الجمهورية الجزائرية الديمقراطية الشعبية
وزارة التعليم العالي و البحث العلمي

Université Badji Mokhtar Annaba
Badji Mokhtar University –Annaba

جامعة باجي مختار- عنابة

Année : 2018

Faculté des sciences de l'ingéniorat
Département d'informatique

T H È S E

En vue de l'obtention d'un Doctorat 3ème Cycle

Approche Hybride pour la Reconnaissance de la Parole

Filière : Informatique

Spécialité : Sciences et Technologies de l'Information et de la Communication

Présentée par

Hamza FRIHIA

Directeur de thèse : Halima BAHI-ABIDET Prof. Université d'Annaba

Devant le Jury

Président:	Yamina MOHAMED BEN ALI	Prof. Université d'Annaba
Examinateur :	AbdelKarim BOUKABOU	Prof. Université de Jijel
Examinateur :	NourEddine DOGHMANE	Prof. Université d'Annaba
Examinateur :	Mohamed REDJIMI	Prof. Université de Skikda
Examinateur :	Nabiha AZIZI	MCA. Université d'Annaba

DEDICACES

Je dédie ma thèse de Doctorat…

*A mes **parents**,*

*A mon encadrante Pr **Halima Bahi***

*A toute ma **famille**.*

*A tous mes **amis***

Remerciements

J'exprime d'abord mes profonds remerciements à Dieu qui m'a donné la force, la volonté et le courage pour accomplir cette thèse de doctorat.

Je voudrais remercier mon encadreur Pr Halima Bahi pour sa disponibilité, sa gentillesse et pour ses suivis continus durant la période de la réalisation de ce travail.

Je tiens à remercie Madame ***Yamina Mohamed Benali*** *Professeur à l'Université de Annaba, pour avoir accepté de présider mon jury de thèse, Monsieur* ***Abdelkrim Boukabou*** *Professeur à l'Université de Jijel, Monsieur* ***Noureddine Doghmane*** *Professeur à l'Université de Annaba, Monsieur* ***Mouhamed Redjimi*** *Professeur à l'Université de Skikda et Madame* ***Nabiha Azizi*** *Docteur à l'Université de Annaba, pour avoir accepté d'examiner cette thèse et pour l'intérêt qu'ils ont porté à ce travail.*

Je tiens à remercier ma famille Fatima zahra, Noureddine, Samir, Dounia, Amina pour m'avoir donné la possibilité de continuer mes études et pour leur soutien moral et leurs encouragements.

Je remercie tous ceux qui ont contribué à faciliter mon travail.

ملخص

يتطلب بناء نظام التعرف على الكلام المستمر قاعدة بيانات ضخمة مجزأة وموسومة. لكن اللغة العربية ككثير من اللغات الأخرى تفتقر إلى مثل هذه الموارد المجزأة، إلا أن استخدام التقسيم التلقائي أثبت أنه بديل جيد لإتاحة المجال لهذه الموارد. في هذه الأطروحة، نقترح الجمع بين نماذج ماركوف المخفية (Modèle de Markov Cachés) وحاملات الهوامش الواسعة (Suppot à Vaste Marges SVM) لتجزئة ووسم موجات الكلام إلى وحدات صوتية صغيرة.
نمادج ماركوف تولد سلسلة من الفونيمات مقسمة و معينة. و تقوم حاملات الهوامش الواسعة (SVM) بتصحيح الحدود وتصحيح التسميات. و من ثم الوحدات المقسمة والمعينة التي تم الحصول عليها بمثابة مجموعة تدريب لتطبيقات التعرف على الكلام.لتجريب هذا النظام نحتاج إلى استخدام البيانات المجزأة والمعينة، لهذا الغرض، قمنا ببناء قاعدة بيانات صوتية أسميناها "ArabPhone".
وأخيرا، فإن النتائج التي تم الحصول عليها هي قريبة من تلك المتحصل عليها في البحوث العلمية الحديثة ويمكن تحسينها من خلال مراعاة خصائص اللسان العربي.

الكلمات المفتاحية: التجزئة الاتوماتكية للكلام, التعرف التلقائي للكلام, اللغة العربية, اس في ام, إش ام ام

Résumé

Le développement de systèmes de reconnaissance de la parole continue et à grand vocabulaire nécessite la disponibilité de grands corpus vocaux segmentés et étiquetés en petites unités (généralement des phonèmes). La langue Arabe étant une langue peu dotée de ressources, la segmentation automatique du signal de la parole semble une bonne alternative pour rendre de telles ressources disponibles. Dans notre travail, notre principale contribution consiste à réaliser un système de segmentation automatique en vue de construire un corpus annoté pour une fin de construction d'un système de reconnaissance. Dans cette approche, les modèles de Markov cachés génèrent les segments phonétiques avec leurs frontières et les SVMs (Support Vector Machines) corrigent ces frontières et éventuellement leurs étiquettes. Enfin, pour valider cette approche nous avons construit un corpus de parole appelé « ArabPhone ». les résultats obtenus sont encourageants et peuvent être améliorés en considérant les spécificités de la langue Arabe.

Mots clés : *Segmentation automatique de la parole, Reconnaissance automatique de la parole, Langue Arabe, SVM, HMM.*

Abstract

Building a large vocabulary continuous speech recognition (LVCSR) system requires a lot of hours of segmented and labelled speech data. Arabic language, as many other low-resourced languages, lacks such data, but the use of automatic segmentation proved to be a good alternative to make these resources available. In this thesis, we suggest the combination of hidden Markov models (HMMs) and support vector machines (SVMs) to segment and to label the speech waveform into phoneme units. HMMs generate the sequence of phonemes and their frontiers; the SVM refines the frontiers and corrects the labels. The obtained segmented and labelled units may serve as a training set for speech recognition applications. The validation of the approach needs the use of transcribed and labelled data, thus we built a dedicated corpus. Finally, the obtained results are close to those described in the literature and could be improved by handling more Arabic speech specificities.

Keys words: *Automatic Speech Segmentation, Automatic Speech Recognition, Arabic language, HMM, SVM.*

Sommaire

Chapitre 1
Introduction générale

Chapitre 2
La reconnaissance automatique de la parole

Chapitre 3
la segmentation de la parole pour la reconnaissance de la parole

Chapitre 4
Hybridation HMM/SVM pour la segmentation de la parole

Chapitre 5
Expérimentations et Résultats

Chapitre 6
Conclusion et perspectives

Bibliographie

Table de Figures

Liste de tables

Chapitre 1
Introduction générale

1.1 Introduction

Les avancées technologiques dans le cadre du traitement de la parole et de l'intelligence artificielle ont conduit à l'émergence d'applications innovantes (parfois commercialisées) relevant de l'apprentissage des langues, du routage des appels téléphoniques, de la traduction de discours, de la recherche d'information dans des bases de données de news, etc. toutes ces applications qui semblent diverses et variées relèvent à la base d'une même technologie qui est celle de la reconnaissance automatique de la parole (RAP). Parmi ces applications, nous trouvons l'invite de Google qui nous permet de formuler une requête de vive voix. En fait, Google inclut un système de RAP qui permet de transcrire une requête audio en une chaîne de caractères ; la recherche se fera ensuite sur des documents textes.

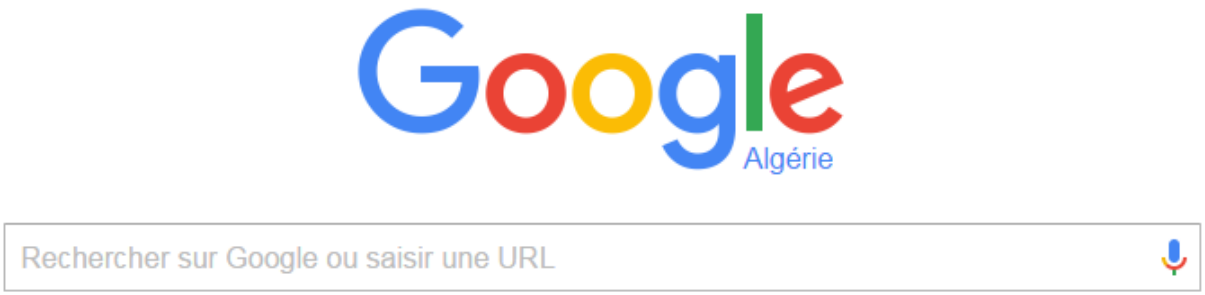


Figure 1.1. Invite vocale de Google

Une fois le signal de la requête « capturé » par le microphone de l'appareil débute une session de reconnaissance classique.

La capture de l'onde sonore de la prononciation produit un signal numérique qui est une succession de valeurs réelles. Le signal de la parole étant complexe et redondant, un premier module du système de RAP consiste en l'extraction des caractéristiques du signal ; Souvent en RAP on utilise des paramètres appelés MFCCs (Mel Frequency Cepstral Coefficients) en guise de caractéristiques. Cette nouvelle représentation du signal sera comparée progressivement à des modèles de référence appris par le système ; Il s'agit souvent de modèles de phonèmes. A chaque fois qu'un phonème est reconnu, le signal est segmenté et on passe à la reconnaissance du phonème suivant. On appelle ce second module : le module de classification.

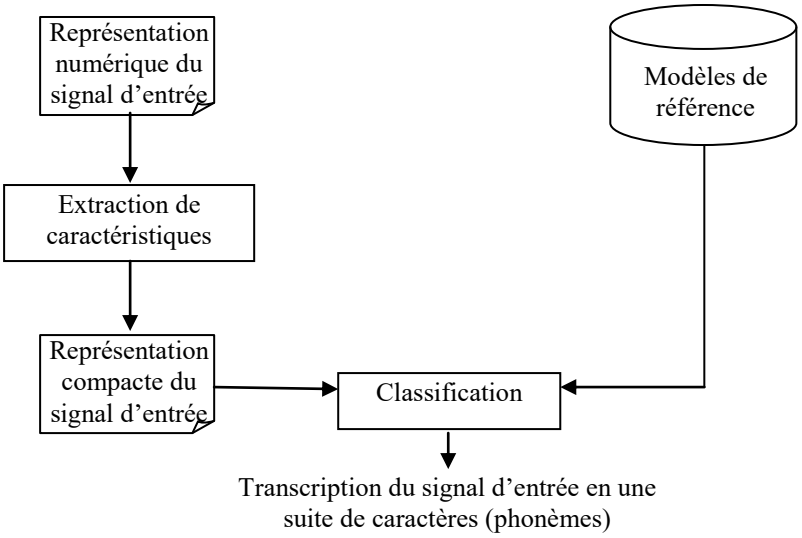


Figure 1.2. Représentation générale des étapes d'un système de RAP

Actuellement, les modèles de référence sont des modèles de Markov cachés (HMM pour Hidden Markov Models) qui représentent l'état de l'art en RAP. Ces modèles sont construits sur la base d'exemples d'apprentissage des différents phonèmes de la langue enregistrés dans différents contextes. La construction de ces modèles dont dépendent les performances du système de RAP nécessite l'existence de grandes bases d'apprentissage où les segments des différents phonèmes sont délimités et étiquetés.

La segmentation consiste en la division du signal d'origine en des unités de base. La figure (1.3) montre la segmentation d'un fichier audio issu de la base IPA[1] (Association International de Phonétique) en des mots.

La segmentation automatique d'un flux de parole se fait principalement selon deux approches. La première se base sur les caractéristiques acoustiques du signal pour opérer la segmentation. Les algorithmes issus de cette approche détectent des discontinuités dans le signal de la parole pour définir les frontières des unités à segmenter. Ainsi une unité est une plage du signal relativement stable. Souvent dans cette approche l'étiquetage des unités n'est pas possible.

[1] https://www.internationalphoneticassociation.org/

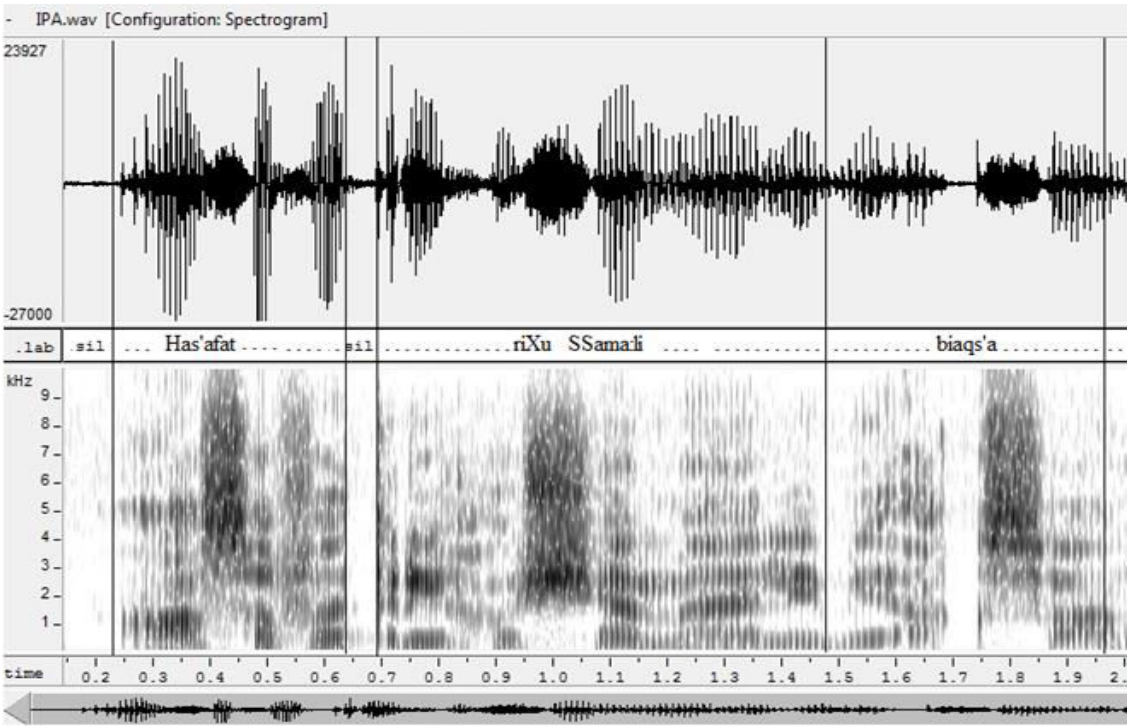


Figure 1.3. Segmentation en mots du début du fichier « 01-Narative.wav » en Arabe de la base IPA

La seconde approche tient compte des particularités des unités phonétiques du langage cible ; il s'agit de l'approche phonétique où on dispose d'une transcription phonétique ou orthographique du signal à segmenter. Elle se base sur un alignement forcé entre cette transcription et les unités du signal. Souvent, ce sont les modèles de Markov cachés qui sont utilisés pour la modélisation des unités. C'est aussi l'approche que nous allons utiliser car ici l'étiquetage des unités accompagne le processus de segmentation et que les HMM sont l'état de l'art en reconnaissance de la parole qui est notre application cible.

1.2 Motivations

Revenons à l'invite vocale de Google, si nous essayons de prononcer un mot en Arabe, et même si nous avons activé la langue Arabe pour la recherche, la transcription que nous obtiendrons ne sera pas en Arabe. En fait, la langue Arabe n'est pas prise en charge par cette invite.

En effet, le développement de systèmes de reconnaissance de la parole continue et à grand vocabulaire (LVCSR : pour Large Vocabulary Continuous Speech Recognition) nécessite la disponibilité de grands corpus vocaux segmentés et étiquetés en petites unités (généralement des phonèmes). Des ressources telles que le corpus TIMIT, où le discours est segmenté par des experts, sont disponibles pour quelques langues tandis que d'autres, comme l'Arabe, peuvent être considérées comme des langues à faible ressources.

La langue Arabe étant une langue peu dotée de ressources, le développement de système de reconnaissance automatique de la parole de type LVCSR n'est pas très répandu vu l'absence de corpus annoté. Le travail que nous menons a pour but de doter la langue Arabe de ressources permettant le développement de diverses applications qui soient basées sur la reconnaissance de la parole.

A l'heure actuelle, il existe d'énormes corpus parole de la langue Arabe, particulièrement ceux issus des news tels ceux d'Al Jazeera, mais cette énorme ressource ne peut être exploitée car le corpus n'est ni segmenté ni étiqueté. Envisager une annotation manuelle de ces ressources semble utopique vu la quantité des données, mais l'alternative de le faire de manière automatique est une alternative prometteuse.

La seconde motivation de notre travail vient du fait qu'en termes d'approches de segmentation la plus répandue est l'approche acoustique, qui est chronologiquement la plus ancienne mais c'est aussi celle où des recherches avancées ne cessent d'être menées. Pour notre cas l'approche phonétique est la plus appropriée. Dans ce cadre la proposition de l'algorithme d'apprentissage embarqué (EL : Embedded Learning) fût un grand tournant. Toutefois cet algorithme souffre de certaines limitations inhérentes à l'initialisation des modèles HMMs que nous allons nous atteler comme d'autres auparavant à les surmonter.

1.3 Contributions

Ce travail de thèse nous a conduit à investiguer le champ de la reconnaissance automatique de la parole et plus spécifiquement celui de la segmentation automatique d'un flux de parole continu. Ainsi, comme premier apport au sujet de cette thèse, nous avons étudié en détail les ressources disponibles en termes de logiciel dans le domaine de la reconnaissance de la parole ; ceci nous a permis de réaliser une étude comparative entre différentes librairies.

Ce travail a été présenté dans le cadre d'une conférence nationale spécialisée (NCSP : National Conference on Speech Processing) en 2014.

Nous avons aussi réalisé un état de l'art sur la segmentation automatique de la parole dont le but est de comprendre le processus de segmentation et les différentes techniques qui y sont utilisées.

En ce qui est de notre principale contribution ; elle consiste en la réalisation d'un système de segmentation performant en vue de construire un corpus annoté pour une fin de construction d'un système RAP. Ce système est basé sur l'hybridation des modèles HMM avec le classifieur SVM (Support Vector Machines) qui est un outil de discrimination puissant. Ce travail a été publié dans le journal « International Journal of Speech Technology ».

Pour pouvoir valider notre proposition, nous avons construit un corpus Arabe annoté « ArabPhone »; ce qui représente une ressource appréciable pour les chercheurs du domaine ; ce travail a été publié dans un chapitre de livre intitulé ‘‘Text, Speech and Dialog’’.

1.4 Organisation de la thèse

Cette thèse traite du thème de la reconnaissance de la parole et particulièrement d'une activité annexe à cette fonctionnalité qui est la segmentation automatique de la parole. La thèse vise à présenter l'essentiel du travail effectué et tente d'offrir une référence aux chercheurs qui se sont investis ou qui souhaitent s'investir dans ce domaine. Elle est structurée comme suit:

Le chapitre 1 consiste en une introduction du travail principalement, au travers de sa problématique et son positionnement dans son contexte.

Le deuxième chapitre introduit la reconnaissance automatique de la parole qui est notre application cible. En particulier, on y présente un panel des outils disponibles avec une comparaison entre ces différentes ressources.

Le chapitre 3 est dédié à la présentation de la segmentation automatique de la parole au travers des deux grandes approches suivies et des différents travaux existants dans le contexte.

Le chapitre 4 présente notre contribution qui consiste en une architecture hybride où un SVM est utilisé pour raffiner le résultat de la segmentation produite par le HMM.

Le chapitre 5 est dédié à la présentation des résultats et des expérimentations. On y trouvera les détails du corpus construit « ArabPhone », les premières expérimentations relatives à l'algorithme de l'apprentissage embarqué. Enfin, on y trouve les détails des résultats de l'approche proposée.

Le chapitre 6 porte sur la conclusion du travail et les perspectives proposées.

Chapitre 2

La reconnaissance automatique de la parole

2.1. Introduction

La communication consiste à établir des liens avec autrui, à communiquer avec l'autre quelle que soit sa nature: un humain, un animal ou une machine ou même avec un hybride. La parole est la manière la plus intuitive et la plus simple utilisée par les humains pour expliquer leurs idées et exprimer leurs besoins.

En intelligence artificielle, on essaie de calquer cette manière naturelle et efficace de communiquer pour rendre plus fluide le dialogue homme/machine. On retrouve ainsi un large panel d'applications qui gravitent autour de la parole, telles que la synthèse vocale qui permet de créer de la parole artificielle à partir de n'importe quel texte (Ching, 2007 ; Patil et al, 2013 ; Shah et al, 2014), la reconnaissance de la parole qui permet à un ordinateur d'identifier les mots dans un flux audio et les convertir en texte (Kaur, 2010), la classification en parole et non parole qui classe et segmente un signal en: parole, musique et le silence (Lu et al, 2002 ; Saadia, 2015), etc

Dans cette thèse, nous nous intéressons au domaine particulier de la reconnaissance de la parole. Les travaux sur la reconnaissance automatique de la parole datent du début du XX^{ème} siècle. En 1952, Le premier système de la reconnaissance de chiffres isolés développé par les laboratoires Bell Labs vit le jour. Ensuite, les recherches s'orientèrent de plus en plus vers la reconnaissance de parole continue. La recherche s'est considérablement accrue durant les années 1970 avec les travaux de Jelinek chez IBM (1972-1993). En 1971, le projet ARPA aux États-Unis (15 millions de dollars) est lancé pour tester la faisabilité de la compréhension automatique de la parole continue avec quelques contraintes. En 1972, un système de reconnaissance de 32 mots vit le jour. En 1986, un projet japonais ATR de téléphone est lancé pour la traduction automatique en temps réel. En 1997, la société « Dragon systems » lance le logiciel « Dragon NaturallySpeaking » ; qui est toujours une référence dans le contexte.

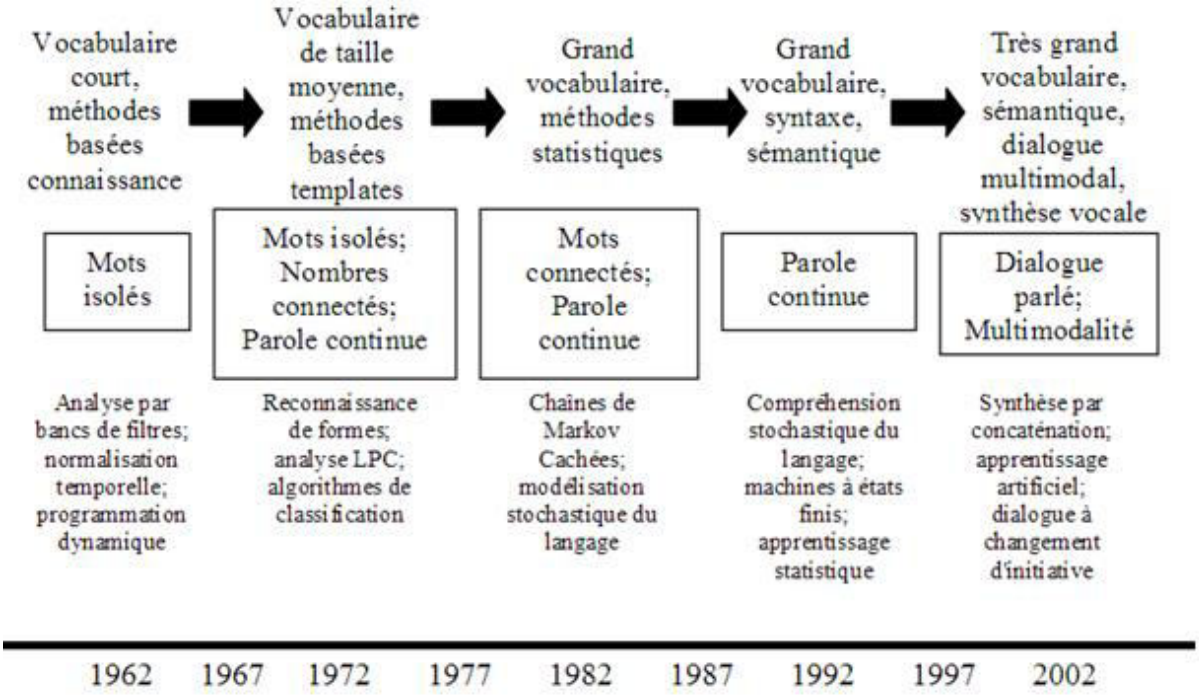


Figure 2.1. Historique du la reconnaissance de la parole (d'après, Le Blouch, 2009)

En 2008, l'invite vocale « Google voice app » est apparue pour lancer des recherches vocales sur le moteur de recherche Google. Et en 2011, une nouvelle version de l'application Iphone appelée « Siri » est apparue. Aujourd'hui, la reconnaissance de la parole est un domaine à forte croissance grâce à la déferlante des systèmes embarqués.

2.2. Reconnaissance de la parole

2.2.1. Définition

La reconnaissance automatique de la parole (RAP) est le processus de conversion d'un signal acoustique d'entrée (entrée au format audio sous forme de mots parlés) en une séquence des « mots » contenus dans le discours. Ces mots reconnus peuvent être des résultats finaux, qui peuvent servir à des instructions de commandes et de contrôle, ou ils peuvent servir de contribution à un traitement ultérieur du langage. Ainsi, la reconnaissance de la parole peut être considérée comme la possibilité de prendre un format audio en tant qu'entrée, puis de générer un format de texte comme sortie d'un système.

Un système de reconnaissance de la parole comprend normalement trois étapes, d'abord une étape d'extraction de caractéristiques, ensuite une étape d'apprentissage et enfin

celle de reconnaissance (ou de classification). Le module d'extraction de caractéristiques sert à transformer le signal en entrée du système en une représentation interne compacte de sorte qu'il soit possible de reconstituer le signal original. La sortie de ce bloc est utilisée pour créer des modèles qui seront utilisés par le module de reconnaissance et qui en général intègre des séquences de phonèmes en des mots.

Le schéma suivant illustre les principales étapes de reconnaissance automatique de la parole (figure 2.2).

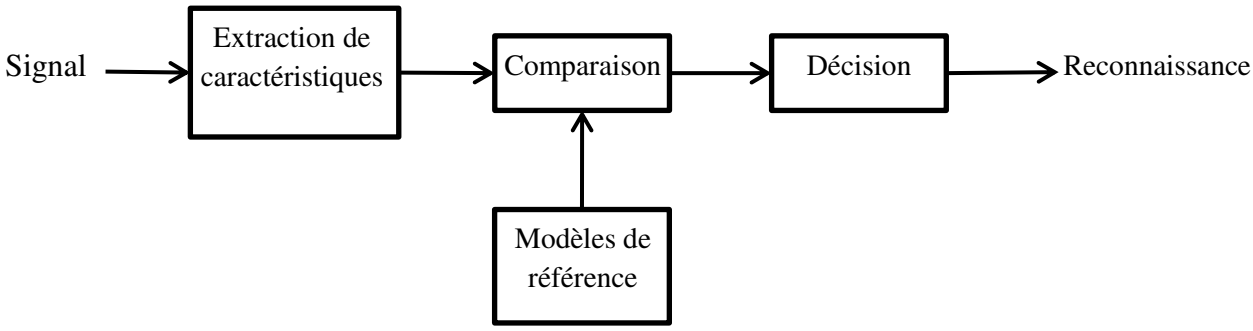


Figure 2.2. Organisation d'un système de RAP

2.2.2. Applications

Toutes les applications de la RAP bénéficient de l'évolution technologique qui se traduit par le fait qu'un système de reconnaissance complet peut désormais être entièrement implémenté sous forme logicielle. Cette évolution a largement contribué au développement de nouvelles applications à des coûts assez faibles. On peut trouver des applications d'un système RAP dans de nombreux domaines

2.2.2.1. Saisie de données

Dans ce cas, les machines à dicter sont utilisées pour dicter des textes à caractère général, par exemple un courrier, ou spécialisé, par exemple, des comptes rendus échographiques. Les dernières versions de ces machines sont intégrées dans le système d'exploitation de la machine hôte, de sorte que les commandes au système peuvent être données oralement.

2.2.2.2. Aide aux handicaps

La reconnaissance de la parole peut intervenir comme outil d'aide pour certains handicaps. Elle permet ainsi à un handicapé moteur un contrôle efficace de son environnement. Ce type d'application reste encore limité mais devrait se développer à l'avenir avec la diminution du coût des systèmes. Un autre domaine est celui de l'aide à l'apprentissage de la langue parlée pour un malentendant. L'idée est de compenser dans un contexte multimodal, le manque d'audition chez le malentendant par un visuel pour l'aider à prononcer les sons de parole et à acquérir la prosodie de la langue. Plusieurs systèmes de ce type ont été réalisés. Cette même idée a été reprise pour aider à l'apprentissage oral d'une langue étrangère.

2.2.2.3. Commande de machines

La commande orale d'un appareil ou d'une machine a été une des premières applications de la reconnaissance automatique de la parole. Ceci s'explique par un contexte favorable à de telles applications où le vocabulaire est limité à quelques dizaines, ou parfois quelques centaines de mots et où les commandes sont composées soit de mots isolés ou enchaînés soit de phrases à structure simple et rigide.

2.2.2.4. Traduction parole-parole

La Traduction parole-parole (TPP) est un axe de recherche très prometteur qui pose plusieurs défis scientifiques importants. Le principe est de permettre à un locuteur de s'exprimer dans sa langue pour s'adresser à un interlocuteur ne parlant pas la même langue. Le système opère une reconnaissance du message, produit une traduction et enfin le résultat est synthétisé en un message oral. Dans ce cas, il est nécessaire de gérer les aspects de reconnaissance et de synthèse de la parole, et du traitement du langage naturel, mais également ceux de la traduction automatique.

2.3. Description du signal de la parole

Le signal vocal est une grandeur physique de nature acoustique ; c'est aussi un support d'informations pourvues de signification : la parole. L'analyse de ce type de signal doit tenir compte de sa complexité et de sa variabilité intra et interlocuteur.

Pour reconnaitre la parole, il est important de connaitre sa structure complexe qui lui permet de transporter facilement de l'information. L'information portée par le signal est contenue dans le spectre et son évolution au rythme de ses différents changements.

2.4. Paramétrisation du signal vocal

2.4.1. Acquisition et modélisation du signal

Comme le signal analogique est un signal continu qui contient une infinité de valeurs et une quantité infinie d'amplitudes, nous sommes obligés de passer au mode discontinu par la transformation en un signal numérique qui est un signal discret dont l'amplitude a été quantifiée. Les opérations effectuées lors de passage d'un signal analogique au numérique sont l'échantillonnage et la quantification.

Échantillonner un signal analogique consiste à prélever des échantillons à la cadence T de façon instantanée. Le choix de la fréquence d'échantillonnage est une tâche importante pour définir la bande passante du signal numérique, cette fréquence doit être supérieure au moins de deux fois la fréquence maximale du spectre du signal analogique (d'après le théorème de Shannon). Par exemple : pour une ligne téléphonique, un signal échantillonné à 8000 Hz contient donc une bande de fréquences allant de 300 à 3400 Hz.

La tâche de quantification consiste à définir le nombre de bits nécessaire sur lesquels la numérisation sera réalisée. Elle permet de mesurer l'amplitude à chaque pas de l'échantillonnage. La quantification se fait en général sur 16 bits.

La qualité du signal numérique dépend de la fréquence d'échantillonnage (plus la fréquence est grande, plus la qualité du signal numérique est bonne) et aussi du nombre de bits choisi pour la phase de quantification.

