avec le moteur de reconnaissance vocale HTK. Dans cette étape, le taux d'erreur est calculé pour chaque fichier séparément. Dans la phase d'analyse, la moyenne pondérée des taux d'erreur est calculée par locuteur, par moteur et profil et par dialecte. On a choisi la moyenne pondérée parce qu'elle est plus significative que la moyenne arithmétique, puisqu'elle prend en considération le nombre de mots dans le fichier.

2.6.6 Indexation

La phase d'indexation consiste d'une part à identifier les fichiers sonores qui contiennent le mot clé recherché, et d'autre part à évaluer la performance de cette recherche. Ainsi, on a calculé la matrice de confusion pour chaque mot clé, et à partir d'elle on a calculé le rappel, la précision et la F-mesure. Ces trois métriques seront par la suite recalculées pour tous les locuteurs ou bien pour chaque locuteur selon le besoin.

2.6.7 Analyse

L'analyse et l'interprétation des données nécessitent une organisation des valeurs des différentes métriques calculées précédemment. Ainsi, on regroupera les résultats dans des

tableaux selon des axes d'analyse bien déterminés. En outre, pour faciliter davantage l'analyse, on présentera les résultats par des courbes selon plusieurs critères.

2.7 Conclusion

Dans ce chapitre on a exposé différents systèmes de reconnaissance vocale. Ainsi, on a fixé des critères de sélection permettant d'identifier les moteurs de reconnaissance vocale qui répondent le mieux à nos conditions expérimentales. On a décidé d'exploiter le moteur de reconnaissance vocale de Dragon NaturallySpeaking et celui de Microsoft livré avec le système d'exploitation Vista pour la réalisation de notre étude. La deuxième partie de ce chapitre a été consacré à la présentation des différentes métriques utilisées dans le cadre de cette étude pour évaluer et comparer la performance des deux moteurs de reconnaissance vocale. On a terminé ce chapitre par la description de notre processus expérimental. Ce dernier englobe toutes les étapes nécessaires pour l'analyse des performances de transcription et d'indexation des deux moteurs. Toutefois, on n'a pas abordé les aspects pratiques qui feront l'objet du chapitre suivant.

CHAPITRE 3

PRÉSENTATION DU SYSTÈME

Ce chapitre couvre les différents éléments qu'on a utilisés pour la mise en place d'un système d'indexation d'annotations vocales. Ce système est développé d'une part pour évaluer et comparer la performance de reconnaissance vocale et celle de l'indexation vocale de deux moteurs de reconnaissance automatique de la parole. D'autre part, pour permettre à l'utilisateur de faire la recherche de mots clés dans des fichiers sonores. En outre, il peut être utilisé comme un outil de démonstration des fonctionnalités de la technologie de reconnaissance vocale.

On présentera dans un premier temps la base de données qu'on a construit pour contenir les données utilisées par notre application. De même, on exposera le modèle relationnel afin de décrire les différentes tables et leurs attributs ainsi que les interactions entre elles. Dans un deuxième temps, on abordera les éléments externes indispensables au bon fonctionnement de notre application. Ainsi, on présentera l'interface de programmation d'application de reconnaissance vocale (SAPI), et l'API Java pour les applications vocales (JSAPI), qui forment les éléments de base de communication de notre système avec les moteurs de reconnaissance vocale. On exposera par la suite les deux types de grammaires supportées par JSAPI. La dernière partie de ce chapitre sera concernée à décrire les principaux éléments et fonctionnalités de l'application développée.

3.1 Base de données

L'étude préalable de notre projet de développement a dévoilé la nécessité de l'implémentation d'une base de données pour le stockage des données d'une façon structurée et avec moins de redondance. En effet, l'analyse des performances des deux moteurs de reconnaissance vocale nécessite la manipulation d'une grande quantité de données. En outre, ces derniers sont diversifiés et proviennent de plusieurs sources. Ainsi, tout au long de notre processus d'expérimentation, la base de données est alimentée de différents types

d'information telles que les références des annotations vocales, les transcriptions, la durée d'annotation, les valeurs des mesures, etc. Ceci permettra un accès simple et rapide à l'information tout en assurant la validité et la cohérence des données. L'utilisation d'une base de données présente un autre avantage dans le cadre de notre étude. En effet, elle permet l'automatisation de certaines tâches comme la transcription automatique, l'évaluation et l'indexation de toutes les annotations vocales. Ainsi, il suffira de cliquer sur un seul bouton pour déclencher l'exécution de ces tâches.

Pour la création et la gestion de notre base de données, on a utilisé le système de gestion de base de données HSQLDB (HSQLDB 2010). Ce dernier est écrit entièrement en Java et il offre un moteur de base de données rapide et léger. En outre, ce système est disponible sous une licence BSD[8] et il est facile de l'intégrer à l'application.

Comme représentée par le modèle relationnel (voir Figure 3.1), notre base de données contient 15 tables. On peut remarquer que ces tables stockent toutes les informations indispensables aux calculs des métriques, et par la suite l'analyse des résultats.

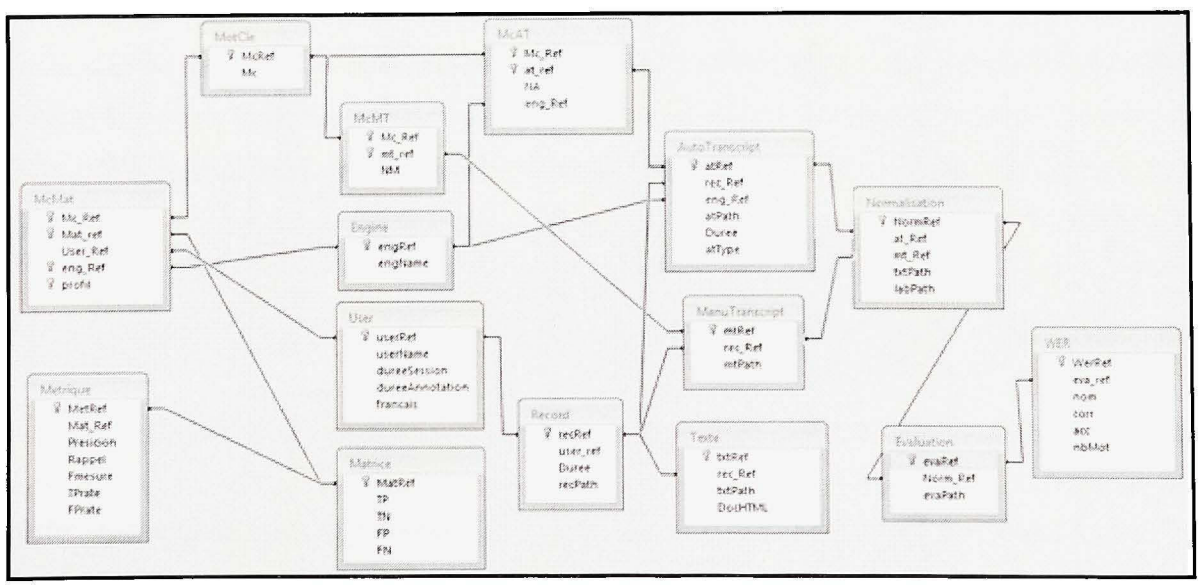

Figure 3.1 Schéma relationnel.

[8]Berkeley Software Distribution: une licence libre utilisée pour la distribution de logiciels.

3.2 SAPI

L'interface de programmation d'application de reconnaissance vocale (Speech Application Programming Interface, SAPI) est un ensemble de classes et fonctions développées par Microsoft, qui permet d'inclure des fonctionnalités de la technologie de la reconnaissance vocale à une application informatique. En effet, le SAPI joue le rôle d'intermédiaire, en fournissant une couche d'interfaçage, entre le moteur de reconnaissance vocale (et/ou le synthétiseur) et l'application. Différentes versions ont été développées, la plus récente est SAPI 5.3 (Microsoft 2010) qui présente une nouvelle architecture[9]. En effet, contrairement aux versions précédentes qui permettent une interaction directe entre l'application et le moteur de reconnaissance, le SAPI 5 communique avec l'application par l'intermédiaire d'un composant (SAPI Runtime) qui propose un ensemble d'interfaces pour assurer cette opération. Pour éliminer la dépendance de l'application à un moteur de reconnaissance particulier, le SAPI passe par une composante appelée Device-Driver Interface (DDI) pour interagir avec le moteur concerné. La Figure 3.2 représente l'architecture de base de SAPI 5.

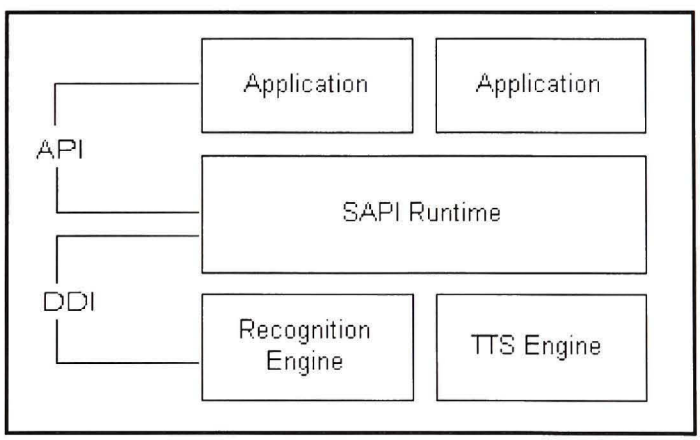

Figure 3.2 Architecture de base de SAPI 5.

Notre application doit communiquer avec deux moteurs de reconnaissance vocale qui se basent sur deux architectures différentes. En effet, le moteur de Dragon NaturallySpeaking

[9] Les quatre premières versions sont développées par Microsoft en se basant sur les travaux de CMU (Carnegie Mellon University).

est compatible avec SAPI 4, alors que celui de Microsoft est basé sur SAPI 5. Ainsi, on a alors décidé d'utiliser une autre API nommé JSAPI (Java speech API) (SUN Microsystems 1998) qui va assurer la communication entre notre application et les deux autres SAPI (SAPI 4 et SAPI5) et par la suite avec les deux moteurs de reconnaissance. Contrairement à SAPI, JSAPI n'est qu'une définition d'interfaces permettant la communication avec les moteurs de reconnaissance vocale et leurs intégrations à des applications développées en langage Java. Son utilisation nécessite l'implémentation et le développement des interfaces déjà définis. Ainsi, différentes compagnies, tels que sourceforce, IBM, Festival, Cloud Garden, etc., ont implémenté, généralement en partie, ce standard.

3.3 Grammaire

Le contrôle de l'activité d'un moteur de reconnaissance automatique de la parole par une application commence par le contrôle de sa grammaire. Cette dernière représente l'ensemble de tous les mots, ou expressions d'une façon plus générale, qui composent le dictionnaire et qui peuvent être prononcés par le locuteur. En plus, il définit la manière dont ces mots doivent être prononcés et les règles d'utilisation de la grammaire appropriée. D'un point de vue technique, une grammaire est un objet de JSAPI qui constitue une composante principale du processus de reconnaissance automatique de la parole. JSAPI supporte deux types de grammaire : grammaire de dictée (*dictation grammar*) et grammaire régulière (*regular grammar*).

3.3.1 Grammaire de dictée

Ce type de grammaire est adopté par les applications vocales utilisant un grand vocabulaire comme les applications de dictée vocale. En outre, la grammaire de dictée n'impose pas beaucoup de contraintes sur ce qui peut être prononcé. Cependant, cette flexibilité nécessite une grande capacité de traitement et tend à générer plus d'erreurs par rapport à l'autre type de grammaire. En plus, la grammaire de dictée gère beaucoup plus de mots entraînés sur de grands corpus de données acoustiques et textuelles en se basant sur des modèles statistiques.

La construction de ce type de grammaire est une opération très complexe et souvent ne se fait pas par le développeur de l'application vocale. Les systèmes de reconnaissance automatique de la parole qui supporte la grammaire de dictée l'incluent à leurs moteurs de reconnaissance. Toutefois, les développeurs peuvent optimiser la grammaire par l'ajout de nouveaux mots.

3.3.2 Grammaire régulière

La grammaire régulière ou encore la grammaire de commande et de contrôle (*command and control grammar*) est une représentation textuelle des mots qui peuvent être prononcés. Cette grammaire est utilisée principalement par les applications de commande vocale basées sur des moteurs de reconnaissance vocale qui ne supportent pas un grand vocabulaire. Contrairement à la grammaire de dictée qui supporte les deux versions de SAPI, la grammaire régulière est supportée seulement par SAPI 5.

L'implémentation de la grammaire régulière nécessite la définition de certaines règles et l'utilisation d'une syntaxe bien définie. Java Speech Grammar Format (JSGF) représente les spécifications de cette grammaire en adaptant le style et les conventions du langage de programmation Java. Pour la description détaillée de JSGF on peut se référer à (Sun 1998).

La grammaire régulière est utilisée principalement pour les systèmes de commande vocale, et par la suite ne fera pas l'objet de notre étude. Par contre, la grammaire de dictée correspond parfaitement à nos conditions d'expérimentations. Ainsi, elle sera exploitée pour le développement de notre système.

3.4 Application

On a développé un système, en langage de programmation Java, qui englobe les différentes étapes de notre processus expérimental. Ce système permet, entre autres, l'évaluation et la comparaison des performances de transcription et d'indexation de deux moteurs de reconnaissance de la parole. Comme représenté dans la section 2.5, notre processus

expérimental commence par la construction d'un corpus d'annotation vocale en utilisant des textes extraits du corpus des lois du Canada. Après leur raffinement, les annotations vocales sont transcrites par un humain puis par les deux moteurs de reconnaissance vocale selon un profil sans apprentissage et un autre avec apprentissage. Ensuite, les différentes transcriptions ont été normalisées afin de calculer, par la suite, les métriques liées au taux de reconnaissance en utilisant l'outil HResults. L'étape d'indexation permet notamment de calculer des métriques pour l'évaluation des performances de détection de mots clés en se basant sur les matrices de contingences. La dernière étape consiste à organiser les résultats dans des tableaux et tracer des courbes pour faciliter l'analyse et l'interprétation de ces résultats selon différents axes d'analyse. Tout au long de ce processus, la base de données est alimentée par les données et les résultats obtenus. La communication de notre système avec les deux moteurs de reconnaissance est assurée par JSAPI qui interagit avec SAPI 4 (pour Dragon NaturallySpeaking) et SAPI 5 (pour Microsoft). La Figure 3.3 représente l'architecture de notre système.

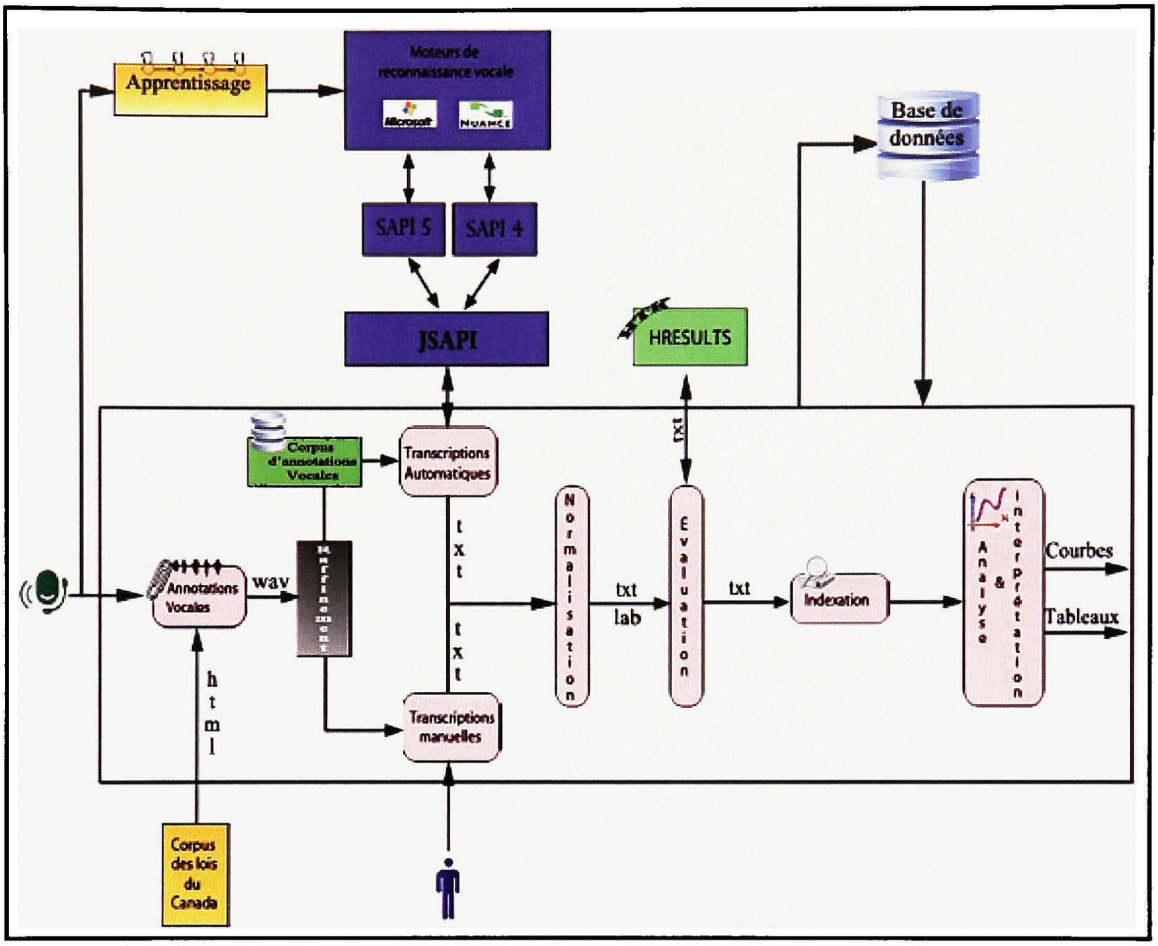

Figure 3.3 Architecture du système développé.

3.4.1 Fenêtre principale

Afin de bien gérer les modules de notre application, on a crée une fenêtre principale souvent connue sous le nom de document à multiple interface (Multiple Document Interface, MDI). Ceci permet de mieux gérer les autres fenêtres enfants de l'application tout en permettant l'échange de données entre ces fenêtres. En outre, la fenêtre MDI inclut une barre de menu facilitant ainsi la navigation dans l'application, et une barre d'état affichant des informations jugées importantes. La Figure 3.4 montre la fenêtre MDI de notre application.

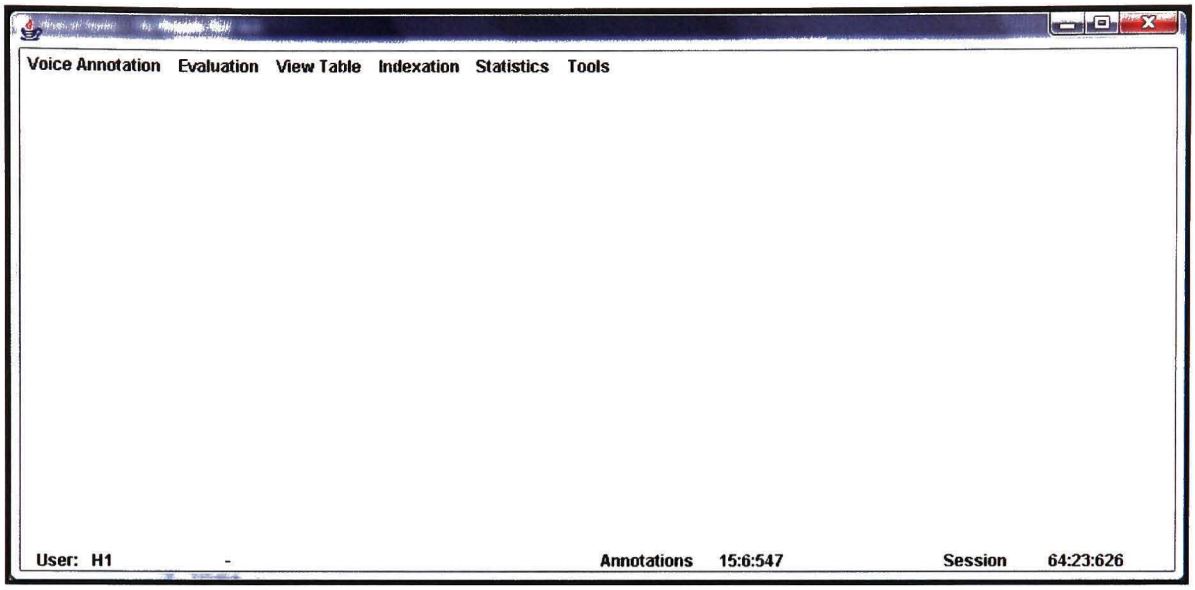

Figure 3.4 Fenêtre MDI.

3.4.2 Fonctionnalités principales

La fenêtre principale de notre application regroupe un ensemble d'outils et de fonctionnalités qui peuvent être utilisés indépendamment de notre processus expérimental. En d'autres termes, ces outils peuvent être utilisés pour exécuter des tâches générales telles que l'enregistrement de séquences audio, lecture de fichier audio, reconnaissance vocale, etc. Comme représentée par la Figure 3.5, cette interface graphique est divisée en trois parties.

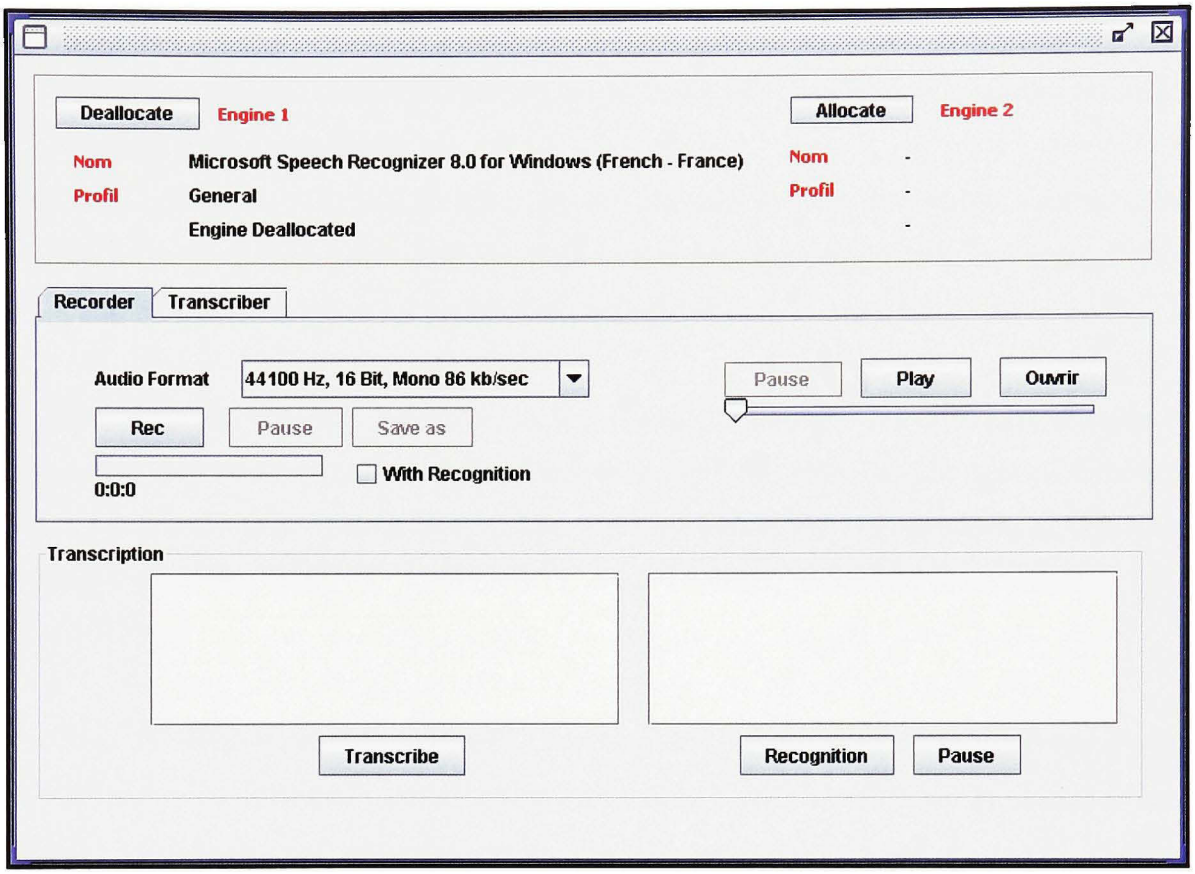

Figure 3.5 Fonctionnalités principales du système.