2.4.1.1. Fenêtrage

La parole est un signal non stationnaire où leurs propriétés changent assez rapidement avec le temps. Aussi, l'utilisation de fonction telles que la transformée de Fourier semble impossible. Toutefois, Les caractéristiques de signal de parole restent stables pendant une courte période de temps. Les méthodes de traitement de la parole considèrent des blocs d'échantillons de taille fixe (fenêtre de 20 à 40 ms) dans lesquels le signal est supposé stationnaire, ensuite les vecteurs d'analyse sont obtenus en déplaçant ces blocs de 10 à 20 ms. Après le fenêtrage, on effectue une pondération de ces fenêtres par des fonctions appropriées ; comme les fenêtres de Hanning, Hamming (figure 2.3) ou triangle, etc.

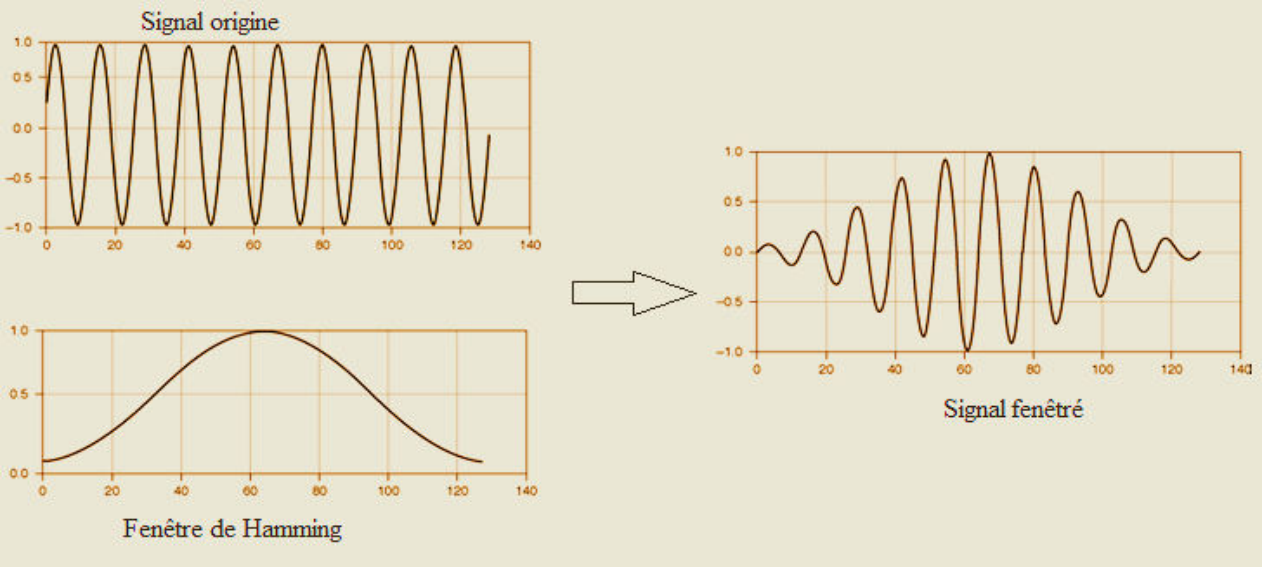


Figure 2.3. Fenêtrage d'un signal sinusoïdal

2.4.2. Extraction de caractéristiques

La phase d'extraction de caractéristiques pour le processus de reconnaissance a pour but d'extraire les coefficients qui représentent au mieux le signal de la parole. Ces paramètres sont calculés à des intervalles réguliers. Le signal de la parole est transformé en une suite de vecteurs acoustiques, ces derniers sont censé modéliser le signal d'origine et doivent extraire le maximum d'informations utiles pour la reconnaissance.

2.4.2.1. Transformée de Fourier

« D'après Joseph Fourier, la série de Fourier représente tout signal périodique par une somme de fonctions trigonométriques. Elle comporte un terme constant et des fonctions sinusoïdales d'amplitudes diverses. Ainsi un son sinusoïdal ne comporte qu'une seule raie spectrale correspondant à la fréquence de sa fonction sinus. Un son complexe est composé d'une multitude de ces raies spectrales qui représentent sa composition fréquentielle » (Vaufreydaz, 2002).

Le calcul de la Transformée de Fourier est donné par la formule (équation 2.1) :

$$F(\omega) = \int_{-\infty}^{+\infty} s(t)^* \, e^{-j\omega t} \, dt \qquad (2.1)$$

Dans le cas d'une séquence discrète d'échantillons, il est possible de calculer une Transformée de Fourier Discrète (TFD).

Cet algorithme étant gourmant en temps de calcul, un autre algorithme de calcul rapide de la transformée de Fourier discrète (FFT pour Fast Fourier Transform) a été proposé par (Cooley et al., 1965). La limitation de cet algorithme est que pour obtenir la FFT on doit considérer une taille de la séquence de puissance de 2.

2.4.2.2. Analyse par prédiction linéaire (LPC)

Le codage prédictif linéaire (LPC) est un outil principalement utilisé pour le traitement du signal vocal et le traitement de la parole pour représenter l'enveloppe spectrale d'un signal numérique de parole sous forme comprimée, en utilisant l'information d'un modèle prédictif linéaire. C'est l'une des techniques d'analyse de la parole les plus puissantes, et l'une des méthodes les plus utiles pour coder une parole de bonne qualité à faible débit et fournit des estimations extrêmement précises des paramètres de la parole. La LPC est basé sur le modèle de filtre source du signal de parole.

Un système de LPC tente d'extraire un ensemble de paramètres qui décrit le signal qu'il analyse. En particulier, il suppose que le signal a été produit par une source qui excite un

ou plusieurs filtres linéaires. Bien qu'il s'agisse d'un modèle imprécis pour les signaux de la vie réelle (parole, radar, signaux sismiques), il constitue néanmoins une bonne estimation de la façon dont ces signaux sont produits. Cela permet une représentation paramétrique compacte de tout signal qui correspond au modèle de production linéaire.

Il existe deux types communs de prédiction linéaire: à court terme (également connu sous le nom de formant) et à long terme (également connu sous le nom de pitch). Le signal de parole passe par le filtre d'analyse de la parole pour supprimer la redondance du signal, l'erreur résiduelle est générée en sortie. Il peut être quantifié par un plus petit nombre de bits comparé au signal d'origine. Donc, au lieu de transférer tout le signal, on peut transférer cette erreur résiduelle et les paramètres de la parole pour générer le signal d'origine. Un modèle paramétrique est calculé en fonction de la théorie de l'erreur quadratique moyenne, cette technique étant connue sous le nom de prédiction linéaire (LP). Par cette méthode, le signal de parole est approximatif en tant que combinaison linéaire de ses p échantillons précédents. Dans cette technique, les coefficients LPC obtenus décrivent les formants. Les fréquences dans lesquelles se trouvent les pics de résonance sont appelées les fréquences des formants. Ainsi, avec cette méthode, les emplacements des formants dans un signal de parole sont estimés en calculant les coefficients prédictifs linéaires sur une fenêtre glissante et en trouvant les pics dans le spectre du filtre LP résultant. Le signal de la parole échantillonné directement à partir du microphone est traité pour l'extraction des fonctions. Les étapes de base du processus LPC comprennent : Le préaccentuation, le blocage de trame, le fenêtrage, l'analyse d'autocorrélation, et l'analyse LPC (Thiang, 2011).

2.4.2.3. Prediction linéaire Perceptualle (PLP)

Le modèle PLP (Perceptual Linear Prediction) développé par Hermansk, modélise la parole humaine en se basant sur le concept de psychophysique de l'ouïe (Hermansky, 1990). La PLP rejette des informations non pertinentes de la parole et améliore ainsi le taux de reconnaissance de la parole. La PLP est identique à la LPC, sauf que ses caractéristiques spectrales ont été transformées pour correspondre aux caractéristiques du système auditif humain.

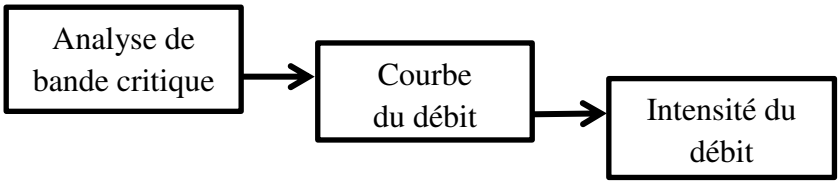


Figure 2.4. Etapes du processus PLP (Dave, 2013)

La PLP se rapproche de trois principaux aspects perceptuels, à savoir: les courbes de résolution de bande critique, la courbe de sonorité égale et la relation force-loi de l'intensité-débit, connues sous le nom de racine cubique (Figure 2.4)

2.4.2.4. Mel-scaled Frequency Cepstral Coefficients (MFCC)

« Les travaux de Stevens (Stevens et Volkman., 1940) ont permis la mise en évidence de la loi de puissance ou loi de Stevens selon laquelle l'intensité de la perception d'un stimulus n'augmente pas linéairement en fonction de sa puissance mais de façon exponentielle en tenant aussi compte des modalités de l'expérimentation »(Vaufraydaz, 2002). A cet effet, dans (Davis et Mermelstein., 1980) les auteurs ont proposé des coefficients qui basés sur une échelle de perception non linéaire appelée Mel ; ce sont les Mel-scaled Frequency Cepstral Coefficients (MFCC). Le passage de l'échelle Hertz à l'échelle Mel se fait via la relation suivante :

$$M_{mels} = x.\log\left(1 + \frac{f_{Hz}}{y}\right) \quad (2.2)$$

L'algorithme de calcul des coefficients MFCCs peut être décrit comme suit :

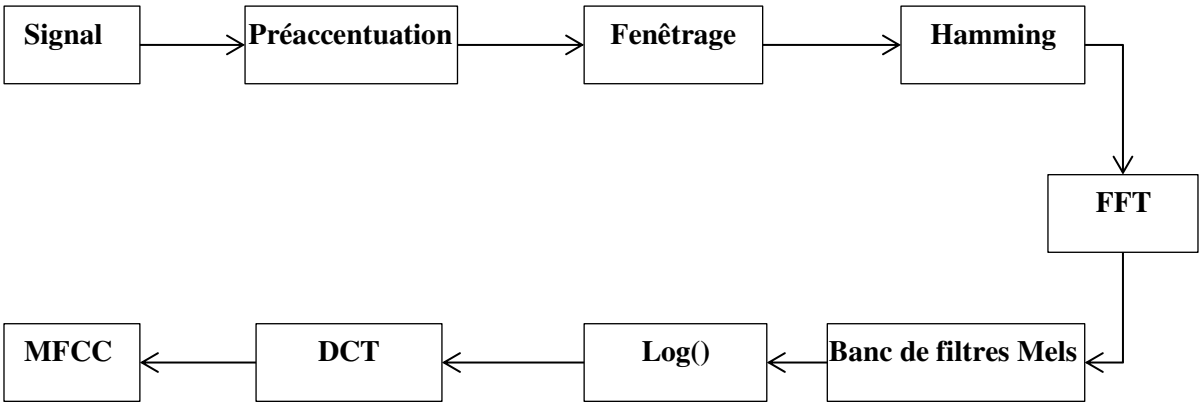


Figure 2.5. Algorithme de calcul des MFCCs

Le nombre de paramètres extraits à l'issu de l'algorithme varie selon l'application cible. Pour notre cas, nous estimons que l'utilisation de 12 coefficients MFCC (en plus de l'énergie) ainsi que leurs dérivées premières et secondes permet de représenter au mieux le signal ; ce choix vient du fait que de nombreux travaux en reconnaissance de la parole ont utilisé ces valeurs et que des travaux antérieurs au sein de notre équipe a adopté ce nombre avec de bons résultats (Necibi, 2015).

2.5. Reconnaissance de mots isolés et de la parole continue

La reconnaissance de mots isolés consiste à retrouver les phonèmes et les mots dans un signal de parole qui est une tâche difficile pour la reconnaissance. De ce fait, séparer tous les mots prononcés par des silences permet de simplifier le problème. Ceci, nous ramène à introduire les approches utilisées en RAP : l'approche globale et l'approche analytique.

Dans l'approche globale, l'unité de base est généralement le mot. Cette méthode fournit une image acoustique de chaque mot à identifier et permet d'éviter l'influence mutuelle des sons à l'intérieur des mots. Elle se limite aux petits vocabulaires.

L'approche analytique tire parti de la structure des mots et identifie les composantes élémentaires (phonèmes, syllabes, ...) du discours. Celles-ci sont les unités de base à reconnaître. Cette approche est plus appropriée que la précédente pour reconnaître de grands vocabulaires, il suffit d'enregistrer dans la mémoire de la machine les principales caractéristiques des unités de base.

Pour la reconnaissance de mots isolés à grand vocabulaire, la méthode globale ne convient plus car la machine nécessiterait une mémoire et une puissance considérable pour respectivement stocker les images acoustiques de tous les mots du vocabulaire et comparer un mot inconnu à l'ensemble des mots du dictionnaire. C'est donc la méthode analytique qui est désormais utilisée : les mots ne sont pas mémorisés dans leur intégralité, mais considérés comme des suites de phonèmes ou syllabes.

La phase d'apprentissage et de reconnaissance de la méthode analytique et globale sont presque les mêmes, ce qui différencie ces deux méthodes est l'entité à reconnaître : pour la première il s'agit du phonème ou syllabes, pour l'autre le mot. On distingue deux phases:

2.6. Approches de la reconnaissance de la parole

Le processus de reconnaissance de la parole consiste à traiter la variabilité de la parole et tenir compte de l'apprentissage de la relation entre l'énoncé spécifique et le mot correspondant. Il y a eu des progrès constants dans le domaine de la reconnaissance de la parole au cours des dernières années dans chacune des trois approches suivantes: l'approche acoustico-phonétique, l'approche reconnaissance de formes et l'approche intelligence artificielle (Dixit et Kaur, 2013).

2.6.1. Approche acoustico-phonétique

L'approche acoustico-phonétique est la première approche de la reconnaissance de la parole (Hemdal et Hughes, 1967) ; elle se base sur la recherche de régions de la parole et l'attribution d'étiquettes appropriées à ces régions. Elle est basée sur la théorie acoustique phonétique qui stipule que dans un langage parlé, il existe un nombre fini d'unités phonétiques distinctes et que ces unités sont caractérisées par un ensemble de propriétés qui se manifestent au travers du signal de parole (Rabiner, 1983). Il s'agit alors de définir une relation entre les caractéristiques spectrales du signal et les unités phonétiques du langage.

Généralement, l'approche acoustico-phonétique comprend trois étapes : La première étape de l'approche acoustico-phonétique est une analyse spectrale de la parole associée à une détection de caractéristiques qui convertit les mesures spectrales en un ensemble de caractéristiques qui décrivent les grandes propriétés acoustiques des différentes unités phonétiques, appelée aussi étape d'extraction des caractéristiques. L'étape suivante est une phase de segmentation et d'étiquetage dans laquelle le signal de parole est segmenté en régions acoustiques stables, puis on attache une ou plusieurs étiquettes phonétiques à chaque région segmentée, ce qui entraîne une caractérisation de la parole en phonèmes. Parmi ces caractéristiques, on utilise fréquemment la nasalité (présence ou absence de résonance nasale), localisation des formants (fréquences des trois premières résonances), voisement ou non

(excitation périodique ou non), …etc. La dernière étape de cette approche tente de déterminer un mot valide (ou une chaîne de mots) à partir des séquences d'étiquettes phonétiques produites par la segmentation à l'étiquetage. Dans cette étape, les contraintes linguistiques (le vocabulaire, la syntaxe et les règles sémantiques) sont utilisées pour accéder au lexique pour le décodage de mots basé sur le réseau de phonèmes. La figure suivante montre un schéma de l'approche acoustico - phonétique.

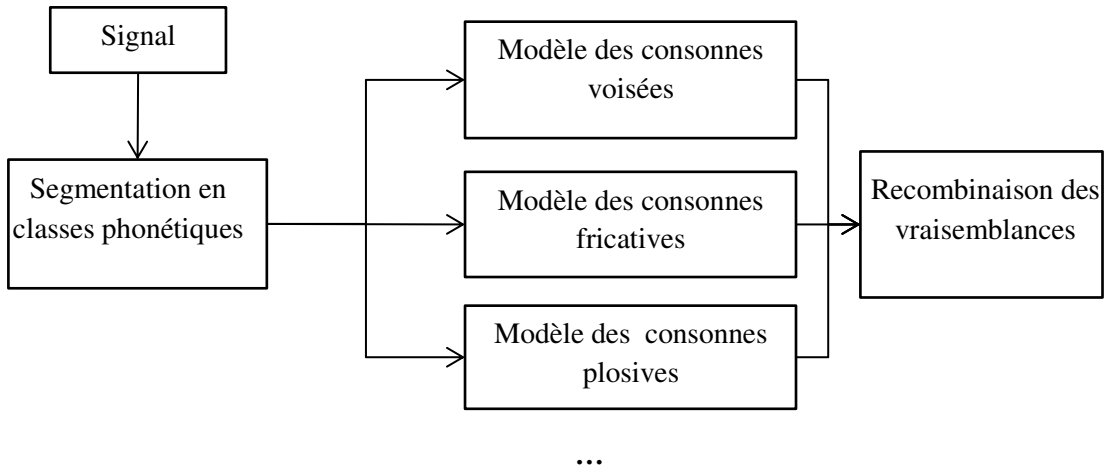


Figure 2.6. Processus de reconnaissance selon l'approche acoustico – phonétique (d'après (Pellegrino, 1998))

L'approche acoustico - phonétique n'a pas été beaucoup utilisée dans la plupart des applications commerciales (Dixit et Kaur, 2013).

2.6.2. Approche reconnaissance de formes

La reconnaissance de formes fait partie du domaine de « machine learning » qui se concentre sur la reconnaissance des formes et les régularités dans les données, plusieurs domaines utilisent cette approche, parmi les applications typiques des techniques de reconnaissance de formes sont la reconnaissance automatique de la parole (Itakura, 1975; Rabiner, 1989; Rabiner et Juang, 1993).

L'approche de la reconnaissance de formes passe généralement par deux étapes essentielles: l'étape d'apprentissage, dans cette phase, on présente au système un ensemble d'exemples et à la fin de cette phase le système devrait être en mesure de les distinguer correctement. Les formes (modèles) ainsi apprises par le système sont appelées formes de

références. La deuxième étape est l'étape de test appelé aussi comparaison. En phase de test, des formes qui n'existaient pas dans le corpus d'apprentissage sont présentées au système, et ce dernier devrait être capable de les caractériser correctement ; cette caractérisation est appelée : classification (Bahi, 2005).

La caractéristique essentielle de cette approche est qu'elle a un fond mathématique bien formulé et elle établit des représentations cohérentes de forme, pour une comparaison des formes fiables, à partir d'un ensemble de données étiquetées via un algorithme d'apprentissage formel. Ensuite, une comparaison directe est faite entre les formes de parole inconnues et les modèles appris dans l'étape d'apprentissage afin de classer les formes inconnues.

Plusieurs méthodes s'apparentent à l'approche de reconnaissance de formes, elles se distinguent principalement, par la manière dont les formes de références sont créées, modélisées et par la méthode qui sert à classer les formes inconnues. Parmi ces techniques on cite : les modèles de Markov cachées (HMM), l'alignement temporel (DTW), les supports à marge vaste (SVM), la quantification vectorielle (VQ), etc (Gamit, 2015).

Une représentation de forme de parole peut être sous la forme d'un modèle de parole ou d'un modèle statistique (par exemple, un HMM) et peut être appliqué à un segment de signal (unité de base : phonème, syllabe, mot).

2.6.2.1. Les Modèles de Markov Cachés

Les modèles de Markov cachés (en Anglais Hidden Markov Models, HMMs) ont été introduits par Baum et al. à la fin des années 60. Les HMMs sont des modèles statistiques très puissants qui trouvent leurs applications dans de multiples domaines. Plus particulièrement, dans la reconnaissance de la parole : il s'agit de faire correspondre une séquence de phonèmes (la séquence d'observations) avec une séquence de phonèmes (la séquence d'états cachés) (Rabiner, 1989). D'autres applications des HMMs existent telles que : l'analyse cryptographique, la traduction automatique, la recherche de virus polymorphe, la reconnaissance d'écriture manuscrite, la bio-informatique, l'analyse financière et bien d'autres... De façon générale, un HMM permet de retrouver des informations cachées que l'on sait liées à des informations observables.

Un HMM est un processus stochastique défini, caractérisé par le quintuplé $\{\pi_i , Q, O , a , b \}$ (Rabiner, 89 ; Rabiner et Juang, 1993) :

- π_i est la distribution de probabilité de l'état initial (dans la reconnaissance vocale le modèle HMM choisi est gauche-droite, donc π_i prend le premier état comme état initial).
- Q est l'ensemble d'états $Q = \{q_1, q_2, \dots q_N\}$.
- *O* est l'ensemble des observations ou les symboles à émettre par les états $O = \{O_1, O_2, \dots O_T\}$.
- *A* est la probabilité de déplacement d'un état q_i a un autre q_j avec $a_{ij} = P(q_j|q_i)$.
- *B* est la fonction de distribution de probabilité des observations O_i pour l'état q_i à l'instant t avec $b_{q_i}(O_t) = P(O_t|q_i)$.

Lorsque nous utilisons les HMM, trois problèmes majeurs apparaissent (Rabiner et Juang, 1993):

Problème d'évaluation : le fait de trouver l'évaluation d'une probabilité $P(O|\lambda)$ de la suite d'observations O selon le modèle λ.

Problème de décodage : l'estimation de la suite d'états cachés appartenant à Q sachant qu'on a l'ensemble des observations O et le modèle λ.

Problème d'apprentissage : c'est le problème d'ajustement des paramètres du modèle λ. pour maximiser la probabilité $P(O|\lambda)$.

Ce dernier problème lié aux modèles de Markov cachés est le problème de ré-estimation (d'apprentissage). Etant donné une séquence d'observation O_T, et un modèle λ, comment on peut calculer la probabilité d'avoir une séquence d'observation sachant le modèle λ ($P(O|\lambda)$). Le but est de trouver un nouveau modèle HMM, λ', à partir d'un modèle initial λ qui maximise la probabilité $P(O|\lambda')$; de telle sorte que $P(O|\lambda') >= P(O|\lambda)$. En 1967, les deux chercheurs Baum et Welch proposent un algorithme d'apprentissage itératif, il s'appelle l'algorithme de Baum-Welch qui est un cas particulier de l'algorithme EM (Expectation – Maximisation) (Dempster et Rubin, 1977). Cet algorithme permet d'estimer les

paramètres du modèle qui maximisent la probabilité $P(O|\lambda)$. L'algorithme de Baum-Welch converge vers un maximum local. Il est basé sur l'algorithme Forward-Backward, il calcul les deux probabilités, Forward $\alpha_t(i)$ et Backward $\beta_t(i)$, pour chaque état $qi\,;\, i \in [1,N]$ d'un HMM et chaque trame $t \in [1,T]$ d'une séquence d'observation $O = O_1, O_2, \ldots, O_T$. La complexité du calcul de ces probabilités est de l'ordre de $O(N^2T)$. Le calcul des probabilités Forward et Backward sert à pondérer les contributions de chaque observation O_t aux paramètres du HMM.

Chaque état émet un symbole (observation), pour cela il faut énumérer toutes les séquences d'états possibles $(Q = q_1 \ldots q_N)$ de longueur égal au nombre d'observation $O = O_1, O_2, \ldots, O_T$, avec $N = T$ et q_1 est l'état initial.

La probabilité d'émission d'une séquence observation O sachant la séquence d'état Q et le modèle λ est :

$$P(O|Q,\lambda) = \prod_{t=1}^{T} P(O|q_t,\lambda) = \prod_{t=1}^{T} b_{q_t}(O_t) \tag{2.3}$$

La probabilité d'avoir la séquence d'état Q sachant le modèle λ se calcule par :

$$P(Q|\lambda) = \pi_i . a_{1,2} . a_{2,3} \ldots a_{T,T-1} \text{ avec } \pi_i = q_1 \tag{2.4}$$

La probabilité conjointe de O et Q est égale à:

$$P(O,Q|\lambda) = P(O|Q,\lambda) . P(Q,\lambda) \tag{2.5}$$

La probabilité d'avoir une séquence d'observation sachant le modèle λ est :

$$P(O|\lambda) = \sum_{q_1 \ldots q_T} P(O|Q,\lambda) . P(Q|\lambda) \tag{2.6}$$

L'algorithme Forward-Backward consiste à calculer la probabilité $P(Q|O)$. A un instant t, on peut diviser la séquence d'observation $O_{1:T}$ en deux parties : $O_{1:t}$ et $O_{t+1:T}$,

$\forall t = 1..T$. L'algorithme Forward sert à calculer la probabilité $P(q_t, O_{1:t})$ et l'algorithme Backward sert à calculer la probabilité $P(O_{t+1:T}|q_t)$.

$P(q_t|O_t)$ est la proportion de $P(q_t, O) = P(O_{t+1}|q_t).P(q_t, O_{1:t})$

On peut décrire l'algorithme Forward au travers des étapes de calcul suivantes (Rabiner et Juang, 1993):

$$\begin{aligned}\alpha_t(i) = P(q_t, O_{1:t}) &= \sum_{q_{t-1}=1}^{N} P(q_t, q_{t-1}, O_{1:t}) \\ &= \sum_{q_{t-1}=1}^{N} P(O_t|q_t).P(q_t|q_{t-1}).P(q_{t-1}, O_{1:t-1})\end{aligned} \tag{2.7}$$

$$\alpha_t(i) = \sum_{q_{t-1}=1}^{N} P(O_t|q_t).P(q_t|q_{t-1}).\alpha_{t-1}(i) \tag{2.8}$$

Avec

$$\alpha_1(1) = P(q_1).P(O_1|q_1) = \pi_1.b_1(O_1) \tag{2.9}$$

A l'instant $t+1$, $\alpha_{t+1}(j)$ est :

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i).a_{i,j}\right].b_j(O_{t+1}) \tag{2.10}$$

Avec $1 \leq t \leq T-1$ et $1 \leq j \leq N$.

En terminant avec la probabilité $P(O|\lambda)$:

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i) \tag{2.11}$$

L'algorithme Backward consiste en le calcul de $P(O_{t+1:N}|q_t,\lambda), \forall t = 1..N$. Il peut être décrit comme suit (Rabiner et Juang, 1993):

$$\beta_t(q_t) = P(O_{t+1:N}|q_t,\lambda) = \sum_{qt+1=1}^{m} P\,(O_{t+1:N}, q_{t+1}|q_t) = \sum_{q_{t+1}} P(O_{t+2:N}|q_{t+1}).P(O_{t+1}|q_{t+1}).P(q_{t+1}|q_t) \quad (2.12)$$

$$\beta_t(q_t) = \sum_{z_{k+1}} \beta_{t+1}(q_{t+1}).P(O_{t+1}|q_{t+1}).P(q_{t+1}|q_t) \quad (2.13)$$

$$\beta_t(i) = \sum_{j=1}^{N} a_{i,j}\,.\,b_j\,(O_{t+1}).\beta_{t+1}(j) \quad (2.14)$$

Avec $t = 1,..N-1$; $i = q_t$; $j = q_{t+1}$; $\beta_T(q_T) = 1, \forall q_t$

Une fois, on a déterminé les deux probabilités Forward et Backward, on doit ré-estimer les paramètres du modèle HMM :

$\varepsilon_t(i,j)$ est la probabilité d'être à l'état i à instant t et à l'état j à l'instant $t+1$

$$\varepsilon_t(i,j) = P(q_t, q_{t+1}|O_t,\lambda) \quad (2.15)$$

La probabilité $\gamma_t(i)$, d'être à l'état i à l'instant t sachant la séquence d'observation et le modèle M, peut être calculée à partir de $\varepsilon_t(i,j)$.

$$\gamma_t(i) = \sum_{j=1}^{N} \varepsilon_t(i,j) \quad (2.16)$$

En utilisant les formules précédentes, on peut calculer les nouvelles probabilités $\hat{a}_{ij}$ et $\widehat{b_j}(O_t)$.

$$\hat{a}_{ij} = \frac{\text{Nombre estimés des transitions de l'état i à l'état j}}{\text{Nombre estimés des transitions de l'état i}} \tag{2.17}$$

Le ratio entre le nombre estimés des transitions entre l'état i et l'état j et le nombre estimés de toutes les transitions à partir de l'état i est :

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \varepsilon_t(i,j)}{\sum_{t=1}^{T-1} \sum_{j=1}^{N} \varepsilon_t(i,j)} \tag{2.18}$$

$$\hat{b}_j(O_t) = \frac{\text{le temps estimé dans l'état j en observant le symbole } O_t}{\text{le temps estimé dans l'état j}} \tag{2.19}$$

Le ratio entre le nombre de fois que l'observation émise à l'état j soit O_t, et le nombre estimé de fois qu'une observation est émise à l'état j est :

$$\hat{b}_j(O_t) = \frac{\sum_{t=1 \text{ s.t.} O_t}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)} \tag{2.20}$$

Etant donné une observation O et un HMM "λ", en phase de reconnaissance, on souhaite estimer la probabilité: p (w|O), où w est une séquence de mots.

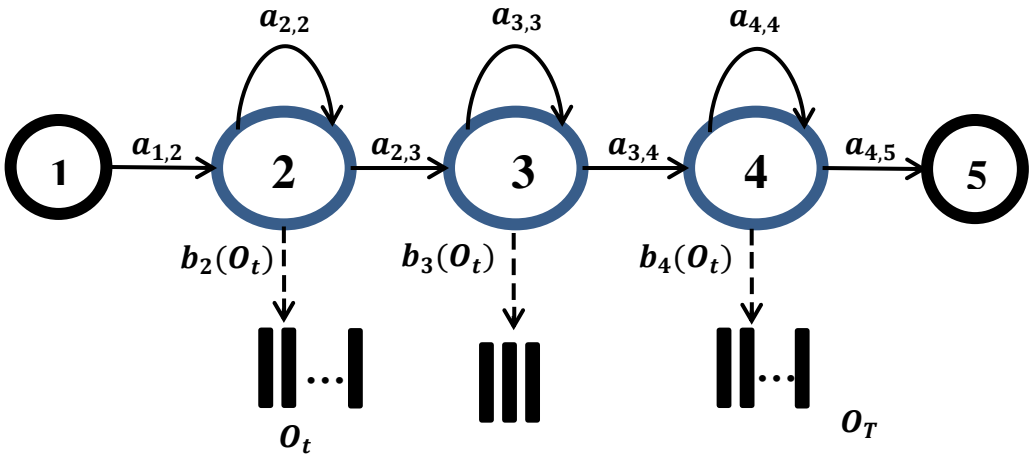


Figure 2.7. Un modèle HMM avec 3 états émetteurs

La vraisemblance P(O|w) est le modèle acoustique préalablement appris (en phase d'apprentissage) et le prior p(w) est déterminé par le modèle de langage. L'unité de base représentée par le modèle acoustique est le phone. La séquence d'état ayant pu générer la séquence observée est estimée via l'algorithme de Viterbi (Rabiner, 1989).

$$p(w|O) = p(O|w)p(w) \tag{2.21}$$

Afin de résoudre ce problème de décodage, l'algorithme de Viterbi est employé. Le critère d'optimalité ici est de rechercher la meilleure suite d'états par la technique modifiée de la programmation dynamique. L'algorithme de Viterbi est un algorithme de recherche parallèle, il recherche la meilleure suite d'états en traitant tous les états en parallèle. Nous devons maximiser P (Q|O, λ) pour détecter la meilleure suite d'états. Soit la probabilité $\delta_t(i)$ qui représente la probabilité maximale le long du meilleur chemin probable d'une suite d'états d'une séquence d'observation donné après t instants et en étant à l'état i ;

$$\delta_t(i) = \max_{q_1, q_2 \dots q_{t-1}, q_t} P[q_1, q_2 \dots q_{t-1}, q_t = S_i, o_1 \dots, o_t|\lambda] \tag{2.22}$$

La meilleure séquence d'états et retournée par une autre fonction $\psi_t(j)$. Cette fonction tient l'index de l'instant t-1, à partir duquel la meilleure transition est faite à l'état actuel.

2.6.2.2. L'alignement temporel

L'alignement temporel (Dynamic Time Warping, DTW), appelé aussi alignement de Viterbi, a été introduit par (Sakoe et Chiba., 1978). Le DTW est une méthode qui recherche une correspondance optimale entre deux séquences données avec certaines restrictions. Les séquences sont déformées de manière non linéaire dans la dimension temporelle. Longtemps, le DTW a été considéré comme la méthode la plus appropriée pour la reconnaissance de la parole car elle tient compte des compressions et extensions temporelles qui sont observées lors de la prononciation plus ou moins rapide d'un mot.

Le principe de base est d'essayer de trouver le chemin optimal à parcourir parmi l'ensemble des distances entre les vecteurs. En traitement de la parole un mot n'est jamais prononcé deux fois de la même manière, c'est pourquoi il est difficile de comparer deux prononciations du même mot. Différentes parties des mots sont allongées ou au contraire compressées de manière à trouver l'alignement qui aboutit à une meilleure concordance possible entre les éléments de test et les vecteurs de référence en fonction des caractéristiques. La mesure de distance locale est la distance entre les caractéristiques d'une paire de trames tandis que la distance globale est celle entre le début de mot et la dernière paire de trames.

Considérons deux séquences de vecteurs de caractéristique dans un espace n-dimensionnel : $x = [x_1, x_i, \ldots, x_n]$, $y = [y_1, y_j, \ldots, y_m]$.

Les deux séquences sont alignées sur les côtés d'une grille, l'une sur le haut et l'autre sur le côté gauche. Les deux séquences commencent en bas à gauche de la grille.

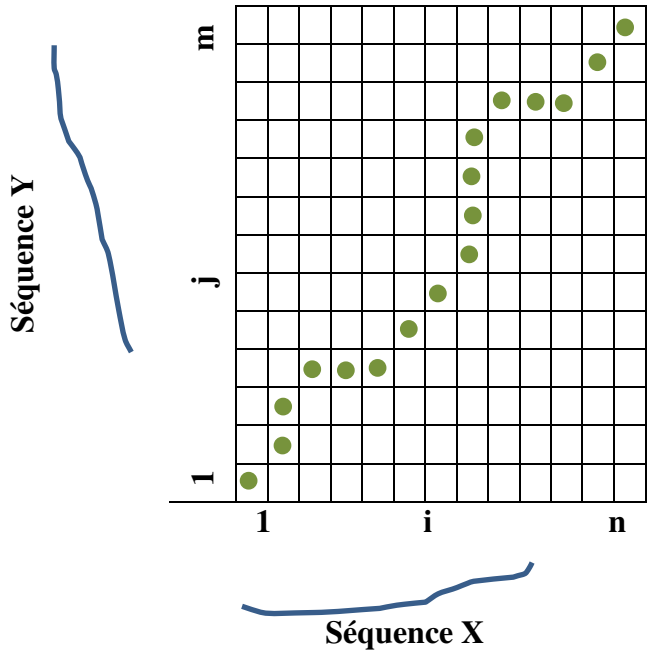


Figure 2.8. Grille de distance globale

Dans chaque cellule, une mesure de distance est placée, en comparant les éléments correspondants des deux séquences.

La meilleure correspondance ou alignement entre ces deux séquences est le chemin à travers la grille, qui minimise la distance totale entre elles, ce qui est appelé Distance Globale

GD. La distance globale est le minimum de la somme des distances (distance euclidienne) entre les éléments individuels sur le chemin divisé par la somme de la fonction de pondération. Pour toute séquence considérablement longue, le nombre de chemins possibles à travers la grille sera très important. La mesure de distance globale est obtenue à l'aide d'une formule récursive. (Subhashini, 2017)

$$GD_{xy} = LD_{xy} + min\,(GD_{x-1\,y-1}, GD_{x-1\,y}, GD_{x\,y-1}) \quad (2.\,23)$$

Où, LD = Distance locale (Distance Euclidienne)

2.6.2.3. La Quantification Vectorielle

Etant donné un grand ensemble de vecteurs, la technique de la quantification vectorielle vise à regrouper ces vecteurs sous la bannière d'un nombre fini de vecteurs représentatifs. Ces vecteurs représentatifs représentent le centre de classes auxquelles sont supposés appartenir les vecteurs de départ. Il est évident que le nombre de classes est largement inférieur à celui des données de départ. Ces vecteurs représentatifs, qui sont les centres des classes définies, sont regroupés dans un dictionnaire (CodeBook). L'utilisation de cette technique induit deux étapes, une étape d'apprentissage dans laquelle est construit le dictionnaire en se basant sur les données de départ, et une étape de classification (reconnaissance) dans laquelle les nouveaux vecteurs sont assignés aux classes sur la base de leur distance euclidienne au centre de chaque classe.

Dans le cas de la reconnaissance de la parole, les vecteurs considérés sont les vecteurs acoustiques issus de l'analyse du signal.

2.6.3. L'approche de l'Intelligence Artificielle

L'approche intelligence artificielle est une hybridation de l'approche acoustico-phonétique et celle de la reconnaissance de formes. Elle exploite les idées et les concepts de méthodes de reconnaissance de formes qui tentent d'automatiser le processus de reconnaissance en s'inspirant de la manière dont l'être humain utilise son intelligence en écoutant, analysant et finalement sur la base de méthodes acoustiques et phonétiques il utilise

des règles de décision. L'idée de base étant d'incorporer différentes techniques et d'intégrer diverses sources de connaissances pour la prise en charge d'un problème donné.

Parmi les méthodes de l'intelligence artificiell, on trouve: les Perceptron multicouches (MLP), les cartes de Kohonen (SOM), les Time-Delay Neural Network (TDNNs), etc.

2.6.3.1. Le Réseau de Neurones Artificiels (RNA)

Un réseau de neurones artificiels (ou Artificial Neural Network (ANN) en anglais) est un modèle de calcul dont la conception est très schématiquement inspiré du fonctionnement du neurone biologique. Les réseaux de neurones sont des processeurs distribués massivement parallèles qui possèdent la propriété de mémoriser des expériences passées en vue de les utiliser dans des procédés non connus à l'avance.

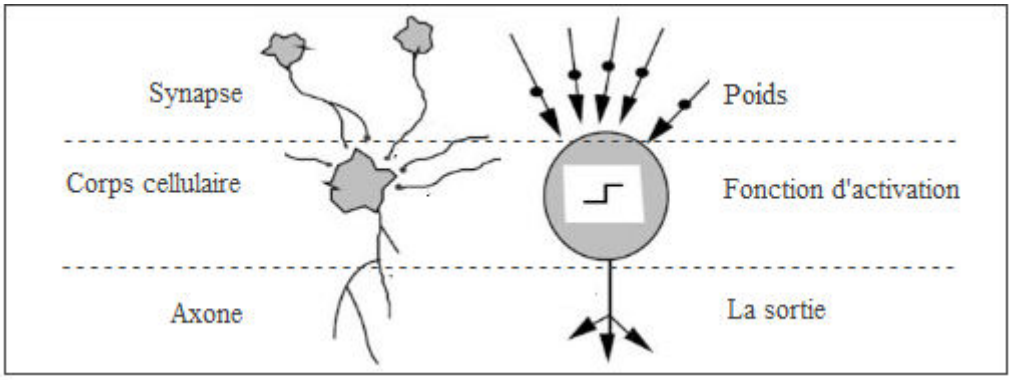


Figure 2.9. Les composons d'un neurone biologique vs artificiel.

Un réseau de neurones artificiel est ainsi constitué de (Bedaux, 2017):

- Des cellules (ou neurones) : connectée entre elles par des liaisons affectées de poids (une pondération).
- Des poids : A chacune de ces entrées est associée un poids w représentatif de la force de la connexion.
- Des entrées : Chaque cellule dispose d'une entrée, qui lui permet de recevoir de l'information des autres cellules.
- D'une fonction d'activation (ou fonction de seuillage, ou encore fonction de transfert) sert à introduire une non linéarité dans le fonctionnement du neurone.
- De sortie (résultats) (Figure 2.9).

Les réseaux de neurones artificiels ressemblent au cerveau par deux aspects: la connaissance est acquise par le réseau au moyen d'un processus d'apprentissage et les intensités des connexions entre les neurones connues par les poids synaptiques sont utilisées pour mémoriser la connaissance. Il existe plusieurs types de règles d'apprentissage qui ont pour rôles de modifier les poids synaptiques du réseau : la correction d'erreurs, Boltzmann, Hebb, l'apprentissage compétitif, etc.

2.6.3.2. Le Perceptron

Le perceptron est créé par (Rosenblatt, 1958), c'est un modèle de réseau de neurones artificiel. Un perceptron linéaire (Figure 2.9) à seuil prend en entrée n valeurs $x_1, \ldots, x_n$ et calcule une sortie O. Il est défini par la donnée de $n+1$ constantes : les coefficients synaptiques $w_1, \ldots, w_n$ et le seuil (ou le biais). La sortie O est calculée par la formule :

$$O = \begin{cases} 1 \text{ si } \sum_{i=1}^{n} w_i x_i > 0 \\ x \\ 0 \text{ sinon} \end{cases} \quad (2.24)$$

Comme le neurone humain, le perceptron est activé lorsque la somme de ces entrées dépasse une certaine valeur seuil donnée. Ce modèle rencontre des difficultés à être utilisé dans de nombreux exemples non linéaires (la fonction du ou exclusif).

2.6.3.3. Le Perceptron Multi Couches (PMC)

Le perceptron multi couches (PMC) est un modèle des réseaux de neurones qui est très répandu, il est définit par :

- Une couche d'entrée qui correspond aux variables d'entrée.
- Une couche de sortie.
- Un certain nombre de couches intermédiaires. Les liens n'existent qu'entre les cellules d'une couche avec les cellules de la couche suivante.

Les cellules (neurones) ne reçoivent des informations que de la couche précédente c'est-à-dire l'information circule d'une couche à sa couche suivante, donc pas de boucle de retour, ils sont « Feed-forward ».

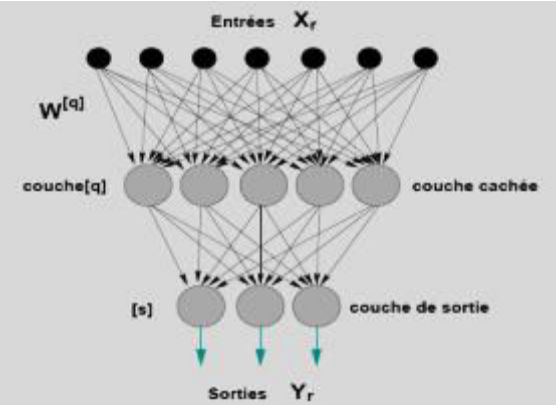


Figure 2.10. Perceptron à deux couches

Pour pouvoir utiliser les réseaux multicouches en apprentissage, deux choses sont indispensables:

1. Une méthode indiquant comment choisir une architecture de réseau pour résoudre un problème donné. (Combien de couches cachées? combien de neurones par couche ?)
2. Une fois l'architecture choisie, un algorithme d'apprentissage qui calcule, à partir de l'échantillon d'apprentissage, les valeurs des coefficients synaptiques pour construire un réseau adapté au problème.

2.6.3.4. Réseaux de neurones à délai temporel (TDNN)

Les réseaux multicouches à délai temporel ou Time-Delay Neural Networks (TDNN) (Waibel, 1988) sont une version avancée du réseau de neurones artificiel. Les réseaux de neurones à délai temporel (TDNN) représentent une tentative efficace pour former un perceptron multicouche statique (MLP) pour le traitement de séquences temporelles. Le TDNN est constitué de sous réseaux agissant comme des extracteurs de formes sur une période définie de la fenêtre d'entrée, chaque sous réseau ayant pour tâche de reconnaitre des séquences.

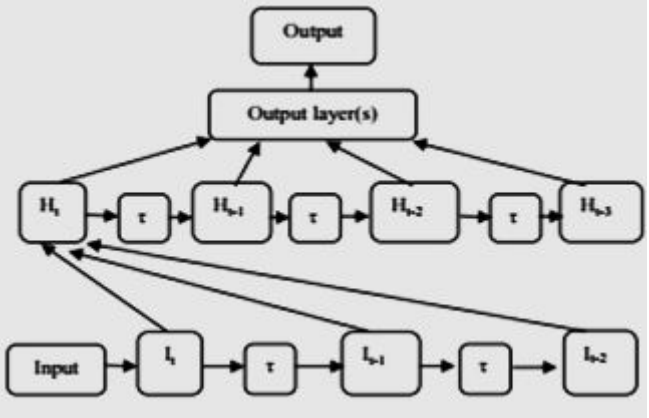


Figure 2.11. Réseau de neurones à délai temporel (TDNN)

Le TDNN se caractérise par :

- Le nombre de couches (Chaque couche a deux directions : direction temporelle et direction caractéristique).
- Le nombre de neurones de chaque couche selon la direction temporelle, fenêtre d'observation.
- Le nombre de neurones de chaque couche selon la direction caractéristique.
- La taille de la fenêtre temporelle qui se traduit par le nombre de neurones de la couche i suivant la caractéristique temporelle vue par un neurone de la couche i + 1.
- Le délai temporel (nombre de neurones) entre deux fenêtres successives dans une couche donnée (Benammar, 2012).

2.7. Outils de segmentation manuelle et d'étiquetage

2.7.1. L'outil Praat

Praat est un outil complet créé pour l'étude du signal de la parole (l'analyse, la manipulation et l'annotation) à l'Institute de sciences phonétiques de l'Université d'Amsterdam en 1996. Praat donne la main de faire plusieurs tâche comme le traçage des graphiques, la construction des grammaires, l'analyses statistiques, la synthèse articulatoire et la simulation des réseaux de neurones. Il est tellement facile à utiliser grâce à ses interfaces graphiques et ses menus simplifiés, toutes personnes n'ont pas de l'expérience en traitement

de la parole peuvent le manipuler facilement (Goldman, 2006). La figure 2.12 montre l'interface de Praat.

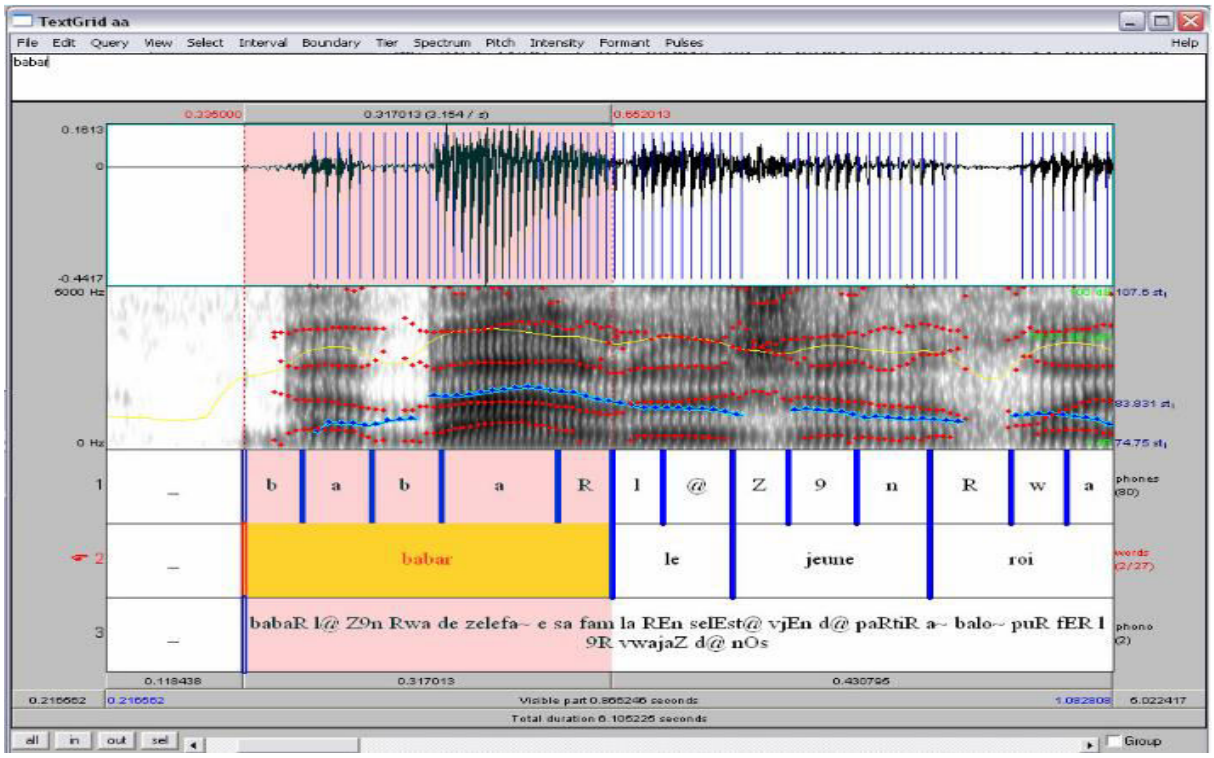


Figure 2.12. La phase d'analyse et d'étiquetage de signal par l'outil Praat(Goldman, 2006)

2.7.2. HSLAB de HTK

HSLAB est un outil d'étiquetage interactif des fichiers sonores, il est intégré dans la librairie HTK (Hidden Markov Model Toolbox). HSLAB charge un fichier sonore sous forme d'onde échantillonnée, il permet aux utilisateurs de marquer les frontières des unités de base et de leurs attribuer des étiquettes. Il permet aussi d'importer des fichiers d'étiquettes existants.

Au démarrage, l'outil HSLAB affiche une fenêtre décomposée en deux parties: une partie d'affichage et l'autre de contrôle (voir la figure 2.13). La première partie contient la forme d'onde de parole ainsi que les étiquettes associées. La partie de contrôle contient des boutons qui servent à manipuler le signal. Les boutons sont groupés selon leur fonction. Le premier groupe en haut contient les boutons de commandes d'entrée/sortie. Les boutons de groupe 2 réservés aux fonctions de lecture, de visualisation et d'enregistrement. Les boutons du troisième groupe sont utilisés pour l'étiquetage.

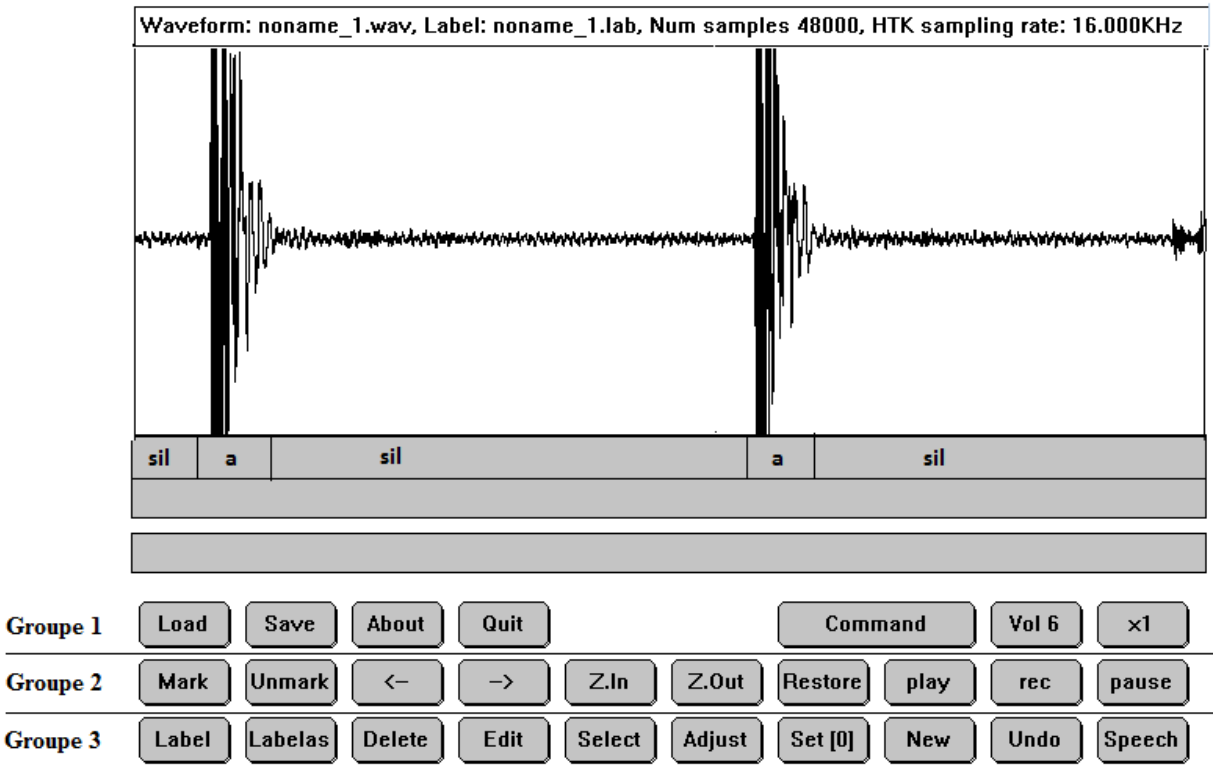


Figure 2.13. La phase d'étiquetage du signal par HSLAB de HTK

2.7.3. WaveSurfer

WaveSurfer est développé au « Center for Speech Technology » de l'Institut Royal de Technologie à Stockholm. Il a été conçu pour l'étiquetage des données audio. Il fonctionne sous Windows, linux. WaveSurfer est un outil de source ouverte.

Lorsqu'un fichier son est ouvert sous WaveSurfer, une fenêtre s'affiche. Chaque fichier son et toutes les données associées peuvent être visualisées. Des volets contiennent le signal, des spectrogrammes, des étiquettes. Il gère plusieurs sons simultanés dans des zones de travail.

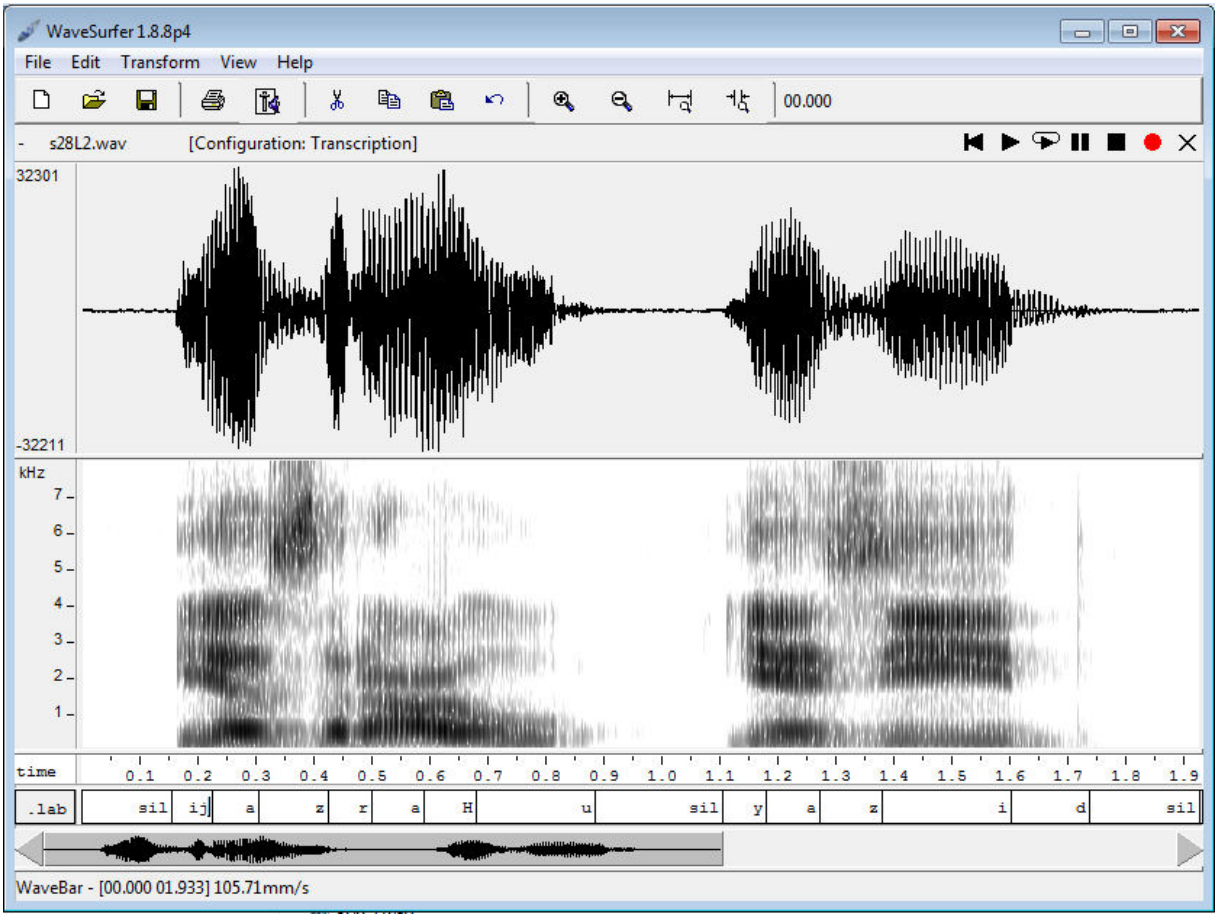


Figure 2.14. Phase d'étiquetage par Wavesurfer de la phrase يزرع يزيد

2.8. Outils de création de systèmes de RAP

Le monde de la reconnaissance automatique de la parole (RAP) est devenu très flexible et pratique grâce au progrès des boites à outils et leurs disponibilités ce qui a permis aux chercheurs d'avancer de manière très rapide ces dernières années. Les premières applications de reconnaissance automatique de la parole (RAP) étaient basées sur la reconnaissance de mots isolés (1970) alors que les applications actuelles s'orientent de plus en plus vers la parole spontanée en passant par la parole continue. Néanmoins, des problèmes restent à résoudre pour accroître la robustesse de ces systèmes et pour étendre leurs capacités de dialogue. Les outils de modélisation les plus utilisés dans le domaine de la reconnaissance vocale étant les Modèles de Markov Cachés (HMM).

La disponibilité des outils de reconnaissance de la parole avec leur architecture modulaire et leur open source permet aux chercheurs et programmeurs d'implémenter et tester de nouveaux algorithmes. En pratique, différentes boite à outils ont été utilisées pour la création des systèmes de reconnaissance vocale avec différents langage, on cite : HTK, Sphinx, Kaldi, boite ASR de Matlab et Java Speech, etc. Pour répondre à la question : quelle est la meilleure librairie, nous nous sommes intéressés à l'étude de quelques outils de construction des systèmes de RAP basés sur les HMMs pour voir leur capacité à gérer des systèmes de grand vocabulaire. Pour cela, nous avons testé les librairies les plus utilisé dans le domaine de la RAP tels que la plateforme open source HTK (Hidden Markov Model ToolKit), Sphinx 4 et la toolbox de Matlab. Nous avons utilisé le corpus TIMIT et nous avons créé deux autres corpus avec l'aide des étudiants du département d'informatique de notre université.

2.8.1. Hidden Markov Model Toolbox (HTK)

La Hidden Markov Model Toolbox (HTK) a été développée à l'université de Cambridge. Cette boite à outils dédiée aux Modèles de Markov Cachés est principalement utilisée pour la reconnaissance de la parole. Elle se compose d'un ensemble de modules et d'outils disponibles gratuitement et téléchargeables à partir du site. HTK est implémenté en langage C et il s'exécute en ligne de commande (figure 2.15), il est capable de mettre en œuvre un grand vocabulaire, indépendamment du locuteur et est applicable sur n'importe quelle langue. La documentation sur HTK est très riche avec des exemples pratiques (vers 300 pages) sur http://htk.eng.cam.ac.uk/.

```
% Phase de préparation de données %

    HSlab -n

    Hcopy -T 1 -C confighcopy -S datatrain.scp

% Phase  d'initialisation et apprentissage %

    HInit -I words.mlf -S train.scp -H hmm/macros -M hmm/hmm0 -T 1 -C config -l sil hmm/sil

    HCompV -C config -f 0.01 -m -S train.scp -M hmm/hmm0 hmm/proto

    HRest -I words.mlf -i 100 -S train.scp -H hmm/hmm0/macros -T 1 -M hmm/hmm1 -C config -l sil hmm/hmm0/sil

    HERest -C config -I phones0.mlf -t 250.0 150.0 1000.0 -S train.scp -H hmm/hmm0/macros -H hmm/hmm0/hmmdefs -M hmm/hmm1 monophones0

% Phase de reconnaisance %

    HVite -H hmm/hmm15/macros -H hmm/hmm15/hmmdefs -S train.scp -l '*' -i recout.mlf -w wdnet -p 0.0 -s 5.0 dict tiedlist

    HResults -I words.mlf tiedlist recout.mlf
```

Figure 2.15. Lignes de commandes de HTK.

L'utilisation de la boite à outils HTK pour mettre en œuvre une application de RAP, induit trois phases principales. D'abord, la phase de préparation de données qui vise à enregistrer, étiqueter et segmenter des données d'apprentissage en utilisant l'outil HSLab. Suivi de l'extraction des vecteurs de caractéristiques après la configuration des paramètres avec l'outil HCopy.

La phase d'apprentissage sert à créer les modèles acoustiques qui représentent les vecteurs de caractéristiques. Durant l'apprentissage, les vecteurs sont d'abord utilisés pour l'initialisation des paramètres des HMMs en utilisant l'outil HInit et HCompV, car les paramètres des HMMs doivent être correctement initialisés. Ensuite, HRest est l'outil qui permet de ré-estimer les paramètres du modèle HMM (c'est l'implémentation de l'algorithme Baum-Welch).

Le processus de reconnaissance se fait avec l'outil HVite. Il existe d'autres outils dans HTK qui sont intéressants comme l'outil de calcul du taux d'erreur et le test des performances du système. La figure ci-dessous montre l'architecture de l'outil HTK (figure 2.16).

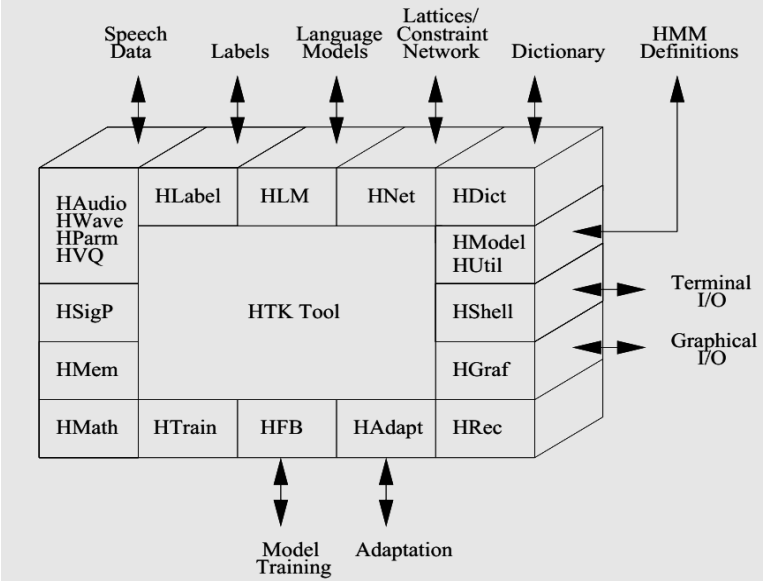


Figure 2.16. Architecture de HTK (d'après, Young et al., 2001).