La première partie, permet l'allocation de ressources pour deux moteurs de reconnaissance vocale. En plus, le nom, le profil et l'état de chaque moteur sont affichés dans cette zone. À un instant donné, un moteur doit être dans un seul état des quatre états suivants :

- ALLOCATED : cet état signifie que les ressources (processeur, mémoire et disque dur) sont réservées pour le moteur concerné. En outre, un accès exclusif au microphone et au haut parleur est accordé au moteur.
- DEALLOCATED : dans le cas où l'application n'a plus besoin du moteur, ce dernier passe à l'état Deallocated. Cela signifie que toutes les ressources accordées au moteur seront libérées pour être utilisées par d'autres applications.

- ALLOCATING_RESOURCES : cet état signifie que les ressources sont en train d'être allouées au moteur. Cette opération est un peu lente, elle peut prendre quelques secondes ou même quelques minutes.
- DEALLOCATING_RESOURCES : cet état signifie que les ressources accordées au moteur sont en train d'être libérées. Pour le bon déroulement de cette opération, il faut s'assurer que le moteur n'est plus utilisé par l'application.

Il existe d'autres sous-états qui facilitent la synchronisation entre les différents moteurs et permettent une bonne utilisation de leurs fonctionnalités. En effet, dans le cas où le moteur est à l'état Allocated, il peut passer dans l'un ou l'autre de ces sous-états : PAUSED, RESUMED, PROCESSING, LISTENING, SUSPENDED, COMMITTED, FOCUS_ON, FOCUS_OFF.

En cliquant sur le bouton ALLOCATE, une nouvelle fenêtre s'affiche et l'utilisateur peut choisir parmi les différents moteurs de reconnaissance vocale installés sur la machine (voir Figure 3.6). Ainsi, les ressources sont allouées au moteur choisi et seront libérées automatiquement à la fin de la transcription d'un fichier audio ou l'arrêt de la reconnaissance en temps réel. L'utilisateur peut utiliser deux moteurs de reconnaissance en même temps.

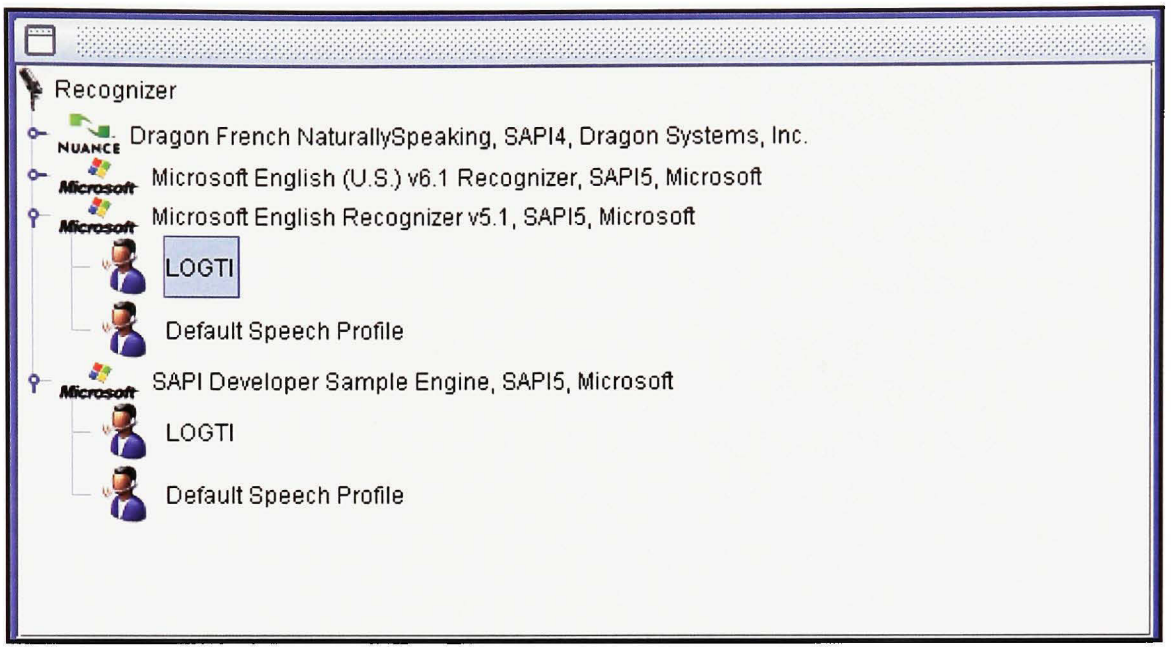

Figure 3.6 Choix du moteur de reconnaissance vocale et du profil.

La deuxième partie permet d'enregistrer des séquences audio et de les sauvegarder sur le disque dur. L'utilisateur peut choisir entre différentes combinaisons de paramètres. En outre, un outil de lecture de fichiers audio est ajouté dans cette zone graphique. Il peut être utilisé notamment pour écouter les enregistrements avant de les sauvegarder sur le disque dur. Pour nos enregistrements, on a fixé la fréquence d'échantillonnage à 44,1 kHz et les amplitudes sont codées sur 16 bits avec un seul canal d'enregistrement. Cependant, les systèmes de RAP utilisés déciment les données acoustiques à des fréquences d'échantillonnages qui leurs sont propres. Ainsi, une conversion doit avoir lieu pour diminuer la fréquence de 44.1 kHz à 16 kHz.

La dernière partie de cette fenêtre est réservée à la transcription des fichiers audio. Ainsi, l'utilisateur doit identifier un moteur de reconnaissance (ou deux au maximum) puis choisir le fichier audio à transcrire. Le résultat de la transcription est affiché dans les zones textes créés à cet effet. De même, l'utilisateur peut effectuer la reconnaissance de la parole en temps réel.

3.4.3 Annotation vocale

Pour la construction de notre corpus vocal, on a développé une interface graphique qui sert à l'affichage des documents à partir du corpus texte. En lisant le document, l'utilisateur peut choisir n'importe quelle zone texte pour l'annoter vocalement. Pour ce faire, il suffit de sélectionner la partie à annoter pour déclencher l'enregistrement de la parole (voir Figure 3.7). Pour arrêter l'enregistrement et terminer l'annotation, il suffit de désélectionner le texte. Par conséquent, une fenêtre graphique est affichée pour indiquer à l'utilisateur s'il accepte ou non la sauvegarde de cette annotation. En cas d'acceptation, l'annotation et le texte annoté sont sauvegardés et toutes les données liées à cette opération sont ajoutées à la base de données. La barre d'état en bas de la fenêtre MDI indique la durée de la session ainsi que la durée totale des annotations pour l'utilisateur courant.

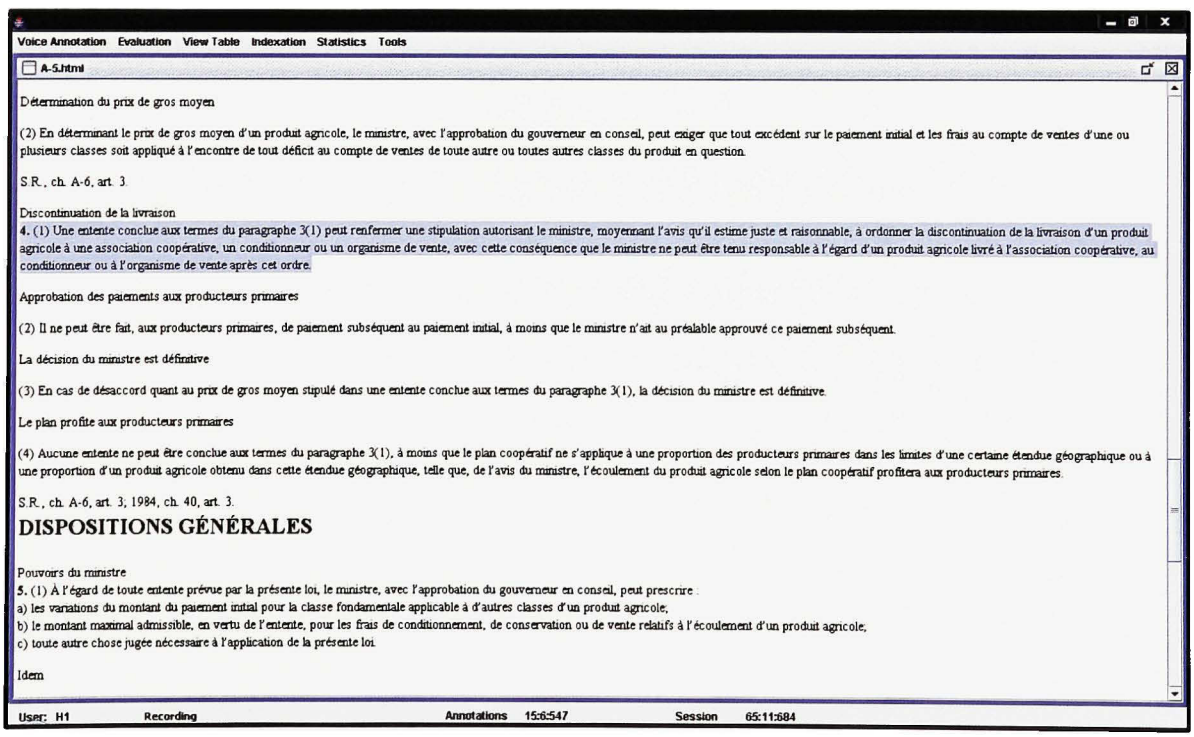

Figure 3.7 Annotation vocale.

3.4.4 Gestion de la base de données

Le module de gestion de la base de données permet principalement la consultation des données de la base. Ainsi, les données sont affichées dynamiquement dans des tables et l'utilisateur peut consulter les informations et supprimer les lignes de la table concernée. Vue la grande quantité de données à ajouter, les nouvelles entrées sont ajoutées à la table par programmation. Par exemple, une fonction est créée pour ajouter des mots clés à la base de données à partir d'un fichier texte.

3.4.5 Modules d'évaluation et analyse de données

Comme pour la transcription automatique des annotations vocales, l'évaluation des performances de reconnaissance peut se faire par fichier ou pour tous les fichiers de la base de données. Ainsi, pour chaque fichier, les métriques liées au taux de reconnaissance sont calculées en comparant dynamiquement la transcription originale et la transcription obtenue par les moteurs de reconnaissance. De même, les différentes métriques d'évaluation de l'indexation sont calculées et ajoutées à la base de données. Plusieurs autres classes ont été implémentées dans le but de représenter les données dans des tableaux et par des courbes. La Figure 3.8 montre l'onglet contenant les mesures de taux de transcription selon différents axes d'analyses.

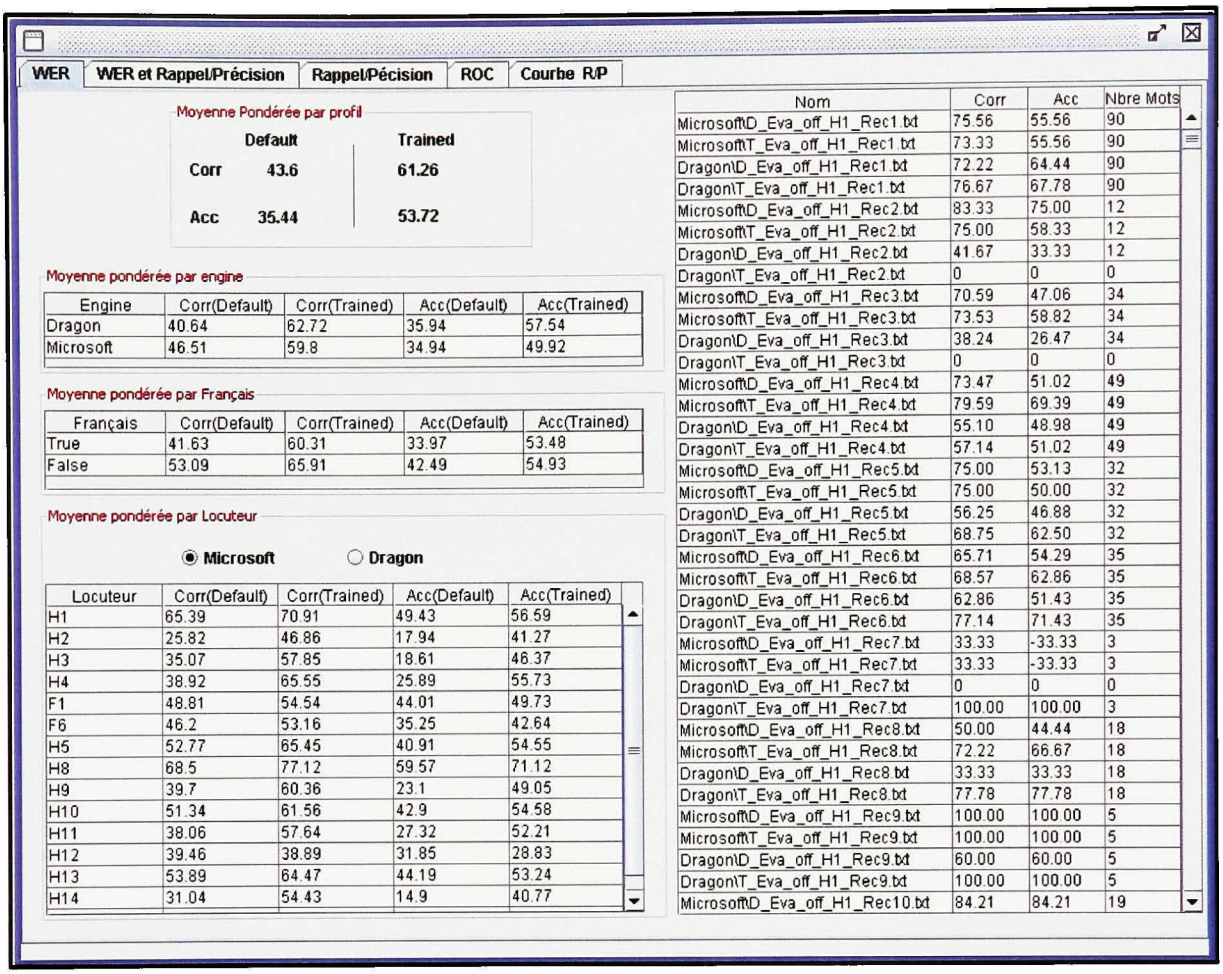

Figure 3.8 Génération des taux de reconnaissance vocale.

3.4.6 Recherche de mots clés

La phase d'indexation permet de calculer les différentes métriques indispensables à l'évaluation de la performance de détection de mots clés dans les fichiers sonores. Ces mesures forment d'une part la base d'un système de comparaison entre le moteur de reconnaissance Dragon et celui de Microsoft. D'autre part, elles servent à étudier l'influence de la performance de transcription automatique sur celle de l'indexation. L'application qu'on a développée permet de calculer ces métriques et de présenter les résultats sous forme de tableaux et de courbes afin de faciliter l'analyse et l'interprétation. En plus, elle inclut un

outil d'indexation permettant la recherche de mots clés dans les annotations vocales en se basant sur leur transcription automatique (voir Figure 3.9).

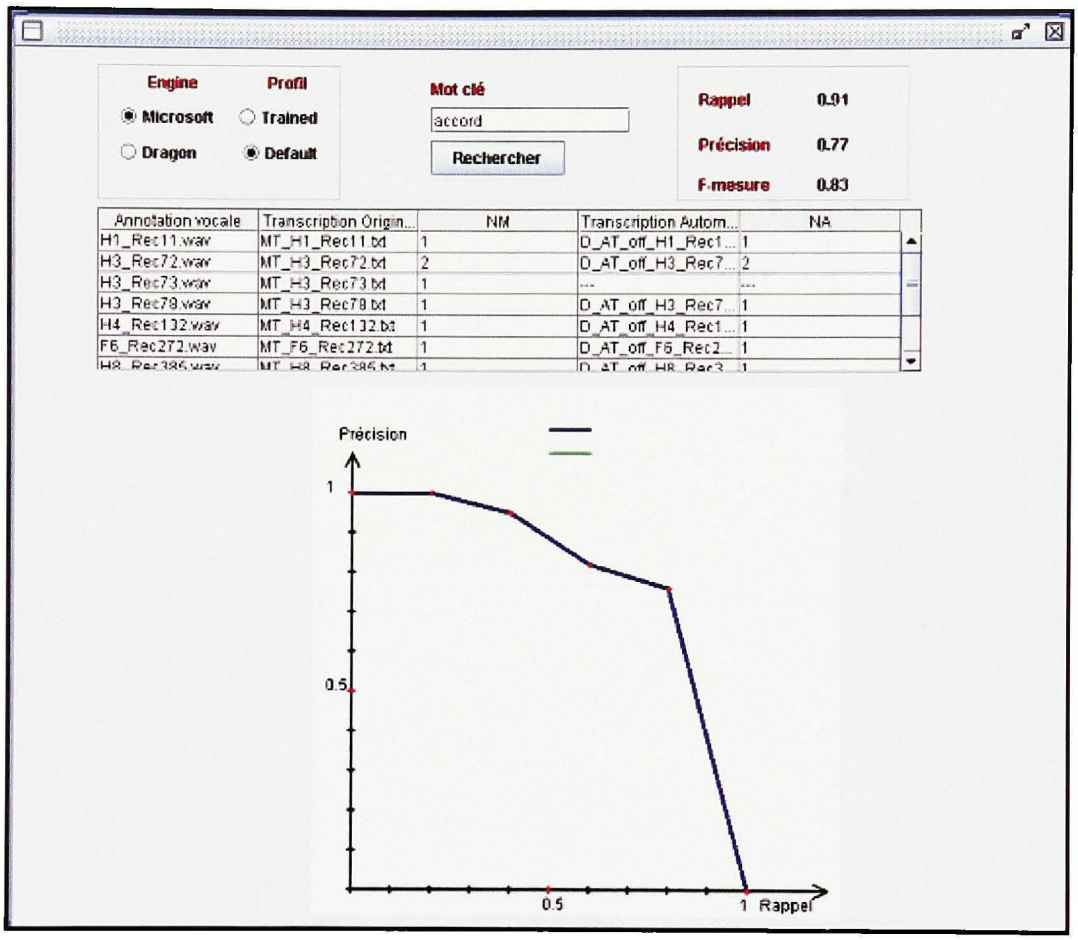

Figure 3.9 Recherche de mot clé.

L'utilisateur peut saisir n'importe quel mot clé dans la zone de texte et appuyer sur le bouton recherche (ou simplement sur la touche entrer) pour afficher toutes les annotations vocales contenant ce mot. De même, tous les fichiers des transcriptions originales et ceux des transcriptions automatiques sont affichés avec le nombre de mots clés dans chaque fichier.

Les fichiers de transcription automatique qui ne contiennent pas le mot clé ne sont pas affichés. Cependant, leurs noms sont remplacés par la chaîne de caractère « --- » indiquant que le mot clé est prononcé dans l'annotation vocale, mais non reconnu par le système. Le

résultat de la recherche est mis à jour à chaque nouveau choix de paramètres. Ces derniers permettent d'identifier le moteur de reconnaissance et le profil (*Default* ou *Trained*) sur lesquels la recherche sera réalisée. En cliquant sur le nom de fichier, celui-ci s'ouvre permettant ainsi à l'utilisateur de lire la transcription ou d'écouter l'annotation vocale.

Au cours de la saisie de mot clé, le champ de saisie propose à l'utilisateur le mot clé contenu dans la base de données et commençant par les caractères déjà saisis. Cet affichage dynamique tend à aider l'utilisateur à identifier le mot clé désiré, et rend cet outil plus convivial. Il est toutefois possible de chercher des mots clés non contenus dans la base de données. Dans ce cas, le système ajoute automatiquement le nouveau mot clé à la base de données et effectue la recherche sur l'ensemble des annotations vocales afin d'afficher le résultat.

Pour chaque mot clé, le rappel, la précision et la F-mesure sont affichés permettant ainsi de comparer la performance d'indexation pour chaque moteur de reconnaissance. En outre, le comportement du système pour cette requête est décrit par la courbe rappel/précision.

3.5 Conclusion

Dans ce chapitre, on a présenté l'application informatique qu'on a développée ainsi que les modules internes et externes qui la composent. Ainsi, on a décrit, en se basant sur le modèle relationnel, les tables de la base de données construite et l'interaction entre elles. On a exposé par la suite les interfaces API utilisées pour assurer la communication de l'application avec les moteurs de reconnaissance. Il est à noter que quelques fonctionnalités ont été utilisées seulement par le moteur de reconnaissance Microsoft. Ceci est dû principalement à l'incompatibilité de SAPI 4 (supporté par Dragon NaturallySpeaking) avec JSAPI.

Dans la dernière partie de ce chapitre, on a détaillé l'architecture de notre système et les différentes fonctionnalités qu'il présente. Ce dernier permet de faire exécuter deux moteurs de reconnaissance vocale en parallèle. Cependant, il est judicieux de mentionner que cette

technique n'a pas fonctionné pour le moteur de Dragon NaturallySpeaking en parallèle avec celui de Microsoft installé sur Vista. En effet, on n'a pas réussi à faire fonctionner le moteur de Dragon, par notre application, sur Vista malgré qu'il est bien installé sur ce système d'exploitation. Par conséquent, on a réalisé la transcription par le moteur de Dragon sur le système d'exploitation XP, alors que la transcription par le moteur de Microsoft est réalisée sur Vista. Toutefois, l'exécution de deux moteurs en parallèle peut se faire sur Vista si on utilise des moteurs de reconnaissance vocale qui traitent un vocabulaire anglais. De même, on peut utiliser les deux moteurs en parallèle sur Windows XP, sauf que le seul moteur de reconnaissance développé par Microsoft qui supporte le français est fourni avec le système d'exploitation Vista.

CHAPITRE 4

ANALYSE ET INTERPRÉTATION

Ce chapitre est entièrement consacré à la présentation et l'interprétation des résultats de nos expériences. Tout d'abord, on présentera les résultats de transcription et d'indexation vocale en employant tout le corpus vocal. On étudiera, par une première expérience, les résultats de transcription selon différents axes d'analyse afin de détecter les sources d'erreurs possibles. De même, les résultats d'indexation sont analysés afin de comparer les performances de détections de mots clés par les deux moteurs utilisés. Par la suite, on établira le lien entre les performances de transcription et celles d'indexation.

Ensuite, on étudiera les causes des mauvaises performances de transcriptions observées lors de la première expérience. On pose comme hypothèse que la parole spontanée est la source principale des mauvais taux de transcription.

On s'intéressera dans la dernière partie de ce chapitre à l'amélioration des performances de transcription par l'entraînement du modèle de langage. Ainsi, on essayera d'adapter le modèle de langage du moteur de reconnaissance de Dragon par différents types de textes.

4.1 Expérience 1

Dans cette première expérience, on a utilisé tout le corpus d'annotations vocales qu'on a construit antérieurement. Les annotations sont d'une part transcrites par un humain, et d'autre part par les deux moteurs de reconnaissance vocale. L'évaluation de la performance de reconnaissance se fait sur la base de ces transcriptions après leur normalisation. Au total, on a 2468 transcriptions automatiques générées par les deux moteurs de reconnaissance vocale pour le profil avec apprentissage et sans apprentissage. Le nombre total de mots dans les transcriptions par un humain est 18678 mots. Dans ce qui suit, on procédera à l'analyse et l'interprétation des résultats de la transcription et de l'indexation des annotations vocales.

4.1.1 Résultat de l'évaluation de la transcription automatique

Le résultat de l'évaluation (par annotation) de la transcription automatique (représenté en partie par le Tableau 4-1) montre que le Taux de Mots Correct (TMC) et le Taux de Précision (TP) varient entre 0% et 100.

Tableau 4-1 Exemples de taux de transcriptions par annotations

Annotation	TMC	TP	Nombre de Mots
Microsoft\D_H1-Rec1	75.56 %	55.56 %	90
Microsoft\T_H1-Rec1	73.33 %	55.56 %	90
Dragoon\D_H1-Rec1	72.22 %	64.44 %	90
Dragon\T_H1-Rec1	76.67 %	67.78 %	90
Microsoft\D_H1-Rec2	83.33 %	75.00 %	12
Microsoft\T_H1-Rec2	75.00 %	58.33 %	12
Dragoon\D_H1-Rec2	41.67 %	33.33 %	12
Dragoon\T_H1-Rec2	0 %	0 %	12
Microsoft\D_H1-Rec3	70.59 %	47.06 %	14
Microsoft\T_H1-Rec3	73.53 %	58.82 %	14
Dragoon\D_H1-Rec3	38.24 %	26.47 %	14
Dragoon\T_H1-Rec3	0 %	0 %	14
Microsoft\D_H1-Rec4	73.47 %	51.02 %	49
Microsoft\T_H1-Rec4	79.59 %	69.39 %	49
Dragoon\D_H1-Rec4	55.10 %	48.98 %	49
Dragoon\T_H1-Rec4	57.14 %	51.03 %	49
Microsoft\D_H1-Rec7	33.33 %	-33.33 %	3
Microsoft\T_H1-Rec7	33.33 %	-33.33 %	3
Dragoon\D_H1-Rec7	0 %	0 %	3
Dragon\T_H1-Rec7	100 %	100 %	3

On remarque que 1.9% des transcriptions automatiques sont 100% correctes avec une précision égale à 100%.