HTK se distingue par l'utilisation exclusive des HMMs, il ne permet pas de faire des combinaisons ou d'hybridations des HMMs avec d'autres classifieurs. En dépit de la richesse de sa documentation, nous avons remarqué que quelques outils dans HTK ont moins d'information du point de vue pratique.

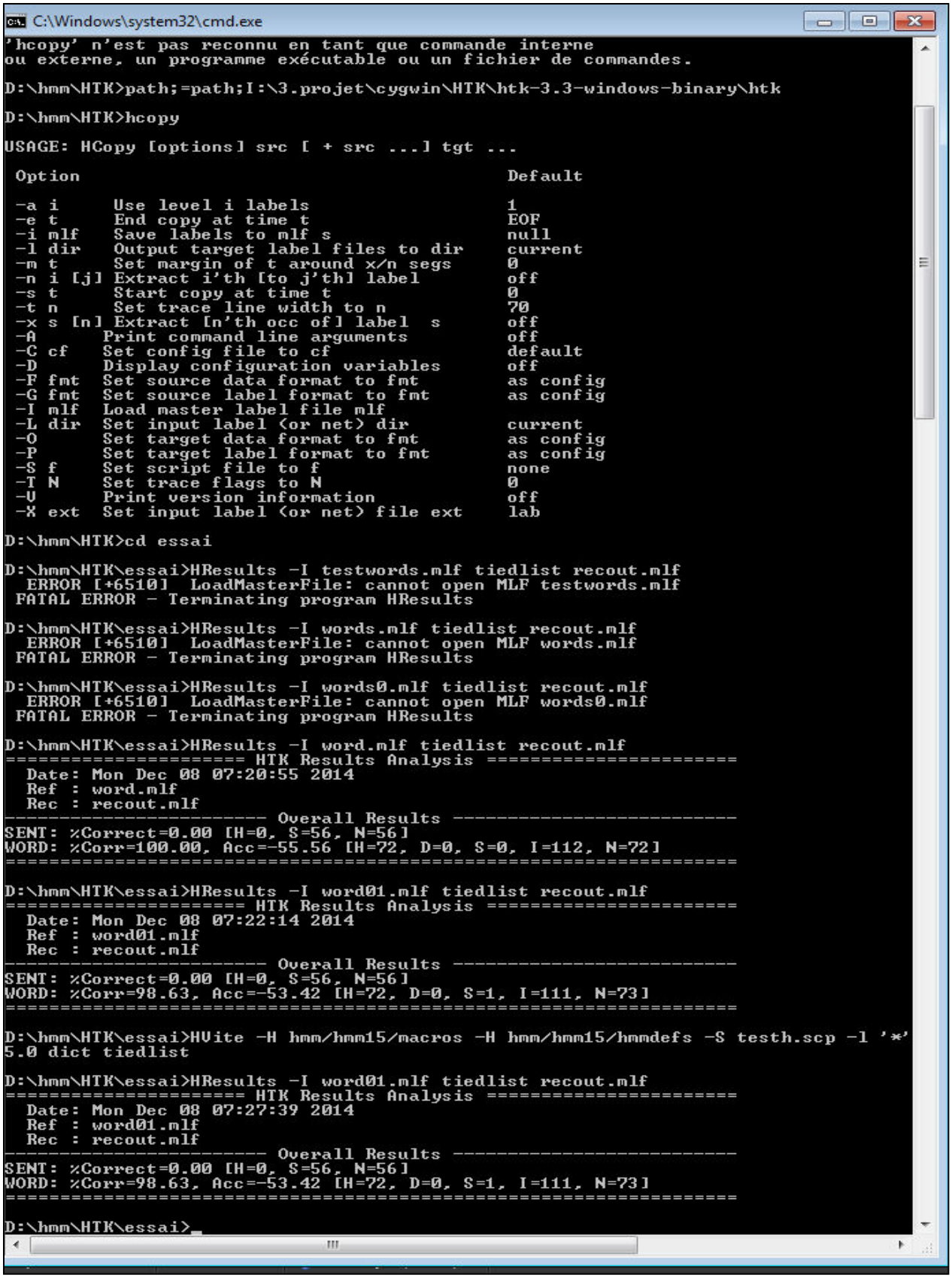


Figure 2.17. Phase de reconnaissance sous HTK

2.8.2. Sphinx

Sphinx est une librairie de reconnaissance vocale gratuitement téléchargeable, avec la possibilité de modifier le code source, il a la capacité d'implémenter des systèmes avec un large vocabulaire, indépendants du locuteur. Sphinx a quatre versions (1, 2, 3 et 4), les premières versions de Sphinx (1,2 et 3) sont écrites en langage C, mais la version récente sphinx4 est écrite en Java. Sphinx 1, 2, 3 et 4 sont des décodeurs, l'outil d'apprentissage s'appelle SphinxTrain. Sphinx4 est très souple dans sa configuration, avec une architecture modulaire qui permet aux programmeurs et chercheurs de tester de nouveaux algorithmes. La figure 2.18 représente l'architecture du Sphinx 4 qui s'articule autour de trois modules principaux (Lamere et al., 2003).

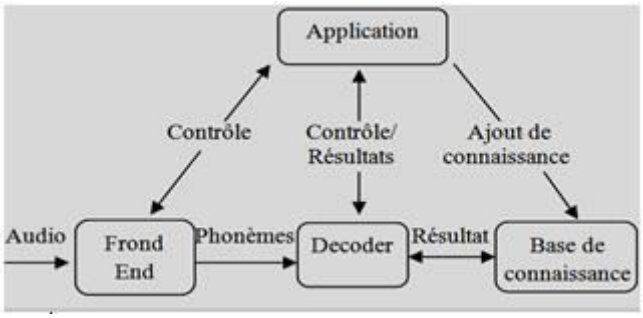


Figure 2.18. Architecture d'application de Sphinx4 (d'après, Lamere et al., 2003)

L'outil de création des modèles acoustiques SphinxTrain est écrit en langage C et l'outil de reconnaissance Sphinx 4 est écrit en Java. La documentation sur Sphinx n'est pas assez riche par rapport à HTK et il n'existe pas assez d'exemples sur l'outil de création des modèles acoustiques « SphinxTrain ».

Le tableau comparatif suivant (Table 2.1) est issue d'une expérience de l'institut HPI (Hasso Plattner Institut) de l'Université de Potsdam (Yang et al., 2011) ; il montre que les performances de Sphinx4 sont meilleures que HTK.

Table 2.1. Comparaison entre Sphinx4 et HTK (d'après, Yang et al., 2011)

2Critère	Sphinx 4	HTK
Taux de reconnaissance (60% de score total)	6.5	6
Indépendance (15%)	8	8
Coût (5%)	10	7
Modularité (15%)	10	0
Actualité (5%)	7	6
Score total	7.45	6.35

Des expériences montrent que la qualité des modèles acoustiques créés par Sphinx est meilleure que celle de HTK (Samu et Baort, 2003), si on ne prend en considération que la base TIMIT. La figure suivante montre une exécution de sphinx 4 sous Java éclipse.

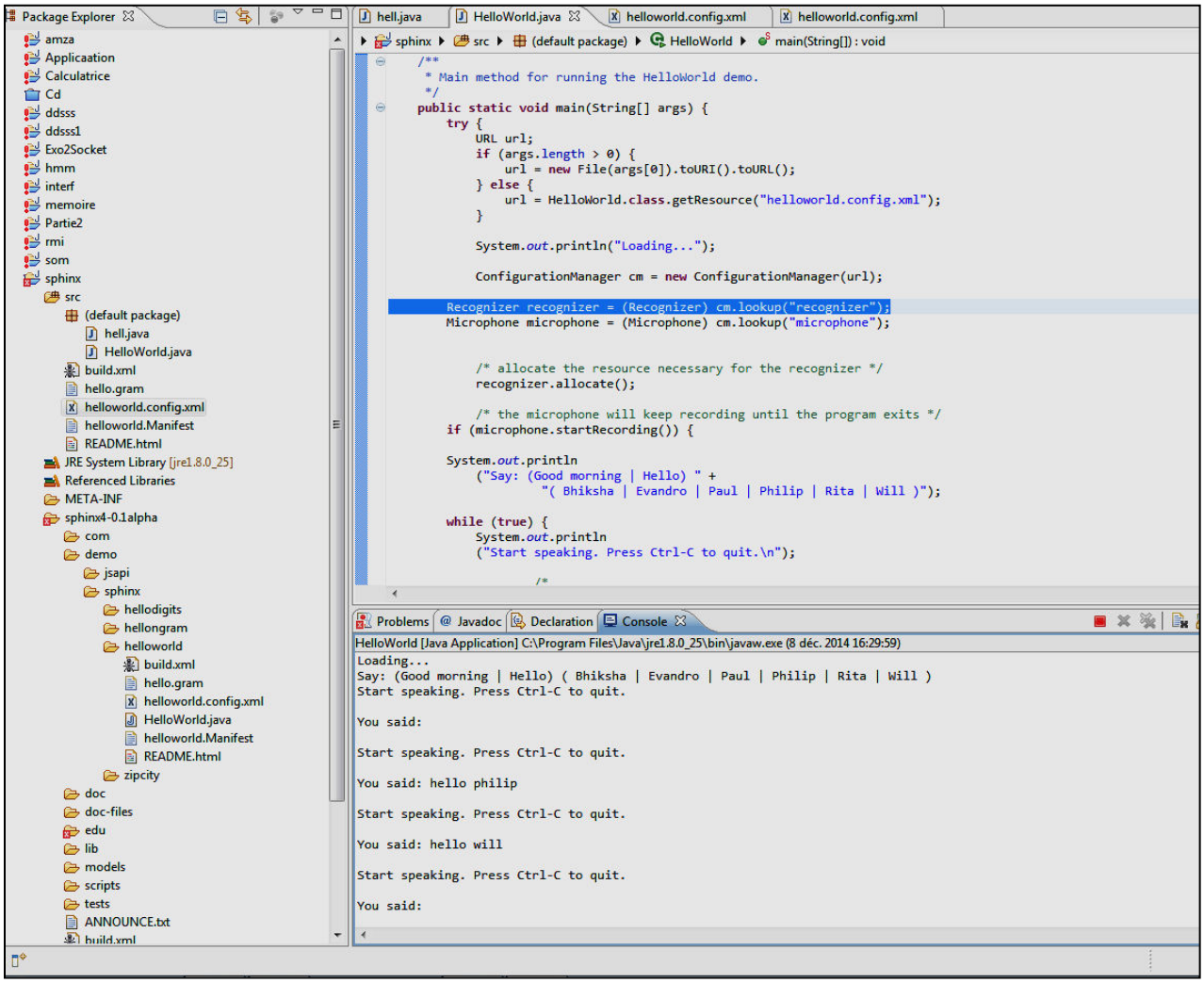


Figure 2.19. Phase de reconnaissance sous Sphinx 4

2.8.3. Matlab

Matlab possède une boite à outils incluant des algorithmes d'apprentissage artificiel basés sur les modèles de Markov cachés et des algorithmes de détection des séquences temporelles hors ligne et en ligne www.mathworks.com.

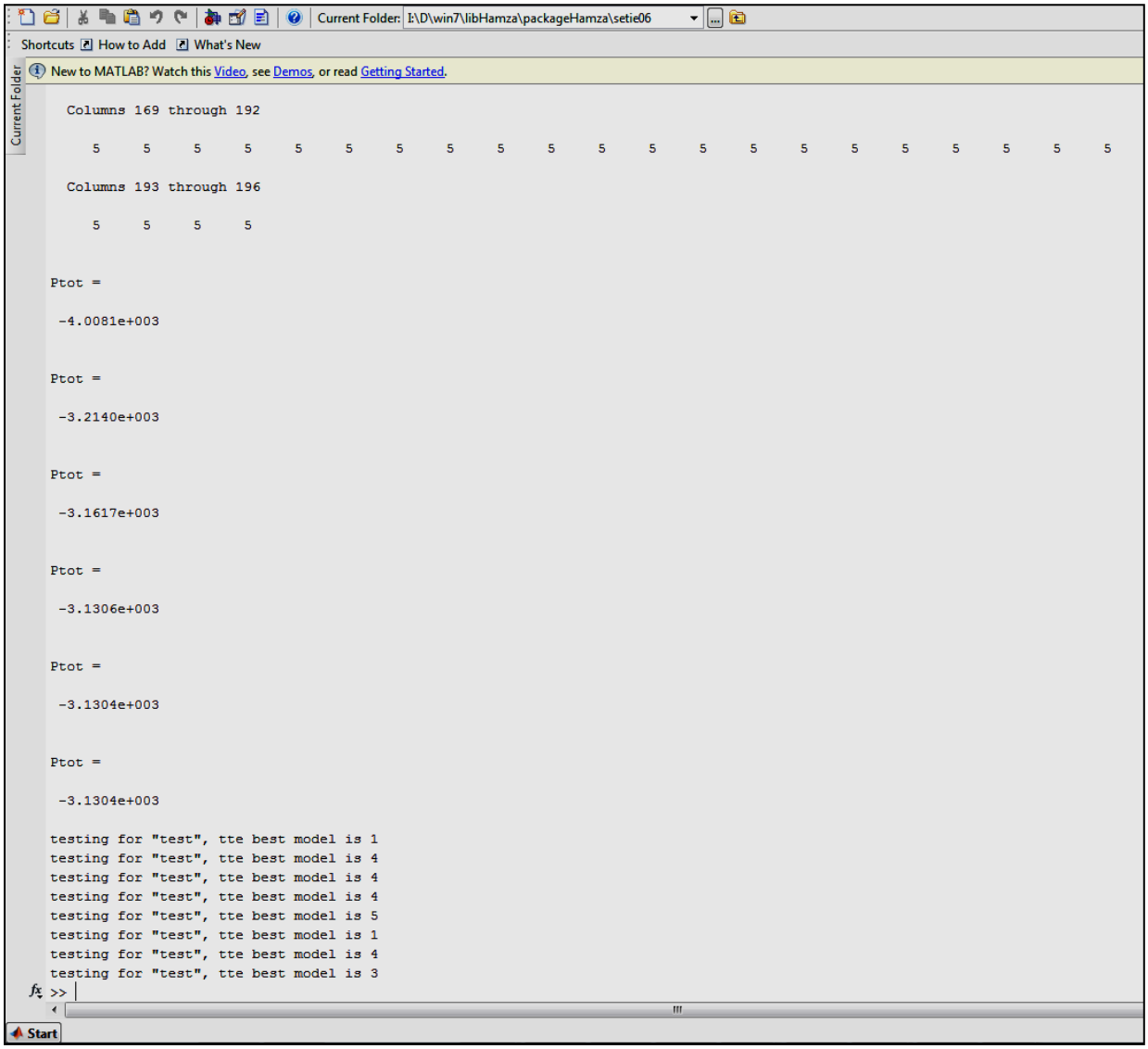


Figure 2.20. Phase de reconnaissance sous Matlab

2.8.4. La boite à outil Kaldi

Kaldi est une boîte à outils open source pour la reconnaissance de la parole écrite en C++. L'objectif de Kaldi est d'avoir un code moderne et flexible, facile à comprendre, à modifier et à étendre. Kaldi est disponible sur SourceForge (voir http://kaldi.sf.net/).

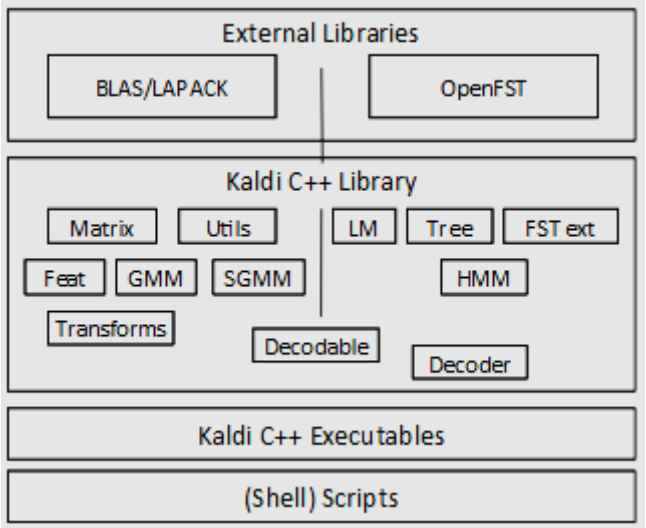


Figure 2.21. Les différentes composantes de Kaldi[1].

```
root@ubuntu:/home/safa/kaldi-trunk/egs/tidigits/s5# local/tidigits_data_prep.sh
/export/corpora5/LDC/LDC93S10/rn_comp
Tidigits directory /export/corpora5/LDC/LDC93S10/rn_comp does not have expected
format.
find: `/train': No such file or directory
Unexpected number of training files 0 versus 8623
find: `/test': No such file or directory
Unexpected number of test files 0 versus 8700
Data preparation succeeded
root@ubuntu:/home/safa/kaldi-trunk/egs/tidigits/s5# cd data
root@ubuntu:/home/safa/kaldi-trunk/egs/tidigits/s5/data# cd local/dict
bash: cd: local/dict: No such file or directory
root@ubuntu:/home/safa/kaldi-trunk/egs/tidigits/s5/data#
```

Figure 2.22. Exécution de Kaldi sous Unix

[1] http://www.it610.com/article/4689003.htm

2.9. Comparaison des outils

Nous avons effectué une étude comparative entre les librairies HTK, Sphinx et Matlab. Le tableau suivant montre les différences entre ces librairies.

Table 2.2. Tableau comparatif entre HTK, Sphinx et Matlab

Critères	HTK	Sphinx	Matlab
Structure	Un système complexe	Bien structuré (modulaire)	Assez Simple
MAJ	Mise à jour régulière	Mise à jour régulière	Mise à jour régulière
Documentation	Il est bien documenté à la fois théorique et pratique.	La documentation relativement pauvre	Documentation à la fois théorique et pratique
Langage	Langage C	Java	Langage C
Traitement du signal	Banc de filtres, MFCC, LPC, PLP, LPreflexC, ClpC, IREFC et MELSPEC.	MFCC, PLP, spectre.	MFCC, Utilise HTK pour extraire des caractéristiques (MLF)
Formats d'entrée	WAV, TIMIT, NIST, OIG, AIFI…	WAV, MP3…	WAV…
Apprentissage	HRest / HERest	SphinxTrain	Forward Backward, Vitreestim
Décodeur	HVite	Sphinx 4	Viterbi_log

Pour mener nos expériences, sous avons demandé à un groupe d'étudiants (8 étudiants) de prononcer cinq mots. Les données obtenues sont 40 échantillons au format wav.

Ces données sont utilisées pour tester les trois librairies HTK, Sphinx et Matlab. Donc, nous avons réalisé trois systèmes de RAP. Pour le premier système nous avons utilisé la librairie HTK avec un corpus contenant des fichiers sonores qui sont des mots Anglais prononcé par des étudiants Algériens. Pour le deuxième système nous avons utilisé la toolbox Matlab avec la base précédente. Le troisième système a été créé avec Sphinx4 en utilisant les

modèles acoustiques créé sur la base TIMIT (contient des mots prononcés par des personnes natives Anglais).

Pour la préparation des données nous avons extrait les coefficients MFCC et leurs dérivées premières et secondes du signal (39 coefficients). L'apprentissage se fait à l'aide de l'algorithme Baum-Welch, le nombre de gaussiennes est 8. Enfin la phase de reconnaissance est réalisée avec l'algorithme de Viterbi. Le résultat de l'étape de reconnaissance est montré ci-dessous.

Table2.3. Les taux de reconnaissance pour chaque système par rapport au corpus

	Taux de reconnaissance des systèmes créés par		
	Sphinx	HTK	Matlab
Corpus test prononcé par des Anglais	**100 %**	**20 %**	**0 %**
Corpus test prononcé par des Arabes	**40 %**	**100 %**	**80 %**

Le tableau ci-dessus montre que les performances d'un système de reconnaissance de la parole sont fortement liées à l'accent des locuteurs (la langue maternelle) en phase d'apprentissage et également à la qualité des modèles construits. Le taux de reconnaissance de Sphinx4 est imbattable dans le cas où le corpus de test comprend des mots prononcés par des Anglais, ceci peut s'expliquer par les modèles qui sont construit à partir de la base TIMIT, qui est mondialement connue. HTK est meilleur si le corpus est celui des Algériens car les modèles construit sont basés sur des prononciations Algériennes. Pour la Toolbox Matlab le taux de reconnaissance est 80% sur le corpus des étudiants Algériens, par contre pour la base des Anglais le taux de reconnaissance est 0 % parce que les modèles construites sous Matlab sur un corpus prononcé par des (L1 : première langue) Arabes et testé par le corpus des Anglais.

Nous remarquons que la précision du système de HTK est meilleure que celle de sphinx et Matlab. Ceci peut s'expliquer par une meilleure modélisation et donc un apprentissage plus précis.

2.10. Conclusion

Les recherches dans le domaine de la reconnaissance de la parole sont devenues moins difficiles grâce à la disponibilité des librairies gratuites et open source. Dans ce chapitre, nous avons vu en particulier les différences entre les librairies les plus utilisées dans le RAP. Nous avons réalisé trois applications avec HTK, Sphinx et Matlab pour voir les performances de reconnaissance pour chaque librairie. Nous pouvons dire notre préférence pour HTK vu, son ouverture et sa rapidité d'exécution.

Chapitre 3

Etat de l'art sur les méthodes de segmentation de la parole

3.1 Introduction

« L'un des exercices les plus périlleux du traitement automatique de la parole consiste à déterminer les frontières des différentes unités phonétiques contenues dans un énoncé. Cette difficulté tient à la nature même de la parole continue : les unités sont fortement co-articulées, et l'on passe souvent de l'une à l'autre de manière continue. Pourtant, on cherche à poser des frontières strictes, puisque la plupart des modèles actuels fonctionnent avec des unités discrètes et petites. » (Pellegrino, 1998)

La tâche de segmentation de la parole consiste à diviser le signal de la parole d'entrée en un ensemble d'unités de base et d'identifier le début et la fin de ces unités (les mots, les syllabes ou phonèmes, etc). L'étiquetage consiste à leur affecter des étiquettes (Labels). Les opérations de segmentation et d'étiquetage représentent l'annotation de corpus.

Dans le traitement automatique de la parole, la segmentation peut être induite à des fins diverses et variées, telles que la reconnaissance ou la synthèse de la parole, la détection de mots clés, le suivi de locuteur, etc. Selon l'application cible, on peut envisager différentes unités de segmentation (Table 3.1).

Table 3.1 : Quelques applications et leurs unités de segmentation

Niveau de segmentation	Mot	Syllabe	Phonème
Détection de mots clés	x		
Synthèse de la parole			x
Identification et vérification de locuteur	x	x	
Reconnaissance de la parole	x	x	X
Indentification de langage	x	x	

Dans cette thèse, nous nous intéressons à la segmentation automatique de la parole pour la reconnaissance de la parole. En effet, la création d'un classificateur pour la reconnaissance de formes suppose l'existence d'ensembles de données suffisamment informés pour pouvoir établir des relations entre les caractéristiques des formes et les classes auxquelles elles appartiennent à l'aide d'algorithmes d'apprentissage. Ceci est particulièrement

vrai pour la reconnaissance de la parole où beaucoup d'heures d'enregistrement sont nécessaires en phase d'apprentissage.

La construction d'un système de RAP nécessite un corpus de parole bien segmenté et étiqueté (souvent l'annotation se fait par un expert phonéticien). Ces frontières qui séparent les segments de parole sont utilisées pour définir les paramètres des modèles de Markov.

Il existe deux grandes familles des méthodes de segmentation. La première classe des méthodes de segmentation est indépendante du texte, il s'agit de segmenter le flux de parole lors de la détection d'un changement acoustique. Dans la seconde classe, la segmentation se fait en accord avec la transcription phonétique du flux de parole. Cette deuxième famille d'algorithmes de segmentation est plus liée à l'apparition des modèles de Markov cachés.

Dans ce chapitre, nous allons d'abord présenter en détail les deux grandes approches qui existent pour la segmentation automatique d'un flux de parole. Cette présentation va inclure un état de l'art des travaux dans les deux approches. Nous présenterons ensuite, les mesures qui sont utilisées pour l'évaluation des performances d'un système de segmentation. Enfin, on reviendra sur les travaux qui traitent de la segmentation de la parole dans le cadre de la reconnaissance de la parole, en particulier ceux dédiés à la parole Arabe.

3.2 Les approches de segmentation de la parole

La segmentation consiste en le découpage d'un signal continu en un ensemble d'unités selon un critère donné. Il existe deux grandes familles des méthodes de segmentation : une approche implicite et une approche explicite. La première classe des méthodes de segmentation est indépendante du texte, il s'agit de segmenter le flux de parole lors de la détection d'un changement acoustique. Dans la seconde classe, la segmentation se fait en accord avec la transcription phonétique du flux de parole.

Les algorithmes issus de l'approche implicite, également appelée approche acoustique, segmentent le signal de parole sans aucune connaissance préalable des informations linguistique (propriétés internes). Cette segmentation est indépendante du locuteur, du texte et de la langue. Ces algorithmes de segmentation se basent sur le traitement de signal en exploitant les caractéristiques dans le domaine temporel (telles que : l'énergie à court terme, le

taux de passage par zéro, etc) et celles du domaine fréquentiel (telles que : le centroïde spectral et le flux spectral). Dans cette approche des méthodes de seuillage sont utilisées pour détecter le début et la fin de chaque unité.

Dans la seconde approche, les algorithmes de segmentation utilisent une certaine connaissance préalable des modèles ou du langage cible (propriétés externes). Ce type de segmentation repose sur l'analyse des caractéristiques externes du signal. Ces méthodes de segmentation se basent sur l'intelligence artificielle: HMM, Méthodes floues, Réseaux de neurones, etc. La segmentation porte sur plusieurs unités : mot, phonème, allophone, diphone, N-gram, syllabe,etc.

Table 3.2. Caractéristiques de la segmentation implicite vs explicite (van Hemert, 1991)

La segmentation implicite	La segmentation explicite
La méthode ne produit pas toujours le nombre adéquat de segments.	La méthode produit le nombre de segments donné par la transcription phonétique.
Les segments ne sont pas étiquetés.	Les segments sont étiquetés en fonction de la transcription phonétique.
Les frontières des segments sont déterminées avec précision pour la segmentation des diphones.	Les frontières des segments peuvent ne pas être précises vue la possible de faible ressemblance entre le spectre de référence et celui du test.

3.2.1 L'approche acoustique

Dans les algorithmes issus de la mouvance acoustique en segmentation du signal, la détection des frontières des segments est basée sur le traitement de signal. On y utilise aussi bien les caractéristiques du signal issues du domaine temporel que celles issues du domaine fréquentiel. Dans les deux domaines, des méthodes de seuillage sont utilisées pour détecter le début et la fin de chaque segment. On va dans ce qui suit présenter les plus connues de ces caractéristiques ainsi que quelques de travaux issus de cette approche.

3.2.1.1 L'énergie à court terme (Short Term Energy STE)

L'énergie associée au signal de la parole varie au cours du temps. Elle fournit une représentation des variations d'amplitude du signal (Jayasankar, 2011). Ainsi, tout traitement de la parole va s'intéresser à la manière dont cette énergie varie. De la nature même de la parole, le signal de la parole consiste en des zones voisées, non voisées ou du silence. Les zones voisées ont une énergie bien plus grande que celles des zones non voisées, tandis que le silence a une énergie presque nulle. Ainsi, l'énergie à court terme (STE) est utilisée pour la classification du signal de la parole en zone voisée, non voisée et en silence.

L'énergie à court terme du signal de la parole est calculée dans le domaine du temps où le signal est fenêtré et en calculant la moyenne sur les échantillons élevé au carré. Cette énergie (STE) est donnée par:

$$E_n = \frac{1}{N}\sum_{m=1}^{N}[x(m)w(n-m)]^2 \qquad (3.1)$$

Avec x(m) l'amplitude de l'échantillon m du signal.
N, la longueur de la fenêtre temporelle

La figure suivante montre l'énergie à court terme du mot [musafir] en fonction du nombre de fenêtres.

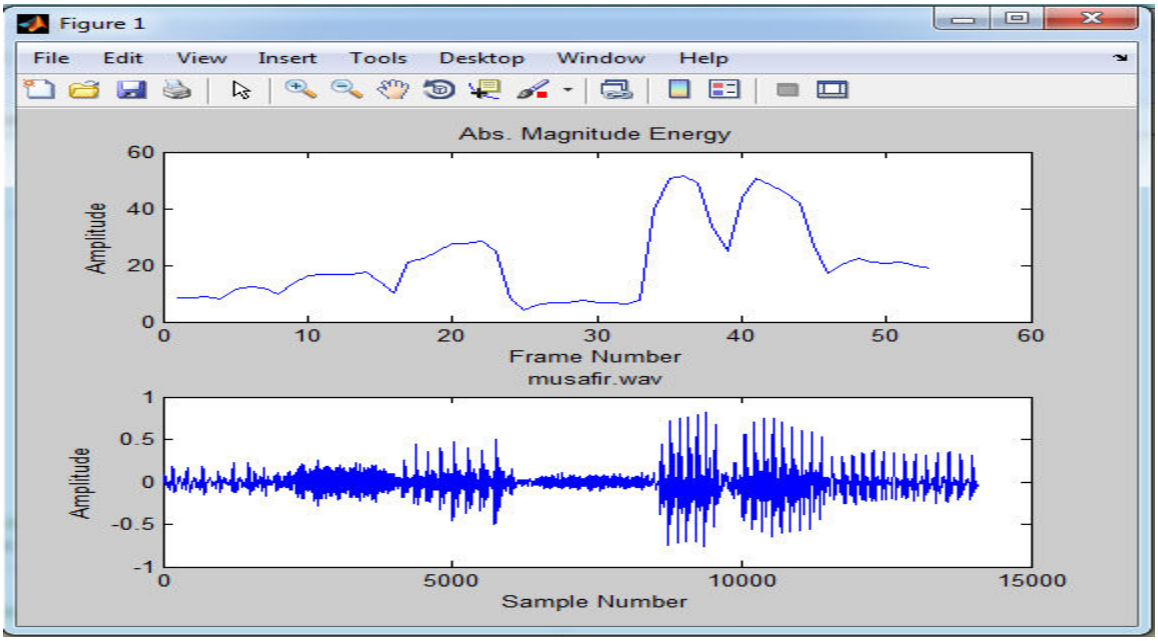


Figure 3.1. L'énergie à court terme en fonction du nombre de fenêtres du mot [musafir]

Le paramètre d'énergie a été utilisé dans la détection de points frontières depuis les premières années. Parmi les travaux qui ont utilisé la STE pour la segmentation, on cite : (Kaur, 2010a) qui utilise la STE pour segmenter le parole en syllabes ainsi qu'un seuil pour détecter le début et la fin des syllabes.

3.2.1.2 Le taux de passage par zéro (Zero Crossing Rate, ZCR)

Le taux de passage par zéro donne des informations sur le nombre de passages par zéro dans un signal de parole. Si le nombre de passages par zéro est grand dans un signal, le signal est en train de changer rapidement et en conséquence le signal peut contenir des informations de haute fréquence. Contrairement, si le nombre de passage par zéro est petit alors cela signifie que le signal est en train de changer lentement et en conséquence le signal peut contenir des informations de basse fréquence (Kalamani, 2014). Le taux de passage par zéro d'un échantillon de signal de parole numérique est défini par:

$$Z_n = \frac{1}{2}\sum_{m=1}^{N} |sgn[x(m)] - sgn[x(m-1)]| w(n-m) \qquad (3.2)$$

Avec

$$sgn[x(m)] = \begin{cases} 1 & x(m) > 0 \\ -1 & x(m) < 0 \end{cases} \qquad (3.3)$$

Le taux moyen de passage par zéro d'un signal de parole peut être utilisé en conjonction avec l'énergie à court terme (ou amplitude) pour une discrimination entre la parole voisée, non voisée et le silence.