Exemple :

- Transcription manuelle : « Loi sur le mariage civil »
- Transcription automatique : « loi sur le mariage civil »

Toutefois, on remarque que la majorité des transcriptions automatiques, présentant des taux de 100%, contiennent seulement quelques mots (73% des annotations vocales contiennent moins de 10 mots). On remarque aussi que plus de la moitié de ces transcriptions sont générées par le moteur de Dragon avec le profil entraîné. Quoique, ce dernier est incapable de donner le taux 100% avec le profil sans apprentissage, contrairement à celui de Microsoft.

D'autre part, 5.26% des transcriptions automatiques sont entièrement incorrectes et généralement avec un TP égal à 0%.

Exemple :

- Transcription manuelle : « un centre de règlement des différends sportifs »
- Transcription automatique : « les récentes de l'examen des différentes portes »

Ces taux de transcription sont obtenus, dans 90% des cas, avec des annotations vocales contenant moins de 10 mots. Le moteur de Dragon génère beaucoup plus de transcription présentant un TMC égal à 0% que celui de Microsoft (74.6% des cas sont générés par Dragon). Toutefois, on remarque que 76.9% des TMC nuls sont obtenus par le profil sans apprentissage.

D'un autre côté, 2.4% des transcriptions automatiques présentent un TP négatif pour des TMC, généralement, inférieurs à 35%. Les valeurs négatives sont obtenues dans le cas où le

nombre d'insertions est plus grand que le nombre de mots dans le fichier. Ceci peut se produire avec les fichiers qui contiennent seulement quelques mots comme avec ceux qui contiennent un nombre important de mots. On remarque aussi que 88% de ces cas sont générés par le moteur de Microsoft, dont 83% avec le profil sans apprentissage.

Performance de transcription par profil

L'analyse de l'ensemble des transcriptions montre une différence importante entre les valeurs des moyennes du TMC et celles du TP pour le profil sans apprentissage et celui avec apprentissage (voir Tableau 4-2). En effet, on constate un écart de 17.81% de TMC entre le profil avec apprentissage et celui sans apprentissage. Pour le taux de précision, cet écart est de 18.39%, en faveur du profil avec apprentissage pour les deux taux.

Tableau 4-2 Moyennes pondérées
du TMC et TP par profil

	Sans apprentissage	**Avec apprentissage**
TMC	43.27 %	61.08 %
TP	35.17 %	53.56 %

Performance de transcription par moteur et par profil

Afin de comparer la performance de transcription des deux moteurs de reconnaissance, on a calculé les moyennes des TMC et TP pour les deux profils. D'après le Tableau 4-3, on remarque que les deux taux sont meilleurs avec le profil entraîné pour les deux moteurs. En outre, la moyenne du TMC du moteur de Microsoft est meilleure de 6% par rapport à celle du moteur de Dragon pour le profil sans apprentissage. Pour le même profil, la valeur de la moyenne du taux de précision est presque identique pour les deux moteurs. Pour le profil avec apprentissage, le moteur de Dragon présente les meilleurs résultats pour le TMC et le TP. En outre, l'écart des moyennes entre les deux moteurs est plus important pour le taux de

précision que pour le taux de mot correct (7.35% pour TP contre 2.64% pour TMC). D'une façon générale, on constate que la performance des deux moteurs est relativement faible. En effet selon la publicité de Nuance, la précision de transcription fournie par Dragon peut atteindre 99% pour la dictée des documents et des e-mails. On essayera de comprendre les sources de cette mauvaise transcription par l'expérience 2.

Tableau 4-3 Moyennes pondérées du TMC et TP par moteur et profil

	Sans apprentissage		**Avec apprentissage**	
	TMC	*TP*	*TMC*	*TP*
Dragon	40.05 %	35.42 %	62.4 %	57.24 %
Microsoft	46.5 %	34.93 %	59.76 %	49.89 %

Performance de transcription par dialecte et par profil

Notre corpus vocal est construit par 12 personnes parlant le français québécois et 3 personnes parlant le français international. Il est clair que le nombre des participants parlant le français international est relativement petit. Toutefois, on a essayé d'avoir une idée, même partielle, sur l'influence du dialecte sur la transcription par nos deux moteurs. Le Tableau 4-4 présente les moyennes de TMC et de TP calculées selon le profil et le dialecte des locuteurs.

Tableau 4-4 Moyennes pondérées du TMC et TP par dialecte et profil

	Sans apprentissage		**Avec apprentissage**	
Français / ***Taux***	*TMC*	*TP*	*TMC*	*TP*
Québécois	41.29 %	33.7 %	60.19 %	53.38 %
International	52.89 %	42.31 %	65.37 %	54.48 %

On remarque que la transcription automatique des annotations vocales effectuées par des locuteurs parlant le français international est plus appropriée que celles réalisées par des locuteurs parlant le français québécois. Ce résultat est vrai quelque soit le type d'apprentissage et pour le taux de mots correct et le taux de précision. Cependant, l'écart des moyennes diminue en passant du profil sans apprentissage à celui avec apprentissage. En effet, l'écart passe de 11.6% pour le TMC avec le profil sans apprentissage à 5.18% avec celui entraîné. De même, l'écart de TP passe de 8.61% avec le profil sans apprentissage à 1.1% avec le profil entraîné.

Performance de transcription par locuteur, par moteur et par profil

Les moyennes des TMC et TP pour chaque locuteur sont principalement exploitées pour étudier l'influence de la performance de transcription sur celle de l'indexation. Toutefois, on peut les utiliser pour détecter quelques anomalies et essayer de comprendre les sources de la mauvaise transcription. Le Tableau 4-5 représente les moyennes des TMC et TP pour chaque locuteur, par moteur et selon le profil.

Tableau 4-5 Moyennes pondérées du TMC et TP par locuteur, par moteur et par profil

	Microsoft				Dragon			
	Sans apprentissage		Avec apprentissage		Sans apprentissage		Avec apprentissage	
Locuteur / Taux	***TMC***	***TP***	***TMC***	***TP***	***TMC***	***TP***	***TMC***	***TP***
H1	65.39	49.43	70.91	56.59	58.83	49.21	69.94	60.77
H2	25.82	17.94	46.86	41.27	30.97	28.24	52.45	48.84
H3	35.07	18.61	57.85	46.37	32.06	26.14	64.99	57.75
H4	38.92	25.89	65.55	55.73	40.29	37.79	74.48	71.83
F1	48.6	43.82	54.31	49.53	27.36	26.52	55.99	53.8
F6	46.2	35.25	53.16	42.64	28.97	25.37	44.18	40.98
H5	52.77	40.91	65.45	54.55	42.16	38.32	65.94	61.27
H8	68.5	59.57	77.12	71.12	54.8	51.64	70.27	66.57
H9	39.7	23.1	60.36	49.05	44.66	36.44	65.24	56.62
H10	51.34	42.9	61.56	54.58	43.71	41.72	70.89	68.02
H11	38.06	27.32	57.64	52.21	31.05	28.18	62.86	60.51
H12	39.46	31.85	38.44	28.5	31.27	29.96	55.69	53.93
H13	53.89	44.19	64.47	53.24	50.9	43.97	63.96	54.12
H14	31.04	14.9	54.43	40.77	26.08	20.7	54.85	49.47
F7	44.52	26.99	53.63	35.14	42.76	31.9	60.99	35.14

L'analyse des résultats du Tableau 4-5 ne permet pas d'établir une relation entre les différents locuteurs. En effet, les valeurs de TMC et de TP varient largement d'un locuteur à un autre. On remarque un grand écart entre les meilleures valeurs et les plus faibles. Par exemple, pour le moteur de Microsoft cet écart est égal à 44.67% pour le TP avec le profil

sans apprentissage. Cet écart est un peu moins important avec le moteur de Dragon. En effet, l'écart maximal est égal à 36.69% pour le taux de précision et avec le profil entraîné.

D'autre part, on constate que le fait qu'un utilisateur présente le meilleur résultat avec un moteur donné ne lui garantit pas le meilleur résultat avec l'autre moteur. Par exemple, le locuteur H8 présente les meilleures valeurs de TMC et de TP pour les deux profils avec le moteur de Microsoft. Cependant, avec le moteur de Dragon il partage les meilleurs résultats avec les locuteurs H1 et H4.

D'un autre point de vue, on remarque que la performance de transcription d'un même locuteur peut augmenter ou bien diminuer en passant du moteur de Microsoft à celui de Dragon. Toutefois, la différence de la performance entre les deux moteurs est parfois flagrante pour certains locuteurs surtout avec le profil sans apprentissage. Par exemple, la moyenne de TMC pour le locuteur F1 a chuté de 48.6% avec le moteur de Microsoft à 27.36% avec celui de Dragon.

Discussion

Il est à noter que les valeurs des TMC sont toujours meilleures par rapport à celles des TP vu que ce dernier prend en compte les insertions. À part ça, on ne peut pas établir une relation entre ces deux métriques. Ainsi, on remarque que le TP peut augmenter même si le TMC diminue, comme dans le cas du locuteur H14 (en comparant les deux moteurs). Ceci peut s'expliquer par la différence de capacité de gestion des erreurs par les deux moteurs.

En ce qui concerne la transcription, on a identifié plusieurs sources d'erreurs causant la dégradation de la performance de transcription :

- Mauvaise prononciation : la mauvaise prononciation d'un mot génère habituellement une erreur de substitution.

- Répétition: un mot est prononcé plus qu'une fois consécutivement. Ceci peut engendrer la modification du contexte de la phrase, et par la suite la génération d'erreurs d'insertion et/ou de substitution.
- Troncation : la production d'un mot s'arrête avant la fin. Dans le meilleur cas, le mot tronqué est remplacé par un autre mot (erreur de substitution). Il est possible aussi que le mot tronqué soit remplacé par un groupe de mots. Dans ce cas, on aura une erreur de substitution et une ou plusieurs erreurs d'insertion.
- Hésitation : les hésitations correspondent à certaines productions du locuteur traduisant son incertitude à ce qu'il va prononcer (euh, eh, hum, etc.). Parfois les hésitations sont ignorées par les moteurs de reconnaissance. Cependant, elles conduisent généralement à des erreurs de substitution et d'insertion.
- (Auto)-correction : le locuteur cherche à rectifier une erreur dans l'énoncé qu'il est en train de produire. Ce phénomène peut causer la modification du contexte de la phrase, et par la suite la génération d'erreur.
- Allongement d'une unité phonétique : le locuteur allonge une unité phonétique plus que la normale. Ceci peut causer une erreur de substitution et/ou d'insertion.

On remarque que dans plusieurs cas, une erreur de substitution engendre une autre erreur de substitution. En effet, le moteur de reconnaissance automatique de la parole cherche le mot qui convient plus au contexte de la phrase. Ainsi, si la première erreur change le contexte, alors elle va influencer le mot suivant celui qui a causé cette erreur. L'exemple suivant illustre cette situation :

- Expression prononcée : **Bon**, **donc** le gouvernement…
- Expression reconnue : **Dans banque** le gouvernement…

Dans cet exemple, le mot « *Bon* » a été reconnu « *Dans* », ce qui a diminué la probabilité d'avoir « *Dans donc* ». En effet, la probabilité d'avoir « *Dans banque* » est supérieure à la probabilité d'avoir « *Dans donc* ». On peut simuler ce cas en faisant la recherche des deux expressions dans le moteur de recherche Google :

- « *Dans banque* » : 45 500 résultats.
- « *Dans donc* » : 28 800 résultats.

De même, une erreur d'insertion peut engendrer d'autres erreurs d'insertions et/ou de substitutions. Prenons cet exemple :

- Expression prononcée : Faciliter **euh l**'application…
- Expression reconnue : Faciliter **heures d**'application…

Dans cet exemple, l'hésitation a causé une erreur d'insertion et a changé par la suite le contexte de la phrase. Ainsi, il est plus probable d'avoir l'expression « heures d'application » que l'expression « heures l'application ».

4.1.2 Résultat de l'indexation vocale

L'analyse principale de la performance d'indexation vocale est réalisée sur la base des métriques de rappel et de précision. Pour cette expérience, on a utilisé un ensemble de 87 mots clés, dont 54 mots ont été prononcés dans les annotations vocales. Chaque mot clé représente une requête exécutée sur l'ensemble des annotations. Ainsi pour chaque mot clé, les métriques de rappel et de précision sont calculées. Le Tableau 4-6 représente les moyennes de rappel et de précision, de tous les mots clés, pour les deux moteurs et selon le profil.

Tableau 4-6 Moyennes du Rappel et précision par moteur et par profil

	Sans apprentissage		**Avec apprentissage**	
	Rappel	*Précision*	*Rappel*	*Précision*
Dragon	0.52	0.91	0.8	0.96
Microsoft	0.64	0.9	0.79	0.94

On remarque qu'il y a une différence, relativement importante, entre les résultats obtenus avec le profil sans apprentissage et ceux avec le profil entraîné notamment pour les valeurs du rappel. Pour le moteur de Dragon, l'écart entre les deux profils est de 0.28 alors qu'il est égal à 0.15 pour le moteur de Microsoft. Toutefois, en comparant les résultats d'indexation des deux moteurs on constate que les valeurs sont très proches. La seule différence remarquable est observable au niveau du rappel avec le profil sans apprentissage. Ainsi, le moteur de Microsoft fournit un rappel supérieur de 0.12 par rapport à celui de Dragon. Toutefois, le moteur de Dragon fournit les meilleurs résultats dans tous les autres cas avec un écart maximal de 0.02.

De même, on peut analyser les résultats d'indexation en se basant sur la métrique F-mesure qui combine le rappel et la précision. Comme présentée par le Tableau 4-7, la valeur de F-mesure du moteur de Microsoft est supérieure à celle du moteur de Dragon de 0.09 pour le profil sans apprentissage. Cependant, le moteur de Dragon présente une F-mesure supérieure de 0.02 par rapport au moteur de Microsoft pour le profil avec apprentissage.

Tableau 4-7 Moyennes du F-mesure
par moteur et par profil

Moteur / *Profil*	*Sans apprentissage*	*Avec apprentissage*
Dragon	0.66	0.85
Microsoft	0.75	0.87

En comparant la performance d'indexation par rapport à celle de transcription, on constate que le résultat de l'indexation est assez satisfaisant vue la performance, relativement faible, de la transcription. En outre, on remarque que chaque moteur garde son rang pour les deux cas. Par exemple, pour le profil sans apprentissage Microsoft présente le meilleur résultat dans le cas du TMC, et un TP nettement moins bon que celui de Dragon. Pour le même profil, Microsoft présente le meilleur résultat pour le rappel avec une précision nettement moins bonne que celle de Dragon. On peut voir ceci comme une conséquence normale,

puisque la bonne transcription conduit à une bonne performance de détection de mots clés. Cependant, on va voir dans une section ultérieure qu'il existe des cas où la performance d'indexation diminue même avec la diminution du taux d'erreur.

Courbe rappel et précision

La Figure 4.1 illustre les quatre courbes moyennes rappel/précision pour les deux moteurs et selon le profil.

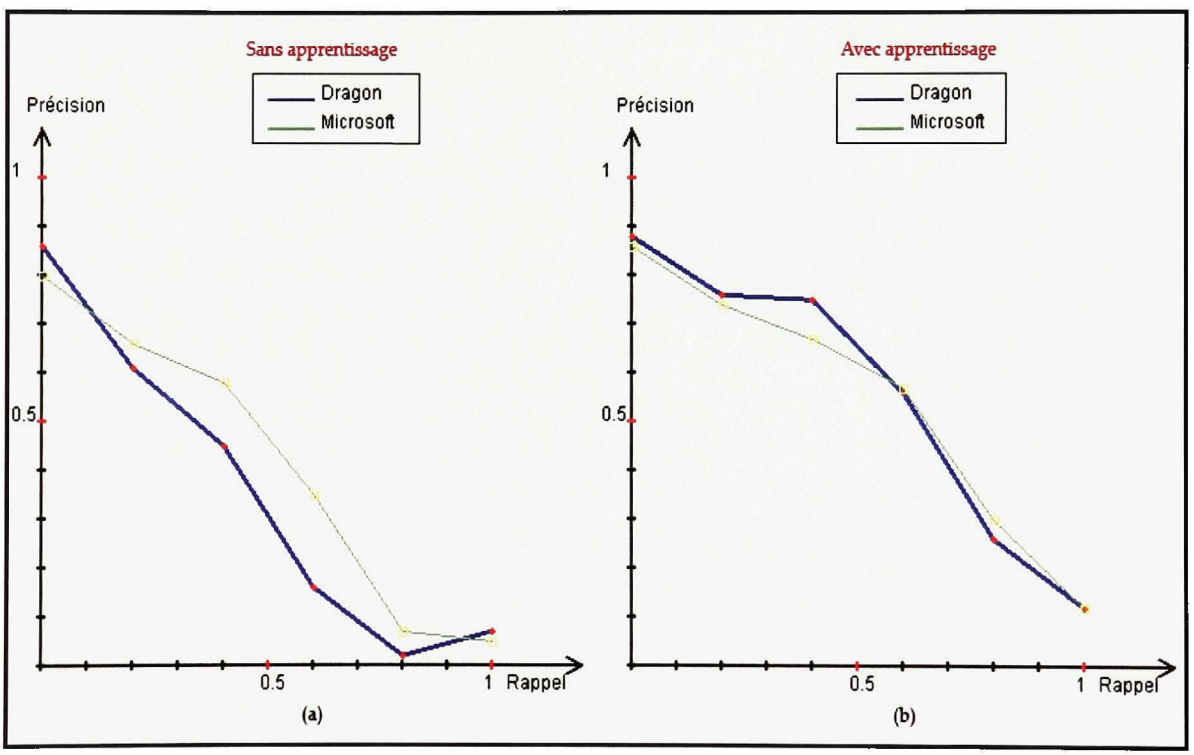

Figure 4.1 Courbes rappel/précision.

Pour le profil sans apprentissage, on remarque que la courbe de Microsoft est généralement au-dessus de celle de Dragon. Ceci indique que le système de Microsoft est plus performant que celui de Dragon dans le cas de ce profil. En outre, la différence de précision est relativement importante entre les deux moteurs pour le rappel 0.4 et 0.6. Dans le cas du profil avec apprentissage, le rapport entre le rappel et la précision est pratiquement identique pour

les deux moteurs. Toutefois, la seule différence de précision remarquable est observée avec un rappel de 0.4.

Graphe Roc

La capacité de détection des mots clés par un système peut être représentée par le graphe ROC, où l'abscisse représente le taux de Faux Positif et l'ordonnée représente le taux de Vrai Positif. Ainsi, on a calculé ces deux taux pour chaque système sur l'ensemble des mots clés. La performance de chacun est représentée par un point sur l'espace du graphe ROC (voir Figure 4.2).

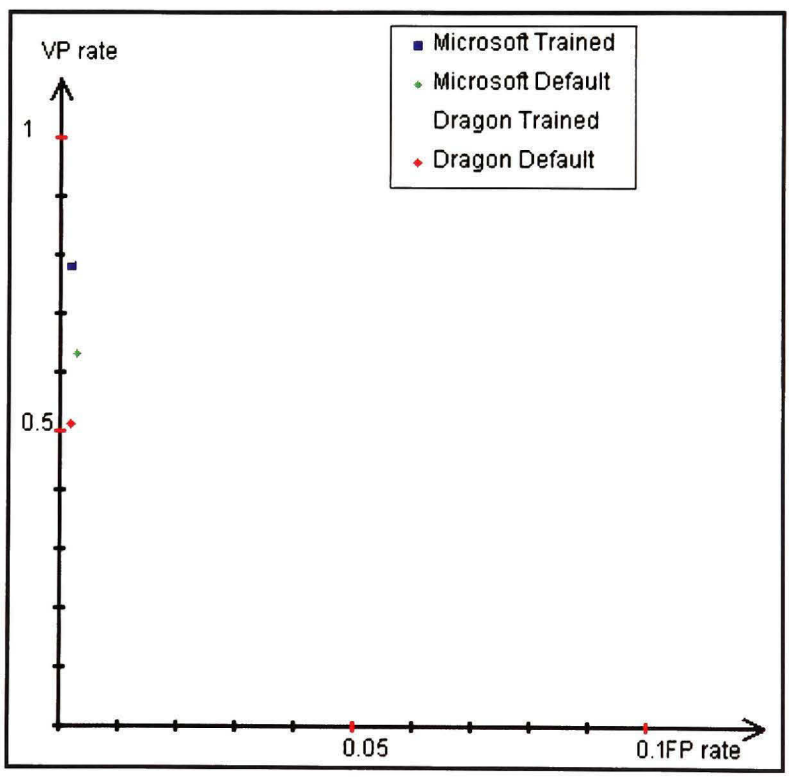

Figure 4.2 Taux de Vrai positif en fonction du taux de faux positif.

On remarque que les quatre points sont situés dans la région de performance conservative du graphe ROC. En effet, le taux de faux positif est infiniment faible par rapport au taux de vrai

positif, et ça pour les quatre cas[10]. Ceci indique que les deux systèmes génèrent peu de faux positif avec le profil sans apprentissage et celui entraîné. Cependant, on remarque une importante différence entre le taux de vrai positif du profil sans apprentissage par rapport au taux du profil avec apprentissage. En effet, le résultat avec le profil entraîné se rapproche plus du point optimal (0,1) avec une légère différence entre le moteur de Dragon et celui de Microsoft. En outre, le moteur de Microsoft présente un taux de vrai positif plus important que celui de Dragon avec le même profil.

Le graphe ROC est utilisé également pour représenter la performance de classification des documents non pertinents par rapport aux documents pertinents non retournés par le système. Comme représenté par la Figure 4.3, le taux de faux négatif est relativement faible pour les deux systèmes et avec les deux profils. En effet, le taux de faux négatif le plus important est égal à 0.02 obtenu par le moteur de Dragon avec le profil sans apprentissage. En outre, on constate que le taux de vrai positif tend vers 1 pour les quatre cas, cela signifie que les deux systèmes classent bien les documents non pertinents comme tels.

[10] Remarquer que le point maximal de l'axe des abscisses pour le taux de faux positif est égal à 0.1.

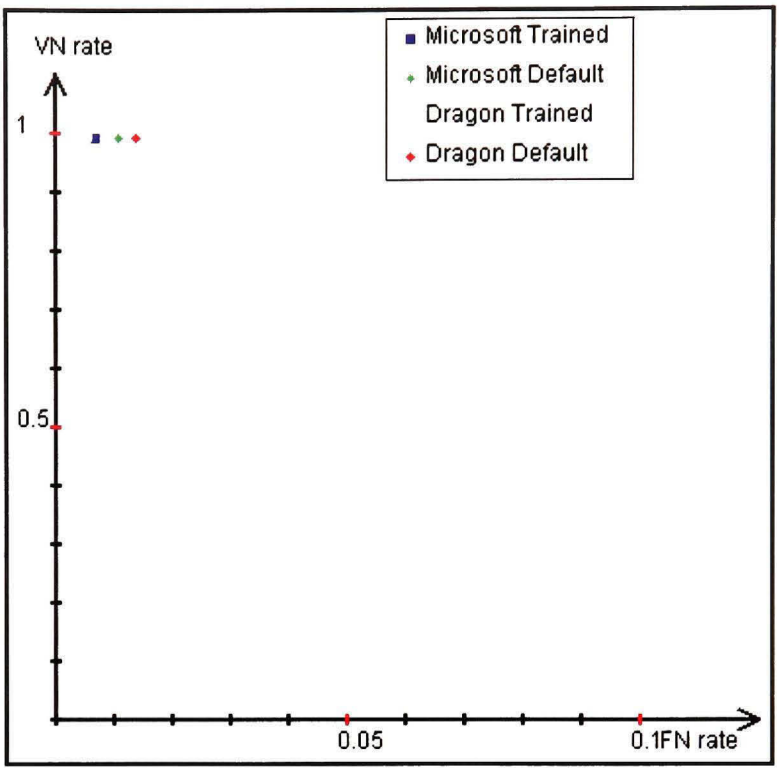

Figure 4.3 Taux de vrai négatif en fonction du taux de faux négatif.