En effet dans (Rabiner et Sambur, 1975), les auteurs proposent un algorithme de segmentation acoustique du signal continu de la parole en des zones délimitées par le silence qui se base sur deux mesures extraites du signal, il s'agit de l'énergie à court terme et du taux de passage par zéro.

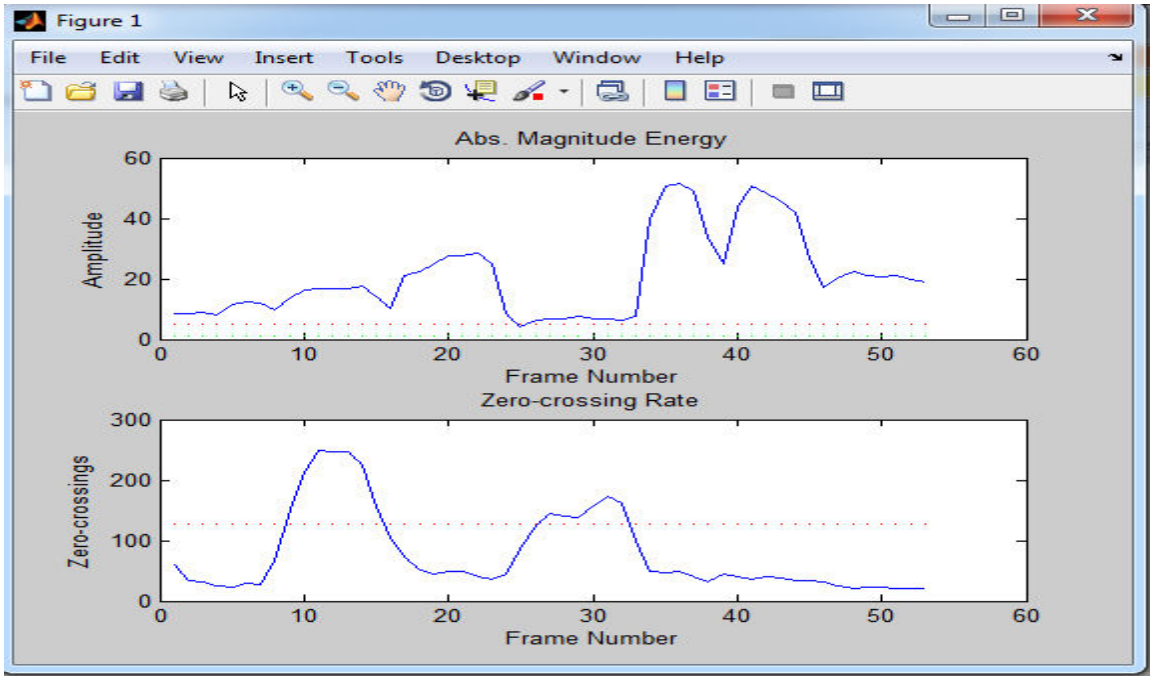


Figure 3.2. L'énergie et le taux de passage par zéro pour une occurrence du mot [riH]

L'opération de segmentation se fait en deux étapes, d'abord ils effectuent la segmentation en utilisant l'énergie pour obtenir une segmentation préliminaire ensuite il s'agit de la raffiner en utilisant le taux de passage par zéro. La segmentation effective se fait sur la base d'un seuil.

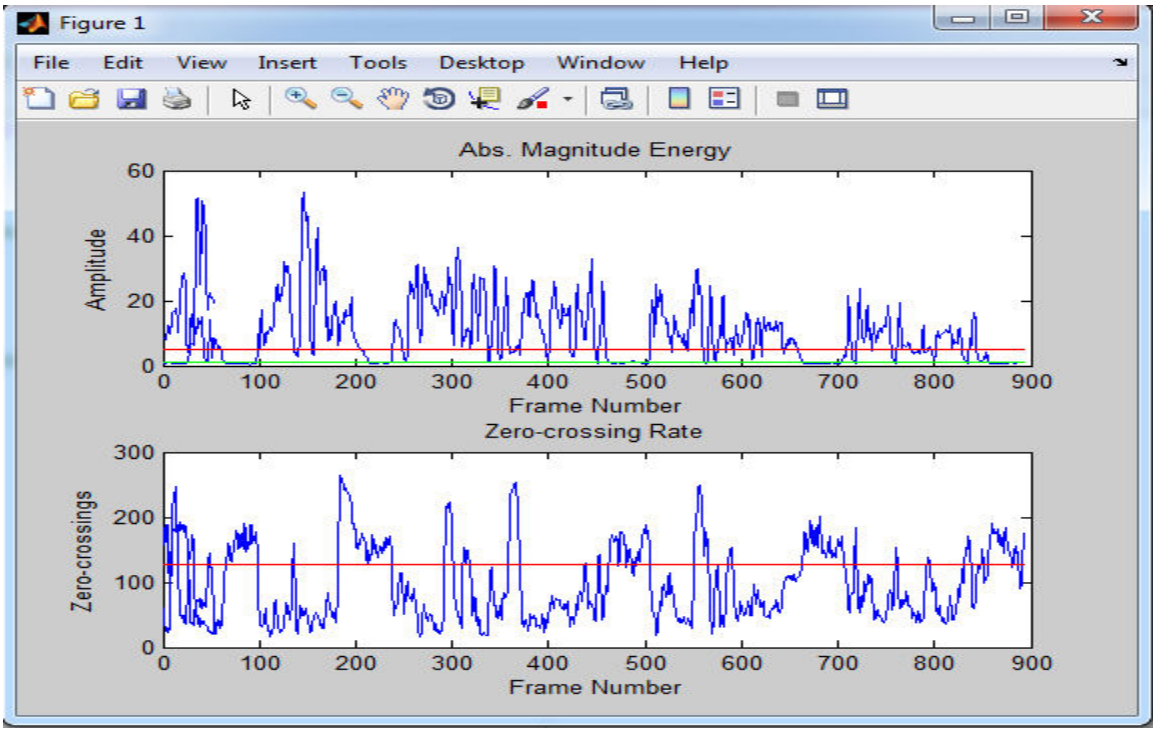


Figure 3.3. Le seuil de segmentation et le seuil minimal

En 2012, (Sangeetha et Jothilakshmi, 2012) ont également combiné la STE avec le ZCR pour que les processus de détection puissent être très précis. D'abord, L'énergie à court terme et le taux passage à zéro sont calculés. Ensuite, un seuil dynamique est généré. Les signaux ayant une valeur inférieure à ce seuil seront modifiés à 0, ainsi le signal de la syllabe aura une valeur de données supérieur au seuil. Puis, le signal qui a été vérifié pour la valeur non nulle et supérieure à un seuil et ce point sera marqué comme emplacement de départ de la frontière. Après avoir obtenu l'emplacement de départ, les valeurs nulles du signal sont vérifiées et s'il y a un nombre adéquat de zéros continus alors il sera défini comme la fin de la limite. Une fois un point final a été détecté, ils procèdent à la recherche de la position de départ de la prochaine extrémité.

3.2.1.3 Le centroide spectral

Le centroide spectral renseigne sur la position du centre de gravité du spectre d'un signal. Cette caractéristique est une mesure de la position spectrale avec des valeurs élevées, elle indique des sons brouillants. Le centroide est calculée comme la moyenne pondérée des fréquences présentes dans le signal. Ces dernières sont calculées en utilisant la transformée de Fourier, avec leur magnitudes comme les poids. Le centroide spectral est donné par:

$$SC_i = \frac{\sum_{m=0}^{N-1} f(m) X_i(m)}{\sum_{m=0}^{N-1} X_i(m)} \quad (3.4)$$

Avec :

f(m)- Frequence Centrale

Xi (m)-Amplitude du signal

3.2.1.4 Revue de littérature

(Rahman et al., 2012) présentent des méthodes simples d'extraction de caractéristiques pour segmenter la parole continue, ils se basent sur la combinaison des domaines temporel et fréquentiel, ensuite, un seuil dynamique est défini pour détecter les frontières. Les auteurs ont utilisé la STE et le controide spectral pour la segmentation, ils calculent ces deux paramètres pour chaque fenêtre de signal. Ensuite ils calculent l'histogramme pour trouver le maximum local pour calculer le seuil. Ils ont utilisé le k-means, Fuzzy k-means et l'algorithme Otsu pour définir les seuils.

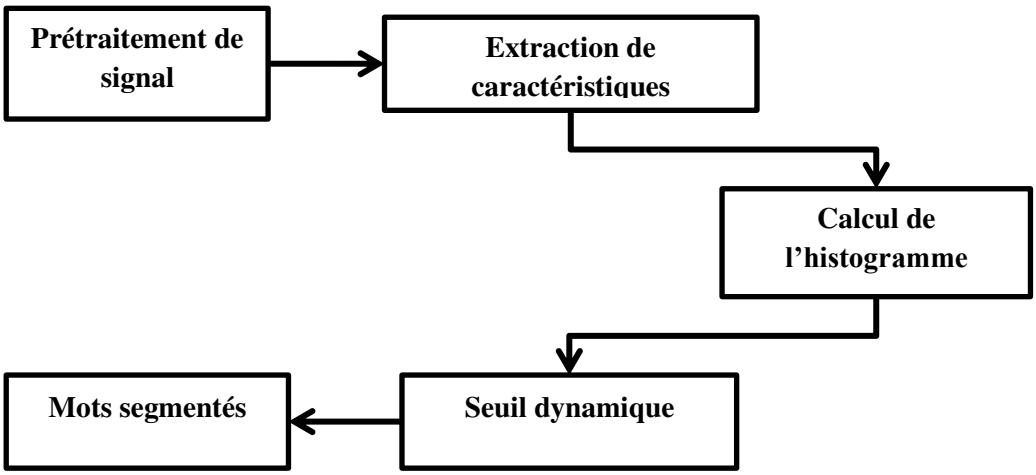


Figure 3.4. Diagramme en bloc des différentes étapes (Rahman et al., 2012)

L'article de (Pekar et Tsikhanenka, 2010) propose un algorithme de segmentation basé sur l'analyse de la densité spectrale de puissance normalisée PSD qui détermine la structure phonétique de la parole prononcée sans aucune information a priori sur le signal. Cet algorithme est indépendant du locuteur et sa complexité de calcul est rendue faible. Le principe de l'algorithme est de calculer la distance entre la PSD après normalisation dans les

segments adjacents de courte durée et de vérifier à base de la vitesse des changements de valeurs d'analyse de l'énergie du signal de courte durée. Ensuite, donner un seuil à la distance entre PSD et de supprimer les petites valeurs des caractéristiques.

(Prasad et al., 2004) et (Nagarajan et al., 2003) ont proposé une approche basée sur le traitement de « group delay function » du spectre d'amplitude pour déterminer les limites des segments dans le signal de parole. La fonction Groupe Delay est la dérivée négative de la transformée de Fourier phase et est définie comme le retard de groupe. L'approche segmente le signal de parole en syllabes qui se base sur le traitement de la fonction d'énergie à court terme du signal de parole. Cette approche utilise uniquement les informations sur le nombre approximatif de segments voisées présents dans le signal de parole (Figure 3.5).

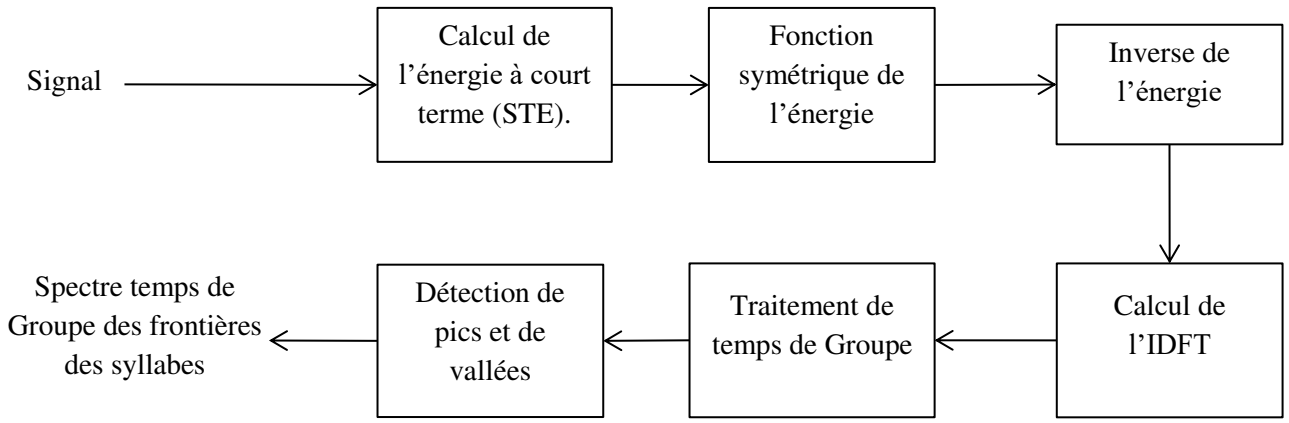


Figure 3.5. Différentes étapes de segmentation en syllabes d'après (Prasad et al., 2004)

L'algorithme de segmentation de (Lakshmi et Murthy, 2006) utilise un signal minimum de phase dérivée de la fonction d'énergie à court terme (STE) comme si elle était un spectre d'amplitude. Les régions de haute énergie dans la fonction de STE correspondent à des noyaux de syllabes tandis que les vallées aux deux extrémités des noyaux déterminent les frontières de la syllabe. Un algorithme de segmentation à deux niveaux sur la base du groupe delay est proposé afin d'extraire les unités de syllabes précises à partir des données de parole.

Les auteurs (Salam et al., 2010) ont proposé l'utilisation du Zero Crossing Propety avec l'algorithme de divergence (Abrecht, 1988) pour réduire les points d'insertion. L'algorithme de Divergence est une méthode statistique qui utilise l'approche implicite avec

fenêtres non-fixe sans connaissance préalable du signal acoustique. Les paramètres de propriétés ZCR sont calculés après le processus d'algorithme de divergence. Il prend les points de segmentation par la méthode statistique. Ensuite, il définit le point de pivot T (seuil) et à partir de ce seuil il obtient deux points temporaires T1 et T2 avec la distance W. Puis, il calcule la ZCR de T1(W1) à T2(W2). Finalement, il calcul une mesure A et compare Si A> Seuil, le point est gardé sinon il est supprimé. Il continue le processus jusqu'à ce que tous les points soient vérifiés. La figure (3.6) décrit de manière synthétique l'algorithme.

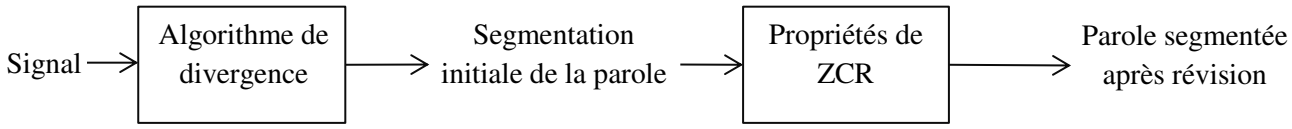


Figure 3.6. Etapes de l'algorithme de segmentation de (Salam et al., 2010)

L'algorithme proposé dans (Malcangi, 2009) utilise trois caractéristiques: l'énergie, le taux de passage par zéro et le pitch rate pour chaque fenêtre. Il combine un moteur d'inférence basée sur la logique floue pour séparer le signal de parole en phonèmes. L'énergie à court terme est combinée avec le taux de passage par zéro pour classer les segments voisés et non voisés. Sur les segments de la parole obtenus on calcule le Pitch de fréquence. L'énergie, le taux de passage à zéro et la hauteur sont combinés pour inférer la classe de chaque segment. A la fin, une couche logique de décision évalue les points d'extrémité des segments. La figure (3.7) décrit les différentes étapes induites par cet algorithme.

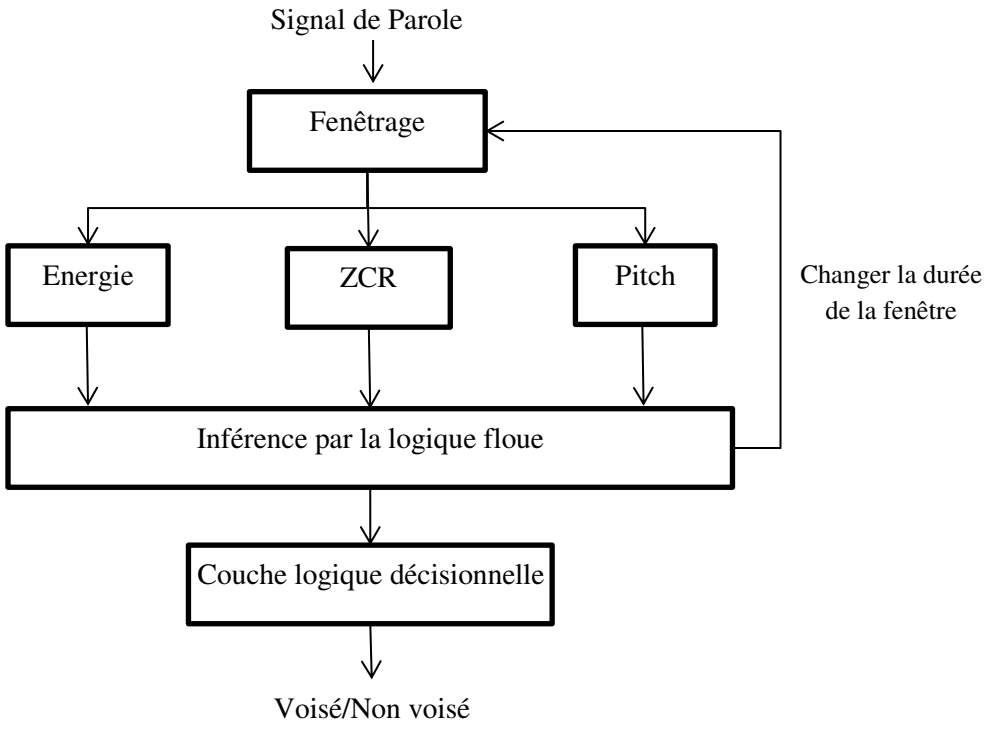


Figure 3.7. Le processus de segmentation de la parole en segments voisés/non voisés (Malcangi, 2009)

L'algorithme proposé dans (Sarkar et Sreenivas, 2005) utilise le LCR (Level crossing rate) à un point de l'échantillon. Ensuite, il calcule le taux de passage de niveau moyen (ALCR) à chaque point par sommation de la LCR sur tous les niveaux. Il utilise l'information temporelle (LCR) pour la segmentation car lorsqu'il y a un changement d'un phonème à l'autre, il y a un changement dans la configuration de l'appareil vocal.

Enfin en guise de revue de l'approche acoustique, dans (Kalamani et al., 2014), on trouve un algorithme de segmentation qui effectue une comparaison entre les caractéristiques temporelles et fréquentielles en terme de segmentation de mots. Les caractéristiques du domaine temporel sont la STE et le ZCR tandis que dans le domaine de fréquence, ce sont le centroïde spectral et le flux spectral. La méthode de segmentation est simple et basé sur un seuil dynamique.

Table 3.3. Revue de quelques algorithmes de segmentation acoustique

Ref.	Unités	Caractéristiques	Langage cible	Application finale
Amanpeet et Tarandeep, 2010	Syllabes	STE	Punjabi	Reconnaissance de la parole
Anwar et al., 2006	Consonne/Voyelle	ZCR, PSD	Arabe	
Awais et al., 2006	Consonne/Voyelle	FFT/ Spectrogramme	Arabe	Reconnaissance de la parole
Lakshmi et Murthy, 2006	Syllabes	STE	Tamil	Reconnaissance de la parole
Kaur et Singh, 2013	Syllabes	STE	Punjabi	Préparation de bases de données parole Punjabi
Nagarajan et al., 2003	Syllabes	Groupe delay	OGI_MLTS corpus	Reconnaissance de la parole
Prasad et al., 2004	Syllabes	Groupe Delay	Anglais (TIMIT, TIDIGITS corpora)	
Rahman et Bhuiya, 2012	Mots / sous mots	STE,ZCR, Centroide spectral et le flux spectral	Bangalais	Reconnaissance de la parole
Sangeetha et Jothilakshmi, 2012	Mots / groupe de mots	STE, ZCR	Indien (Tamil, Telugu, Hindi et Malayalam)	-Reconnaissance de la parole -Traduction parole/parole
Shah et al., 2014	Phonèmes	PLP	Gujarati (une des langues officielles de l'Inde)	Synthèse de la parole
Tolba et al., 2005	Consonne/Voyelle	Ondelettes	Arabe	Reconnaissance de la parole

3.2.2 L'approche phonétique

Etant donnée un signal de parole et une transcription phonétique de ce signal, il s'agit de retrouver une association entre cette description phonétique et la séquence des fenêtres

acoustiques de ce signal. Cette transcription est constituée d'une séquence de symboles linguistiques, typiquement des phonèmes.

L'objectif des algorithmes issus de l'approche phonétique consiste à obtenir une séquence de segments acoustiques contigus et définis par des instants temporels de début et de fin. Le nombre de segments acoustiques délimités doit être égal au nombre d'étiquettes présentes dans la transcription phonétique (voir table 3.2). Dans ce cas, chaque segment acoustique délimité est muni d'une étiquette issue de cette transcription.

La méthode la plus utilisée pour la segmentation phonétique consiste à effectuer un alignement forcé entre le flux de parole à segmenter et la transcription orthographique ou phonétique en utilisant les HMM (Toledano et Gomez, 2002). En 2002, Dines et al. ont écrit: "Différentes méthodes ont été employées pour la segmentation phonétique de la parole, les techniques les plus répandues reposant sur le modèle de Markov caché"[1] et "Le HMM constitue actuellement l'épine dorsale de la plupart des systèmes modernes de reconnaissance de la parole et, intuitivement, il semble également bien adapté à la tâche de segmentation de la parole"[2]. En 2016, dans (Brognaux et Drugman, 2016), les auteurs ont écrit: "Plusieurs techniques ont été proposées pour fournir automatiquement la segmentation des fichiers vocaux. Parmi ceux-ci, nous pouvons essentiellement distinguer entre les méthodes basées sur les algorithmes Dynamic Time Warping (DTW) et celles utilisant les HMM .... Si les algorithmes DTW ont donné des résultats acceptables, la modélisation acoustique basée sur les HMM a été soulignée comme étant la technique la plus fiable pour l'alignement phonétique automatique. C'est actuellement la technique la plus utilisée pour l'alignement forcé "[3].

Ainsi, à l'heure actuelle, la tendance explicite continue de croître grâce aux outils disponibles pour l'apprentissage des HMM et le concept de l'apprentissage embarqué, dans

[1] Traduction de : "Various methods have been employed for the phonetic segmentation of speech, the most prevalent being techniques that rely on the hidden Markov model"

[2] Traduction de :" The HMM currently forms the backbone of most modern speech recognition systems and, intuitively, it also seems well suited to the task of speech segmentation"

[3] Traduction de : "Several techniques have been proposed to automatically provide the segmentation of speech files. Among these, we can essentially distinguish between methods based on Dynamic Time Warping (DTW) algorithms and those using HMMs….While DTW algorithms were shown to provide acceptable results, HMM-based acoustic modeling has been pointed out as being the most reliable technique for automatic phonetic alignment. It is currently the most widely-used technique for forced alignment"

lequel la segmentation et la reconnaissance interagissent. Cet algorithme est devenu l'état de l'art dans la discipline. Nous présentons dans ce qui suit son principe.

3.2.2.1 L'algorithme de l'apprentissage embarqué

Dans les approches d'apprentissage classiques, les HMM sont construits avec des frontières fixes selon la segmentation manuelle, tandis qu'avec l'algorithme de l'apprentissage embarqué (EL pour Embedded Learning) les limites sont alignées itérativement par alignement-forcé (Young et al., 2002). L'idée est que le re-étiquetage est fait dans chaque itération. Par conséquent, le classifieur dans la prochaine itération aura des ensembles plus cohérents de données étiquetées (Figure 3.8).

L'apprentissage embarqué est le nom donné au processus d'apprentissage des modèles acoustique de manière itérative, puis en utilisant les modèles pour faire un nouveau étiquetage de l'ensemble d'apprentissage via l'alignement forcé, puis l'apprentissage d'une nouvelle reconnaissance de ces étiquettes, puis ré-aligner à nouveau jusqu'à ce que le système résultant converge.

L'apprentissage embarqué utilise régulièrement le corpus de parole comme source des données d'apprentissage et simultanément pour réestimer l'ensemble des modèles HMM. Pour chaque signal en entrée, il a besoin d'une transcription (liste des phonèmes associée à chaque signal). Cet apprentissage revoit ensuite l'ensemble de tous les HMM de phonèmes correspondant à cette liste de phonèmes pour faire un seul HMM composite. Cet HMM composite est utilisé pour acquérir les statistiques nécessaires à la ré-estimation. Lorsque tous les signaux d'apprentissage ont été traités, l'ensemble total de statistiques accumulées sont utilisées pour ré-estimer les paramètres de tous les HMM de phonèmes. Les transcriptions sont nécessaires uniquement pour identifier la séquence de phonèmes dans chaque signal. Aucune information de frontières de phonèmes n'est nécessaire.

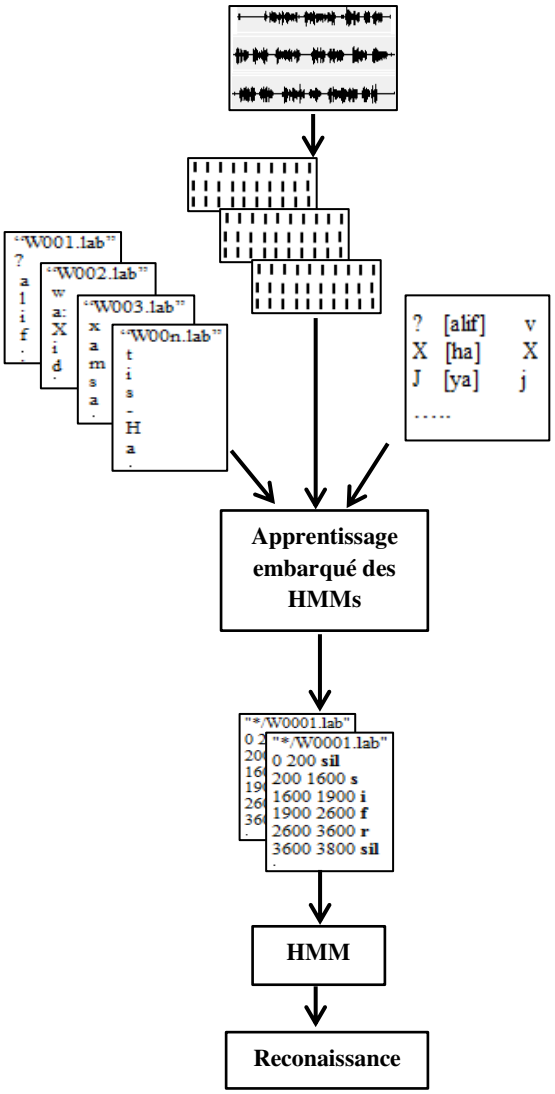


Figure 3.8. Schéma synthétique de l'apprentissage embarqué (Frihia et Bahi., 2015)

L'initialisation apriori de l'ensemble des HMMs de phonèmes par la ré-estimation embarquée peut être réalisée de deux manières. La première consiste à utiliser un sous-ensemble de données segmentées manuellement pour initialiser chaque HMM de phonème individuellement. Lorsqu'il est utilisé de cette manière, l'algorithme utilise l'étiquette pour extraire tous les segments de parole correspondant à chaque HMM de phonèmes pour réaliser l'apprentissage de modèles. La seconde utilise la technique « Flat Start» qui consiste au départ à segmenter chaque signal d'apprentissage uniformément. Etant donné un signal de parole de l'ensemble d'apprentissage, ce dernier sera segmenté uniformément selon le nombre de phonèmes dans la transcription phonétique correspondante. Ainsi, si la transcription comprend N phonèmes, le signal sera segmenté en N segments de même taille. Chaque

segment sera initialement étiqueté comme le phonème correspondant dans la séquence de la transcription.

Ensuite, les modèles de phonèmes seront alignés avec les réalisations actuelles de ce phonème de sorte que sur les itérations suivantes, les modèles s'aligneront comme prévu. Le concept de l'apprentissage embarqué met à jour simultanément l'ensemble des modèles HMM à un système utilisant l'ensemble de données d'apprentissage. Pour chaque fichier d'apprentissage, il utilise la transcription associé à la construction d'un HMM composite qui couvre le signal d'entrée. Cet HMM composite est constitué par la concaténation des instances des modèles HMM de phonème correspondant à chaque étiquette dans la transcription. L'algorithme Forward-Backward est ensuite appliqué et les sommes nécessaire pour former les moyennes pondérées accumulées de façon normale. Lorsque tous les fichiers d'apprentissage ont été traités, de nouveaux paramètres estimés sont formés à partir de ces sommes pondérées.

L'apprentissage embarqué utilise la même procédure Baum-Welch comme pour le cas isolé, mais au lieu que l'apprentissage se fasse sur chaque modèle individuellement, tous les modèles sont appris en parallèle. L'algorithme opère selon les étapes suivantes[4]:

1- Attribuer zéro accumulateur pour tous les paramètres de tous les HMM.
2- Obtenir le prochain signal d'apprentissage.
3- Construire un HMM composite en rejoignant en séquence les HMM correspondant aux symboles de transcription du signal d'apprentissage.
4- Calculez les Forward - Backward probabilités pour le HMM composite. L'intégration d'états intermédiaires non émetteurs dans le modèle composite nécessite quelques changements au calcul des probabilités avant et arrière, mais ceux-ci sont seulement mineurs.
5- Utilisez les probabilités avant et arrière pour calculer les probabilités d'occupation de l'état à chaque trame de temps et mettre à jour les accumulateurs de la manière habituelle.
6- Répétez de 2 jusqu'à ce que tous les signaux d'apprentissage aient été traités.
7- Utilisez les accumulateurs pour calculer de nouvelles estimations des paramètres pour tous les HMM.

[4] http://www.eecs.yorku.ca/course_archive/2007-08/W/6328/Reading/htkbook31_part1.pdf

Ces étapes peuvent être répétées alors autant de fois qu'il est nécessaire pour réaliser la convergence requise. Bien que l'emplacement des limites des unités phonétiques dans les données d'apprentissage ne soit pas nécessaire pour cette procédure, la transcription phonétique de chaque énoncé d'apprentissage est nécessaire.

3.2.2.2 Revue de littérature

Les algorithmes issus de l'approche phonétique sont basées sur l'intelligence artificielle et utilisent habituellement les modèles de Markov cachés (HMM) avec un alignement forcé (Brognaux et Drugman 2016; Shanmugam et Murthy2014; Toledano et Gómez 2002; Brugnara et al. 1993).

Une méthode automatique en deux étapes de production d'un diphone est proposée par (Taylor, 1990). Il a utilisé les modèles de Markov cachés pour localiser les frontières de phonèmes, puis un algorithme de minimisation de discontinuité spectrale permet de choisir les limites de diphones.