Rapport entre la performance de transcription et la performance d'indexation

Afin d'étudier le rapport entre la performance de transcription automatique et celle de l'indexation, on a calculé les moyennes des rappels et des précisions pour chaque locuteur sur l'ensemble des mots clés. De même, la moyenne des taux d'erreur pour chaque locuteur est calculée afin de tracer des courbes représentant d'une part le rappel par rapport au taux d'erreur, et d'autre part la précision en fonction du taux d'erreur (voir Figure 4.4). L'analyse des résultats obtenus des deux moteurs et avec les deux profils montre que le rappel diminue considérablement en augmentant le taux d'erreur. Par exemple, pour le moteur de Microsoft et avec le profil sans apprentissage le rappel diminue de 0.82 à 0.4 quand le taux d'erreur augmente de 0.4 à 0.85. Toutefois, on remarque que le rappel peut augmenter de nouveau même si le taux d'erreur augmente. Ce cas est observé quand la différence entre les taux

d'erreurs est faible. Par exemple, le rappel passe de 0.58 à 0.71 alors que la le taux d'erreur augmente de 0.65 à 0.68 (courbe (a) profil sans apprentissage). Cette légère différence peut être expliquée par les erreurs de transcription automatique elles-mêmes. En d'autres termes, si le moteur fait des erreurs de transcription au niveau des mots clés recherchés, alors le rappel sera plus faible par rapport à celui obtenu avec un taux d'erreurs touchant plus les mots non recherchés même si ce dernier est plus élevé.

Les faits présentés ci-dessus sont aussi observés dans le cas du rapport entre le taux d'erreur et la précision. Quoique, la différence des écarts est moins importante que celles observées dans les cas du rappel.

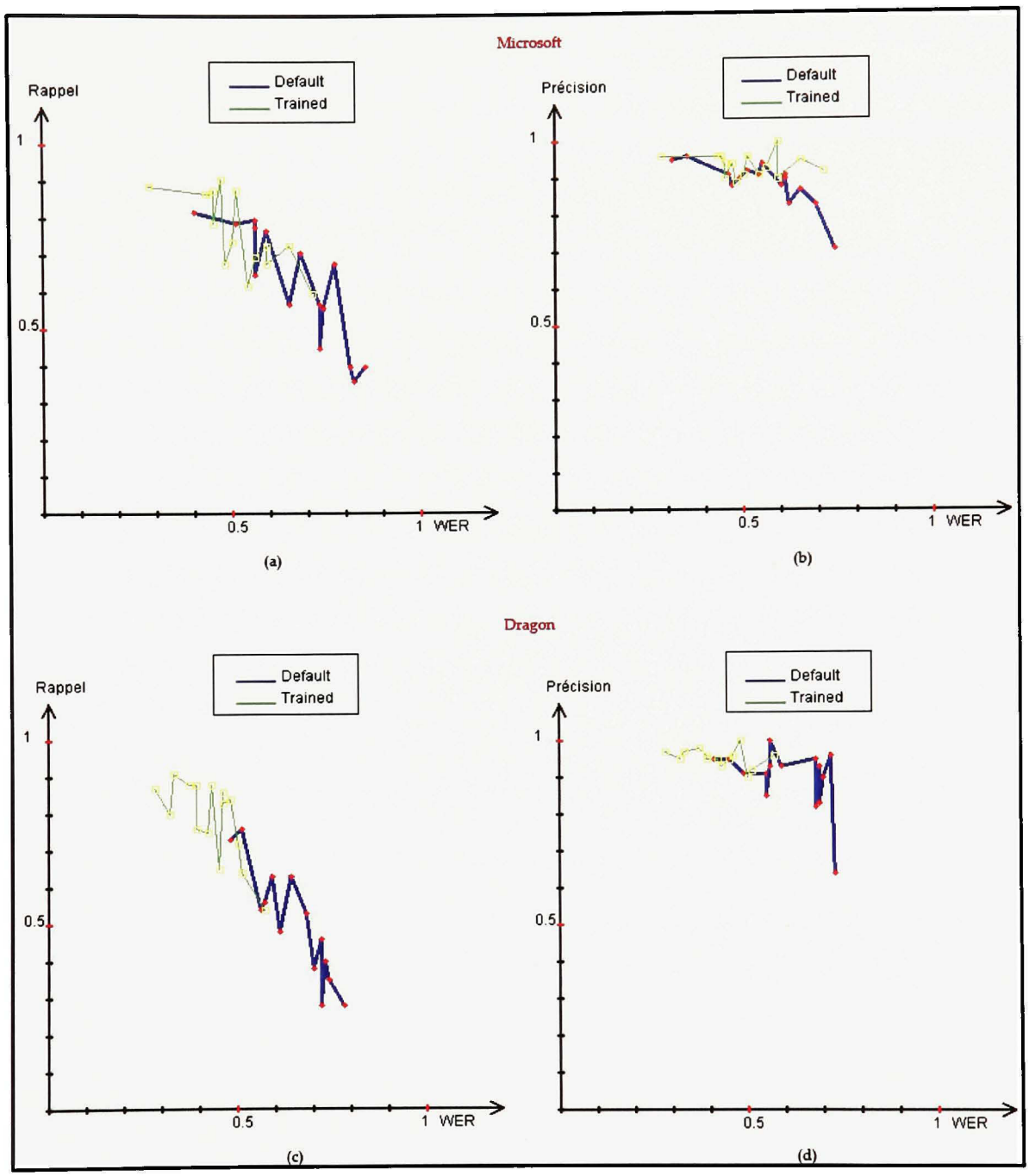

Figure 4.4 Rapport entre TE et la performance d'indexation.

4.2 Expérience 2

Cette expérience a pour objectif d'étudier et de comprendre la cause des mauvais taux de transcription qu'on a obtenus lors de la première expérience. On peut poser comme

hypothèse que les mauvais taux de transcription sont une conséquence de la parole spontanée. On a donc fait une seconde expérience qui consistait à faire relire les annotations par deux locuteurs. Ainsi, on se retrouvait non pas avec de la parole spontanée mais de la parole lue de la parole spontanée.

Le corpus de test utilisé dans cette expérience est formé des annotations vocales des deux locuteurs H1 et H2. Le nombre d'annotations pour les deux locuteurs est égal à 60 annotations totalisant 25 minutes. Chaque locuteur doit relire les transcriptions originales de ses annotations obtenues lors de la phase de construction du corpus initial, avec les deux moteurs. Il est à noter que le participant répète exactement ce qui a été prononcé incluant les répétitions, les mots tronqués et les mauvaises prononciations. Le résultat obtenu est considéré comme un cas de transcription de lecture de la parole spontanée (LPS), alors que, le résultat de la transcription de la première expérience est considéré comme un cas de transcription de la parole spontanée (PS).

4.2.1 Résultats

Cette expérience est réalisée seulement avec le profil entraîné des deux locuteurs H1 et H2. Le résultat des deux locuteurs ensemble pour la transcription de PS et LPS sont donnés par le Tableau 4-8.

Tableau 4-8 Taux de transcription global
pour la PS et LPS

	Parole spontanée	**Lecture de la parole spontanée**
TMC	59.09 %	78.43 %
TP	51.2 %	75.18 %

On remarque que le résultat de la transcription de LPS est beaucoup supérieur au résultat obtenu par la transcription de PS. En effet, on constate une amélioration de 19,34% pour le

TMC, et une amélioration de 23.98% pour le TP. Ce résultat peut être confirmé par l'analyse des taux de transcription, de chaque moteur séparément, données par le Tableau 4-9.

Tableau 4-9 Taux de transcription par moteur pour PS et LPS

	Parole spontanée		**Lecture de la parole spontanée**	
	TMC	*TP*	*TMC*	*TP*
Dragon	60.39 %	54.19 %	82.48 %	79.94 %
Microsoft	57.78 %	48.2 %	74.38 %	70.42 %

Comme on peut le constater, les taux de transcription de la parole spontanée sont considérablement améliorés pour les deux moteurs. Toutefois, l'amélioration des résultats obtenus par le moteur de Dragon est plus importante que celle du moteur de Microsoft.

D'un autre point de vue, on remarque que la différence de performances entre les deux locuteurs est relativement importante pour la transcription de la parole spontanée (voir Tableau 4-10). Cependant, cette différence est moins remarquable dans le cas de la lecture de la parole spontanée. En effet, les résultats du locuteur H2 se sont beaucoup plus améliorés par rapport à ceux de H1.

Tableau 4-10 Taux de transcription
par moteur et par locuteur pour PS et LPS

	Microsoft				**Dragon**			
	PS		**LPS**		**PS**		**LPS**	
Locuteur / ***Taux***	***TMC***	***TP***	***TMC***	***TP***	***TMC***	***TP***	***TMC***	***TP***
H1	70.98	56.67	81.72	77.62	70.02	60.84	82.69	80.23
H2	46.8	41.15	68.27	64.43	52.38	48.66	82.3	79.7

4.2.2 Discussion

Tous les résultats obtenus dans cette expérience montrent que la performance de transcription de LPS est meilleure que celle de PS. On rappelle que cette expérience consiste à relire fidèlement les transcriptions originales des annotations obtenues lors de la construction du corpus initial. Il est à noter que la parole est préparée mais le texte ne l'est pas, puisqu'il résulte de la transcription de la parole spontanée. Toutefois, contrairement à la parole spontanée, le locuteur n'improvise pas ce qu'il va prononcer en lisant le texte. Par conséquent, les phrases prononcées seront plus longues et plus significatives. Ceci n'explique que partiellement la différence de performance entre PS et LPS. En effet, dans le cas de la parole spontanée le locuteur fait beaucoup d'hésitations. Ces dernières ne sont pas marquées dans la transcription originale, et par la suite elles sont ignorées dans le cas de la lecture de la parole spontanée. Si elles sont prononcées, ces hésitations causent généralement des erreurs d'insertion.

D'autre part, contrairement à LPS, dans le cas de PS on rencontre souvent le phénomène d'allongement des unités phonétiques. Dans ce cas, le moteur de reconnaissance automatique de la parole peut fournir comme résultat un mot (souvent une erreur de substitution) et une erreur d'insertion. Par exemple, si le locuteur prononce « *par unnne maladie* », le moteur peut reconnaître « *par il ne maladie* ».

4.3 Expérience 3

Cette expérience a pour objectif d'améliorer la performance de transcription en adaptant le modèle de langage au contexte de l'annotation. En d'autres termes, on entraînera le modèle de langage avec des textes qui sont en relation avec le contexte de l'annotation. Dans un premier lieu, on utilisera tout le corpus des lois du Canada pour entraîner le modèle de langage. Ceci permettra au système, d'une part, d'ajouter les mots qui apparaissent dans ces documents et qui ne sont pas contenus dans son vocabulaire. D'autre part, le modèle de langage s'adaptera au style d'écriture employé dans ces documents. Dans un deuxième lieu,

on entraînera le système avec seulement les documents du corpus des lois du Canada qui sont utilisés dans la phase d'annotation. Dans cette expérience, on se limite qu'au locuteur H1 et H2, et avec seulement le moteur de reconnaissance de Dragon avec le profil entrainé pour réaliser la transcription. Les résultats de ces deux tests sont présentés dans le Tableau 4-11.

Tableau 4-11 Résultat de l'adaptation avec les documents du corpus

Adaptation / *Taux*	**TMC**	**TP**
Sans adaptation	59.09 %	51.2 %
Tout le corpus des lois du Canada	55.0 %	50.79 %
Textes de lois utilisées	61.95 %	55.24 %

Comme on peut le remarquer, la performance de transcription a baissé en utilisant tout le corpus des lois du Canada. Par contre, on constate une importante amélioration dans le cas de l'apprentissage avec seulement les documents utilisés pour l'annotation. Il est à noter que le nombre des documents annotés est très petit par rapport au nombre des documents constituant le corpus. Ainsi, on peut conclure que la pertinence des documents utilisés pour entraîner le modèle de langage est plus importante que la taille du corpus d'entraînement.

Les documents du corpus des lois du Canada présentent des textes structurés et préparés. Toutefois, l'adaptation du modèle de langage avec ces documents (cas des documents annotés) a conduit à une amélioration de performance de transcription. Ainsi, on a posé comme hypothèse que la performance s'améliora davantage en entraînant le modèle de langage avec des textes résultant de la parole spontanée. On a alors répété la même expérience, mais cette fois-ci avec les transcriptions manuelles des annotations. En effet, on a utilisé dans un premier test toutes les transcriptions manuelles des annotations réalisées par tous les locuteurs. Dans le deuxième test, on a utilisé les mêmes annotations mais sans inclure celles de H1 et H2. Les résultats obtenus sont présentés dans le Tableau 4-12.

Tableau 4-12 Résultat de l'adaptation
avec les transcriptions manuelles

Adaptation / *Taux*	TMC	TP
Sans adaptation	59.09 %	51.2 %
Transcriptions manuelles de tous les locuteurs (inclus H1 et H2)	77.33 %	71.6 %
Transcriptions manuelles (exclus H1et H2)	63.41 %	56.63 %

Les résultats de ces deux tests montrent que les performances de transcription sont améliorées dans les deux cas. En effet, on constate que le TMC s'est amélioré de 18,24% et le TP de 20.4% lorsqu'on a utilisé les transcriptions manuelles de tous les locuteurs. De même, on constate une amélioration de 4.32% pour le TMC et de 5.43% pour le TP lorsqu'on a adapté le modèle de langage avec les transcriptions manuelles sans celle de H1 et H2. D'autre part, on remarque que les résultats obtenus avec les transcriptions manuelles de tous les locuteurs sont meilleurs que les résultats obtenus en exemptant les transcriptions de H1 et H2. Ainsi, on peut constater que les performances de transcription peuvent s'améliorer considérablement en adaptant le modèle de langage avec les données propres de chaque locuteur.

Afin de confirmer ce qu'on vient de constater, d'autres annotations vocales ont été produites par le locuteur H1. Au total, on a obtenu 26 annotations d'une durée totale de 13 minutes. Ensuite, on a utilisé les transcriptions manuelles de ces annotations pour entraîner le modèle de langage. On a effectué par la suite la transcription des premières annotations générées par le locuteur H1 (pas celles utilisées pour l'apprentissage). Le Tableau 4-13 présente les résultats de transcriptions de ce test.

Tableau 4-13 Résultat de l'adaptation du modèle de langage de H1

Adaptation / *Taux*	TMC	TP
Sans adaptation	70.02 %	60.84 %
Transcriptions manuelles de H1	74.86 %	63.73 %

Comme on peut le constater, la performance de transcription est améliorée en adaptant le modèle de langage avec les transcriptions manuelles des annotations de H1. On insiste ici sur le fait que l'adaptation est réalisée avec peu de données. Le corpus d'apprentissage est formé de transcriptions manuelles de 13 minutes d'annotations. On suppose alors qu'un apprentissage avec un grand corpus de textes résultant de la transcription manuelle des annotations faites par un locuteur donné, améliora les performances de transcription pour ce dernier. De même, les performances de transcription pour un locuteur donné, peuvent s'améliorer à force d'utiliser le système.

4.4 Conclusion

Dans ce chapitre, on a présenté les différentes expériences qu'on a réalisées ainsi que les résultats obtenus par chacune d'entre elles. On a constaté que les meilleurs résultats, de transcription et d'indexation, ont été observés avec le moteur de reconnaissance de Microsoft avec le profil sans apprentissage. Dans les mêmes conditions, mais avec le profil entraîné, le moteur de Dragon fournit les meilleurs résultats. De façon globale, le moteur de Dragon offre les meilleures performances.

D'autre part, on a remarqué que les moteurs de reconnaissance automatique de la parole sont plus performants avec la parole lue que la parole spontanée. Toutefois, on a prouvé que la performance peut être améliorée en adaptant le modèle de langage à la façon de parler du locuteur.

CONCLUSION

Le travail de ce mémoire a consisté principalement à une étude portant sur l'indexation des annotations vocales dans un contexte de gestion documentaire. On s'est intéressé à l'indexation à base de la reconnaissance à grand vocabulaire. En d'autres termes, la détection de mots clés est réalisée en se basant sur le résultat de la transcription fournie par un moteur de reconnaissance automatique de la parole continue.

Dans la littérature, l'indexation à base de la reconnaissance à grand vocabulaire a donné lieu à une grande variété de réalisation. En effet, cette technique est utilisée pour la catégorisation des discussions téléphoniques (Gillick, Baker et al. 1993), l'indexation des conférences universitaires (Park, Hazen et al. 2005), etc. Outre la détection de mots clés dans les fichiers sonores, cette méthode permet de fournir la transcription de ces fichiers avec une précision qui peut dépasser 90% dans certains contextes.

Cependant, la performance de transcription se dégrade considérablement dans le cas de la parole spontanée. Les travaux qu'on a menés dans ce mémoire ont confirmé le lien entre la parole spontanée et les mauvais taux de transcription. En effet, la transcription de la parole lue de texte spontanée a donné une amélioration de 19,34% pour le TMC, et une amélioration de 23.98% pour la TP par rapport à la transcription de la parole spontanée. Cette différence de performance est causée principalement par le phénomène d'allongement des unités phonétiques et les hésitations observés fréquemment dans le cas de la parole spontanée. En effet, l'hésitation engendre des erreurs d'insertion et change souvent le contexte de la phrase, ce qui produit en plus des erreurs de substitution. Ainsi, on envisage de modéliser les hésitations afin de les détecter et par la suite les éliminer. Ceci permettra de diminuer le nombre d'erreurs et par la suite augmenter les taux de transcription.

D'une façon plus générale, il existe divers phénomènes qui sont à l'origine des erreurs de transcription. En effet, l'analyse des résultats de transcription a permis d'identifier plusieurs sources d'erreurs comme la mauvaise prononciation, la répétition, la troncation, l'hésitation,

l'(auto)-correction et l'allongement des unités phonétiques. Chacun de ces phénomènes entraine la génération d'une ou plusieurs erreurs de transcription. Pire encore, il est possible que l'erreur générée change le contexte de la phrase, ce qui entraine la génération d'autres erreurs. Un bon traitement de ces phénomènes conduit à une bonne gestion des erreurs de transcription, et par la suite une bonne performance de transcription.

En établissant le lien entre la performance de transcription et la performance d'indexation, on a constaté que l'augmentation du taux d'erreur entraine la dégradation de la performance d'indexation. Toutefois, même avec un mauvais taux de transcription on a pu avoir une bonne performance de détection de mots clés. En effet, pour un taux d'erreur de 50.11% on a obtenu un rappel de 0.79 et une précision de 0,96 avec le moteur de Microsoft avec le profil entraîné.

D'autre part, la comparaison des performances du moteur de reconnaissance de Dragon avec celui de Microsoft montre que ce dernier est plus performant dans le cas du profil sans apprentissage. Pour le profil avec apprentissage, le moteur de Dragon présente les meilleures performances de transcription et d'indexation. Toutefois, la différence de performance entre les deux moteurs est minimale surtout pour l'indexation. À part les performances de transcription, le système de Dragon présente des outils non fournis par Microsoft, tel que l'outil de transcription des fichiers audio. Toutefois, le moteur de reconnaissance de Microsoft est fourni gratuitement avec le système d'exploitation. En se basant sur ces différences, l'utilisateur pourra choisir entre l'un ou l'autre selon ses besoins.

Visant à améliorer la performance de transcription, on a entraîné le modèle de langage de Dragon avec diverses sortes de textes. Dans un premier lieu, on a utilisé les documents du corpus employé pour réaliser l'annotation. On a constaté que le taux de transcription augmente en utilisant seulement les documents du corpus qui ont été annotés. Toutefois, la performance de transcription se dégrade en utilisant tout le corpus pour faire l'apprentissage. Ainsi, on assume que la pertinence des documents est plus importante que la taille du corpus pour faire l'adaptation. Ainsi, on suggère de préparer un grand corpus de documents

pertinents au contexte de l'annotation pour l'utiliser à l'adaptation. Dans un deuxième lieu, le modèle de langage est entraîné par les transcriptions manuelles des annotations vocales. On a remarqué que les taux de transcription ont augmenté considérablement notamment dans le cas où les transcriptions des locuteurs concernés par cette expérience sont utilisées. En effet, l'utilisation des transcriptions manuelles d'un locuteur donné pour entraîner son profil permet d'adapter le modèle de langage à la façon de parler de ce locuteur. Par conséquent, cette adaptation permet au système de s'adapter au contexte de l'annotation.

ANNEXE I

ETAPES DE TRAITEMENT ACOUSTIQUE

A.1 Production de la parole

La production de la parole est une opération complexe qui implique la coordination de plusieurs organes de notre système phonatoire (voir Figure-A I-1). En effet, les poumons agissent comme un générateur d'air pour alimenter le larynx. Ce dernier se situe au niveau du cou, son mouvement entraîne la production des sons implosifs ou bien des éjectifs. Le larynx comprend les cordes vocales dont la vibration au contact de l'air produit le son, plus particulièrement les voyelles. Finalement, l'air passe par le conduit vocal et la cavité nasale pour arriver aux organes d'articulation (langue, lèvre, mâchoires, etc.) qui produisent le signal acoustique. Pour plus de détails, on peut consulter (Calliope 1989).

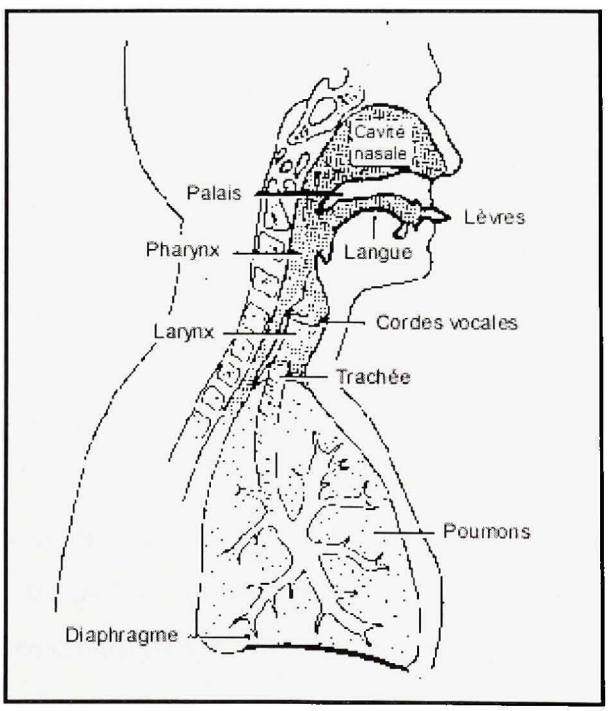

Figure-A I-1 Schéma des organes impliqués dans la production de la parole.
Tirée de BIGOT (1998)

A.2 Filtrage

Après son acquisition par un microphone, le signal acoustique doit subir un filtrage analogique pour éliminer les fréquences non perceptibles par l'oreille. En effet, l'oreille humaine est censée percevoir des fréquences comprises entre 20 Hz et 20 000 Hz. Ainsi, toutes les fréquences en dehors de cette bande passante doivent être éliminées. Cette opération peut se réaliser en appliquant un filtre qui ne laisse pas passer les informations non significatives. En outre, d'autres filtres peuvent être appliqués pour traiter le phénomène de masquage temporel. Ce dernier, s'aperçoit lorsqu'un son de forte intensité masque un autre de faible intensité. Dans ce cas, c'est mieux de ne pas coder le son faible puisque l'oreille ne va pas le percevoir.

Le filtrage analogique permet de réduire la quantité d'informations à coder sans modifier l'information pertinente du signal. Ceci est très utile pour la transmission du code audio via des canaux à bande passante limitée. En effet, dans certaines architectures de reconnaissance automatique de la parole, le signal traité est transmis à un serveur pour réaliser l'opération de reconnaissance.

A.3 Conversion analogique numérique

Cette étape consiste à convertir le signal analogique en un signal numérique exploitable par une machine. La numérisation et le résultat de deux opérations successives sur le signal d'entrée, soit l'échantillonnage et la quantification :

- L'échantillonnage consiste à capturer à temps régulier des valeurs du signal analogique (continu) afin de le transformer en un signal numérique (discret). Cette conversion est nécessaire puisque les ordinateurs ne peuvent pas traiter un signal continu. La fréquence

d'échantillonnage est déterminée par le théorème de Shannon[11] pour que le signal soit correctement échantillonné.

- La quantification permet l'approximation des valeurs instantanées du signal par des valeurs discrètes. Le signal quantifié sera plus fidèle au signal original en augmentant le nombre des valeurs instantanées. De même, on peut perdre des informations utiles si ce nombre de valeurs est petit.

A.4 Extraction des paramètres

Cette étape permet de transformer le signal numérisé en une suite de vecteurs de coefficients. Il s'agit d'extraire le minimum d'informations nécessaires à une bonne reconnaissance de la parole. Les coefficients les plus utilisés dans la reconnaissance automatique de la parole sont les MFCC (Mel Frequency Cepstrum Coefficients). Comme représentée par la Figure-A I-2, l'extraction de ces paramètres est une suite d'étapes. Brièvement, le processus commence par une étape de préaccentuation afin de ressortir les hautes fréquences. Le signal est ensuite segmenté en fenêtre de 30 ms toutes les 10 ms. Un transformé de Fourier rapide (Fast Fourier Transform, FFT) est appliqué à chacune des trames obtenues. Ensuite, un banc de filtres triangulaires sur l'échelle de Mels est crée. Ces filtres sont appliqués pour mieux simuler le fonctionnement de l'oreille. Après la conversion des trames en échelle de Mel, on obtient les coefficients MFCC par l'application d'une DCT (Discrete Cosinus Transform). Finalement, on applique la méthode suppression de la moyenne cepstrale (Cepstral Mean Subtraction, CM5).