(Ting et al., 2007) propose une approche de segmentation basée sur l'algorithme de Viterbi qui utilise la technique de l'alignement-forcé. Il utilise des HMM à densité continue (CDHMM) avec un mélange gaussien. Il présente une méthode de raffinement limite implicite qui est incorporé dans l'alignement phonétique de Viterbi. Dans cette approche, toutes les données ont été introduites dans le système pour effectuer l'initialisation. L'initialisation sert à répartir équitablement le nombre de trames pour tous les phonèmes. Le modèle HMM est formé avec des jetons de phonèmes avec leurs limites étendues aux phonèmes. Cela augmente la capacité des HMM dans les limites de modélisation des phonèmes et fournit un effet de raffinement lorsqu'il est utilisé dans l'alignement phonétique pour réduire les erreurs de segmentation.

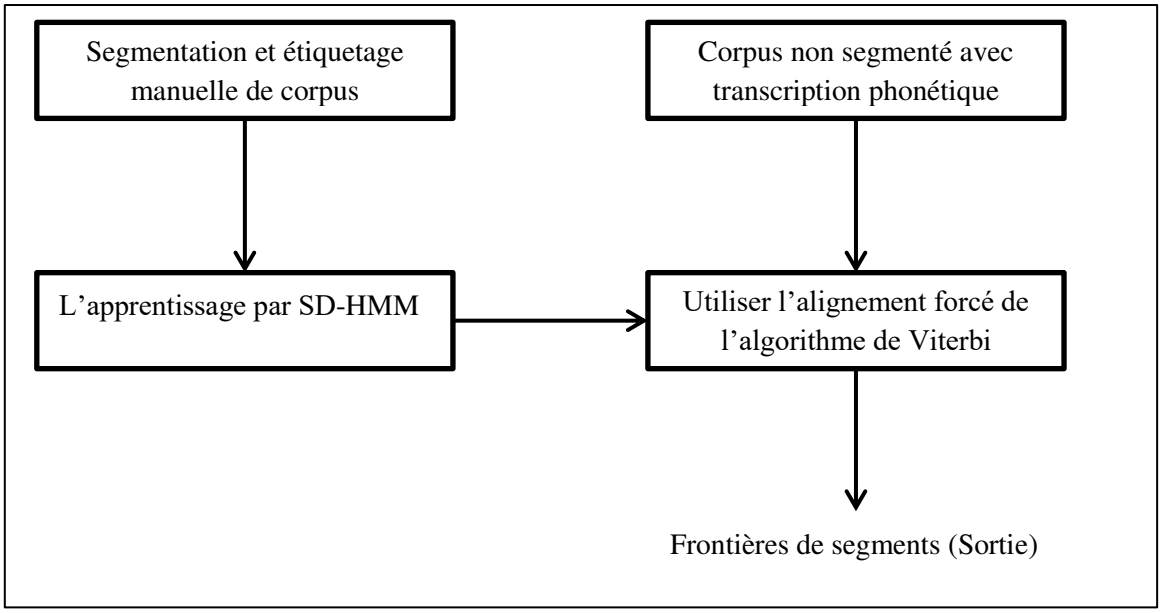


Figure 3.9. Architecture du système de segmentation de (Ting et al., 2007)

Globalement, les algorithmes issus de cette approche exigent l'existence d'un cas isolé des unités désirées; heureusement, l'apprentissage embarqué (EL) permet le développement des unités de parole annotées sans nécessité d'avoir leurs formes unitaires isolées. Cependant, quelques problèmes surviennent lorsque l'apprentissage embarqué est utilisé; ces problèmes sont associés à l'utilisation de l'initialisation en « flat start » (Mporas et al. 2008).

3.3 Les mesures d'évaluation

Pour évaluer les performances d'un système de segmentation, un ensemble de mesures peuvent être calculées. Ces mesures sont issues de deux classes : Les mesures orientées objectif et les mesures de l'objectif (Gałka et Ziółko, 2007).

3.3.1. Les mesures orientées objectif

La segmentation de la parole fait généralement partie d'un système complexe de reconnaissance, d'annotation, de synthèse, de compression ou de traitement de la parole. Dans ce cas, le résultat de l'activité du système est mesuré avec des critères habituellement bien définis dans le domaine, ces mesures peuvent aussi permettre de mesurer les performances du système de segmentation. Ces mesures peuvent être :

- Le taux de reconnaissance par mot

- Le taux d'erreur par mot (Word error rate : WER)
- Le taux de reconnaissance par phonème
- Le taux d'erreur par phone (Phone error rate : PERR)
- Le ratio signal/bruit (Signal-to-noise ratio SNR)
- Le taux de compression
- etc…

Cette approche est largement utilisée car les résultats peuvent être facilement comparés et sont en fait les critères d'efficacité les plus importants, lorsque la segmentation est une composante d'un système plus complexe.

Dans le cadre des applications de reconnaissance de la parole, le WER (Word Error Rate) est le score le plus fréquemment utilisé. Une telle performance est calculée en comparant une transcription de référence avec la sortie de la transcription par le système de RAP. A partir de cette comparaison, il est possible de calculer le nombre d'erreurs, qui appartiennent généralement à trois catégories:

- Insertions I (Lorsque dans la sortie du système de RAP il est présent un mot non présent dans la référence)
- Omissions O (un mot manque dans la sortie du système de RAP)
- Substitutions S (un mot est remplacé par un autre)

$$WER = \frac{S + O + I}{N} \tag{3.5}$$

Où, N est le nombre de mots dans la transcription de référence.
Une autre mesure est également utilisée dans ce contexte, il s'agit de la précision (Accuracy).

$$Précision = 1 - WER \tag{3.6}$$

3.3.2 Les mesures de l'objectif

Lorsque les sorties de l'algorithme de segmentation sont comparées à une référence, un ensemble de mesures peut être calculé. Les insertions sont détectées lorsqu'une ou plusieurs frontières créées par l'algorithme de segmentation ne correspondent à aucune frontière de référence ou s'il existe plusieurs frontières générées à proximité d'une seule frontière de

référence. Les omissions sont notées lorsqu'une limite est marquée dans la référence, mais l'algorithme ne produit aucune frontière correspondante. Enfin, les frontières détectées correctement sont considérées comme des alarmes (hits).

En utilisant ces mesures, la qualité de segmentation peut être évaluée et analysée à l'aide de trois scores «partiels»: le taux de réussite (HR : Hit Rate), le taux de sur-segmentation (OS : Over Segmentation) et le taux de fausses alertes (FA : False Alarm).

Le taux de réussite (HR) représente le pourcentage des frontières de référence correctement détectées. Etant donnée une section finie de parole, soit Nhit le nombre de frontières correctement détectées et soit Nref le nombre total de frontières dans la référence. Le HR peut alors être calculé en utilisant l'équation suivante :

$$HR = \frac{N_{hit}}{N_{ref}} \times 100 \tag{3.7}$$

Une autre mesure centrale, en particulier dans le cas des méthodes implicites, est le taux de sur segmentation (OS), qui est le rapport entre le nombre total de frontières détectées Nf et le nombre de frontières dans la référence Nref. Le score OS montre de combien est supérieur (ou inférieur) le nombre total de frontières détectées par l'algorithme, par rapport au nombre total de frontières de référence prises à partir de la transcription manuelle.

$$OS = \left(\frac{N_f}{N_{ref}} - 1\right) \times 100 \tag{3.8}$$

La troisième mesure partielle est le taux de fausse alarme (FA). Il représente le pourcentage de détections incorrectes de l'algorithme.

$$FA = 1 - \frac{N_{hit}}{N_f} \tag{3.9}$$

Afin d'évaluer la qualité globale d'une méthode de segmentation, une mesure globale qui prend simultanément en compte ces scores est nécessaire. La Précision (3.10) décrit la probabilité de la fréquence à laquelle l'algorithme identifie une frontière correcte chaque fois qu'une frontière est détectée. Le Rappel (3.11) est identique au HR (3.7), sauf qu'il n'est pas

échelonné pour être un pourcentage. Afin de décrire la performance d'un algorithme par une valeur scalaire, la F-mesure (3.12) peut être calculée à partir de la précision et du rappel.

$$PRC = \frac{N_{hit}}{N_f} \tag{3.10}$$

$$RCL = \frac{N_{hit}}{N_{ref}} \tag{3.11}$$

$$F = \frac{2.0 \times PRC \times RCL}{PRC + RCL} \tag{3.12}$$

3.4 Segmentation de la parole Arabe

Les recherches pour la segmentation de la parole Arabe sont en harmonie avec celles présentes à travers le monde pour d'autres langues. Ainsi, nous rencontrons des systèmes de segmentation basés sur des approches implicites et explicites.

Dans (Ramban et al., 2015), les auteurs utilisent un algorithme similaire à celui décrit dans (Rabiner et Sambur, 1975) pour segmenter le discours Arabe en phonèmes.

Dans (Awais et al., 2006), les auteurs décrivent un algorithme de segmentation de phonèmes qui utilise la FFT pour segmenter un discours Arabe continu en phonèmes. Pour cela, les valeurs de seuil pour les zones de fréquence sont définies pour séparer les phonèmes en deux classes. Ensuite, l'intensité, la durée du phonème et les plages de fréquence permettent le raffinement de la segmentation dans chaque classe. La figure 3.10 détaille l'algorithme de segmentation.

1. Le spectre de la FFT et les valeurs d'intensité pour le signal de parole sont calculés.

2. Au cours de la première phase, une valeur de seuil particulière de la zone de fréquence de 200Hz-600Hz pour le spectrogramme est utilisée pour séparer les phonèmes en deux classes.

A: /t/, /t'/, /?/, /k/, /q/, /f/, /T/, /s/, /S/, /s'/, /x/, /X/, pause.
B: /w/, /j/, /r/, /l/, /m/, /n/, /D/, /D'/, /z/, /?\/, /G/, /h/, /b/, /d/, /d'/, /dZ/, /a/, /i/, /u/.

3. Ensuite, les pauses sont séparées du reste des consonnes de la classe A à l'aide de trois indices: intensité, durée du phonème et spectrogramme

4. La classe B est affinée en utilisant une valeur de seuil spécifique du spectrogramme dans la zone de fréquence de 500Hz - 1000Hz en :

B.1: /l/,/m/,/n/,/D/,/D'/,/z/,/b/,/d/,/d'/,/dZ/, /i/.
B.2: /w/, /j/, /?\/, /G/, /h/, /r/, /a/, /u/.

5. La classe B.1 est affinée pour séparer la voyelle / i / en utilisant une valeur de seuil spécifique du spectrogramme dans la zone de fréquence de 2000Hz - 4000Hz en :

***B.1.1*:** /l/, /m/, /n/,/D/, /D'/,/z/,/b/,/d/,/d'/,/dZ/.
***B.1.2*:** /i/.

6. Des valeurs de seuil spécifiques pour les zones de fréquences de spectrogramme dans 1000Hz-2000Hz et 2500Hz-4000Hz sont utilisées pour séparer les voyelles / a / et / u / respectivement. Par conséquent, la classe B.2 est subdivisée en :

***B.2.1*:** /w/, /j/, /?\/, /G/, /h/, /r/.
***B.2.2*:** /a/, /u/.

7. Dans la dernière étape, les deux consonnes / h / et / r / sont recherchées dans les régions des voyelles en utilisant le signal d'intensité, et si elles sont trouvées, elles en seront séparées.

Figure 3.10. Algorithme de segmentation (Awais et al., 2006)

(Anwar et al., 2006) et (Abdo et Kandil, 2016) ont présenté des algorithmes de segmentation de la parole consacrés à l'apprentissage de la langue Arabe et en particulier au Coran. En particulier (Anwar et al., 2006) ont déclaré que: "Les propriétés liées au phonème sont intégrées dans différentes propriétés du signal qui peuvent servir de repères pour la segmentation des phonèmes. Il pourrait y avoir une combinaison différente de ces indices qui peuvent générer des résultats différents avec des niveaux de précision différents ". Ainsi, ils préconisent des combinaisons différentes de ces indices.

Le tableau 3.4 montre les résultats rapportés dans les travaux précédents en termes de frontières correctes et de WER ou de précision.

Table 3.4. Performances de quelques algorithmes de segmentation en Arabe

Auteurs	Méthode de Segmentation	Unités cibles	Corpus	Logiciel	Précision / WER	Hit rate
Awais et al., 2006	Spectrogramme FFT	Phonèmes	10 fichiers avec 2346 phonèmes (différents locuteurs)		95.39%	
Nofal et al, 2003	Apprentissage embarqué	Phonèmes	10 heures avec 100 locuteurs	HTK	3.21% 5.4%	35ms:7.75% 70ms:0.78% 100ms:0.78%
Tolba et al., 2005	Ondelettes	Consonne / voyelle	20 mots (6 fois)	Matlab	88.3%	
Anwar et al., 2006	ZCR, PSD	Phonèmes Consonne / voyelle		C++	89%	
Abdo et al, 2016	Maximum local à partir de la dérivée des premiers MFCC	Syllabes	23 phrases par 12 locuteurs (2544 syllabes)		91.5%	

Chapitre 4

Hybridation HMM/SVM pour la segmentation et l'étiquetage de parole Arabe appliqué à la reconnaissance

4.1. Introduction

Comme nous l'avons souligné dans le précédent chapitre, l'apparition de l'algorithme de l'apprentissage embarqué (EL pour embedded learning) a permis aux chercheurs et aux développeurs dans le domaine de la RAP de s'affranchir des corpus annotés manuellement (en particulier pour les langues où ces corpus ne sont pas disponibles) car il permet la segmentation phonétique en se référant à la transcription du corpus à annoter.

(Malfrère et al, 2003) ont utilisé le principe de l'algorithme EL, mais au lieu d'utiliser l'algorithme Baum-Welch pour le raffinement de la segmentation, ils ont utilisé l'algorithme de Viterbi. Les auteurs ont souligné que: "Le principal problème de cette méthode est qu'une première segmentation est nécessaire pour initialiser le processus d'apprentissage"[1]. En tant que solution, ils ont suggéré: "C'est pourquoi les paramètres HMM sont généralement initialisés en utilisant des bases de données segmentées manuellement (au moins partiellement)"[2]. L'initialisation par « Flat start » est considérée comme une limitation de l'algorithme EL (Mporas et al., 2008). Par conséquent, certaines expériences ont été menées pour améliorer les performances de cet algorithme. (Mporas et al. 2008) ont utilisé l'algorithme EL de manière hybride en le combinant avec l'apprentissage des unités isolées.

[1] Traduction de : "The main problem of this method is that a first segmentation is required to bootstrap the training process"

[2] Traduction de : "That is why HMM parameters are usually initialized using (at least partially) hand-labelled databases"

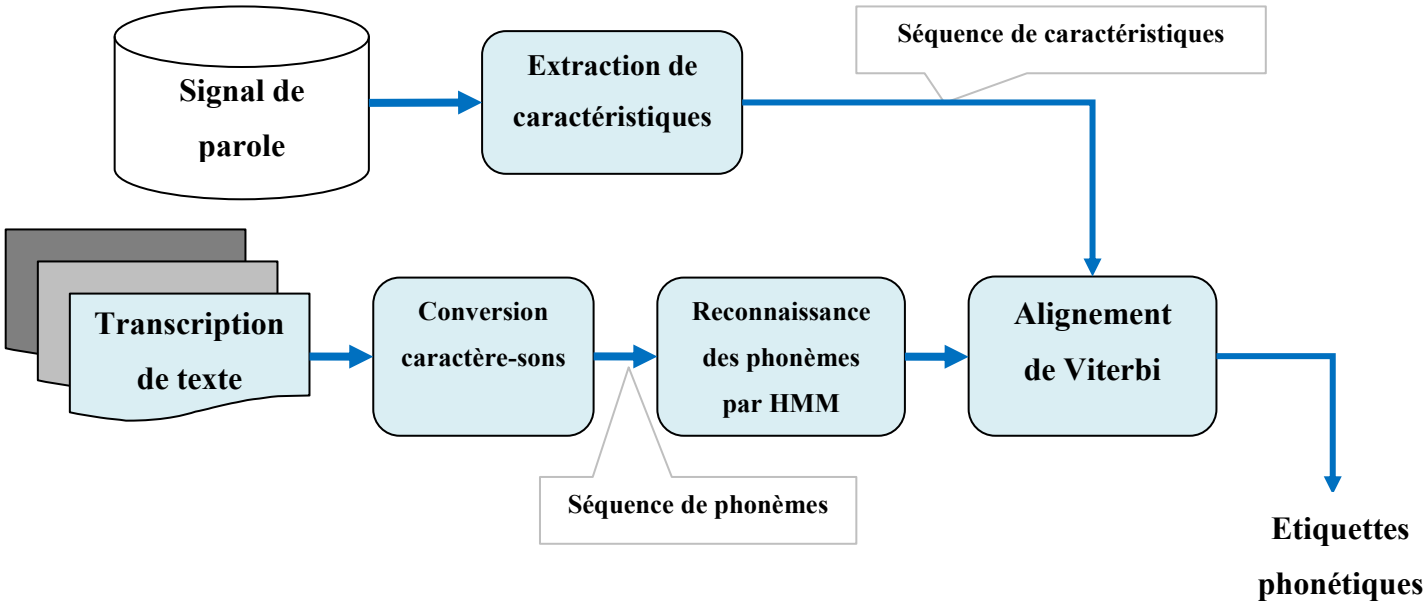


Figure 4.1. Diagramme en bloc du système de segmentation (d'après Mporas et al., 2008)

Dans (Brognaux et Drugman, 2016), les auteurs ont fait trois suggestions principales pour améliorer les performances de l'algorithme EL qu'ils ont déjà mis en œuvre dans leur outil Train & Align (T & A) (Brognaux et al., 2012).

Leur première suggestion est directement liée à l'initialisation uniforme du «Flat Start». Ils ont utilisé l'algorithme de détection d'activité vocale pour détecter les segments non parole. Ces segments sont utilisés pour initialiser uniquement les modèles de silence. Tous les autres phonèmes sont initialisés avec la stratégie « Flat Start ». Les auteurs préconisent que cette extension permet de mieux initialiser les modèles de silence et contribue à une convergence itérative vers de meilleures frontières des segments de silence.

Deuxièmement, les auteurs ont considéré que le vecteur acoustique typique est généralement composé des 12 MFCC et de l'énergie, avec leurs coefficients delta et accélération (39 coefficients) et « que les humains utilisent des indices supplémentaires, qui ne sont pas nécessairement représentés dans les coefficients de MFCC, pour décider de l'emplacement précis des frontières des unités »[3]. Ainsi, les auteurs ont fait quelques expériences pour sélectionner un ensemble d'indices supplémentaires afin d'améliorer les

[3] Traduction de : "that humans make use of additional cues, which are not represented in the MFCC coefficients, to decide the precise location of the boundaries"

résultats de la segmentation par l'algorithme EL. Ils sélectionnèrent formellement la périodicité, la turbulence et l'intensité; Ils obtinrent un nombre total de 42 paramètres.

La troisième extension a exploité le son inversé dans le temps où le signal de parole ainsi que sa transcription sont inversés et soumis à l'algorithme EL. Les auteurs soutiennent que dans le cas de l'alignement correct, les deux corpus devraient fournir les mêmes résultats, sinon « pour des limites incertaines, calculer la moyenne entre les deux alignements devrait fournir des estimations lissées, réduisant ainsi les erreurs avec un seuil de tolérance élevée ».

L'objectif de notre travail est de mettre à la disposition des chercheurs et des développeurs dans le domaine de la RAP et en particulier pour la parole Arabe, des corpus segmentés et étiquetés de manière entièrement automatique, dédié à la création de systèmes de RAP. Ainsi, nous proposons une approche hybride pour la segmentation automatique du corpus parole. L'idée vient du fait que le modèle génératif (Modèle de markov cachées HMM) va être combiné avec un modèle discriminatif (Support Vector Machine SVM). Les HMM génèrent les séquences de phonèmes ainsi que leurs frontières. Puis, le SVM est appliqué pour ajuster ces frontières et étiqueter les segments mal segmentés par les HMMs. Finalement, nous construisons un système de reconnaissance basé HMM via le nouveau corpus segmenté.

4.2. Machines à vecteurs de support (SVM)

Les séparateurs à vaste marge, connus par l'acronyme anglais SVM (Support Vector Machines), ont été développés dans les années 1990 (Cortes et Vapnik, 1995; Vapnik, 1995). Les SVMs constituent une classe d'algorithmes basée sur le principe de minimisation du «Risque structurel » décrit par la théorie de l'apprentissage statistique qui utilise la séparation linéaire. Les séparateurs à vaste marge sont initialement conçus pour les problèmes de classification binaires. Les SVM permettent de séparer linéairement les exemples positifs des exemples négatifs dans un ensemble d'apprentissage par un hyper-plan qui garantisse un maximum de marge (figure 4.2).

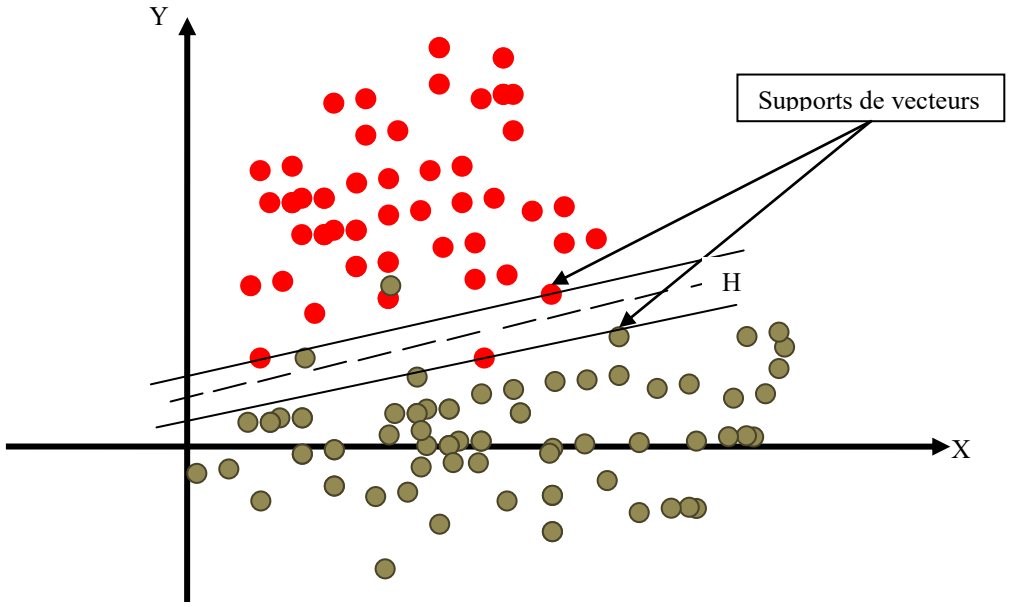


Figure 4.2. Séparateurs à vaste marges

4.2.1. Les données linéairement séparables

Considérons « N » points $\{(X_1, Y_1), (X_2, Y_2), \dots \dots (X_N, Y_N)\}, X_i \in R$ Avec : $i = 1 \dots N$ et $Y_i \in \{\pm 1\}$. Ces points sont classées en utilisant une famille de fonctions linéaires définis par:

$$\langle w, x \rangle + b = o \qquad (4.1)$$

L'opérateur $\langle . \rangle$ est le produit scalaire, w et b sont des paramètres à estimer. Avec $w \in R^n$ et $b \in R$ de telle sorte que la fonction de décision concernant l'appartenance d'un point à l'une des deux classes soit donnée par :

$$f(x) = \mathrm{sgn}(\langle w, x \rangle + b) \qquad (4.2)$$

La fonction (4.1) représente l'équation de l'hyperplan *H*. La fonction de décision va donc observer de quel côté de *H* va se trouver l'élément de x.

On appelle la marge d'un élément la distance euclidienne prise perpendiculairement entre *H* et x. Si on prend un point quelconque t sur *H*, cette marge peut s'ecrire comme :

$$M_x = \frac{w}{\|w\|} (x - t) \qquad (4.3)$$

La marge de toutes les données est définie comme étant :

$$M = \min_{x \in E} M_x \quad (4.4)$$

L'approche de classification par SVM tend à maximiser cette marge pour séparer le plus clairement possible deux classes. Une marge qui soit la plus large possible assure mieux le processus d'affectation d'un nouvel élément à l'une des classes.

Un classifieur à marge maximale est un classifieur dont l'hyperplan optimal séparant deux classes est une solution du problème d'optimisation mathématique suivant (forme primale) :

$$MIN\ \frac{1}{2}\|w\|^2(\langle w, x\rangle + b)z \geq x \in E \quad (4.5)$$

La fonction objective de ce problème est le carré de l'inverse de la double marge qu'on veut maximiser. La contrainte unique correspond au fait que les éléments x doivent être bien placés. La résolution de ce problème nécessite de fixer les paramètres w et b qui constituent les variables de l'algorithme d'apprentissage.

Les classifieurs à marge maximale donnent de bons résultats lorsque les données sont linéairement séparables.

4.2.2. Les données non linéairement séparables

En pratique, il est assez rare d'avoir des données linéairement séparables. Afin de traiter également des données bruitées ou non linéairement séparable, les SVMs ont été généralisées grâce à deux outils : la marge souple (soft margin) et les fonctions noyau (kernel functions).

Le principe de la marge souple est d'autoriser des erreurs de classification. Le nouveau problème de séparation optimale est reformulé comme suit :

$$MIN_{w,b,\epsilon}\ \frac{1}{2}w^T W + c\sum_{i=1}^{l}\epsilon_i\,, C \geq 0 \quad (4.6)$$

Sous les contraintes :

$$y_i(\langle w, x\rangle + b) \geq +1 - \epsilon_i \quad (4.7)$$

$$\epsilon_i \geq 0 \; pour \; i = 0, \ldots \ldots l$$

Un terme de pénalité est introduit dans la formule (4.6), le paramètre C est défini par l'utilisateur, et il peut être interprété comme une tolérance au bruit de classification.

Pour les données non linéairement séparable, l'idée de base est de projeter l'espace d'entrée (espace des données) dans un espace de plus grande dimension appelé espace de caractéristiques (feature space) afin d'obtenir une configuration linéairement séparable (à l'approximation de la marge souple près) des données, et d'appliquer alors l'algorithme SVM. Le processus de recherche d'une fonction de classification en utilisant les SVMs non linéaires se décompose en deux étapes :

Tout d'abord, les vecteurs d'entrée sont transformés en vecteurs de caractéristiques de grande dimension où les données d'apprentissage peuvent être linéairement séparées.

Puis, les SVMs sont utilisés pour trouver l'hyperplan de marge maximale dans le nouvel espace de caractéristiques. L'hyperplan de séparation devient une fonction linéaire dans l'espace de caractéristiques transformé, mais une fonction non linéaire dans l'espace d'entrée d'origine.

4.2.3. La fonction noyau

Le choix du noyau a un impact majeur sur la performance des SVM. Quelques méthodes ont été suggérées pour sélectionner un bon noyau, mais il s'agit encore d'un sujet de recherche actif. En général, le noyau gaussien est souvent préféré, puisqu'il donne de bonnes performances dans toutes sortes de contextes. La Table 4.1 présente les noyaux les plus fréquemment utilisés:

Table 4.1. Les noyaux les plus fréquemment utilisés

Nom	Noyau	
Linéaire	$K(x_i, x_j) = (x_i \cdot x_j)$	(4.8)
Polynomial de degré d	$K(x_i, x_j) = (x_i^T \cdot x_j + 1)^d$	(4.9)
Gaussien	$K(x, y) = exp\left(-\frac{\|x-y\|^2}{2\sigma^2}\right)$	(4.10)
Multi quadratique inverse	$K(x_i, x_j) = \frac{1}{\sqrt{(xi - xj) \cdot (xi - xj) + \beta}}$	(4.11)

4.2.4. SVM multi-classes

Les séparateurs à vaste marge ont été développés pour traiter des problèmes binaires mais ils peuvent être adaptés pour traiter les problèmes multi-classes. Une classification multi-classes peut être mise en œuvre en utilisant la méthode de couplage par paire et ceci par la combinaison de multiple classifieurs binaires.

Le problème multi-classes peut être géré par deux stratégies: « un contre un » ou « un contre tous ». La technique « un contre un » consiste à construire un classificateur pour chaque deux classes. La technique « un contre tous » consiste à construire autant de classifieurs que de classes (Crammer et Singer 2001 ; Hsu et Lin 2002).

4.3. Hybridation HMM / SVM pour la segmentation automatique de la parole

Notre but est de mettre en œuvre pleinement la segmentation automatique de la parole et un système d'étiquetage pour les langues à faibles ressources, nous choisissons l'algorithme de l'apprentissage embarqué dans un contexte de la langue Arabe. Cependant, nous voudrions améliorer ses performances en ajoutant la puissance discriminative des SVM. Les SVMs ont prouvé leur efficacité dans les tâches de classification, mais comme ils sont des modèles non

structurels pour permettre la mise en œuvre de systèmes de reconnaissance à large vocabulaire, nous avons proposé de les utiliser dans l'étape de segmentation pour raffiner les limites et corriger les étiquettes produites par l'algorithme EL (Frihia et Bahi, 2017).

4.3.1. Architecture du système proposé

L'algorithme de l'apprentissage embarqué (EL) est l'état de l'art pour la segmentation et l'étiquetage entièrement automatiques de corpus de parole. Comme nous l'avons expliqué précédemment, notre suggestion vise à améliorer ses performances ; dans ce contexte, nous proposons une méthode de segmentation automatique en appliquant l'apprentissage discriminant sur des données séquentielles étiquetées et générées par une autre technique d'apprentissage générative. Pour cela, on a choisi la combinaison de deux algorithmes performants d'apprentissage les SVM multi-classes et les HMMs, car ces deux algorithmes ont montré leur efficacité dans leurs domaines (classification et génération). L'approche consiste à aider les modèles de Markov cachés lors de la tâche de génération des états. D'abord, on laisse les HMM faire la segmentation des données, puis les SVMs interviennent pour raffiner la segmentation (figure 4.3).

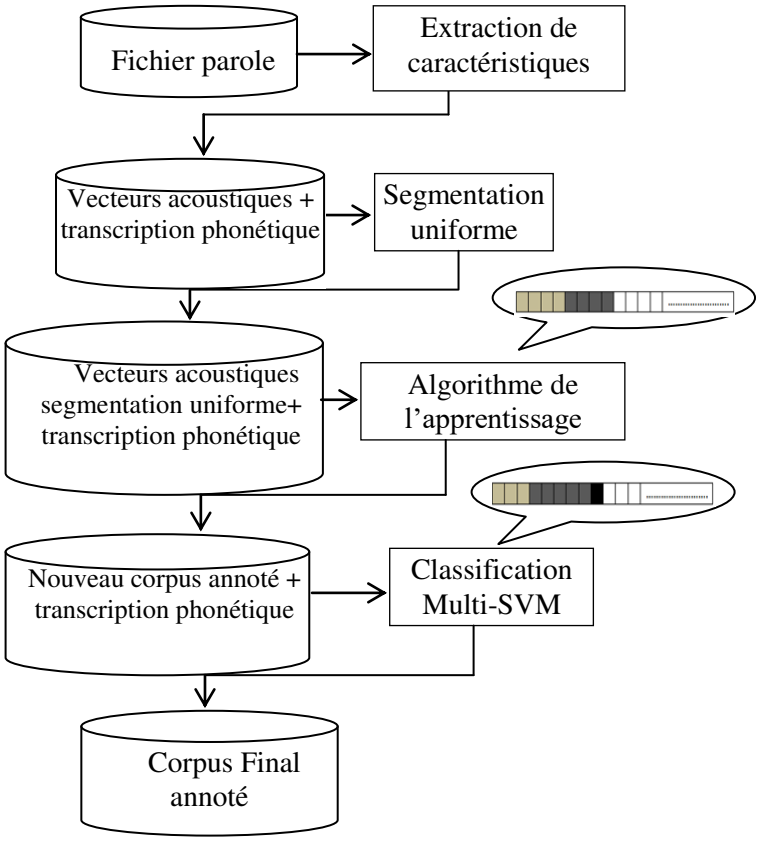


Figure 4.3. Architecture de la combinaison HMM /SVM pour la segmentation de la parole

Etant donné un corpus de signal de parole non segmenté, la première phase consiste à appliquer une méthode d'extraction de caractéristiques (dans notre cas on utilise les coefficients MFCC), cette étape est réalisée en utilisant la fonction HCopy de HTK. Dans un deuxième temps, l'ensemble des vecteurs acoustiques sont uniformément segmentés et étiquetés selon le nombre de phonèmes fournis par la transcription phonétique. Une fois l'initialisation « Flat Start » faite, les segments obtenus (phonèmes avec frontières) et la transcription phonétique sont soumis à l'algorithme EL.

On a utilisé le modèle triphone dans la phase d'étiquetage de segments pour augmenter la probabilité d'avoir la bonne étiquette. A la fin de cette phase, on obtient une première version de corpus segmenté et étiqueté. Les segments obtenus (étiquetés avec les frontières) sont la concaténation du nombre fixe de caractéristiques du vecteur xi : (x1 ... xi,

$i \in 1, N)$ et une étiquette yj, $(j = 1, M), c - à - d : ((x1 \ \ldots \ xi, \ yj))$. Cette sortie sera envoyée au SVM pour raffiner les frontières des segments et corriger les étiquettes mal placées.

4.3.2. Préparation de données pour le SVM

Les sorties de l'algorithme EL seront les entrées du SVM pour raffiner les frontières et corriger les étiquettes mal placées. Les segments résultant de la segmentation EL n'ont pas les mêmes tailles car chaque segment est une suite de vecteurs acoustique $(x1 \ \ldots \ xi, i \in 1, N)$ tel que (chaque xi est un vecteur de 39 coefficients MFCC) qui a une étiquette yj, (j=1,M)soit: $((x1 \ \ldots \ xi, \ yj))$, or les entées du SVM doivent être de taille fixe et identique. Pour cela, on a affecté à chaque xi de la suite $(x1 \ \ldots \ xi)$ l'étiquette yj ; soit $(x1, yj), (x2, yj), \ldots (xi, yj)$. La (figure 4.4) montre plus en détail la préparation de données pour le SVM.

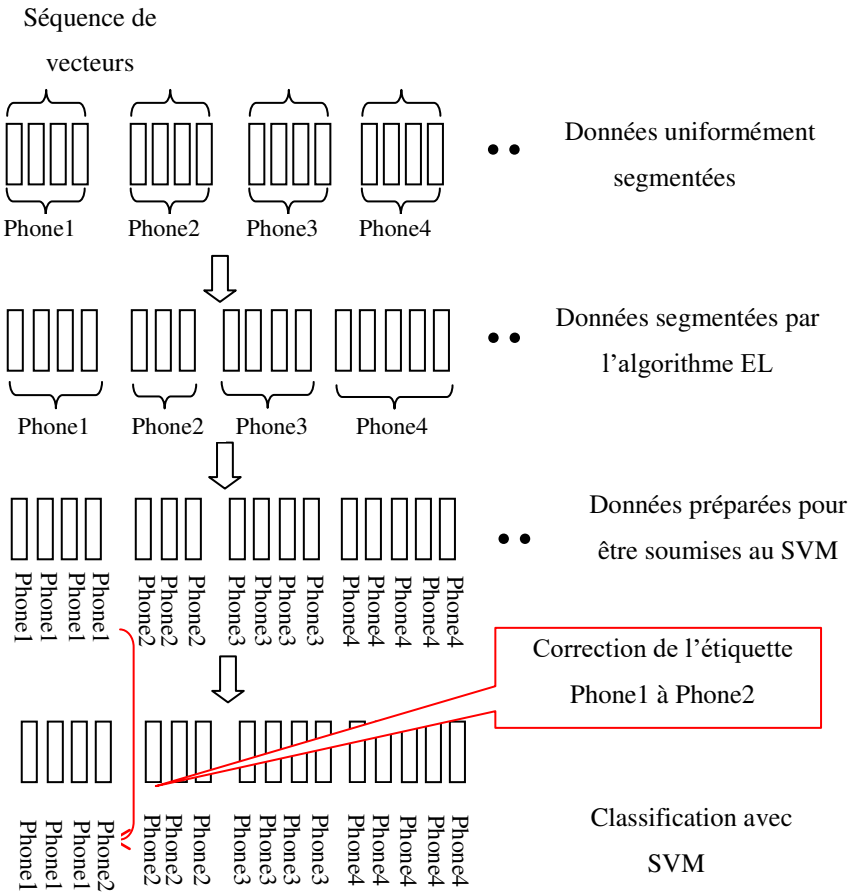


Figure 4.4. La préparation de données pour le SVM

4.3.3. La construction de système de RAP

Une fois que nous avons obtenu notre corpus annoté, la dernière tâche consiste à construire un système de reconnaissance de la parole basé sur ces nouvelles données segmentées pour pouvoir évaluer la qualité de l'algorithme de segmentation. La figure ci-dessous explique en détail l'architecture du système (figure 4.5).

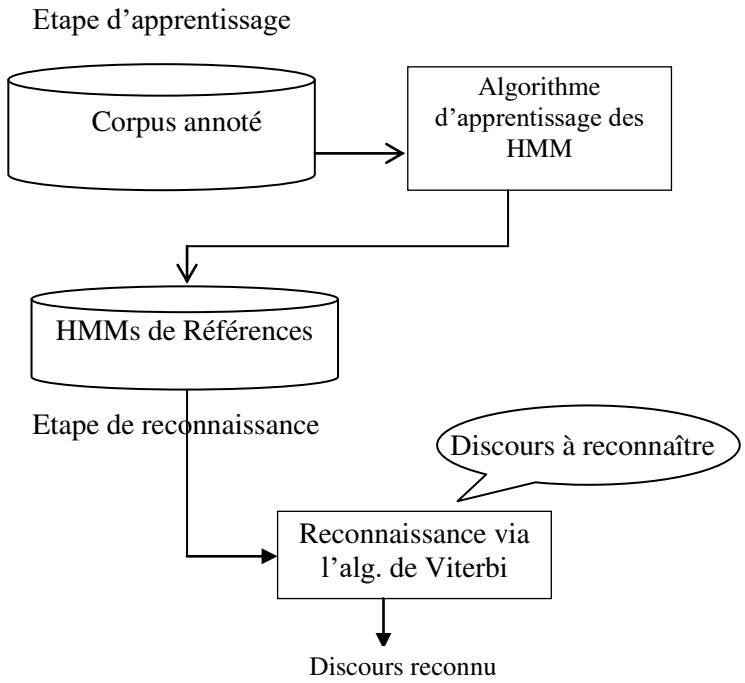


Figure 4.5. L'architecture du système RAP

Chapitre 5
Expérimentations et Résultats

5.1 Introduction

Tout au long du travail de cette thèse, nous avons été amenés à réaliser un certain nombre d'expérimentations. Les premières étaient relatives au développement d'un système de RAP, nous avons présenté une partie des résultats dans le chapitre 2, lorsque nous avons expliqué notre motivation pour le choix de HTK comme librairie de développement. Ensuite, nous avons investigué la faisabilité de l'algorithme EL pour la segmentation de corpus et l'impact possible sur le système de reconnaissance ; ces expériences nous les avons menées sur le corpus « ArabicDigits » puis sur notre corpus « ArabPhone ». Enfin, une fois notre proposition finalisée, nous l'avons évalué en suivant le protocole suivant.

Nous avons procédé à la construction de trois (3) systèmes de RAP de sorte que: les modèles HMM de références du premier système soient basés sur une annotation manuelle du corpus d'apprentissage. Les modèles de référence du deuxième système sont construits sur la base d'un corpus annoté via l'algorithme EL. Les modèles de référence du troisième système sont construits sur la base d'un corpus annoté selon notre proposition (HMM/SVM).

Dans ce chapitre nous allons d'abord présenter les deux corpus utilisés. Ensuite, nous présentons nos tests pour la mise en place de l'algorithme EL en utilisant ces deux corpus. Enfin, nous présentons les évaluations relatives à la proposition HMM/SVM.

5.2 Corpus utilisés

Pour mener nos expérimentations en segmentation, nous avons d'abord utilisé un corpus de chiffres Arabe. Ensuite, nous avons construit notre corpus plus adapté à notre problématique.

5.2.1 Arabic Digits Corpus

Arabic Digits est un ensemble de données créé par (Hemmami et Bedda, 2010) et qui est publié sur le site web d'apprentissage UCI[1] (Lichman, 2013). Cet ensemble de données se compose des dix chiffres Arabes standards. Il comprend 8800 fichiers; chacun représente les vecteurs de coefficients MFCC (Mel Frequency Cepstral Coefficients) extraits d'un chiffre

[1] http://archive.ics.uci.edu/ml/index.php

donné, dont 6600 fichiers représentant les données d'apprentissage et les 2200 restant les données de test.

Les enregistrements des fichiers « wav » proviennent de 88 locuteurs, chacun prononçant les dix chiffres dix fois. Dans ce corpus, seuls les coefficients MFCC sont disponibles. Les coefficients MFCC sont regroupés par bloc. Ainsi, nous avons pour un chiffre donné un ensemble d'occurrences en tant que fichier de caractéristiques. Nous n'avons pas la transcription phonétique des mots ni les limites entre ses différents phonèmes. Nous avons utilisé la transcription orthographique (que nous avons traduit en transcription phonétique) des chiffres comme entrée de l'algorithme EL pour essayer de définir les frontières des phonèmes automatiquement.

5.2.2. Le Corpus « ArabPhone »

L'indisponibilité de corpus de parole segmenté et étiqueté pour la langue Arabe pour la création de systèmes de reconnaissance de parole, nous a encouragé à créer le corpus « ArabPhone » dédié à la langue Arabe Standard (Frihia et Bahi, 2016). En effet, la création de système de parole à grand vocabulaire nécessite un corpus segmenté en unités de base (généralement les phonèmes).

Dans une première version, le corpus « ArabPhone » contient 28 phrases prononcées en langue Arabe Standard (MSA : Modern Standard Arabic) par 30 adultes algériens des wilayas d'Annaba, Jijel, Taref (22 hommes et 8 femmes) âgés entre 20 et 40 ans; (28 phrases * 30 locuteurs = 840 phrases, environ de 2520 mots et 12000 phonèmes). L'enregistrement des fichiers « wav » est fait dans différents environnements pour plus de variété des échantillons. La durée des « wav » varie entre 2 et 6 secondes. Le choix des 28 phrases s'est fait sur la base du nombre de phonèmes dans l'alphabet Arabe. Pour chaque phonème, nous avons associé une phrase qui contient ce dernier dans les trois positions possibles dans un mot (au début, milieu et fin) (Figure.5.1).

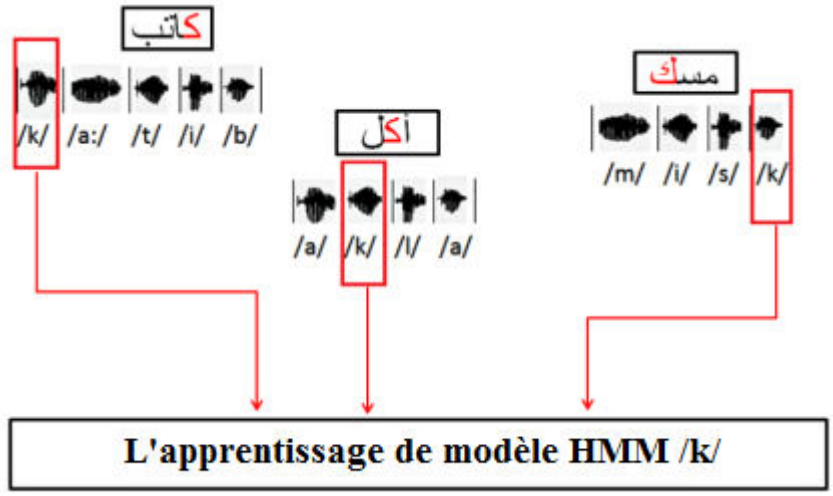


Figure 5.1. Le phonème /k/ dans différentes positions (d'après (Frihia et Bahi, 2016))

Les conditions d'enregistrement sont comme suit: la fréquence d'échantillonnage est de 16000 Hz, la quantification à 16 bits. Pour le fenêtrage on a choisi la fenêtre de Hamming. L'extraction de 39 caractéristiques MFCC se fait en prenant à chaque fois des trames de taille 25 ms de signal avec un glissement de 10 ms. Le vecteur de caractéristiques est le résultat de 12 coefficients MFCC plus le paramètre normalisé de l'énergie ainsi que les premières et les secondes dérivées de ces coefficients pour chaque trame.

Le tableau suivant présente les phrases de corpus ainsi que leurs transcriptions phonétiques selon la norme SAMPA (Speech Assessment Methods Phonetic Alphabet).

Table 5.1. Phrases du corpus « ArabPhone » transcrites selon la norme SAMPA

Alphabet	Phrases en Arabe	Phrase transcrit sous SIMPA
أ	الْجَنَّةُ تَحْتَ أَقْدَامِ الْأُمَّهَاتِ	al-Zannatu_taX-ta_?aq-da:mi_l-?ummaha:t
ب	يَا مُقَلِّبَ الْقُلُوبِ ثَبِّتْ قُلُوبَنَا	ja:_muqalliba_l-qulu:bi_Tabbit_qulu:bana:
ت	اِشْتَرَتْ فَاطِمَهْ تِينًا وَ زَيْتُونًا	?iS-tart_fa:t`imah_ti:nan_wa_zaj-tu:na:

ث	وَرَثَ ثَابِتْ ثَلَاثَةُ ثِيَابٍ	waraTa_Ta:bit_Tala:Tatu_Tija:b
ج	جَاءَ نَجِيبُ مَعَ الْحُجَّاجِ	Za:?a_naZi:bu_maHa_l-XuZZa:Z
ح	جَرَحَ الْحَجَرُ حَافِرَ الْحَيَوَانِ	ZaraXa_l-XaZaru_Xa:fira_l-Xajawa:n
خ	أَخَذَ خَالِدٌ خَاتَمَ خَدِيجَهْ	?axaDa_xa:lidun_xa:tama_xadi:Zah
د	صَدِيقَةُ وِدَادْ عِنْدَهَا دُمْيَهْ	s`adi:qatu_wida:d_Hin-daha:_dum-jah
ذ	ذَرَى الفَلَّاحُ القَمْحَ بِالمِذْرَاةِ	Dara_l-falla:Xu_l-qam-Xa_bil-miD-ra:h
ر	لَا تَزِرُ وَازِرَةٌ وِزْرَ أُخْرَى	la:_taziru_wa:ziratun_wiz-ra_?ux-ra:
ز	زَارَ عَزَّامْ جَزِيرَةَ الْكَرَزِ	za:ra_Hazza:m_Zazi:rata_l-karaz
س	السَّمْعُ وَ الْبَصَرُ مِنَ الْحَوَاسِ	?a-ssam-Hu_wa_l-bas`aru_mina_l-Xawa:s
ش	لاَ أَشْرَبُ الشَّايَ بَعْدَ الْعَشَاءِ	la:_?aS-rabu_SSa:ja_baH-da_l-HaSa:?
ص	سَرَقَ اللُّصُوصُ صُنْدُوقَ الصَيَّادِ	saraqa_llus`u:s`u_s`un-du:qa_s`s`ajja:d
ض	ضَرَبَ الضَّابِطُ الضَرْبَةَ الْقَاضِيَهْ	d`araba_d`d`a:bit`u_d`d`ar-bata_l-qa:d`ijah
ط	الطَائِرُ الْوَطْوَاطُ نَشِيطٌ	?a-t`t`a:?iru_l-wat`-wa:t`u_naSi:t`
ظ	أَظَافِرُ مَحْفُوظْ نَظِيفَةٌ	?aD`a:firu_maX-fu:D`_naD`i:fah
ع	بَاعَ عَادِلٌ الْعَرَبَةَ لِعِمَادْ	ba:Ha_Ha:dilun_l-Harabata_liHima:d
غ	غَابَ بَلِيغْ عِنْدَ الْغُرُوبِ	Ga:ba_bali:G_Hin-da_l-Guru:b

ف	فَرِيدٌ وَ عَفَافٌ بَيْنَ الْصُّفُوفِ	fari:dun_wa_Hafa:fun_baj-na_s`s`ufu:f
ق	إِقْتَرَبَتِ الْقَافِلَةُ مِنَ الْسُّوقِ	?iq-tarabati_l-qa:filatu_mina_ssu:q
ك	كُنْ مُفَكِّرًا يَهَابُكَ عَدُوُّكَ	kun_mufakkiran_jaha:buka_Haduwwuk
ل	لَا إِلَاهَ إِلَّا اللهْ	la:_?ila:ha_?illa_lla:h
م	خَاتَمُ مَرْيَمْ جَمِيلٌ وَ ثَمِينٌ	xa:tamu_mar-jam_Zami:lun_wa_Tami:n
ن	لَا يَنْفَعُ عَمَلٌ بِدُونِ إِيمَانْ	la:_jan-faHu_Hamalun_bidu:ni_?i:ma:n
ه	دَارُ الْهُدَى مَنَارَتُهَا هَادِيَهْ	da:ru_l-huda:_mana:ratuha:_ha:dijah
و	وَضَعَتْ وِدَادٌ الْوُرُودَ بِالْوِعَاءِ	wad`aHat_wida:dun_l-wuru:da_bil-wiHa:?
ي	يَزْرَعُ يَزِيدٌ	jaz-raHu_jazi:d

5.3 Expérimentations avec l'algorithme EL

Dans nos premières expériences, nous avons utilisé l'algorithme EL pour tester sa faisabilité avec un corpus de langue Arabe et s'assurer que les résultats qu'il produit en phase de reconnaissance ne sont pas obsolètes. Dans cette partie, notre protocole d'évaluation consiste à comparer les résultats en phase de reconnaissance produits par un système dont les modèles HMM sont construits sur la base d'une segmentation manuelle à ceux produits par un système dont les modèles HMMs sont construits sur la base d'une annotation de l'algorithme EL (Figure 5.2).

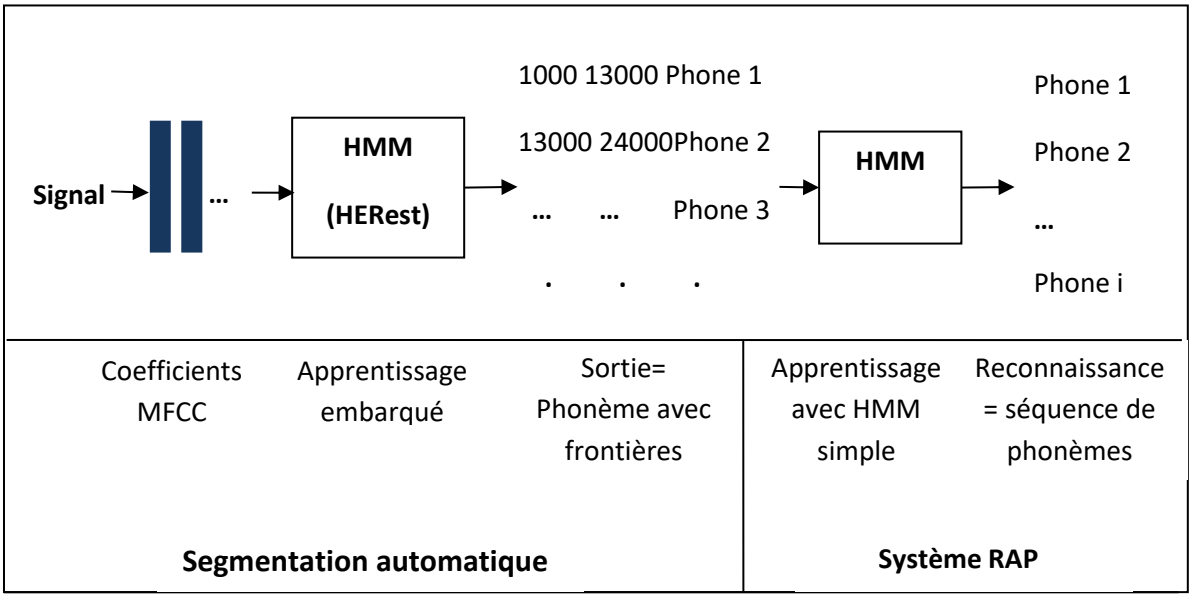


Figure 5.2. Architecture de l'algorithme EL pour la segmentation de la parole

En pratique, la segmentation de la parole se fait comme suit: étant donné la transcription phonétique connu du flux de la parole et de ses coefficients MFCC, nous avons construit un modèle initial HMM pour chaque phonème en utilisant la fonction HcompV de la boîte à outils HTK, cette fonction n'a pas besoin d'étiquetage préalable de fichiers.

HCompV -C config -f 0.01 -m -S train.scp -M hmm/hmm0 hmm/proto

Ensuite, nous alimentons la fonction HERest avec une liste de séquences de phonèmes et nous commençons une session d'apprentissage embarqué. Une fois la phase de segmentation terminée, une deuxième étape consiste à apprendre les modèles.

HERest -C config -I phones0.mlf -t 250.0 150.0 1000.0 -S train.scp -H hmm/hmm0/macros -H hmm/hmm0/hmmdefs -M hmm/hmm1 monophones0

L'initialisation des modèles HMMs se fait avec les données segmentées et étiquetées à l'aide de la fonction HInit. Ensuite, la phase d'apprentissage commence par la fonction HRest.

HInit -I words.mlf -S train.scp -H hmm/macros -M hmm/hmm0 -T 1 -C config -l sil hmm/sil

HRest -I words.mlf -i 100 -S train.scp -H hmm/hmm0/macros -T 1 -M hmm/hmm1 -C config -l sil hmm/hmm0/sil

Plus tard, la fonction HVite qui effectue la reconnaissance nous fournit la séquence des phonèmes à partir du signal d'entrée.

HVite -H hmm/hmm15/macros -H hmm/hmm15/hmmdefs -S train.scp -l '*' -i recout.mlf -w wdnet -p 0.0 -s 5.0 dict tiedlist

HResults -I words.mlf tiedlist recout.mlf

5.3.1. L'apprentissage embarqué avec le corpus « ArabicDigits »

Considérons le premier ensemble de données pour tester la contribution de l'approche d'apprentissage embarqué dans la segmentation, nous avons d'abord effectué une étape de reconnaissance de la parole sur l'ensemble des mots selon l'approche globale, où le mot est reconnu comme une entité. (Table 5.2)

Table 5.2. Taux de reconnaissance par mot.

Chiffre		Taux de reconnaissance
0	s'ifr	94,85
1	WaXid	100
2	ITnan	97,42
3	Tala:Ta	97,27
4	ArbaHa	87,42
5	Xamsa	93,94
6	Sitta	96,67
7	SabHa	85,91
8	Tamanja	99,33
9	TisHa	92,58
Moyenne		94,539

Ensuite, au lieu de prendre les mots comme unité de base, on a considéré les phonèmes comme unité de base ; c'est l'approche analytique. Le tableau 5.3 montre le taux de

reconnaissance du système RAP en termes de séquence de phonèmes par chiffre sur le corpus d'apprentissage et sur le corpus de test.

Table 5.3. Résultats de la reconnaissance au niveau phonèmes

Chiffre		**Taux de reconnaissance**	
		Apprentissage	**Test**
0	s'ifr	92.24	90.76
1	waXid	93.60	91.49
2	iTnan	81.89	83.83
3	Tala:Ta	57.49	61.82
4	arbaHa	75.61	77.85
5	Xamsa	80 .02	82.14
6	Sitta	97.52	97.35
7	sabHa	74.42	75.19
8	Tamanja	66.92	67.98
9	tisHa	87.80	85.71
Moyenne		79,896	80,233

Les taux de reconnaissance pour les chiffres «zéro», «un» et «six» sont bons. Pour les chiffres «deux», «quatre» et «neuf», ils sont acceptables. Le taux de reconnaissance des chiffres trois (3) et huit (8) est inférieur aux autres. Cela s'explique par la présence du phonème / T / (ث). En effet, le phonème / T / est parfois prononcé comme / t / (ت) et puisque nous n'avons pas les fichiers audio, nous ne pouvons pas savoir comment le chiffre était prononcé (/Tala:Ta/ ou /tala:ta/).

Les résultats de Table 5.2 sont meilleurs par rapport avec ceux de la Table 5.3, parce que dans l'approche globale, nous n'avons pas considéré les segments du signal. En effet, un ou deux des phonèmes constituant le chiffre peuvent être mal reconnus alors que le mot entier est bien reconnu.

Le taux de reconnaissance des chiffres «trois» et «huit» dans la Table 5.2 est élevé par rapport aux résultats de la Table 5.3. Ceci est dû à l'effet des deux phonèmes / T / et / t /. Dans

l'approche globale, leur effet est caché, tandis que dans la reconnaissance de phonèmes, leur effet est sensible.

Même si l'ensemble de données est limité et que les mots sont isolés, les résultats montrent l'avantage de l'approche de segmentation considérée pour une étape de reconnaissance. En raison des limites de ce corpus (mots isolés) et en particulier de l'absence de transcription fiable des fichiers acoustiques, nous avons testé l'approche sur le corpus « ArabPhone », de plus à notre connaissance, il n'existe pas de corpus annoté pour la parole Arabe.

5.3.2. L'apprentissage embarqué avec le corpus « ArabPhone »

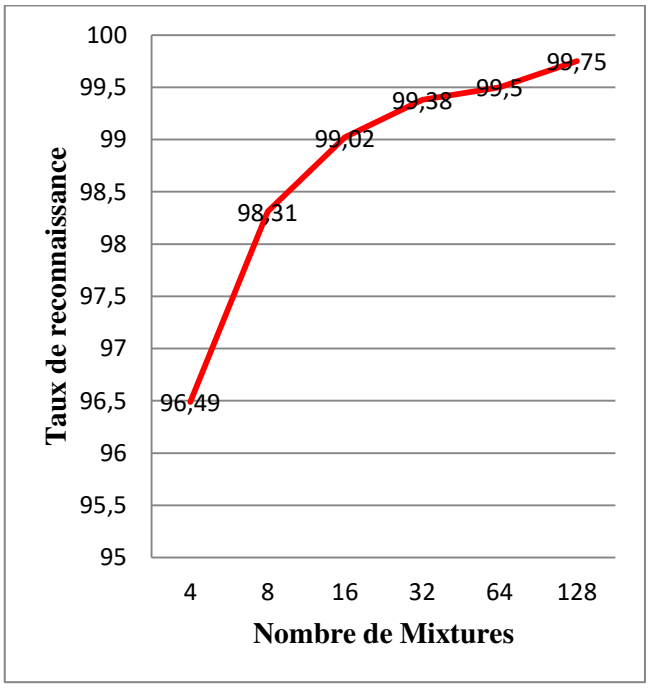
Nous allons montrer l'utilisation de la segmentation automatique de corpus d'apprentissage en utilisant la technique d'apprentissage embarqué pour la construction d'un système de reconnaissance de la parole sur la base « ArabPhone ». La construction des modèles HMMs basées phonèmes s'est faite sur 7 phrases, qui contiennent 21 mots et 35 phonèmes. Nous avons créé 35 HMMs à 3 états de gauche à droite grâce à la segmentation via l'apprentissage embarqué.

Dans un premier temps, on conduit les tests pour déterminer le nombre de mixture nécessaire pour présenter les données pour chaque état de HMM. Les résultats reportés dans la Table 5.4 et la figure 5.3 montrent les résultats obtenus en terme de taux de reconnaissance.

Table 5.4. Taux de reconnaissance du système pour chaque nombre de mixture

	Corpus			
	Apprentissage		Test	
Nombre de mixture de Gaussiennes	Taux de reconnaissance	Précision	Taux de reconnaissance	Précision
4	96,49	90,09	89,66	82,76
8	98,31	96,75	88,79	82,76
16	99,02	98,03	89,43	84,55
32	99,38	99,01	76,15	69,23
64	99,50	99,38	66,92	61,65
128	99,75	99,75	60,77	53,08

Le taux de reconnaissance de Corpus Apprentissage



Le taux de reconnaissance de Corpus Test

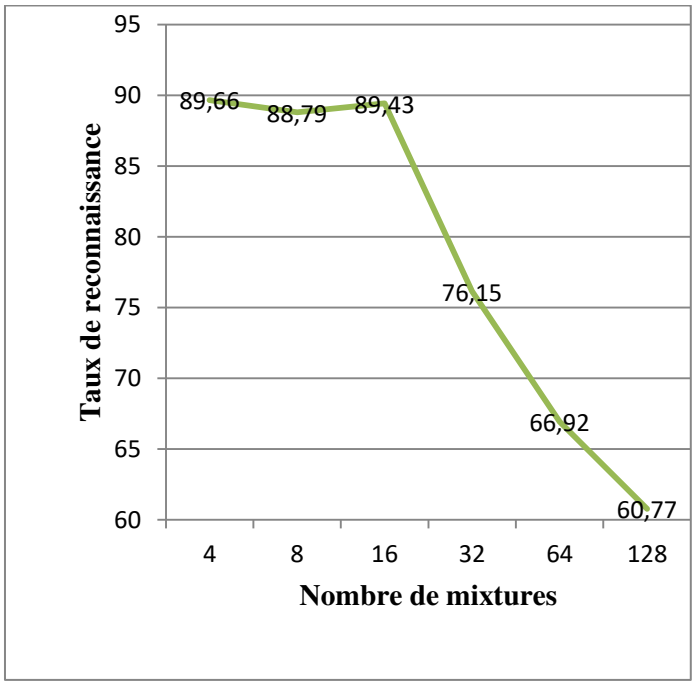


Figure 5.3. Taux de reconnaissance des bases d'apprentissage et tests

Nous remarquons que le taux de reconnaissance des données d'apprentissage augmente en augmentant le nombre de mélange de gaussiennes et en même temps le taux de reconnaissance des données de test diminue. Ce qui peut s'expliquer par le sur apprentissage. Par exemple le taux de reconnaissance sur les données d'apprentissage du modèle avec 128

composantes du mélange est « 99.75 %» par contre le taux sur les données de test est de « 60,77 % ». Nous refaisons les tests mais pour calculer le taux d'alignement à différents intervalles de tolérance (Table 5. 5).

Table 5.5. Taux d'alignement correct (en %) en fonction du nombre de Gaussiennes

Nombre de Gaussiennes	**≤10ms**	**≤20ms**	**≤35ms**	**≤45ms**
4	73.72	79.20	90.67	95.10
8	59.02	72.62	84.76	90.45
16	58.27	71.40	79.79	86.58
32	56.19	66.67	79.85	88.72

Au vu des résultats obtenus avec ces tests et les précédents, on retiendra la valeur de 4 comme nombre de mixtures pour nos Gaussiennes et on poursuit les expérimentations à la base de ce nombre.