---

[11] Théorème de Shannon : toutes les fréquences du signal inférieures à la moitié de la fréquence d'échantillonnage seraient correctement restituées.

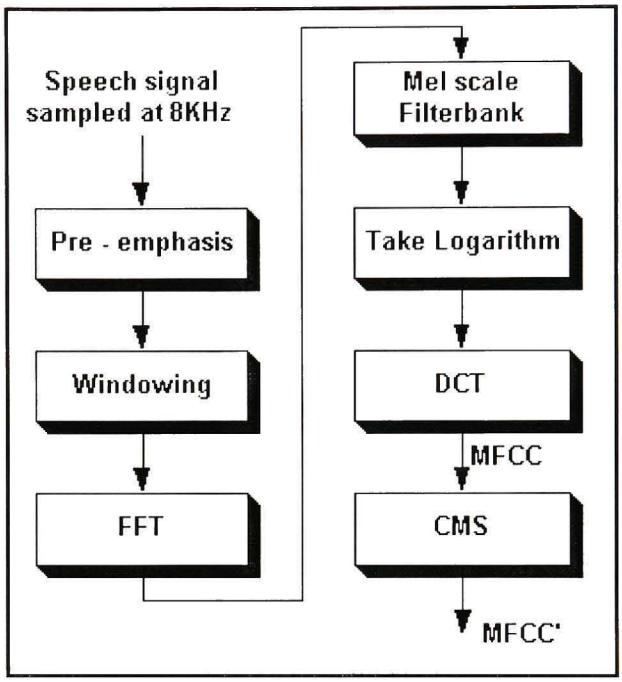

Figure-A I-2 Étape d'extraction des MFCC.
Tirée de (Unjung 2010)

ANNEXE II

QUELQUES METHODES DE MODELISATION

B.1 Comparaison dynamique

La comparaison dynamique est une approche géométrique qui permet de comparer une forme inconnue à une forme de référence. Elle se base sur un principe très simple utilisé avec les premiers systèmes de reconnaissance des mots isolés. Le schéma de fonctionnement de cette méthode est donné par la Figure-B II-1 :

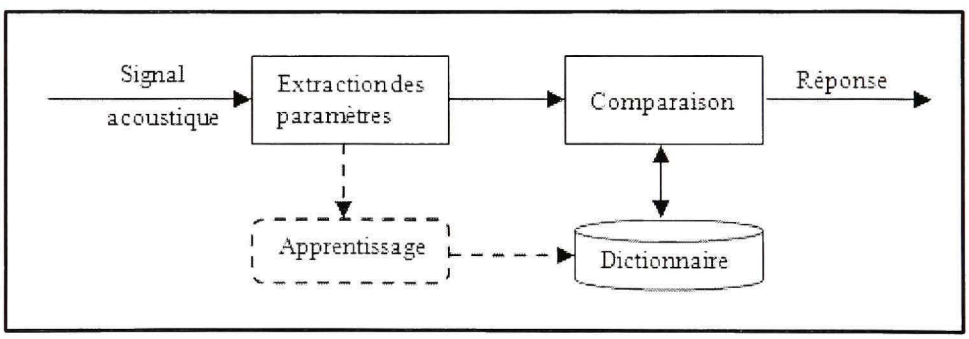

Figure-B II-1 Schéma de fonctionnement de la comparaison dynamique.

La mise en ouvre d'un système de reconnaissance de la parole basée sur cette approche nécessite la construction d'un dictionnaire. En effet, le locuteur doit prononcer tous les mots constituant le vocabulaire à reconnaître pour créer les formes de références. L'apprentissage dans cette méthode est très simple, elle consiste à répéter la prononciation des mots par le même locuteur plusieurs fois. Ainsi, chaque mot prononcé sera représenté par plusieurs formes, chacune d'elle modélisé par une séquence de vecteurs. Ces derniers sont obtenus dans la phase d'extraction des paramètres basée sur des techniques de traitement du signal. En phase de reconnaissance, le système calcule la distance[12] qui sépare le mot prononcé avec toutes les références dans le dictionnaire, celui le plus proche représente le mot cherché.

---

[12] Exemple de distance utilisée : distance euclidienne, distance de Malhabolis, etc.

Une difficulté majeure réside dans le fait qu'un même mot prononcé par un locuteur donne deux spectrogrammes différents. En effet, la variabilité intra-locuteur, associée à d'autres facteurs, agit à ce niveau et engendre des différences non linéaires dans le temps rendant difficile la comparaison des formes. Pour faire face à ce problème, Vintsjuk (VINTSJUK T.K. 1968) a introduit la technique d'alignement temporel dynamique (DTW: Dynamic Time Warping). Cet algorithme est fondé sur la programmation dynamique désignée pour la première fois par Bellman (Bellman 1957). L'algorithme réalise un alignement temporel en recherchant, parmi tous les alignements possibles, celui qui correspondant à l'alignement de coût minimal. La Figure-B II-2 représente les vecteurs des paramètres de la forme acoustique de référence I (i=1, … , I) alignés, sur une matrice, avec les vecteurs des paramètres de la forme inconnue.

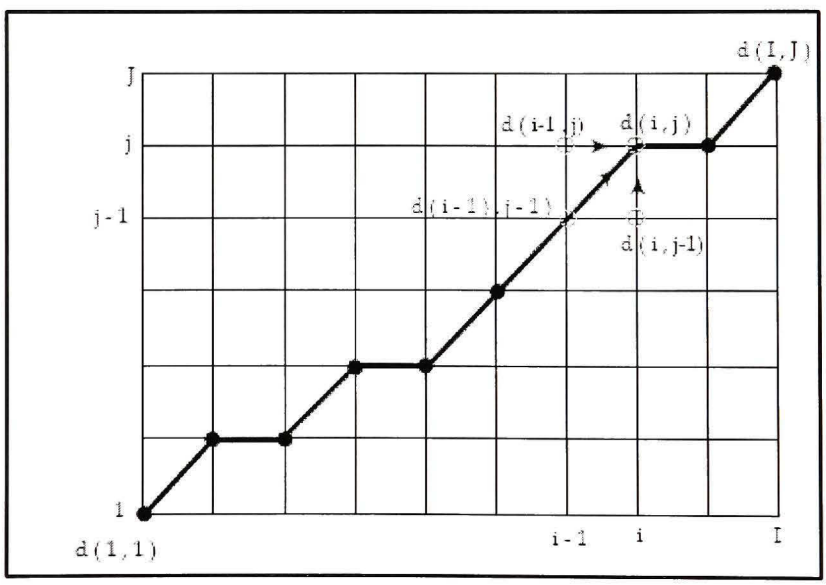

Figure-B II-2 Algorithme d'alignement temporel dynamique (DTW).

La recherche se fait on calculant la distance d(i,j) entre deux vecteurs mis en correspondance :

$$g(i, j) = \min \begin{cases} g(i-1,\ j) + d(i,\ j) \\ g(i-1,\ j-1) + 2d(i,\ j) \\ g(i,\ j-1) + d(i,\ j) \end{cases} \qquad \text{(AII.1)}$$

La distance optimale entre les deux formes est alors donnée par :

$$D = \frac{g(I,\ J)}{(I + J)} \qquad \text{(AII.2)}$$

Ce processus permet non pas seulement de déterminer la distance entre deux formes, mais aussi de chercher le chemin optimal. Cette fonctionnalité peut être utilisée dans la phase d'apprentissage en appliquant des techniques d'agrégation (Tou and Gonzalez 1974).

L'approche de comparaison dynamique a connu un succès pour la reconnaissance de mots isolés. Cependant, elle est peu efficace pour la reconnaissance de la parole continue et reste limitée à l'utilisation d'un petit vocabulaire. En effet, la modélisation des formes acoustiques pour construire le dictionnaire reste encore sensible à l'ensemble des variabilités acoustiques. Néanmoins, cette technique est plus adaptée à un contexte d'utilisation mono-locuteur en environnement peu bruité (Linares 2003).

B.2 Modèles neuromimétiques

Les modèles neuromimétiques[13] sont utilisés depuis une dizaine d'années pour résoudre les problèmes liés à la reconnaissance automatique de la parole. Cependant, ces modèles ont été utilisés depuis longtemps pour la classification et la reconnaissance des formes (Mendel and Fu 1970). Le modèle connexionniste est inspiré des neurones biologiques. En effet, il est composé de plusieurs cellules simples mais fortement interconnectées les uns des autres. Chaque cellule élémentaire est appelée neurone car elle imite le fonctionnement d'un neurone biologique. La sortie de chaque cellule est une fonction non linéaire de la somme pondérée de toutes ses entrées (Haton 2006). La Figure-B II-3 illustre le principe basique de fonctionnement d'un neurone artificiel :

---

[13] Désignés aussi par les réseaux de neurones artificiels, modèle connexionniste ou encore Parallel Distributed Processing Model (modèle PDP).

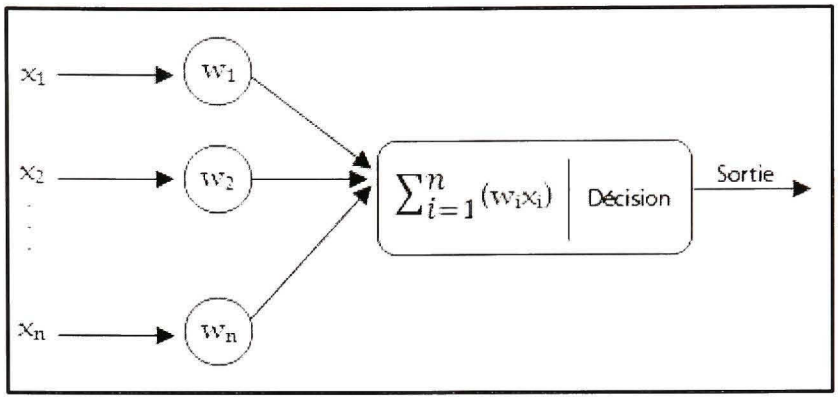

Figure-B II-3 Principe de base d'un neurone artificiel.

B.2.1 Typologies de réseaux de neurones artificiels

La façon dont les différentes cellules sont interconnectées est une caractéristique très importante dans les modèles connexionnistes. Pour la reconnaissance automatique de la parole on distingue essentiellement trois topologies de réseau : perceptrons multicouche, les réseaux récurrents et les cartes auto-organisatrices.

B.2.1.1 Les perceptrons

Le perceptron est la topologie de réseau de neurones artificiels la plus simple caractérisée par une classification linéaire. Ce type de réseau neuronal est un réseau sans contre-réaction et ne contient aucun cycle. Le modèle utilisé est un réseau multicouches et il représente la version évoluée des perceptrons monocouche inventé par Rosenblatt (Rosenblatt 1962). Le perceptron multicouche est formé de couches reliées entre elles par l'intermédiaire des neurones qui les composent Figure-B II-4.

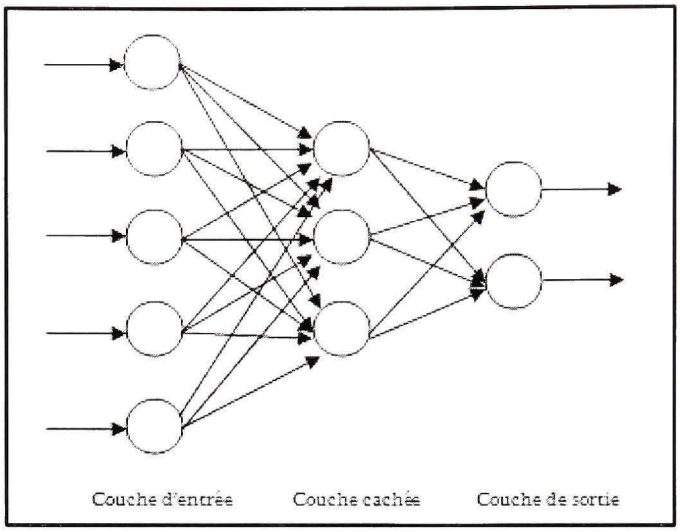

Figure-B II-4 Structure d'un perception à trois couches.

L'apprentissage dans un perceptron multicouche est de type supervisé c'est-à-dire que l'on procède par correction d'erreur. Pour une forme d'entrée donnée, on impose une réponse en sortie du réseau afin de calculer l'erreur commise par le réseau. Ainsi, pour chaque vecteur d'entrée on calcule l'erreur de sortie par rapport à une sortie désirée. Cette opération permet de recalculer les poids en fonction de l'erreur et par la suite construire des paramètres plus représentatifs. Plusieurs travaux ont été réalisés pour proposer de nouvelles méthodes d'attribution de poids plus sophistiquées (Karouia 1996).

D'une façon générale, pour l'apprentissage de type supervisé on distingue deux types d'erreur correspondant à deux critères d'optimisation :

- Le critère de moindres carrés: fondé sur l'utilisation d'une distance euclidienne entre les formes.
- Le critère d'entropie croisée: les sorties du réseau sont considérées comme des distributions de probabilité sur les variables d'entrée.

B.2.1.2 Les réseaux récurrents

Les entrées d'un neurone dans les réseaux récurrents peuvent être aussi bien des entrées que des sorties d'autres neurones. Dans la reconnaissance automatique de la parole on utilise souvent ces réseaux en les construisant à partir des perceptrons. Ainsi, les neurones de la couche cachée réactivent les entrées afin de prendre en compte l'écoulement temporel. Comme illustré par la Figure-B II-5 l'état d'un neurone est une fonction du vecteur d'entrée à l'instant *t* et de l'état du réseau à l'état *t-1*.

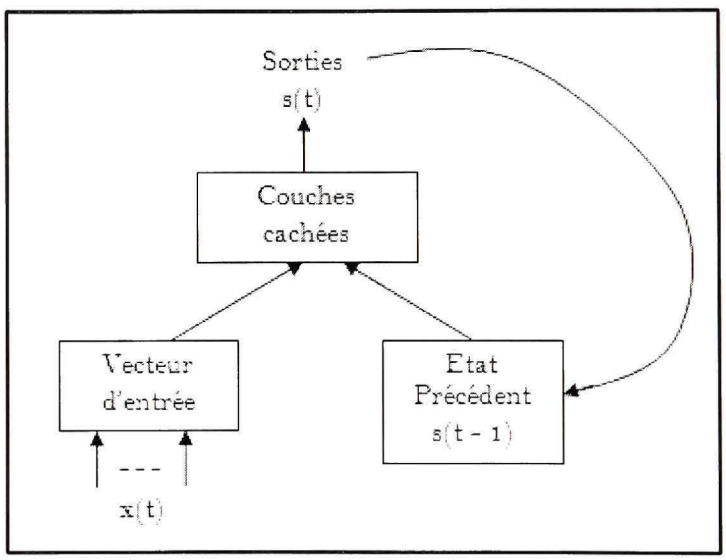

Figure-B II-5 Principe d'un réseau récurrent.

Les réseaux récurrents présentent une limitation dans la reconnaissance automatique de la parole dans le cas d'un grand vocabulaire. Pour résoudre ce problème, Chen propose d'utiliser une architecture hiérarchique en modélisant des sous-réseaux récurrents pour l'identification des syllabes constituant un mot (Chen 1995).

B.2.1.3 Les cartes auto-organisatrices

La carte auto-organisatrice[14] est une autre topologie de réseau de neurones artificiel inventée par le statisticien Kohonen (Kohonen 1982). Ce modèle est inspiré de l'organisation neuronale du cerveau des vertébrés où des stimuli de même nature excitent une région du cerveau bien particulière. Les unités du réseau sont interconnectées latéralement selon une structure non hiérarchique. Cette dernière, se compose habituellement de deux couches de neurones, une couche d'entrée et une autre représentant une carte de paramètres. Cette structure permet aux cellules de recevoir les données d'une part de la couche d'entrée, et d'autre part des cellules voisines de la deuxième couche.

L'apprentissage dans les cartes auto-organisatrices est de type non supervisé. Plusieurs modèles ont été proposés pour ce type d'apprentissage. Par exemple, on peut utiliser la règle d'apprentissage de Hebb, qui suggère que lorsque deux neurones sont excités conjointement, cela renforce le lien qui les réunit (Hebb 1949). Cette règle vient à attribuer un poids fort à la connexion entre deux cellules si elles sont simultanément actives, et un poids faible dans le cas contraire.

Afin de mieux adapter l'approche connexionniste à la reconnaissance de la parole, plusieurs méthodes ont été proposées. On peut citer, à titre d'exemple, les réseaux multicouches à retard, les classifieurs dynamiques et les modèles biologiques qui sont le fruit de l'avancement des recherches en neurophysiologie.

B.2.2 Avantages des modèles neuromimétiques

Les modèles neuromimétiques présentent plusieurs avantages supplémentaires dans la reconnaissance automatique de la parole. Voici quelques points forts mentionnés dans (Boite 1987; Haton 2006) :

---

[14] Désignée aussi par le terme anglais *self organizing map* (SOM), ou encore carte de Kohonen

- Ces modèles se caractérisent par un apprentissage discriminant permettant l'amélioration de la reconnaissance d'une classe de formes, et le rejet des autres classes. En outre, ils peuvent être interprétés comme une généralisation non linéaire des fonctions discriminantes. Ce dernier point leur permet d'obtenir un bon résultat de classification par rapport à d'autres approches de classification.
- L'entraînement est basé sur une modélisation non paramétrique ce qui permet d'estimer des distributions statistiques de n'importe quelle forme. En effet, ce modèle ne nécessite pas d'hypothèses sur les propriétés statistiques des données en entrée.
- Plusieurs vecteurs acoustiques peuvent être fournis facilement à l'entrée du réseau de neurones.
- L'implémentation de modèle sous forme logicielle ou matérielle est facilitée par ces structures parallèles régulières.

B.3 Machine à vecteur support

Les machines à vecteurs de support (Vapnik 1998) (en anglais Support Vector Machine, SVM) sont une généralisation des classifieurs linéaire destinée à résoudre des problèmes de discrimination[15] et de régression[16]. Les SVM permettent de créer une surface de décision entre deux classes définies dans un même espace. En effet, une frontière de décision est construise afin de séparer linéairement les deux classe, le critère d'optimisation est la largeur de la marge entre les deux classes. Dans le cas d'un problème linéairement séparable (le cas le plus simple), le choix de l'hyperplan séparateur pose un problème. Il existe une infinité de plans séparateurs, d'où la nécessité d'identifier un seul qui maximise la marge entre les échantillons et l'hyperplan séparateur.

L'utilisation des machines à vecteurs de support dans la reconnaissance automatique de la parole est efficace dans certains cas. En effet, elle était parmi les techniques utilisées dans ces

---

[15] Décider à quelle classe appartient un échantillon.
[16] Prédire la valeur numérique d'une variable.

dernières années pour l'identification du locuteur, la reconnaissance des formes acoustiques, la détection de mots clés ainsi que pour la conception des formes acoustiques hybrides (Haton 2006).

B.4 Modèles de Markov

Les signaux acoustiques se caractérisent par un aspect non stationnaire à variation semi-continue rendant très difficile la séparation des formes à reconnaître. Pour certains systèmes de reconnaissance vocale, tel que la reconnaissance de la parole continue, cet aspect pose un problème majeur puisque les formes à reconnaitre n'ont pas de frontières apparentes. Les modèles de Markov cachés (Hidden Markov Models, HMM) représentent la meilleure solution à ce problème. En effet, ces modèles sont capables de modéliser mathématiquement les variabilités temporelles par des paramètres de transitions. En outre, les paramètres des distributions de sortie permettent de modéliser les variabilités spectrales (Alani and Guellif 1994). Les HMMs ont été introduits à la reconnaissance automatique de la parole dans les années 70 par Baker (Baker 1975) et Jelinek (Jelinek 1997).

Les modèles de Markov cachés doivent leurs succès à leurs conceptions sur des bases mathématiques solides. Une telle conception permet la modélisation des processus stochastiques pour traiter les problèmes à informations incertaines ou incomplètes. Un HMM représente un ensemble de séquences d'observations dont l'état de chaque observation n'est pas observé. En effet, le modèle de Markov caché est un processus doublement stochastique dans lequel les observations sont une fonction aléatoire de l'état, et dont l'état change à chaque instant en fonction des probabilités de transition issues de l'état précédent (Luba and Younes 2005). Dans cette section on va présenter les différents aspects théoriques des HMM en se basant principalement sur (Alani and Guellif 1994; B. Roe 1994; Jelinek 1997; Peinado 2006) .

B.4.1 Formulation mathématique

Pour concevoir un système de reconnaissance de la parole, on a besoin d'une formulation mathématique permettant la discussion et la mise en évidence du problème.

Le système est composé de :

- Entrée : signal acoustique, **A** **avec** $\mathbf{A} = x_1, x_2, \ldots, x_T$; $x_i \in A$
- Sortie : séquence de mots, **W** **avec** $\mathbf{W} = w_1, w_2, \ldots, w_n$; $w_i \in w$

A est la séquence de paramètres, qui définit le signal acoustique en amant du système, composée de T observations, et **W** représente un ensemble de n mots appartenant à un vocabulaire bien connus.

Pour chaque mot dans l'ensemble du vocabulaire le système doit calculer la probabilité d'avoir W sachant que A est prononcé. La décision sera en faveur du mot qui possède la probabilité la plus élevée qu'on le note $\hat{W}$. Par la suite, la règle de décision sera :

$$\hat{W} = \arg\max_{W} P(W|A) = \arg\max_{W} \frac{P(W)P(A|W)}{P(A)} \qquad \text{(AII.3)}$$

Puisque l'objectif est de déterminer le mot $\hat{W}$ (ou plus précisément la séquence de mots) qui maximise le produit P(W) P(A|W), on peut ne pas calculer P(A) et réduire par la suite le temps de calcul. De toute façon la valeur de P(A) n'influe pas le résultat final puisqu'elle est la même valeur pour tous les mots de W. Donc pour une probabilité apriori P(W) le mot le plus probable dépend seulement de P(A|W). Par la suite, $\hat{W}$ sera calculé comme suit:

$$\hat{W} = \arg\max_{W} P(W)\, P(A|W) \qquad \text{(AII.4)}$$

B.4.2 Processus acoustique dans le cas des HMMs

La Figure-B II-7 représente un schéma simplifié du processus acoustique dans le cas d'une modélisation avec les HMMs. Le signal acoustique en provenance d'un microphone est traité dans le module *Traitement de signal* qui génère à des intervalles de temps réguliers des vecteurs de paramètres σ_i. Chaque vecteur représente des échantillons du signal d'entrée obtenus après une succession d'étapes et de méthodes permettant, entre autres, de réduire le nombre de paramètres. Le *Comparateur* compare le vecteur σ_i avec chacun des éléments de l'ensemble **R**= $\{\rho_1, \rho_2, \ldots, \rho_k\}$ qui contient des vecteurs prototypes sauvegardés dans le module de stockage *Prototype*.

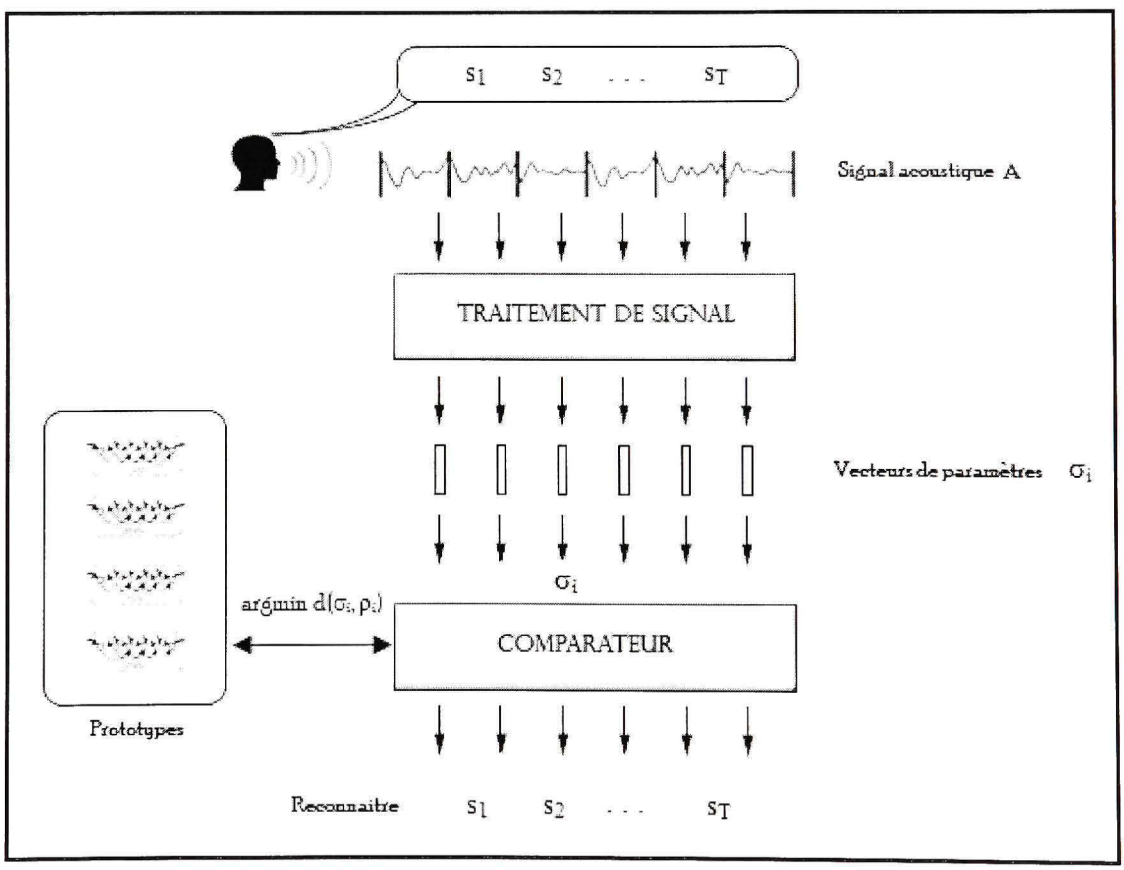

Figure-B II-7 Processus acoustique dans le cas des HMMs.

Les éléments de **R** sont de mêmes types que les vecteurs σ_i ce qui permet au *Comparateur* de calculer la distance entre σ_i et les éléments de **R.** Le vecteur qui possède la distance minimale sera retenu et son index sera le symbole acoustique $\hat{J}$ le plus probable.

$$x_i = \hat{J} = \arg \min_{J=1}^{K} d(\sigma_i , \rho_j) \qquad \text{(AII.5)}$$

La comparaison entre σ_i et les vecteurs de l'ensemble **R** repose sur l'idée d'agrégation. En effet, l'algorithme *Vector Quantization* ou encore *K-means Clustering* est utilisé afin d'identifier le vecteur de **R** le plus proche du vecteur réel.

B.4.3 Principe du modèle de Markov cachée

Afin de modéliser la probabilité P(A|W) on a besoin d'un modèle pour représenter les données. Le modèle de Markov caché est le modèle statistique le plus utilisé pour modéliser P(A|W). C'est un processus doublement stochastique dont une composante est une chaîne de Markov non observable. L'intéressant dans une chaine de Markov est que la probabilité pour chaque évènement de se produire ne dépond que du résultat de l'évènement qui le précède. Ainsi, si X1, X2, …, X_n est une séquence de variables aléatoires prenant ses valeurs d'un ensemble fini $\mathfrak{X}$= {1,2,…, N} alors :

$$P(X_1, X_2, \ldots, X_n) = \prod_{i=1}^{n} P\,(X_i \mid X_{i-1}) \qquad \text{(AII.6)}$$

Le modèle de Markov est un processus stochastique représenté par les paramètres suivants :

- **N**, le nombre des états du modèle :

  $S = \{s_1, s_2, \ldots, s_N\}$

- **T**, le nombre des symboles observations distincts

  $O = \{o_1, o_2, \ldots, o_T\}$

- **B**, la matrice des probabilités de transitions. Elle représente la distribution des probabilités des observations associées avec la transition de l'état s_t à l'état s_{t+1}.

$$B = b_j(k) = P(o_k \mid q_t = s_i) \tag{AII.7}$$

avec j= 1,2, … ,N ; k= 1,2, …, T

- **A**, Une distribution de probabilité de transition :

$$A = a_{ij} = P(q_{t+1} = s_j \mid q_t = s_i) \tag{AII.8}$$

$$\sum_{j=1}^{N} a_{ij} = 1 \tag{AII.9}$$

avec $1 \leq i \leq N$

- **Π**, une distribution des probabilités initiales des états :

$$\pi_i = P(q_1 = s_i) \tag{AII.10}$$

$$\sum_{i=1}^{N} \pi_i = 1 \tag{AII.11}$$

avec $1 \leq i \leq N$

Les distributions de probabilité A, B, Π construisent le modèle HMM désigné par λ:
λ= {A, B, Π}.

Le modèle HMM peut être utilisé comme un générateur pour donner une suite d'observations. La Figure-A I-8 montre un exemple de génération de six observations, de o_1 à o_6, par un HMM construit de six états.

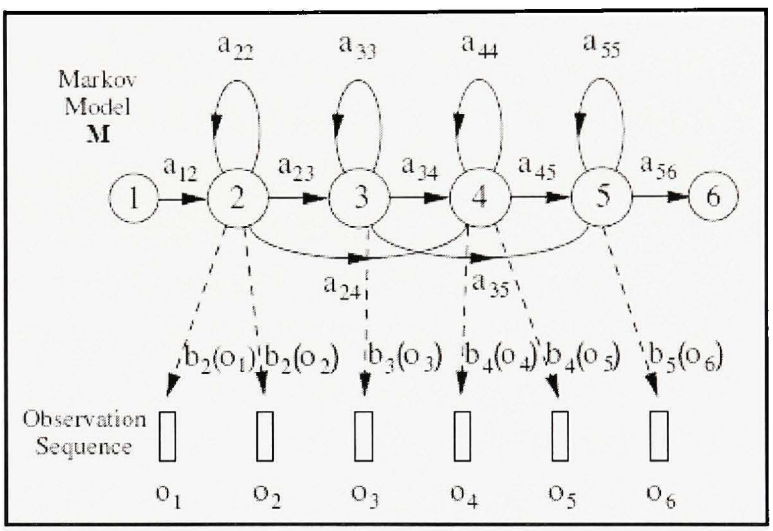

Figure-A I-8 Génération d'observation par un HMM
Tirée de Young, Odell et all. (1995, p. 4)

la séquence d'observation $O= o_1,o_2 \ldots,o_6$ peut être générée par différentes séquences d'états Q. Ainsi, la probabilité $P(O|\lambda)$ doit être calculée pour toutes les séquences $Q=q_1,q_2, \ldots, q_6$ possibles.

Dans le cas général, la séquence d'observation $O= o_1,o_2,..o_T$ la plus probable générée par le modèle λ, en se déplaçant parmi les états $Q= q_1,q_2 \ldots,q_T$, est calculée en multipliant les valeurs de probabilités de transitions a_{ij} par les probabilités d'observations $b_j(o_t)$. Ceci est donné par la formule suivante :

$$P(O,Q|\lambda) = b_{q1}(o_1)\, b_{q2}(o_2), \ldots, b_{qT}(o_T) \qquad \text{(AII.12)}$$

Toutefois, seulement la séquence d'observations O est connu alors que la séquence d'états est cachée, pour cette raison le modèle est appelé modèle de Markov caché. La séquence d'observations sachant un modèle de Markov sera alors calculée comme suit :

$$\sum_{q=1}^{T} \pi_{q_1} a_{q_1 q_2} b_{q_1}(o_1) a_{q_2 q_3} b_{q_2}(o_2) \ldots a_{q_{T-1} q_T} b_{q_1}(o_T) \qquad \text{(AII.13)}$$

Cependant, c'est mieux de ne pas utiliser cette formule puisqu'elle consomme beaucoup de mémoire et effectue trop de calcul. En effet, avec T observations et N états dans le modèle, il existe N^T séquences d'états possible, ce qui nécessite environ $2TN^T$ opérations. Ce problème peut être résolu en utilisant l'algorithme Forward qui sera détaillé dans ce qui suit.

Le modèle HMM peut être représenté aussi sous forme d'un treillis où chaque nœud est une observation générée dépendamment de la transition entre les états (voir Figure-B II-9). Ainsi, la probabilité $P(o_1, o_2, \ldots, o_T)$ peut être calculée en mettant en évidence l'évolution, dans le temps, du processus pour générer la séquence $o_1, o_2, \ldots, o_T$. Schématiquement chaque niveau représente une observation, ainsi il aura autant de niveaux que de différentes observations.

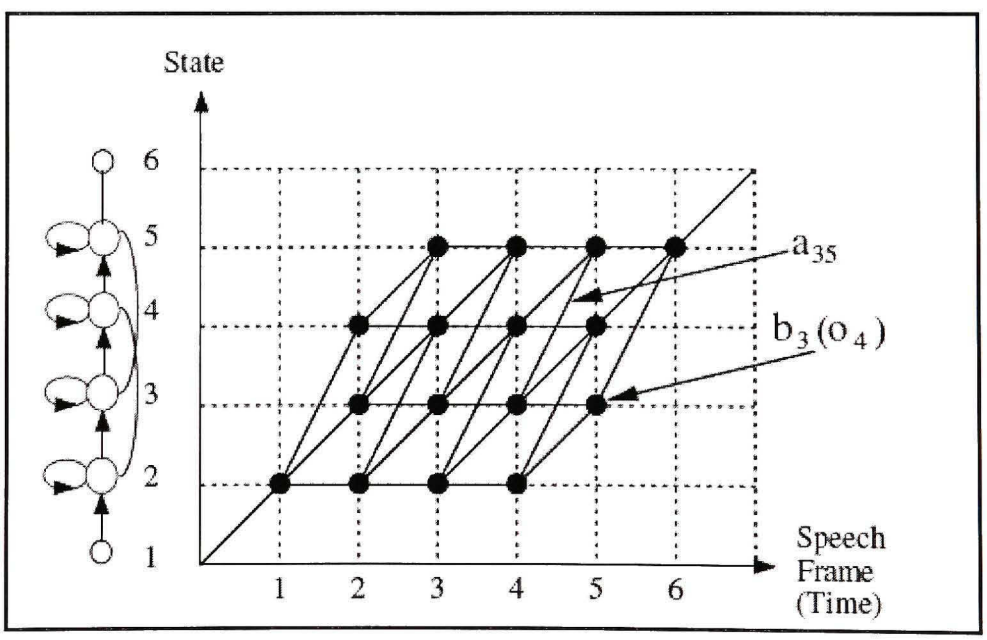

Figure-B II-9 Représentation d'HMM sous forme de treillis.
Tirée de Young, Odell et all. (1995, p. 10)

La probabilité $P(o_1, o_2, \ldots, o_T)$ est égale à la somme de tous les chemins (depuis l'état initial $s_0 = 1$ jusqu'à l'état final) dans le treillis.

B.4.4 Les problèmes fondamentaux d'un HMM

Le modèle de Markov caché présente trois problèmes fondamentaux au nivaux de son design. La résolution de ces problèmes permet l'application des HMMs dans des applications réalistes de reconnaissance automatique de la parole.

B.4.4.1 Évaluation du modèle

Étant donnée une suite d'observations $O= o_1,o_2,..o_T$, comment peut-on évaluer le modèle $\lambda= \{A, B, \Pi\}$ afin de déterminer parmi plusieurs, le mieux adapté pour générer cette suite d'observation. À part la méthode directe utilisant la formule AII.13, d'autres méthodes peuvent être utilisées pour calculer $P(O|\lambda)$:

L'algorithme Forward

L'algorithme Forward calcule les probabilités partielles $\alpha_t(i)$ de chaque état, cette dernière correspond à la probabilité de tous les chemins précédant menant à cet état (Figure-B II-10). Les probabilités partielles correspondent à la probabilité d'atteindre les états concernés en parcourant tous les chemins possibles.

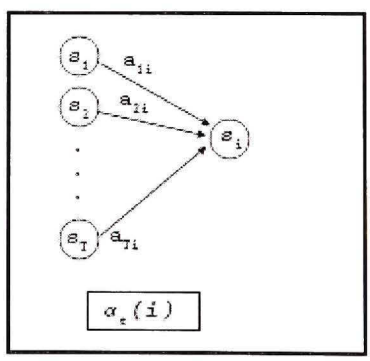

Figure-B II-10 Algorithme Forward.

La somme de ces probabilités correspond à la probabilité de la séquence d'observation sachant le modèle HMM en question. Les étapes de l'algorithme Forward sont :

1) Initialisation :

$$\alpha_1(i) = \pi_i b_i(o_1) \tag{AII.14}$$

2) Induction :

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij} \right] b_j(o_{t+1}) \tag{AII.15}$$

3) Terminaison :

$$P(O \mid \lambda) = \sum_{i=1}^{N} \alpha_T(i) \tag{AII.16}$$

avec $1 \leq i \leq N$, $1 \leq j \leq N$, $1 \leq t \leq T - 1$

L'algorithme Backward

L'algorithme de Backward se base sur le même principe que l'algorithme Forward sauf que les probabilités partielles seront calculées dans la direction inverse Figure-B II-11.

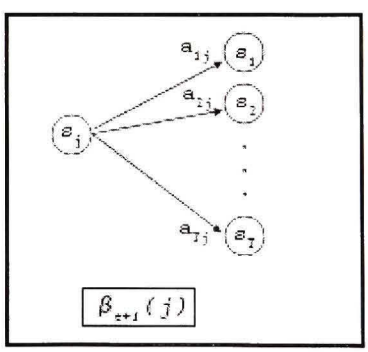

Figure-B II-11 Algorithme de Backward.

Voici les étapes de cet algorithme :

1) Initialisation :

$$\beta_1(\mathrm{i}) = 1 \qquad \text{(AII.17)}$$

2) Induction :

$$\beta_t(\mathrm{i}) = \sum_{\mathrm{j}=1}^{\mathrm{N}} a_{\mathrm{ij}} \mathrm{b}_{\mathrm{j}}(\mathrm{o}_{t+1}) \beta_{t+1}(j) \qquad \text{(AII.18)}$$

3) Terminaison :

$$P(O \mid \lambda) = \sum_{i=1}^{N} \pi_i b_i(o_i) \beta_1(i) \qquad \text{(AII.19)}$$

avec $1 \leq \mathrm{i} \leq \mathrm{N}$, $1 \leq \mathrm{j} \leq \mathrm{N}$, $\mathrm{t}=\mathrm{T}-1, \mathrm{T}-2, \ldots, 1$

Une autre méthode peut être utilisée pour résoudre ce problème en se basant sur l'algorithme de Viterbi. Ce dernier va être présenté dans le paragraphe suivant comme une solution pour le deuxième problème d'un HMM.

B.4.4.2 Estimation de la suite d'états cachés

Dans la reconnaissance automatique de la parole il est important d'associer à une séquence d'observations une séquence d'états étant donné un modèle HMM. L'algorithme de Viterbi vient pour faire face à ce problème de décodage en cherchant la séquence d'états qui maximise la probabilité $P(O,Q|\lambda)$. En effet, cet algorithme fonctionne de la même façon que l'algorithme Forward avec la seule différence que ce premier remplace la sommation des probabilités partielles par la probabilité maximale. L'idée de base est de chercher pour chaque état *s* du niveau *i* du treillis la séquence la plus probable $s_1(s)$, …, $s_{i-1}(s)$ menant à l'état *s*, puis chercher la séquence la plus probable $s_{i+1}(s)$, …, $s_k(s)$ suivant l'état *s*. Finalement, déterminer l'état *s* dans le niveau i dont la séquence complète $s_1(s)$, …, $s_{i-1}(s)$,

s_i=s, s_{i+1}(s), …, s_k(s) présente la probabilité la plus élevé. Tout ce calcul peut être réalisé d'une façon récursive en se basant sur l'algorithme de Viterbi.

Soit :

- $\delta_t(i)$: la probabilité maximale d'une séquence au temps t allant à l'état s_j.
- $\psi_t(j)$: la variable contenant le chemin le plus court correspondant au nœud q_t=s_i.

L'algorithme de Viterbi fonctionne comme suit :

1) Initialisation :

$$\delta_1(\mathrm{i}) = \pi_i b_i(o_1) \tag{AII.20}$$

$$\psi_1(i) = 0 \tag{AII.21}$$

2) Recursion :

$$\delta_t(\mathrm{j}) = \max_{1\le i\le N} \left[\delta_{t-i}(i) a_{ij}\right] b_j(o_t) \tag{AII.22}$$

$$\psi_t(\mathrm{j}) = \operatorname*{argmax}_{1\le i\le N} \left[\delta_{t-i}(i) a_{ij}\right] \tag{AII.23}$$

3) Terminaison :

$$P^* = \max_{1\le i\le N} \left[\delta_T(i)\right] \tag{AII.24}$$

$$q_t^* = \operatorname*{argmax}_{1\le i\le N} \left[\delta_T(i)\right] \tag{AII.25}$$

4) Meilleure séquence :

$$q_t^* = \psi_{t+1}(q_{t+1}^*) \qquad \text{(AII.26)}$$

B.4.4.3 Apprentissage

Le modèle HMM doit passer par une phase d'apprentissage au cours de laquelle les paramètres du modèle $\lambda = \{A, B, \Pi\}$ seront entraînés afin de maximiser la probabilité $P(O|\lambda)$. On a donc besoin d'échantillons de données, ceci est nécessaire pour construire un modèle permettant une représentation de nouvelles données de mêmes types. Les éléments de bases de ce modèle sont la structure de transitions et les paramètres statistiques du HMM. La bonne connaissance de la situation et l'intuition sont les deux facteurs qui permettent la bonne conception de ces éléments et l'estimation des valeurs des paramètres initiaux, car il n'existe aucune méthode universelle pour le faire. Après l'estimation initiale des paramètres, on peut utiliser l'algorithme Baum-Welch. Comme le montre la Figure-B II-12, ce dernier se sert de l'algorithme Forward-Backward.

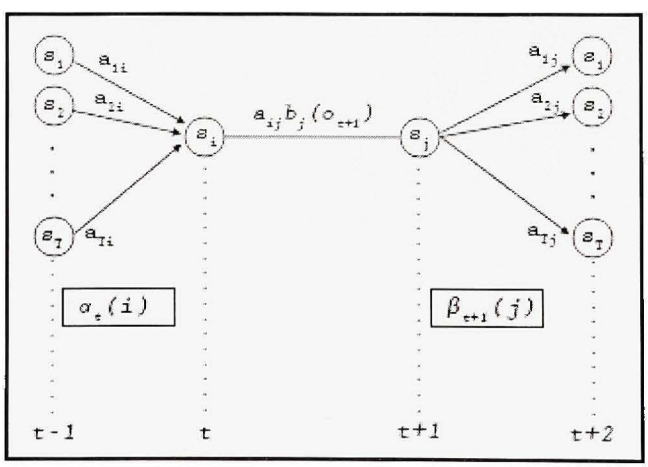

Figure-B II-12 Algorithme Baum-Welch.

Soit $\hat{\lambda}$ le modèle optimisé de λ qui donnera une bonne représentation statistique des données O observées. Il est judicieux de dire qu'il n'est pas évident d'avoir des paramètres $\hat{\lambda}$

optimaux, mais plutôt satisfaisants, car ceci dépend fortement de la représentativité des données d'entrainement. Pour utiliser l'algorithme Baum-Welch, définissions la probabilité de transiter de i vers j au temps t sachant l'observation O et le modèle λ par :

$$\xi_1(i,j) = P(q_t = s_i, q_{t+1} = s_J \mid O, \lambda) = \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{P(O \mid \lambda)} \qquad \text{(AII.26)}$$

À partir de l'équation AII.27 on pourra ré-estimer les paramètres comme suit :

- Probabilité des états initiaux :

$$\hat{\Pi} = \sum_{j=1}^{N} \xi_1(i,j) \qquad \text{(AII.27)}$$

- Probabilité de transition :

$$\hat{A} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \sum_{j=1}^{N} \xi_t(i,j)} \qquad \text{(AII.28)}$$

- Probabilité d'observation :

$$\hat{b}_j(k) = \frac{\sum_{t:o_t=x_k}^{T-1} \sum_{i=1}^{N} \xi_t(j,i)}{\sum_{t=1}^{T-1} \sum_{j=1}^{N} \xi_t(j,i)} \qquad \text{(AII.29)}$$

Algorithme

1) Initialisation : estimer les paramètres initiaux de $\lambda = \{A, B, \Pi\}$

2) Calculer les probabilités :

$$\xi_1(i, j) \tag{AII.30}$$

$$\alpha_t(i) \tag{AII.31}$$

$$\beta_t(i) \tag{AII.32}$$

3) Ré-estimer les paramètres.

4) Répéter les étapes 2 et 3 jusqu'à la convergence.

B.4.5 Modèle acoustique composite

Dans le paragraphe précédant on a vu que chaque unité acoustique peut être modélisée par un HMM. Dans la pratique, chaque mot w_i va être modélisé en concaténant plusieurs petits modèles HMMs afin de former un modèle global représentant la séquence de mots W. la méthode la plus utilisée pour concevoir un modèle acoustique de toute la séquence de mots est le modèle acoustique phonétique (Jelinek 1997). La Figure-B II-13 illustre un cas général pour un modèle acoustique de la séquence de mots W.

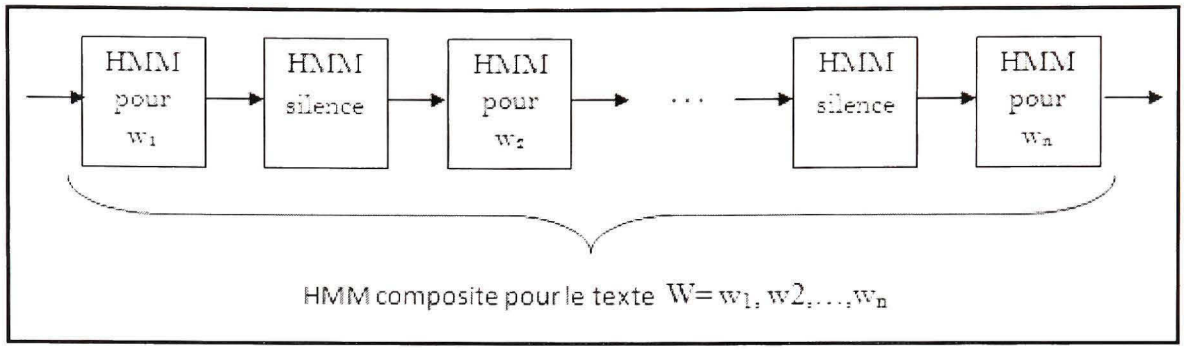

Figure-B II-13 HMM composite.

Le modèle acoustique phonétique peut être représenté comme une succession d'étapes :

1) Créer un dictionnaire phonétique pour le vocabulaire en question mettant ainsi en évidence la correspondance entre chaque mot et une séquence de symboles bien déterminée.
2) Associer pour chaque symbole de l'alphabet phonétique un modèle *élémentaire* HMM commençant par un état initial et se terminant par un état final.
3) Concaténer les différents modèles élémentaires HMM pour former un mot.
4) Pour avoir un modèle composite de la transcription W à partir des données acoustiques A, il faut concaténer les différents mots w_i générés et insérer entre eux un modèle élémentaire HMM représentant un silence.
5) Utiliser l'algorithme Baum-Welch pour estimer les paramètres statistiques du modèle HMM.

L'étape 5 nécessite beaucoup plus d'attention et doit être étudiée plus en profondeur, pour plus de détails voir (Jelinek 1997).

ANNEXE III

MODÈLE DE LANGAGE

C.1 Le modèle de langage trigramme

Il est très difficile voir impossible d'évaluer la probabilité P (w_i| w_1, …, w_{i-1}) pour n>3, donc il faut chercher une façon réalisable pour calculer la probabilité de prononcer le mot w_i sachant un historique donné. De toute façon, si on prend en compte tout l'historique on risque d'avoir une probabilité très faible ou même nulle. Cependant, on peut utiliser un modèle trigramme qui prend en compte juste les deux derniers mots qui précédent le mot actuel. La probabilité P(W) sera calculée alors comme suit :

$$P(w) = \prod_{i=1}^{n} P(w_i \mid w_{i-2}, w_{i-1}) \qquad \text{(AIII.1)}$$

et

$$P(w_3 \mid w_1, w_2) = \frac{c(w_1, w_2, w_3)}{c(w_1, w_2)} \qquad \text{(AIII.2)}$$

Même cette méthode de calcul peut donner des probabilités nulles en-cas où le trigramme n'existe pas dans le corpus d'entraînement. En effet, une étude menée par des chercheurs de IBM en 1970 montre que 23% des trigrammes qui ont apparu dans la phase de test, n'existent pas dans le corpus d'entraînement (Jelinek 1997). Donc, la meilleure façon sera d'introduire différents modèles en même temps : trigrammes, bigrammes, unigrammes :

$$P(w_3 \mid w_1, w_2) = \lambda_3 f(w_3 \mid w_1, w_2) + \lambda_2 f(w_3 \mid w_2) + \lambda_1 f(w_3) \qquad \text{(AIII.3)}$$

avec $\lambda_1 + \lambda_2 + \lambda_3 = 1$

Plusieurs méthodes, tel que la technique de lissage linéaire (*deleted interpolation*), peuvent être utilisées pour calculer les poids λ_1, λ_2 et λ_3. Cette méthode consiste à diviser les données d'entraînement en deux portions inégales. La première, qui est la plus grande, sera utilisée pour estimer la fréquence relative f(|). Alors que la deuxième ne nécessite pas beaucoup de données relativement à la première, et elle est utilisée pour estimer les poids λ_i.

Cette technique introduit de nouvelles probabilités de transitions d'où les relations :

$$\lambda_1 = \lambda'_4 * \lambda'_1$$

$$\lambda_2 = \lambda'_4 * \lambda'_2 \qquad \text{(AIII.4)}$$

$$\lambda_3 = \lambda'_3$$

avec $\lambda'_2 = 1 - \lambda'_1$ et $\lambda'_4 = 1 - \lambda'_3$

Les valeurs de λ'_i peuvent être ensuite évaluées par l'algorithme de Baum en mettant en évidence les transitions nulles. Pour plus de détails se référer à (Jelinek 1997).

C.2 Évaluation d'un modèle de langage

La meilleure façon pour évaluer un modèle de langage est de le tester dans un système de reconnaissance automatique de la parole. Cependant, cette approche est très difficile est nécessite l'entraînaient du modèle de langage à tester ainsi que le modèle acoustique. Pour éviter cette méthode, on peut utiliser la perplexité pour mesurer la qualité d'un modèle de langage. Le calcul de cette dernière peut se faire par la formule suivante donnée par Jelinek (Jelinek 1997) :

$$PP_M(T) = 2^{-\frac{1}{N}\sum_e \log_2(P_M(e))} \qquad \text{(AIII.33)}$$

$PP_M(T)$ représente la qualité de prédiction d'un événement *e* par le modèle M sur le corpus T contenant N mots. Plus cette valeur est petite, plus la capacité de prédiction du modèle de langage est bonne. Des expériences ont montré que lorsque la taille du corpus d'apprentissage augmente la perplexité décroît de manière uniforme (Chen, Gauvain Jean-Luc et al. 2001).

ANNEXE IV

ARCHITECTURE DE SPHINX-4

L'architecture de Sphinx-4 (Walker, Lamere et al. 2004) peut être décrite comme étant un ensemble d'éléments interagissant les uns avec les autres. Chaque élément, ou module, opère d'une façon indépendante et se comporte comme un petit système qui reçoit les données comme entrée pour les traiter et ensuite fournir un résultat. Cette vue orientée objet ajoute une grande flexibilité au système global en permettant les chercheurs de traiter chaque module à part et de le modifier sans influencer les autres modules. Sphinx-4 comporte trois modules principaux : Le Front-End (*FrontEnd*), le Décodeur (*Decoder*) et la Base de connaissances (*Linguist*). La Figure-D IV-1 illustre son architecture de base :

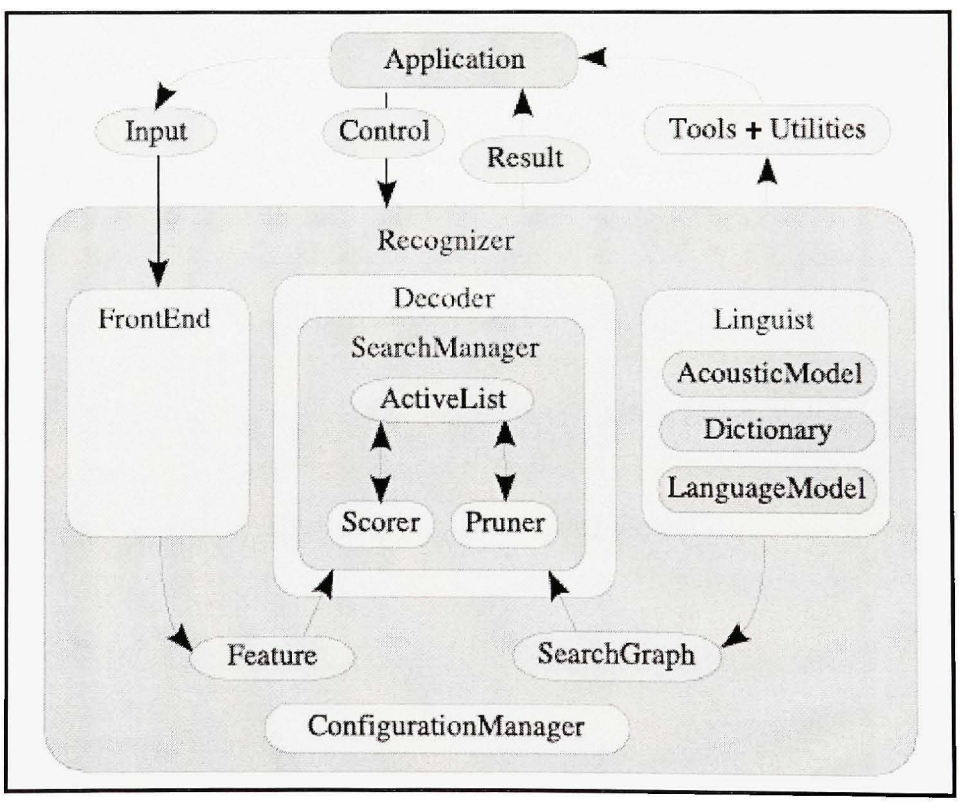

Figure-D IV-1: Architecture de Sphinx-4.
Tirée de Walker et al. (2004, p. 2)

Comme représenté par la Figure-D IV-1, chaque module reçoit en entrée des données et fournit en sortie un résultat qui va servir comme une entrée pour un autre module. En effet, le module Front-End acquiert le signal acoustique de l'application de l'utilisateur pour le traiter et fournir une séquence de paramètres. Le module *base de connaissances* comprend trois éléments coopérant ensemble afin de produire le graphe recherché (*SearchGraph*). Le troisième module, le décodeur, utilise le graphe et les paramètres fournis par les deux autres modules pour les traiter et ensuite générer un résultat à l'application. Ces trois modules, décrits ci-dessus, construisent la base du système Sphinx-4. Cependant, d'autres éléments accomplissent des tâches secondaires nécessaires à l'administration, la configuration et le contrôle du système. En effet, des contrôles peuvent êtres exercés sur chaque module à tout moment du processus. En outre, les différents paramètres du système peuvent être ajustés par le gestionnaire de configuration (*ConfigurationManager*) d'une façon dynamique et en temps réel. Par exemple, il est possible de changer la configuration par défaut du module Front-End en changeant la sortie des paramètres. Par la suite, ce dernier au lieu de produire des coefficients Mel-Frequency Cepstral Coefficient (MFCC), il produira des coefficients Perceptual Linear Prediction (PLP) sans modifier le code source du système. Sphinx-4 fournit aussi un ensemble d'outils permettant de faire des calculs statistique, calcul de la vitesse d'exécution, la consommation de la mémoire, etc. Ces outils peuvent êtres utilisés d'une façon dynamique au cours de l'exécution du système, permettant ainsi de gagner de temps. En effet, au cours du processus de reconnaissance, le décodeur peut envoyer des évènements à l'application alors qu'il est en train de travailler sur la recherche. Ces évènements permettent à l'application de bien superviser le processus de reconnaissance et d'interagir avec le décodeur avant même la fin du processus. Le système fournit également des utilitaires qui soutiennent l'application au niveau du traitement des résultats de la reconnaissance.

D.1 Le module Front-End

La fonction principale du module Front-End est de traiter les signaux d'entrées afin de fournir une séquence de paramètres. En effet, ce module a été développé de façon à lui

permettre de décoder simultanément différents types de paramètres tels que MFCC, PLP ou même des paramètres du signal vidéo. Comme illustré par la Figure-D IV-2, le module Front-End est composé d'une ou plusieurs chaînes de communications appelées *DataProcessor*. Ces derniers sont reliés les uns avec les autres par des objets appelés *Data* qui encapsulent les données traitées ainsi que des marqueurs indiquant les données de classification des évènements notamment le détecteur du point final.

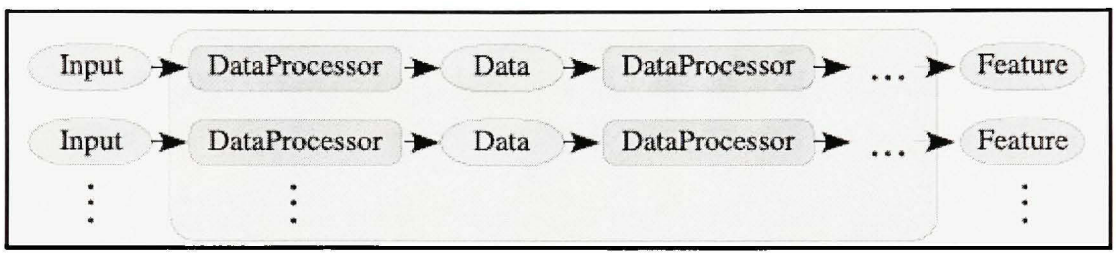

Figure-D IV-2: Chaine de communication dans le module Front-End.
Tirée de Walker et al. (2004, p. 3)

Cette conception présente beaucoup d'avantages :

- En plus de la possibilité de traiter simultanément plusieurs types d'entrés, et par la suite fournir différents types de paramètres, le nombre de flux parallèle est arbitraire.
- La communication entre les différents éléments de ce module a été conçue pour que le *DataProcessor* demande les données, de l'élément précédant, qu'en cas de besoin (*pull design*[17]).

La Figure-D IV-3 montre plus clairement le processus de traitement de données et d'extraction des paramètres acoustiques dans ce module.

[17] Contrairement à pull design, Push design permet au *DataProcessor* de diffuser ses données de sortie dès qu'il est généré.

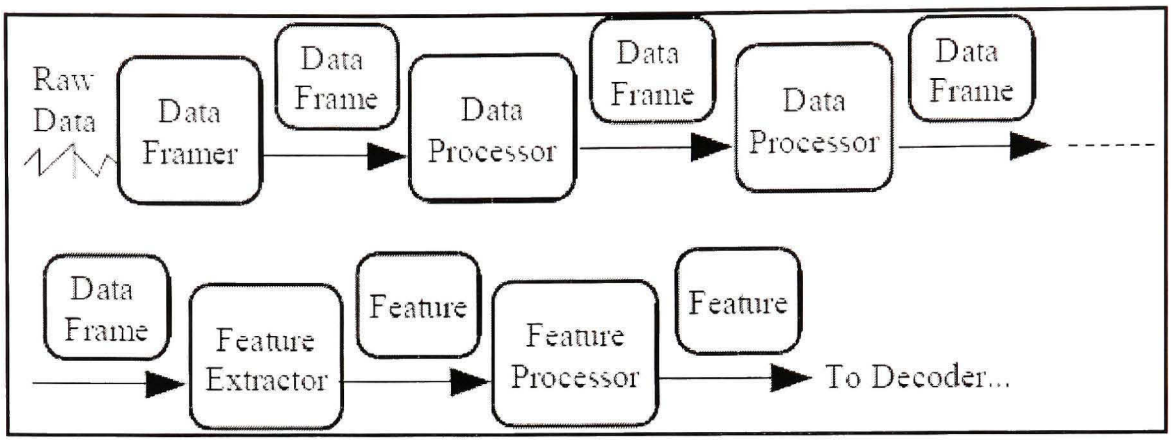

Figure-D IV-3: Traitement des données et extraction des paramètres.
Tirée de Gouvea, Kwok et al. (2002)

Le Front-End reçoit en entrée des données brutes qui seront analysées et traitées pour fournir en sortie des paramètres acoustiques. En effet, Le *Data Framer* détecte les flux de données afin de les encapsuler dans des packages annotés *Data Frame*. Ces derniers, contiennent des informations concernant les paquets de données telle que la nature des données, si c'est un segment de début ou de fin, etc. Des techniques de traitement du signal sont implémentées à l'intérieur du *DataProcessor* et sont appliquées sur les données. Ces implémentations sont des techniques de: fenêtrage du signal, de pré-emphase, Transformée de Fourier Discrète (FFT), etc. Une phase d'extraction de paramètres succède la fin de cette phase de traitement. En effet, la composante *Feature Extractor* extrait les paramètres acoustiques (*Feature*) nécessaires à l'accomplissement du travail du décodeur.

D.2 Le module base de connaissances

Le module base de connaissance génère un graphe (*SearchGraph*) qui sera utilisé par le module décodeur afin de déterminer les expressions prononcées. Ce dernier lui envoie des feedbacks lui permettant ainsi de s'ajuster en se basant sur les résultats de recherches successives. Il est possible que la génération du graphe soit faite par un sous-module contenu dans le module décodeur (Paul, Philip et al.). Cependant, le principe reste le même et le résultat est obtenu en exploitant les données contenues dans le module *Base de Connaissances* qui est composé de ces trois éléments :

- **Modèle de Langage** : il permet de modéliser ce qui peut être dit dans un contexte bien déterminé en fournissant une structure représentée par un certain nombre d'implémentations.
- **Dictionnaire** : le dictionnaire fournit la prononciation de chaque mot contenu dans le module Modèle de Langage. Les prononciations placent les mots dans une séquence d'unités contenues dans le Modèle Acoustique.
- **Modèle Acoustique** : le Module Acoustique permet de faire une correspondance entre une unité de la parole prononcée et un modèle de Markov caché (HMM) à partir des paramètres fournis par le module Front-End.

D'une façon générale, le module *Base de Connaissance* transforme chaque mot, contenu dans le vocabulaire actif, en une séquence de sous-unité de mot dépendamment du contexte. Le Modèle Acoustique reçoit ensuite ces unités avec leurs contextes associés et fournit par la suite le graphe HMM correspondant à ces unités. En se basant sur le modèle de langage et le modèle HMM, il est possible de construire le graphe représentant l'expression prononcée. La Figure-D IV-4 illustre un exemple de graphe résultant du module *Base de Connaissance* qui représente le premier élément utilisé par le module décodeur dans le processus de décodage.

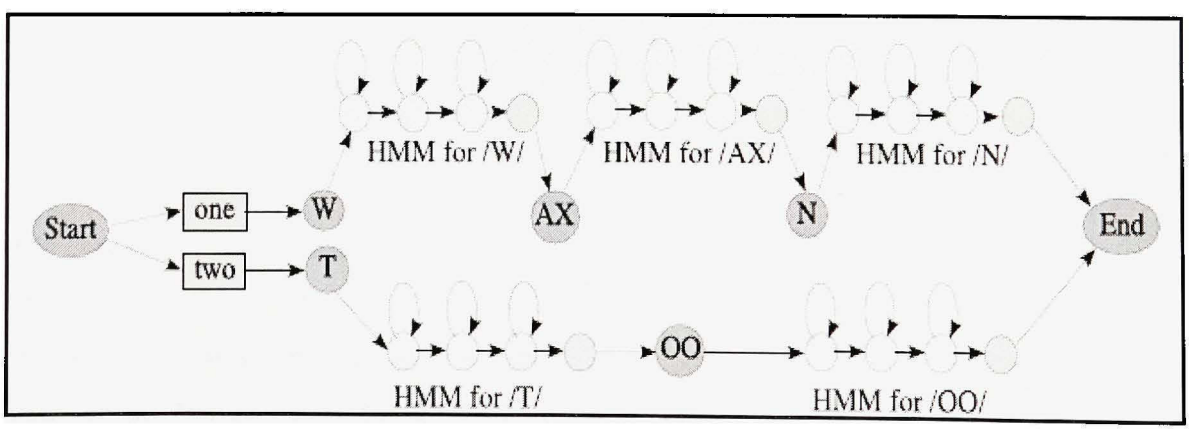

Figure-D IV-4: Exemple d'un graphe résultant du module Base de connaissance.
Tirée de Walker et al. (2004, p. 5)

Chaque nœud dans le *SearchGraph*, appelé *SearchState*, peut être un nœud émetteur ou non-émetteur dépendamment des caractéristiques acoustiques. Les mots dans les rectangles (voir

Figure-D IV-4) sont des composants du modèle de langage. Ces mots sont fragmentés en quelques unités pour formés le dictionnaire, ceux-ci sont représentées par les lettres dans les cercles sombres. Des HMMs élémentaires modélisant ces unités contenues dans le dictionnaire représentant ainsi le modèle acoustique. Les modèles HMMs sont représentés dans la mémoire comme des structures statiques dans la majorité des systèmes de reconnaissances vocales. Cependant, Sphinx-4 représente les HMMs élémentaires comme étant des graphes d'objets. Chaque nœud correspond donc à un état du modèle HMM et chaque arc représente la probabilité de transition d'un état à un autre dans ce modèle. Cette conception permet d'implémenter un modèle acoustique flexible qui peut fournir des HMM avec différentes typologies.

D.3 Le décodeur

En se basant sur les paramètres acoustiques fournis par le module Front-End, le graphe résultant du module base de connaissance et les feedback échangés avec l'application, le module décodeur génère le résultat de la reconnaissance de l'expression prononcée. Le résultat (*Result* sur la Figure-D IV-1) est obtenu suite à des recherche pour déterminer la séquence de mot la plus probable qui peut correspondre aux vecteurs de paramètres acoustique. Ce module est composé de plusieurs éléments dont principalement le *SearchManager*. Ce dernier piloté par le décodeur, génère à chaque étape du processus un objet *Result* qui contient tout les chemins qui mènent à un état non émetteur. Cet objet peut être traité ensuite par les utilitaires afin de déterminer les scores de confidence par exemple. *SearchManager* est implémenté en se basant sur le principe de jeton comme décrit par Young (Young, Russell et al. 1989). Dans Sphinx-4, un jeton est représenté par un objet associé avec un nœud *SearchState* et contient : l'ensemble des scores acoustique et de langage du chemin à un point donnée, une référence du nœud *SearchState*, une référence à l'entrée des paramètres acoustiques et d'autres informations pertinentes. Le *SearchManager* peut construire plusieurs jetons au cours du processus, ces derniers seront ajoutés à une liste

appelée *ActiveList* en utilisant le composant *Prruner*[18]. Le composant *Scorer* communique avec le *SearchManager* afin de lui fournir, sur demande, des informations sur la valeur de densité des états. En effet, ce dernier peut calculer le score d'un état donné à un instant donné en accédant au vecteur de paramètres.

[18] Son implémentation est simplifiée par l'utilisation du Garbage Collector de la plateforme Java. Ce dernier peut nettoyer un chemin simplement en enlevant le jeton terminal de la liste *ActiveList*.

ANNEXE V

ARCHITECTURE DE HTK

E.1 Vue générale

Cette section est conservée pour la présentation de l'architecture de HTK en se basant sur la le manuel d'utilisation de HTK (Young, Odell et al. 1995).

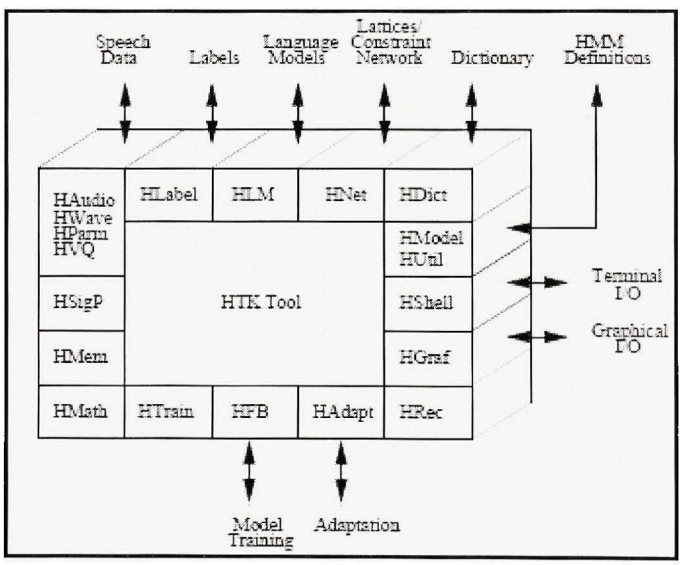

Figure-E V-1: Architecture de HTK.
Tirée de Young et al. (1995, p. 15)

Comme représenté par la Figure-E V-1, le système HTK est composé d'une vingtaine de modules encapsulant des fonctions et des fonctionnalités différentes. Les interactions avec le milieu extérieur sont assurées par des liens entre HTK et les données et modules externes. En effet, les interactions avec le système d'exploitation sont gérées par le module HShell, alors que les entrées/sorties graphiques sont dirigées par le module HGraf. D'autres modules fournissent des interfaces de gestion et de manipulations : HLabel pour les fichiers d'étiquettes, HLM pour les fichiers du modèle de langage, HNet pour le réseau, HDict pour

les dictionnaires, HVQ pour les VQ *codebook* et HModel pour la définition des HMMs. En outre, toutes les entrées/sorties de nature vocale sont gérées par le module HWave et paramétrés via HParm. Toutefois, les flux de données audio directs sont transmis par le module HAudio. Comme le module HLabel, HWave supporte différents types de fichiers permettant la communication avec d'autres systèmes. Des utilités pour la manipulation des HMMs sont regroupées dans le module HUtil, alors que les outils et les supports d'entraînements sont fournis par HTrain et HFB. En outre, le module HAdapt renforce les différents outils d'adaptation de HTK. Le dernier module Hrec est le plus important puisqu'il contient les fonctions principales du processus de reconnaissance.

L'exécution des outils de HTK se fait d'une façon traditionnelle avec des lignes de commandes sous DOS. Ainsi, le paramétrage des outils est garanti par les arguments qui peuvent modifier le fonctionnement de l'outil. En effet, chaque commande nécessite un (ou plusieurs) argument(s) principal (s) et peut être complétée par des arguments optionnels. Ces derniers sont désignés par des lettres en majuscules pour dire qu'ils ont le même fonctionnement pour toutes les commandes. D'une façon générale, les arguments optionnels sont suivis par leurs valeurs et sont séparés par un espace. Cependant, il y des arguments qui ne nécessitent pas de valeurs, car ils agissent comme un drapeau pour activé ou désactivé quelques fonctionnalités de l'outil en question.

Il existe un autre moyen pour le paramétrage des outils qui se résument dans les fichiers de configurations. Ces derniers sont utilisés principalement pour le contrôle du comportement des modules de bibliothèques dont tous les outils de HTK dépendent. Par exemple, la commande Hfoo permet le chargement des paramètres contenus dans le fichier de configuration config. En outre, l'option –C permet de charger de multiples fichiers de configuration.

E.2 Étapes de reconnaissance de la parole dans HTK

Le processus de reconnaissance de la parole continue dans HTK comprend 4 phases principales. La Figure-E V-2 illustre ces 4 phases ainsi que les outils utilisés dans chacune d'entre elles.

E.2.1 Phase de préparation des données

Cette phase concerne la préparation et l'organisation des données qui vont être utilisées pour l'entraînement et le test du système. Ainsi, toutes les données doivent être converties dans un format unique et leurs transcriptions associées doivent être bien échiquetées. Les données vocales peuvent être obtenues à partir des bases de données appropriées ou bien en utilisant l'outil HSlab qui permet l'enregistrement et l'étiquetage manuel des signaux acoustiques. L'outil HCopy permet de rassembler plusieurs types de fichier en un seul fichier. Par exemple, différents fichiers audio, un fichier de configuration et un autre de script peuvent êtres rassemblée dans un seul fichier MLF. L'outil HList est exploité principalement pour examiner le contenu des fichiers vocaux. HLEd est un outil utilisé pour manipuler et convertir rapidement et efficacement les fichiers d'étiquetages. Des statistiques concernant ces derniers peuvent êtres réalisées par l'outil HLStats. Finalement, les VQ *codebook* qui sont utilisées pour construire un système de probabilité discrète sont préparées par l'outil HQuant.

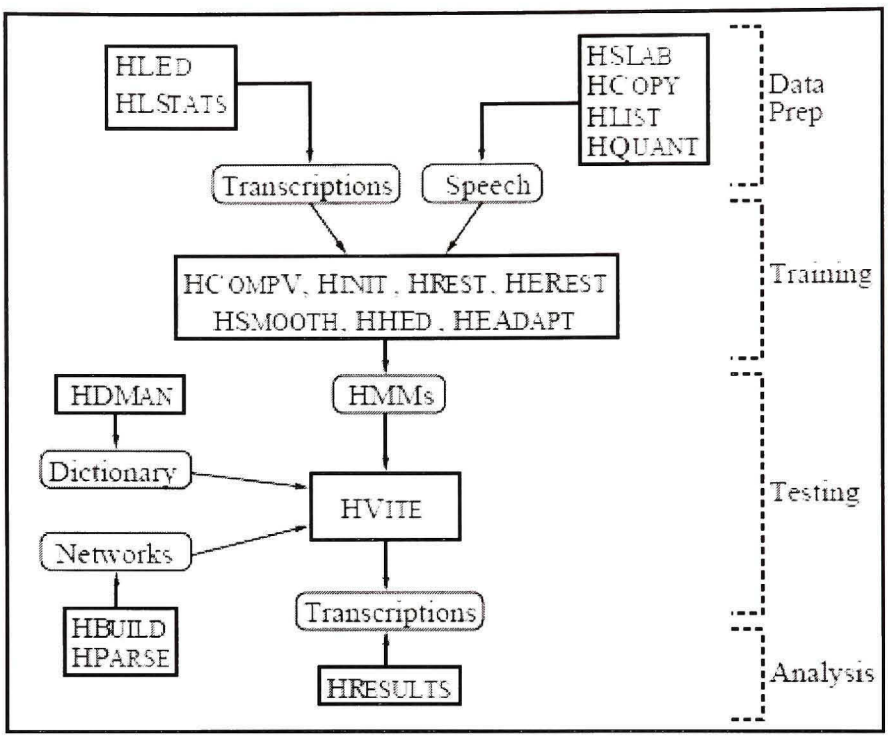

Figure-E V-2: Processus de reconnaissance de la parole continue dans HTK.
Tirée de Young et al. (1995, p. 17)

E.2.2 Phase d'entraînement

Cette phase assure la définition des topologies nécessaires à la modélisation de chaque HMM du modèle acoustique du système. Par défaut la distribution standard de HTK inclut plusieurs prototypes de HMM et de scripts qui permettent de générer la majorité des topologies automatiquement. Toutefois, il est possible de définir une topologie selon les besoins et la sauvegarder dans un fichier texte afin de les manipuler ultérieurement.

La Figure-E V-3 illustre le processus d'entraînement ainsi que les différents outils de HTK nécessaires au bon déroulement de cette phase. L'entraînement du système est un processus récursif qui débute par la création d'un ensemble initial de modèles. À ce niveau, le choix d'outils qui seront utilisés dépend des données existantes. En effet, si les données ne sont pas étiquetées l'outil HCompV sera utilisé afin d'initialiser tous les modèles. Cette initialisation

rend identiques tous les modèles et affecte à leurs moyenne et variance les valeurs de ceux du système global. Dans le cas où les données sont entièrement étiquetées et les extrémités de chaque mot sont bien définies, les outils HInit et HRest seront utilisés afin de générer individuellement chaque modèle HMM. En effet, HInit fournit des estimations des paramètres des modèles d'une façon itératives en utilisant les algorithmes k-means et Viterbi. Ensuite, l'algorithme Baum-Welch est utilisé par l'outil HRest afin de ré-estimer les paramètres.

Après la génération d'un premier ensemble de données, l'outil HERest doit être utilisé afin de ré-estimer les paramètres. L'algorithme Baum-Welch est utilisé cette fois aussi, mais la ré-estimation correspond à l'ensemble de tous les modèles HMMs simultanément. Ainsi, les modèles HMMs des phonèmes correspondant à une expression donnée sont reliés ensemble. Puis l'algorithme forward-backward est utilisé pour calculer les variables statistiques notamment la moyenne et la variance, qui seront utilisées par la suite pour la ré-estimation des paramètres des modèles HMMs.

La construction d'un système de reconnaissance de la parole pour un contexte donné avec HTK nécessite l'utilisation de l'outil HHEd à la fin de cette phase. Cet outil permet de cloner les modèles HMMs déjà générés puis les modifier, en ajoutant/supprimant des transitions, pour enfin ré-estimer de nouveau les paramètres pour un contexte donné. Le problème principal dans ce cas est l'insuffisance des données. Cependant, l'outil HSmouth présente une solution à ce problème par l'application d'un lissage à la distribution.

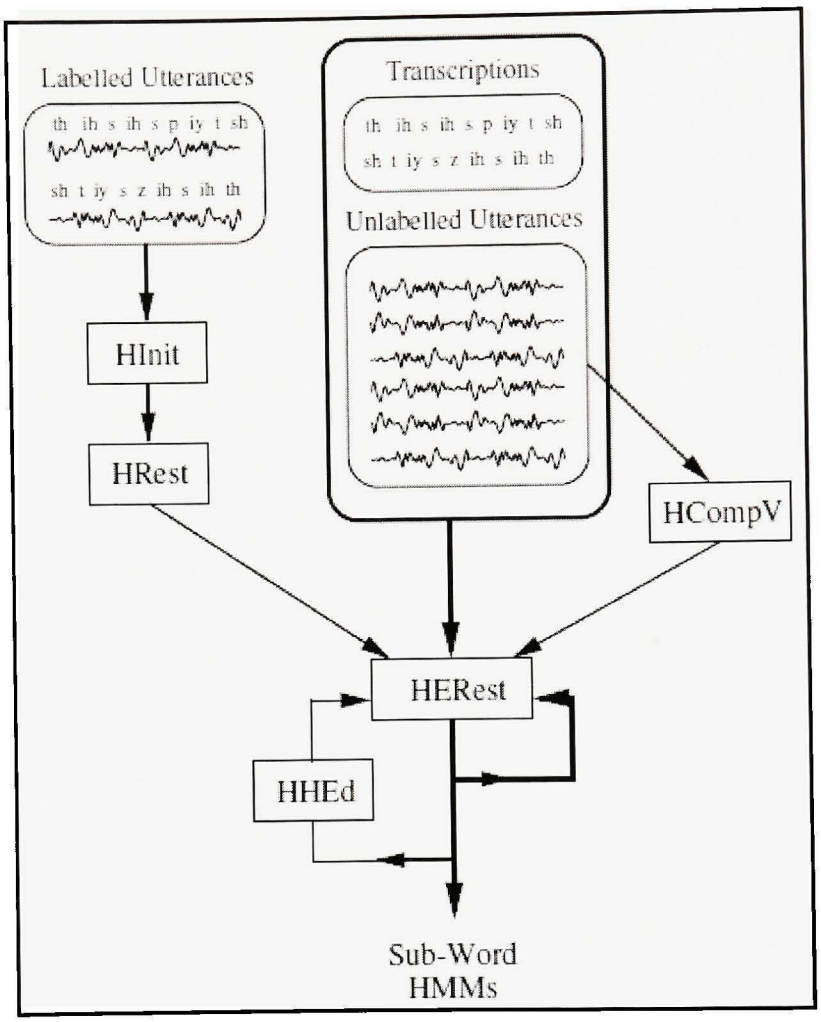

Figure-E V-3: Phase d'entraînement dans HTK.
Tirée de Young et al. (1995, p. 18)

E.2.3 Phase de reconnaissance

Cette phase représente la reconnaissance de la parole proprement dite et elle exploite quelques outils de HTK notamment HVite, HLRescore et HDecode.

- **HVite** : cet outil prend comme entrée un réseau de mots (modèle de langage) décrivant les relations possibles entre les mots, un dictionnaire définissant la prononciation de chaque mot et un modèle acoustique. HVite commence par convertir le réseau de mots en un réseau de phonèmes puis cherche et attache la définition de HMM à chaque phonème

correspondant. Une bonne reconnaissance vocale nécessite la construction de grands dictionnaires résultants de plusieurs sources et subissant plusieurs transformations. La construction de ces dictionnaires et un processus relativement difficile qui nécessite l'utilisation d'un gestionnaire de dictionnaire comme HDMan. D'autres outils tels que HBuild, HParse et HSGen sont très utiles pour la création et la manipulation des modèles de langage.

- **HLRescore** : cet outil est développé pour la manipulation des treillis généralement générés par HVite. Il permet de chercher le meilleur chemin dans le treillis, enrichir le treillis par de nouveaux modèles de langage, convertir le treillis en un réseau de mots, etc.
- **HDecode** : c'est un décodeur développé comme extension à HTK adaptée pour les systèmes de reconnaissance de la parole grand vocabulaire. Il fonctionne pratiquement de la même façon que l'outil HVite en se basant sur un ensemble de modèles HMMs et un dictionnaire.

E.2.4 Phase d'analyse

Cette phase s'assure de la qualité de la reconnaissance vocale et évalue la performance du système. La façon la plus ordinaire est de soumettre le système à des tests pour transcrire des fichiers vocaux. Ensuite, la transcription résultante du système doit être comparée avec la transcription exacte du fichier transcrit. Cette opération est réalisable par l'outil HResults qui permet l'évaluation de la performance des systèmes de reconnaissances de la parole.

ANNEXE VI

FORMULAIRE DE CONSENTEMENT

Titre du projet : Indexation d'annotations vocales dans un contexte de gestion documentaire

Étudiant chercheur : Ouali, Chahid

Directeur de recherche : Dumouchel, Pierre

Superviseur : Sévigny, Martin

Avant d'accepter de participer à cette recherche, veuillez prendre le temps de lire les renseignements qui suivent. Ce formulaire de consentement vous explique les buts de ce projet de recherche et ses procédures. Il indique les coordonnées de la personne avec qui communiquer au besoin. Nous vous invitons à poser toutes les questions que vous jugerez utiles à la personne qui vous présente ce document.

Pourquoi faisons-nous ce projet?

Nous mettons en œuvre ce projet afin d'évaluer les performances de quelques moteurs de reconnaissance de la parole dans un contexte d'annotations vocales. Nous voulons aussi étudier l'impact des résultats fournis par ces moteurs de reconnaissance de la parole sur les performances d'indexation de fichiers sonores.

Comment allons-nous nous y prendre?

L'expérience aura lieu dans une salle de réunion de l'entreprise Irosoft et sera divisée en deux parties. La première partie, d'une durée de 10 minutes, concerne la création de votre profil puis l'entraînement de deux moteurs de reconnaissance de la parole. Durant cette étape vous lisez quelques paragraphes afin que le moteur de reconnaissance de la parole adapte ses modèles à votre voix. La deuxième partie, d'une durée d'une heure, concerne l'enregistrement de plusieurs fichiers audio. Durant cette étape vous consulterez des textes extraits du corpus des lois du Canada et vous annoterez vocalement des parties de ces textes. Vos annotations seront par la suite enregistrées et transcrites manuellement et transcrites automatiquement par les moteurs de reconnaissance de la parole.

Participation volontaire et retrait préventif

Votre participation dans cette recherche doit être volontaire. Vous êtes libre de vous retirer en tout temps par avis verbal, sans préjudice et sans devoir justifier votre décision. Si vous

vous retirez de la recherche, les renseignements personnels et les données de recherche vous concernant et qui auront été recueillis au moment de votre retrait seront détruits.

Confidentialité et vie privée

Dans cette étude nous ne vous demanderons aucune information personnelle permettant de vous identifier. Votre nom, prénom et signature dans ce formulaire ne vont pas êtres utilisés. Un code est attribué à chaque participant permettant de l'identifier et de l'associer à ses enregistrements.

Risques reliés à votre participation

Les risques anticipés sont minimaux pour votre participation à ce projet de recherche. Vos risques se limitent à un engagement volontaire de votre temps d'une durée maximale de 90 minutes.

Utilisation des données

Il est anticipé que vos annotations vocales seront transcrites manuellement par un humain et automatiquement par des moteurs de reconnaissance de la parole. En outre, les résultats (et non vos données) de cette recherche peuvent êtres partagés dans des articles publiés ou dans des présentations à des conférences.

Remerciements

Votre collaboration est très précieuse pour cette recherche et on vous remercie vivement d'y participer.

CONSENTEMENT

Votre signature atteste que vous avez clairement compris les renseignements concernant votre participation au projet de recherche et indique que vous acceptez d'y participer. Vous ne devez jamais hésiter à demander des éclaircissements ou de nouveaux renseignements au cours du projet. Pour tout renseignement sur le projet de recherche, veuillez communiquer avec :

Chahid Ouali : chahid.ouali@gmail.com , (514) 584 6403

Je suis d'accord que mes enregistrements vocaux seront utilisés dans cette recherche.

Nom : _____ Prénom : _____

Signature _____ date: _____

IMPORTANT : Un exemplaire du formulaire de consentement signé doit être remis au participant.

BIBLIOGRAPHIE

Alani, T. and H. Guellif (1994). Modeles de Markov caches-theorie et techniques de base, INRIA. Research report No. 2196.

B. Roe , D. (1994). Voice communication between humans and machines, Washington, National Academy Press.

Baker, J. K. (1975). Stochastic modeling for automatic speech understanding, Academic Press.

Bellman, R. E. (1957). Dynamic Programming, Courier Dover Publications, 2003.

Ben Ayed, Y. (2004). Détection de mots clées dans un flux de parole. Laboratoire Lorrain de Recherche en Informatique et ses Applications, Ecole Nationale Supérieure des Télécommunications. Ph.D.: 138

Boite, R. (1987). Traitement de la parole, PPUR presses polytechniques, 2000.

Calliope (1989). La parole et son traitement automatique. Paris, Paris : Masson.

Chelba, C. and A. Acero (2005). Position specific posterior lattices for indexing speech. Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics. Ann Arbor, Michigan, Association for Computational Linguistics: 443-450.

Chen, C. H. (1995). Fuzzy logic and neural network handbook, McGraw-Hill, Inc.

Chen, L., Gauvain Jean-Luc, et al. (2001). "USING INFORMATION RETRIEVAL METHODS FOR LANGUAGEMODEL ADAPTATION." Proc. EUROSPEECH'01: pp 255-258.

Christiansen, R. W. and C. K. Rushforth (1977). "DETECTING AND LOCATING KEY WORDS IN CONTINUOUS SPEECH USING LINEAR PREDICTIVE CODING." IEEE Transactions on Acoustics, Speech, and Signal Processing ASSP-25(Compendex): 361-367.

CMU. (2008). "Sphinx-4: A speech recognizer written entirely in the JavaTM programming language." Retrieved 20 Juillet 2009, from http://cmusphinx.sourceforge.net/sphinx4/.

Ezzaidi, H. (2002). Discrimination parolemusique et étude de nouveaux paramètres et modèles pour un système d'identification du locuteur dans le contexte de conférences

téléphoniques, Chicoutimi : Université du Québec à Chicoutimi, 2002. Thèse de doctorat (Université du Québec à Chicoutimi) ;.

Fawcett, T. (2006). "An introduction to ROC analysis." Pattern Recognition Letters 27(8): 861-874.

Flach, P. and S. Wu (2003). Repairing concavities in ROC curves, Citeseer.

Foote, J. T., G. J. E. Jones, et al. (1995). Talker independent keyword spotting for information retrieval. Eurospeech 95, Madrid. ESCA.

Garofolo, J. S., G. P. Auzanne, et al. (2000). The TREC spoken document retrieval track: a success story. Information Technology: Eighth Text REtrieval Conference (TREC-8), 16-19 Nov. 1999, Gaithersburg, MD, USA, NIST.

Gauvain, J. L., L. F. Lamel, et al. (1994). "Speech-to-text conversion in French." International Journal of Pattern Recognition and Artificial Intelligence 8(Copyright 1994, IEE): 99-131.

Gelin, P. (1997). Détection de mots clés dans un flux de parole : Application à l'indexation de documents multimédia. Systèmes de Communication / Institut Eurécom, Ecole Polytechnique Fédérale de Lausanne. Ph.D.: 219.

Gillick, L., J. Baker, et al. (1993). Application of large vocabulary continuous speech recognition to topic and speaker identification using telephone speech. Proceedings of ICASSP '93, 27-30 April 1993, New York, NY, USA, IEEE.

Grataloup, C. (2007). La reconstruction cognitive de la parole dégradée: étude de l'intelligibilité comme indice d'une capacité cognitive humaine., Université Lumière Lyon 2 & Université Claude Bernard Lyon 1. Ph.D.

Haton, J.-P. (2006). Reconnaissance automatique de la parole : du signal à son interprétation. Paris, Paris : Dunod.

Haton, J. P., A. Bonneau, et al. (1990). "Acoustic-phonetic decoding of speech: problems and solutions." Traitement du Signal 7(Copyright 1991, IEE): 293-313.

Hebb, D. O. (1949). The organization of behavior: a neuropsychological theory.

Higgins, A. L. and R. E. Wohlford (1985). KEYWORD RECOGNITION USING TEMPLATE CONCATENATION. Proceedings - ICASSP 85, IEEE International Conference on Acoustics, Speech, and Signal Processing., Tampa, FL, USA, IEEE.

Hirsch, H. G. and C. Ehrlicher (1995). Noise estimation techniques for robust speech recognition. 1995 International Conference on Acoustics, Speech, and Signal Processing, 9-12 May 1995, New York, NY, USA, IEEE.

HSQLDB. (2010, 19 Mars 2010). "HSQLDB - 100% Java Database." Retrieved 12 Avril 2010, from http://hsqldb.org/.

IASRT. (2010). "Internet-Accessible Speech Recognition Technology." Retrieved 03 Avril 2010, from http://www.isip.piconepress.com/projects/speech/.

IBM. (2010). "Embedded ViaVoice." 05 Avril 2010, from http://www-01.ibm.com/software/pervasive/embedded_viavoice/.

ISIP. (2010). "Institute for Signal and Information Processing." Retrieved 03 Avril 2010, from http://www.isip.piconepress.com/.

James, D. A. and S. J. Young (1994). A fast lattice-based approach to vocabulary independent wordspotting. Proceedings of ICASSP '94. IEEE International Conference on Acoustics, Speech and Signal Processing, 19-22 April 1994, New York, NY, USA, IEEE.

Jelinek, F. (1997). Statistical methods for speech recognition. Cambridge, Mass., MIT Press.

Jones, G. J. F., J. T. Foote, et al. (1994). Video mail retrieval using voice:report on keyword definition and data collection.

Jones, G. J. F., J. T. Foote, et al. (1996). Retrieving spoken documents by combining multiple index sources. 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 18-22 Aug. 1996, USA, ACM.

Jones, K. S., G. J. E. Jones, et al. (1996). "Experiments in spoken document retrieval." Information Processing & Management 32(Copyright 1996, IEE): 399-417.

Karouia, M. (1996). Initialization and architecture determination of multilayer perceptrons, Université de Compiègne. Thesis: 207 p.

Klatt, D. H. (1977). "Review of the ARPA Speech Understanding Project." The Journal of the Acoustical Society of America 62(6): 1345-1366.

Kohonen, T. (1982). CLUSTERING, TAXONOMY, AND TOPOLOGICAL MAPS OF PATTERNS. Proceedings - 6th International Conference on Pattern Recognition., Munich, W Ger, IEEE.

Lausanne, U. d. (2010). "Cours de phonétique." Retrieved 13/03/2010, from http://www.unil.ch/ling/page12580.html.

Lea, W. A. (1980). Trends in Speech Recognition, Prentice Hall PTR.

Linares, G. (2003). "Techniques et applications de traitement de la parole " Dynamic Time Warping, from http://www.technolangue.net/article.php3?id_article=46.

Luba, V. and A. Younes (2005). Modèles de Markov cachés Reconnaissance de la parole, FACULTE POLYTECHNIQUE DE MONS.

Mendel, J. M. and K. S. Fu (1970). Adaptive, learning, and pattern recognition systems: theory and applications. London, UK, Academic Press.

Microsoft. (2010). "Microsoft Speech API (SAPI) 5.3." Retrieved 02 Avril 2010, from http://msdn.microsoft.com/en-us/library/ms723627%28VS.85%29.aspx.

Mills, T. J., D. Pye, et al. (2000). shoebox : a digital photo management system, AT&T Laboratories Cambridge.

Morgan, D. P., C. L. Scofield, et al. (1991). Multiple neural network topologies applied to keyword spotting. ICASSP 91. 1991 International Conference on Acoustics, Speech and Signal Processing (Cat. No.91CH2977-7), 14-17 May 1991, New York, NY, USA, IEEE.

Nuance. (2010). "Nuance Corporate Website." Retrieved 03 Avril 2010, from http://www.nuance.com.

Odell, J. and K. Mukerjee (2007). "Architecture, user interface, and enabling technology in Windows Vista's speech systems." IEEE Transactions on Computers 56(Compendex): 1156-1168.

Park, A., T. J. Hazen, et al. (2005). Automatic processing of audio lectures for information retrieval: Vocabulary selection and language modeling. 2005 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP '05, March 18, 2005 - March 23, 2005, Philadelphia, PA, United states, Institute of Electrical and Electronics Engineers Inc.

Paul, L., K. Philip, et al. Design of the cmu sphinx-4 decoder, unknown: 5.

Pearce, D. and H.-G. Hirsch (septembre 2000). "The AURORA Experimental Framework for the Performance Evaluations of Speech Recognition Systems under Noisy Condidions." ISCA ITRW ASR2000 "Automatic Speech Recognition: Challenges for the Next Millennium".

Peinado, A. (2006). Speech recognition over digital channels : robustness and standards. Chichester, West Sussex, Chichester, West Sussex : John Wiley.

Reddy, D. R. (1966). An approach to computer speech recognition by direct analysis of the speech wave, Stanford University: 150.

Reddy, R. (1976). speech recognition by machine: a review, Proceedings of the IEEE: pp. 501--531.

Renals, S., D. Abberley, et al. (2000). "Indexing and retrieval of broadcast news." Speech Communication 32(Copyright 2000, IEE): 5-20.

Rose, R. C. (1992). Discriminant wordspotting techniques for rejecting non-vocabulary utterances in unconstrained speech. ICASSP-92: 1992 IEEE International Conference on Acoustics, Speech and Signal Processing (Cat. No.92CH3103-9), 23-26 March 1992, New York, NY, USA, IEEE.

Rose, R. C. (1995). "Keyword detection in conversational speech utterances using hidden Markov model based continuous speech recognition." Computer Speech and Language 9(Compendex): 309-333.

Rose, R. C. and D. B. Paul (1990). A hidden Markov model based keyword recognition system. ICASSP 90. 1990 International Conference on Acoustics, Speech and Signal Processing (Cat. No.90CH2847-2), 3-6 April 1990, New York, NY, USA, IEEE.

Rosenblatt, F. (1962). Principles of neurodynamics: perceptrons and the theory of brain mechanisms, Spartan Books.

Sun (1998). Java™ Speech Grammar Format Specification: 37.

SUN Microsystems (1998). Java™ Speech API Programmer's Guide: 138.

T. Bänziger, G. Klasmeyer, et al. (2000). "Améliorer les systèmes de vérification automatique du locuteur en intégrant la variabilité émotionnelle : Méthodes et premières données." XXIIIèmes Journées d'Etude sur la Parole, Aussois,.

Tou, J. T. and R. C. Gonzalez (1974). Pattern recognition principles. Reading, MA, USA, Addison-Wesley.

Unjung, N. (2010). "Mel-Frequency Cepstral Analysis."

Vapnik, V. N. (1998). Statistical learning theory. New York, New York : Wiley.

VINTSJUK T.K. (1968). "Recognition of words of oral speech by dynamic programming." Kibernetica Vol 81, n°8: pp. 81-88.

Vojtko, J., J. Korosi, et al. (2008). Comparison of automatic speech recognizer SPHINX 3.6 and SPHINX 4.0 for creating systems in Slovak language. Systems, Signals and Image Processing, 2008. IWSSIP 2008. 15th International Conference on.

Voxeo. (2010). "Voxeo." Retrieved 05 Avril 2010, from http://www.voxeo.com/.

Walker, W., P. Lamere, et al. (2004). Sphinx-4: A Flexible Open Source Framework for Speech Recognition: 9.

Wikipédia. (2010, 30 janvier 2010 13:38 UTC). "Phonétique " 49416181 Retrieved 13 mars 2010 06:57 UTC from http://fr.wikipedia.org/w/index.php?title=Phon%C3%A9tique&oldid=49416181.

Wilcox, L. D. and M. A. Bush (1992). Training and search algorithms for an interactive wordspotting system. ICASSP-92: 1992 IEEE International Conference on Acoustics, Speech and Signal Processing (Cat. No.92CH3103-9), 23-26 March 1992, New York, NY, USA, IEEE.

Wilpon, J. G., C. H. Lee, et al. (1989). Application of hidden Markov models for recognition of a limited set of words in unconstrained speech. 1989 International Conference on Acoustics, Speech, and Signal Processing, May 23, 1989 - May 26, 1989, Glasgow, Scotland, Publ by IEEE.

Wilpon, J. G., L. R. Rabiner, et al. (1990). "Automatic recognition of keywords in unconstrained speech using hidden Markov models." IEEE Transactions on Acoustics, Speech and Signal Processing 38(Copyright 1991, IEE): 1870-1878.

Young, S., J. Odell, et al. (1995). The HTK book. Cambridge University. 1996.

Young, S., N. Russell, et al. (1989). Token passing: a simple conceptual model for connected speech recognition systems, University of Cambridge, Department of Engineering.

Young, S. J., M. G. Brown, et al. (1997). Acoustic indexing for multimedia retrieval and browsing. 1997 IEEE International Conference on Acoustics, Speech, and Signal Processing, 21-24 April 1997, Los Alamitos, CA, USA, IEEE Comput. Soc. Press.