Pour évaluer les performances de notre segmentation, nous comparons les performances de deux systèmes de RAP en termes de WER. Le premier système RAP a été construit sur des unités phonétiques segmentées automatiquement et le second a été construit sur la base de la segmentation manuelle des phonèmes. Pour la phase d'apprentissage (y compris la segmentation automatique et les modèles acoustiques d'apprentissage), nous utilisons cinq (5) occurrences pour chacune des 28 phrases (corpus 1). L'ensemble de données de test comprend 17 occurrences des phrases (corpus 2).

Table 5.6. Le taux WER des systèmes RAP avec segmentation manuelle vs automatique

	Segmentation Automatique	**Segmentation Manuelle**
Corpus1	0.0278	0.0126
Corpus 2	0.1106	0.0852

Ces résultats montrent un taux d'erreur de mots d'environ 3% pour le corpus d'apprentissage, qui est supérieur d'environ 1% à la valeur issue de la segmentation manuelle; Ce score est très intéressant, vu l'effort fournit lors de la segmentation manuelle. Pour le corpus de test, le WER est d'environ 11%, qui est supérieur d'environ 3% par rapport à la

valeur issue de la segmentation manuelle. Ce score est faible et devrait être amélioré. L'amélioration peut se faire en fournissant l'ensemble de données d'apprentissage avec des enregistrements supplémentaires.

Pour la deuxième classe de mesures, nous calculons le pourcentage de frontières correctes avec une tolérance de temps allant de 10 à 40 ms.

Table 5.7. le pourcentage de frontières correctes avec une tolérance de temps allant de 10 à 40 ms.

	Tolérance			
	10ms	**20ms**	**30ms**	**40ms**
Hit rate	44.83%	65.52%	89.65%	93.10%

Ces résultats sont très prometteurs et nous ont encouragé à poursuivre nos expériences d'amélioration de l'algorithme EL.

5.3.3 Exemples illustratifs

À titre d'illustration des performances de la segmentation EL avec l'initialisation uniforme (Flat Start) des HMM, nous calculons le taux de réussite obtenu pour la phrase 8 avec trois locuteurs: 1 femme (locuteur 8) et 2 hommes (locuteurs 1 et 21) (Figure 5.4).

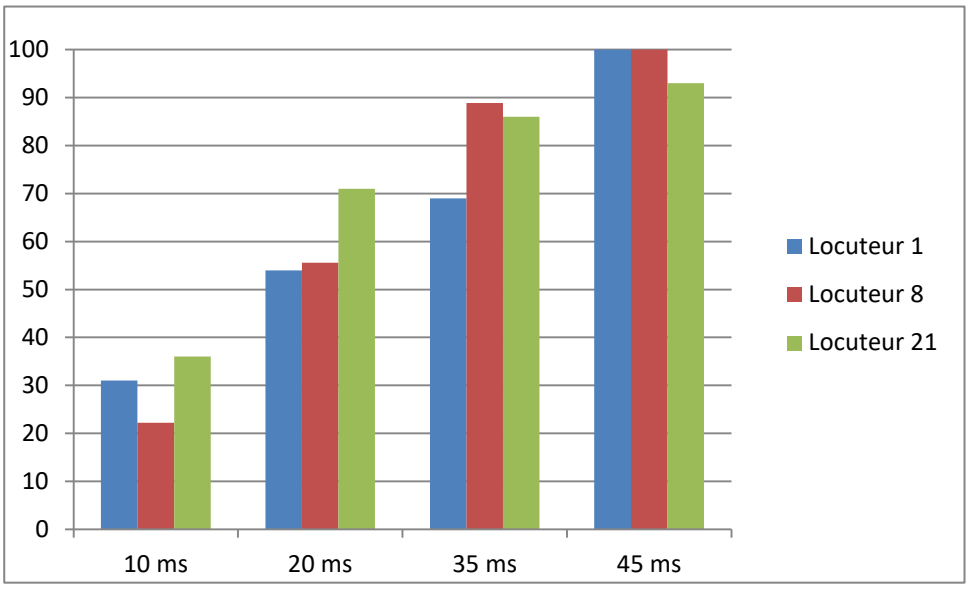


Figure 5.4. Le pourcentage des frontières correctes avec l'initialisation uniforme de l'algorithme EL pour les locuteurs 1, 8 et 21.

De la même manière, nous calculons le taux de réussite de trois phrases pour le locuteur 21.

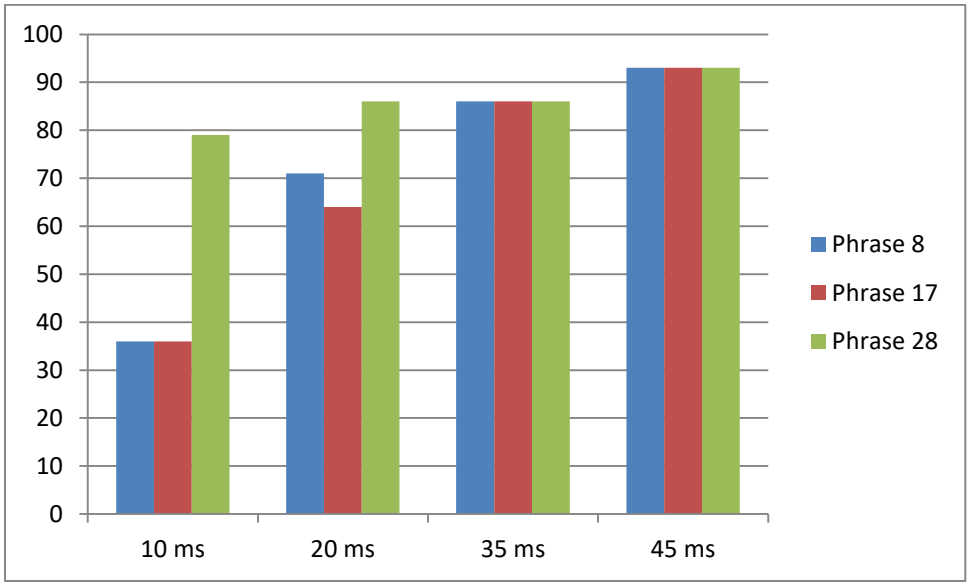


Figure 5.5. Le pourcentage des frontières correctes avec l'initialisation uniforme de l'algorithme EL pour les phrases 8, 17 et 28 du locuteur 21

Phrase 8 صَدِيقَةُ وِدَادْ عِنْدَهَا دُمْيَة

Phrase 17 أَظَافِرُ مَحْفُوظ نَظِيفَة

Phrase28 يَزْرَعُ يَزِيدْ

Avec une tolérance supérieure à 35 ms, le taux de frontières correctes est similaire pour les trois phrases et atteint 93%. A moins de 35 ms, la phrase 28 a le meilleur taux de réussite. La différence s'explique par la brièveté de cette phrase et, en particulier, son faible nombre de voyelles longues. Les voyelles longues étant souvent sous-segmentées.

5.4 L'algorithme HMM/SVM pour la segmentation de la parole

Nous avons établi par les précédentes expérimentations que la segmentation automatique peut produire des résultats proches de ceux de la segmentation manuelle. Dans un souci de réduire l'écart entre les résultats des deux segmentations (écart qui est en faveur de la segmentation manuelle), nous avons proposé une approche hybride dont le but est d'améliorer les résultats de l'algorithme EL.

Pour valider les performances de la proposition de l'algorithme de segmentation HMM/SVM, on considère trois systèmes de RAP appliqués sur le corpus « ArabPhone » : le premier est le modèle de référence basé sur la segmentation manuelle (les données sont manuellement segmentées et étiquetées). Le deuxième utilise les résultats de la segmentation par l'algorithme EL (Frihia & Bahi 2016). Le troisième utilise les résultats de la présente proposition.

À partir de la première classe de mesures, on compare les résultats des trois systèmes de RAP en terme de (WER) le taux d'erreur par mots (Table 5.8).

Table 5.8. Word Error Rate des trois systèmes

Mode de segmentation	WER
Segmentation manuelle	0,0126
Apprentissage embarqué	0,0278
Hybridation HMM/SVM	0,0273

Le premier système de la segmentation manuelle donne les meilleurs résultats en terme de WER, d'autre part la combinaison HMM/SVM améliore les performances de l'algorithme de segmentation EL.

Dans un deuxième temps, on calcule le pourcentage de frontières correctes avec un intervalle de tolérance de 10ms, 20ms, 35ms et 45ms. La Table 5.9 présente les pourcentages de frontières correctes et montre que la combinaison HMM/SVM améliore la qualité de segmentation par rapport à l'algorithme EL dans les intervalles 20 ms, 35 et 45 ms, mais dans la tolérance 10 ms l'algorithme EL est meilleur.

Table 5.9. Taux d'alignment correct (en %)

Mode de segmentation	≤10ms	≤20ms	≤35ms	≤45ms
Apprentissage embarqué	73,72	79,20	90,67	95,10
Hybridation HMM / SVM	61.17	85.88	91.76	96.47

La figure suivante donne un exemple illustratif, où le SVM raffine les frontières obtenues par l'algorithme EL. La voyelle longue /a:/ n'est pas segmenté de manière correcte (lignes jaunes), d'autre part l'hybridation HMM/SVM donne de meilleurs segmentation des voyelles longues.

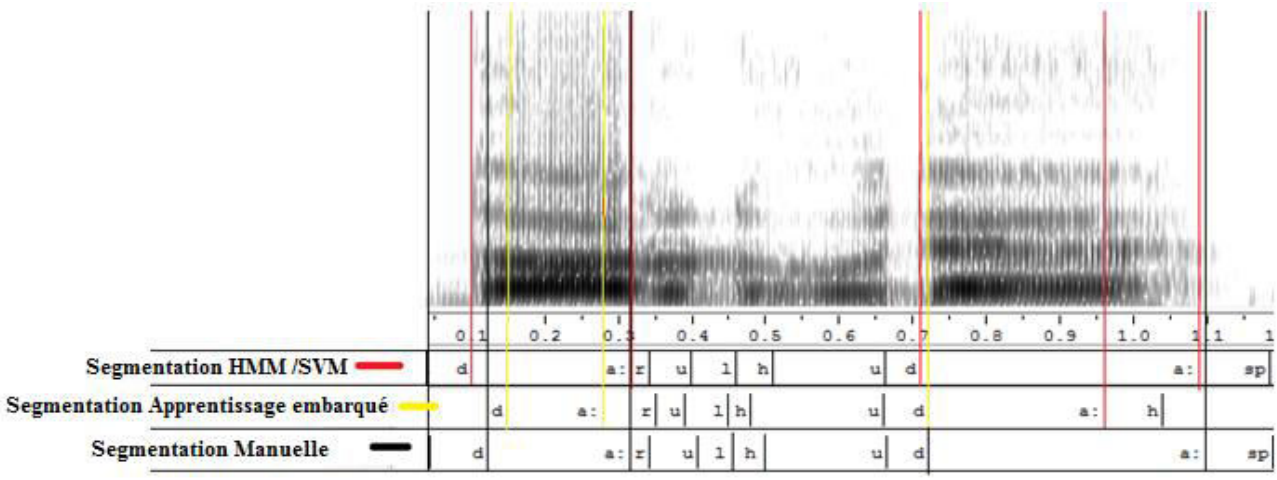


Figure 5.6. Comparaison entre les trois segmentations

Dans la plupart du temps, l'algorithme EL se trompe dans la détection des frontières des voyelles longues, est le SVM raffine la segmentation. Notons aussi que les frontières de segmentation entre deux mots concaténés sans espace (la coarticulation) sont définies de manière assez précise dans notre approche (Figure 5.7).

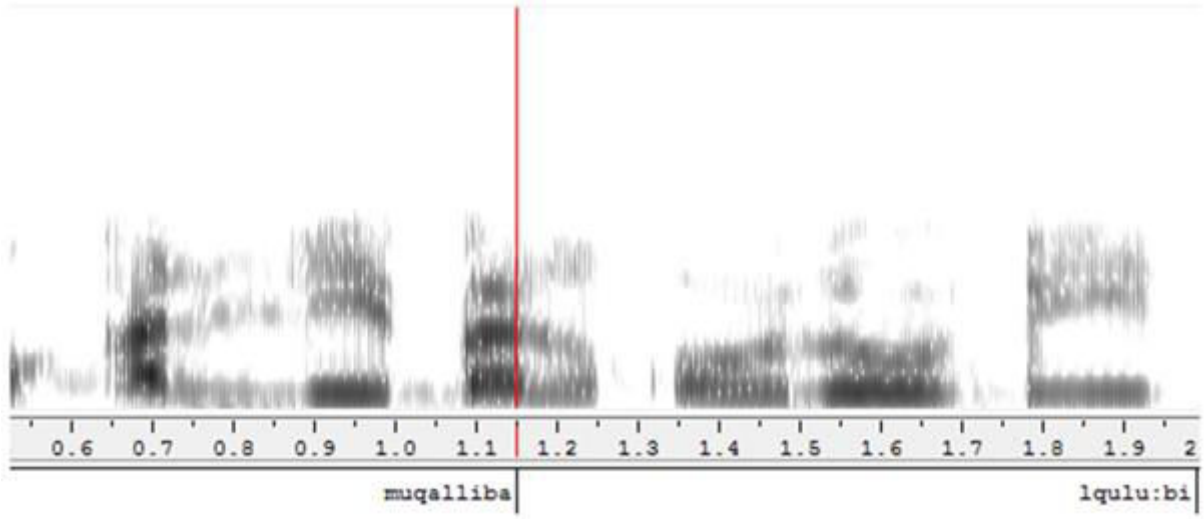


Figure 5.7. Coarticulation entre deux mots

5.5 Conclusion

Dans cette partie, nous avons abordé le problème de la segmentation automatique et de l'étiquetage des données vocales des langues à ressources insuffisantes. L'approche proposée fonctionne en deux phases.

Tout d'abord, l'algorithme d'apprentissage embarqué effectue la segmentation et l'étiquetage initiaux du corpus. Dans une deuxième étape, un SVM multi-classes sert à raffiner les limites des phonèmes et à corriger leurs étiquettes. Pour valider notre approche, nous avons d'abord comparé les segments obtenus avec les références fournies par la segmentation manuelle en termes de frontières correctes et celles fournies par l'algorithme d'apprentissage embarqué et ensuite celles fournies par notre proposition. Enfin, comme le système cible est la reconnaissance de la parole, nous calculons les mesures utilisées pour évaluer les systèmes RAP.

Pour cela, nous avons construit un corpus de parole appelé « ArabPhone ». Au cours de l'étape d'évaluation, nous comparons les résultats issus de reconnaissance la vocale dont les modèles acoustiques des phonèmes ont été construits sur une segmentation manuelle avec ceux de la reconnaissance vocale dont les modèles acoustiques ont été construits sur la base d'une segmentation automatique. Les résultats obtenus par notre proposition en terme de frontières correctes sont en accord avec ceux disponibles dans la littérature (Tableau 5.10),

mais le score WER devrait être amélioré en ajoutant un modèle de langage et par l'augmentation de l'ensemble de données d'apprentissage.

Table 5.10. Tableau comparatif de différents travaux sur la segmentation en langue Arabe

Référence	Méthode de segmentation	Unités	Corpus	Précision (%)	Déviation
Awais et al., 2006	FFT spectrogramme	Consonne/ Voyelle	10 fichiers avec 2346 phonèmes (10 locuteurs)	95.39	
Nofal et al., 2003	EL	Phonèmes	4 phrases avec 258 phonèmes.		7.75 (35ms)
Tolba et al., 2005	Ondelettes	Consonne/ Voyelle	20 words (6 times)	88.3	
Anwar et al., 2006	ZCR. PSD	Consonne/ Voyelle	14300 phonèmes (8 locuteurs)	89	
Abdo et Kandil 2016	maxima local des fonctions Delta des MFCC	Syllabes	23 phrases continues par 12 locuteurs (2544 syllabes)	91.5	
Frihia et Bahi, 2016	EL	Phonèmes	ArabPhone	97.22	6.9 (40ms)
Frihia et Bahi, 2017	EL / SVM	Phonèmes	ArabPhone	97.27	3.53 (45ms)

Conclusion et perspectives

6.1. Bilan

Le développement de systèmes de reconnaissance vocale continue et à grand vocabulaire (LVCSR : pour Large Vocabulary Continuous Speech Recognition) nécessite la disponibilité de grands corpus vocaux segmentés et étiquetés.

Dans notre processus de recherche, on a suivi le cheminement suivant, partant d'une recherche bibliographie, ensuite le choix de l'outil de modélisation de construction de systèmes de reconnaissance de parole (Frihia et Bahi 2014), la création d'une base dédiée pour l'Arabe (Frihia et Bahi 2016), le choix d'une approche automatique pour la segmentation de parole (Frihia et al 2015), et enfin la proposition d'une hybridation HMM/SVM pour améliorer les résultats de la segmentation automatique (Frihia et Bahi 2017).

La langue Arabe étant une langue peu dotée de ressources, le développement de système de reconnaissance automatique de la parole n'est pas très répandu vu l'absence de corpus étiqueté. Le travail que nous menons a pour but de doter la langue Arabe de ressources permettant le développement de diverses applications qui soient basées sur la reconnaissance de la parole.

Comme premier apport au sujet de cette thèse, nous avons étudié les ressources disponibles en termes de logiciel dans le domaine de la reconnaissance de la parole ; ceci nous a permis de réaliser une étude comparative entre différentes librairies. Ce travail a été présenté dans le cadre d'une conférence nationale spécialisée (NCSP : National Conference on Speech Processing) en 2014.

Nous avons aussi pu réaliser un état de l'art sur la segmentation automatique de la parole dont le but était de comprendre le processus de segmentation et les différentes techniques qui y sont utilisés.

En ce qui est de notre principale contribution, elle consiste en la réalisation d'un système de segmentation performant en vue de construire un corpus annoté pour une fin de construction d'un système RAP. Ce système est basé sur l'hybridation des modèles HMMs avec le classifieur SVM (Support Vector Machines) qui est un outil de discrimination puissant ; ce travail a été publié dans le journal « International Journal of Speech Technology ».

Pour pouvoir valider notre proposition, nous avons construit un corpus Arabe annoté « ArabPhone »; ce qui représente une ressource appréciable pour les chercheurs du

domaine ; ce travail a été publié dans un chapitre de livre intitulé ‘‘Text, Speech and Dialog’’ publié par Springer.

6.2. Perspective de l’approche de segmentation

Aujourd’hui, le Deep Learning (DNN, DBN.) est une technique émergente dans le domaine de la reconnaissance de parole, donc notre première perspective consisterait à faire intervenir les DNN dans le processus de segmentation dont le but est d’améliorer les performances de des systèmes de reconnaissance vocal. Notre deuxième perspective est d’implémenter un algorithme d’apprentissage qui effectue l’apprentissage des HMM en prenant en considération les positions de phonème dans le signal de parole

6.3. Perspective du corpus ArabPhone

En partant d’une première version du Corpus ArabPhone on essayera d’atteindre une base de 100 locuteurs et la mettre gratuitement en ligne. On essayera de faire de la collaboration avec d’autres universités algérienne (pour avoir les prononciations de plusieurs régions). Il faut noter que pour le moment nous n’avons que des locuteurs de l’Est Algérien.

Publications

Publications dans des journaux internationaux

Frihia, Hamza et Bahi Halima., HMM/SVM segmentation and labelling of Arabic speech for speech recognition applications. International Journal of Speech Technology (2017). doi:10.1007/s10772-017-9427-z

Conférences nationales

Frihia Hamza et Bahi Halima. Etude Comparative entre les Librairies de Reconnaissance Vocale. National Conference on Speech Processing NCSP'2014, Alger

Conférences internationales

Frihia Hamza et Benouareth Abdellah et Bahi Halima. Embedded Training of HMM for Refinement of the Arabic Speech Segmentation. International conference on Telecommunications and ICT, ICTTelecom-2015, Oran.

Necibi Khaled et Frihia Hamza et Bahi Halima. On The Use of Decision Trees for Arabic Pronunciation Assessment. Proceedings of the International Conference on Intelligent Information 2015.

Frihia Hamza et Bahi Halima. Embedded Learning Segmentation Approach for Arabic Speech Recognition. Sojka P., Horák A., Kopeček I., Pala K. (eds) Text, Speech, and Dialogue. TSD 2016. Lecture Notes in Computer Science, vol 9924. Springer, Cham, Czech.

Bibliographie

Abdo, M. S. and Kandil, A. H. (2016). Semi-automatic segmentation system for syllables extraction from continuous arabic audio signal. International Journal of Advanced Computer Science and Applications, 7, pages 535–540.

Al-Haddad, S. A. R., Samad, S. A., Hussein, A., & Abdullah, M. K. A. (2006). Automatic segmentation and labeling for Malay speech recognition. WSEAS Transactions on Signal Processing, 9.

Andre-Obrecht, R. (1986). Automatic segmentation of continuous speech signals. In *Acoustics,* , IEEE International Conference on Speech, and Signal Processing ICASSP'86., volume 11, pages 2275–2278.

Anwar M. J., Awais M. M., Masud S. et Shamail S. (2006). Automatic arabic speech segmentation system. International Journal of Information Technology, 12, pages 102–111.

Arenas-García, J. et Perez-Cruz, F. (2003). Multi-class support vector machines : a new approach. International Conference on Acoustics, Speech, and Signal Processing (ICASSP'03). 2, pages II–781. Canada.

Awais, M., Masud, S., Shamail, S. et al. (2007). Continuous arabic speech segmentation using fft spectrogram. Innovations in Information Technology, pages 1–6.

Bahi, H., Sellami, M. (2001). Combination of vector quantization and hidden markov models for arabic speech recognition. ACS/IEEE International Conference on Computer Systems and Applications, pages 96–100. Lebanon.

Bahi, H., Sellami, M. (2005). Neural expert model applied to phonemes recognition. International Workshop on Machine Learning and Data Mining in Pattern Recognition, pages 507–515. Germany.

Bahi, H. (2005). NESSR : Un système neuro-expert pour la reconnaissance de la parole thèse. Université d'Annaba.

Bahi, H. et Benati, N. (2009). A new keyword spotting approach. International Conference on Multimedia Computing and Systems (ICMCS'09). pages 77–80. Morocco.

Bansal, P., Pradhan, A., Goyal, A., Sharma, A., et Arora, M. (2014). Speech synthesis-automatic segmentation. International Journal of Computer Applications, 98(4).

Becchetti, C., Ricotti, L. P. (1999). Speech Recognition Theory and C++ Implementation. Snos Ltd, John Wiley.

Bellanger M. (1995). Traitement numérique du signal, Théorie et pratique. Editions Masson.

Benammar R. (2012). Traitement Automatique De La Parole Arabe Par Les HMMs: Calculatrice Vocale. (Doctoral dissertation) Université Abou Bekr Belkaid Tlemcen 2012.

Bilmes, J. A. (2003). Buried markov models : A graphical-modeling approach to automatic speech recognition. Computer Speech & Language, 17(2), pages 213–231.

Brognaux, S., Roekhaut, S., Drugman, T. et Beaufort, R. (2012). Train&align : A new online tool for automatic phonetic alignment. In Spoken Language Technology Workshop (SLT), pages 416–421. Miami.

Brognaux, S., Drugman, T. (2016). Hmm-based speech segmentation : improvements of fully automatic approaches. In IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP), 24(1), pages 5–15.

Brugnara, F., Falavigna, D. et Omologo, M. (1993). Automatic segmentation and labeling of speech based on hidden markov models. Speech Communication, 12 (4), pages 357–370.

Burges, C. J. (1998). A tutorial on support vector machines for pattern recognition. Data mining and knowledge discovery, 2(2), pages 121–167.

Cangemi, F., Cutugno, F., Ludusan, B., Seppi, D., et Van Compernolle, D. (2011). Automatic speech segmentation for italian (assi) : tools, models, evaluation, and applications. Convegno dell'Associazione Italiana Scienze della Voce, Lecce, Italie .

Clarkson, P., Moreno, P. J. (1999). On the use of support vector machines for phonetic classification, International Conference on Acoustics, Speech, and Signal Processing, 2, pages 585–588.

Coelho, L. P., Braga, D. (2006). Automatic phonetic segmentation and labelling of spatanous speech. IV Jornadas en Tecnologia del Habla, pages 369 -372.

Calliope. (1989). La parole et son traitement automatique. (J.P. Tubach, éditeur principal), Collection technique et scientifique des télécommunications, Edition, Masson, Paris.

Chelali, F. Z., et Djeradi, A. (2017). Text dependant speaker recognition using MFCC, LPC and DWT. International Journal of Speech Technology, 20(3), pages 725-740.

Ching P. C., Lee T., Lo W. K. et Meng H. (2007). Cantonese Speech Recognition and Synthesis, Advances in Chinese Spoken Langauge Processing, pages 365-386.

Cooley, J. W., Tukey, J. W. (1965). An algorithm for the machine calculation of complex Fourier series. Mathematics of computation, 19(90), pages 297-301.

Cortes, C. et Vapnik, V. (1995). Support-vector networks. In Machine learning, 20 (3), pages 273–297.

Crammer, K. et Singer, Y. (2001). On the algorithmic implementation of multi-class svms, Journal of Machine Learning Research, 2, pages 265–292.

Dave, N. (2013). Feature Extraction Methods LPC, PLP and MFCC in Speech Recognition, International Journal for Advance Research in Engineering and Technology, 1(VI), pages 1-5.

Davis S., Mermelstein P. (1980). Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences. IEEE Transactions ASSP, 28(4), pages 357-366.

Demuynck, K. et Laureys, T. (2002). A comparison of different approaches to automatic speech segmentation. International Conference on Text, Speech and Dialogue, pages 277–284.

Dines, J., Sridharan, S. et Moody, M. (2002). Automatic speech segmentation with hmm, Proceedings of the 9th Australian Conference on Speech Science and Technology, pages 544–549.

Dixit, R., Kaur, N. (2013). Speech Recognition Using Stochastic Approach: A Review, International Journal of Innovative Research in Science, Engineering and Technology, 2(2).

Douib, O. (2013). Reconnaissance automatique de la parole Arabe par CMU SPHINX 4, Mémoire de Magister, Université de Tebessa.

Dusan, S. et Rabiner, L. R. (2006). On the relation between maximum spectral transition positions and phone boundaries. Proceedings of Interspeech, pages 17– 21.

Ejbali, R., Zaied, M. et Amar, C. B. (2010). Wavelet network for recognition system of arabic word. International Journal of Speech Technology,13 (3), pages 163–174.

Eriksson, L. (2009). Algorithms for automatic segmentation of speech, Working Papers in Linguistics, 35, pages 53–61.

Esposito, A. et Aversano, G. (2005). Text independent methods for speech segmentation. Nonlinear Speech Modeling and Applications, pages 261–290.

Francoeur, D. (2010). Machines à vecteurs de support - Une introduction. Université de Sherbrooke.

Frihia, H., Bahi, H. (2014). Etude comparative entre les librairies de reconnaissance vocale. National conference on speech processing (NCSP'14), Alger.

Frihia, H., Benouareth, A. et Bahi, H. (2015). Embedded training of hmm for refinement of the arabic speech segmentation. International Conference on Telecommunications and ICT (ICT Telecom-2015), Oran.

Frihia, H., Bahi, H. (2016). Embedded learning segmentation approach for arabic speech recognition. International Conference on Text, Speech, and Dialogue, pages 383–390. République de Tchèque.

Frihia. H., Bahi, H. (2017). HMM/SVM segmentation and labelling of Arabic speech for speech recognition applications. Internation Journal of Speech Technology, 20 (3), pages 563-573.

Gamit, M. R., Dhameliya, P. K. et Bhatt, N. S. (2015). Classification Techniques for Speech Recognition: A Review, 5, pages 58-63.

Galka, J., Ziolko, B. (2007). Study of performance evaluation methods for non-uniform speech segmentation. International Journal of Circuits, Systems and Signal Processing, 1, pages 167–172.

Garofolo, J. S., Lamel, L. F., Fisher, W. M., Fiscus, J. G. et Pallett, D. S. (1993). Darpa timit acoustic-phonetic continous speech corpus cd-rom. nist speech disc 1-1.1. NASA STI/Recon, rapport technique, 93.

Gosselin, B. (1996). Application de réseaux de neurones artificiels a la reconnaissance de caractères manuscrits, Thèse de Doctorat, Faculté Polytechnique de Mons.

Guermeur, Y. (2007). SVM Multiclasses Théorie et Applications, Rapport d'Habilitation HDR, Université de Nancy 1.

Guo, G., Li, S. Z. (2003). Content-based audio classification and retrieval by support vector machines. IEEE transactions on Neural Networks, 14 (1), pages 209–215.

Gurban, M., Thiran, J. P. (2005). Audio-visual speech recognition with a hybrid svm-hmm system. 13th European Signal Processing Conference, pages 1–4.

Hacine-Gharbi, A. (2012). Sélection de parametres acoustiques pertinents pour la reconnaissance de la parole, Thèse de Doctorat, Université d'Orléans.

Hosni, H., Sakka, Z., Kachouri, A. et Samet, M. (2009). New approach of speech phonetic segmentation applied at arabic language, Proceedings of SETIT, Tunisie.

Hemdal, J. F., Hughes, G. W. (1967). A feature based computer recognition program for the modeling of vowel perception. In W. Wathen-Dunn, Ed., Models for the Perception of Speech and Visual Form, Cambridge, MA, MIT Press.

Hermansky H. (1990). Perceptual Linear Predictive (PLP) Analysis of speech. Journal of Acoustic Society Am., 87(4), pages 1738-1752.

Hammami, N., Bedda, M. et Farah, N. (2012). Tree distributions approximation model for robust discrete speech recognition, International Journal of Speech Technology, 15(4), pages 455–462.

Hsu, C.-W., Lin, C.-J. (2002). A comparison of methods for multiclass support vector machines, IEEE transactions on Neural Networks, 13(2), pages 415– 425.

Jang, J.S.R. (2017). ASR (Automatic Speech Recognition) Toolbox. Disponible à partir de la page de l'auteur "http://mirlab.org/jang"

Jayasankar, T., Thangarajan, R. et Selvi, J. A. V. (2011). Automatic continuous speech segmentation to improve Tamil text-to-speech synthesis. International Journal of Computer Applications, 25(1), pages 31–36.

Jodouin, J. F. (1994), Les réseaux neuromimétiques, HERMES, Paris.

Kalamani, M., Valarmathy, S., Anitha, S. et Mohan, R. (2014). Review of Speech Segmentation Algorithms for Speech Recognition. International Journal of Advanced Research in Electronics and Communication Engineering, 3(11).

Kalamani, M., Valarmathy, S. et Anitha, S. (2015). Hybrid speech segmentation algorithm for continuous speech recognition. International Journal on Applications of Information and Communication Engineering, 1, pages 39–46.

Kaur, A., Singh, P. et Rattan, D. (2010). Automatic marking of Punjabi syllables boundaries in a sound file, 2nd International Conference on Signal Processing Systems, 3, pages V3-313. Chine.

Kaur, E. A., Singh, E. T. (2010). Segmentation of continuous punjabi speech signal into syllables. In Proceedings of the World Congress on Engineering and Computer Science, 1, pages 20-22. USA.

Kaur, G., Singh, P. (2013). A Technique to Detect Syllable Boundary in a Wave File, International Journal of Computer Science and Communication Engineering, special issue on NCRAET-2013. pages 95-99.

Khanagha, V., Daoudi, K., Pont, O. et Yahia, H. (2014). Phonetic segmentation of speech signal using local singularity analysis. Digital Signal Processing, 35, pages 86–94.

Khawaja, M. A., Haider, N. G. (2007). Segmentation of sindhi speech using formants, IEEE International Conference on Signal Processing and Communications, pages 796–799. UAE.

Kim, D. K., Kim, N. S. (2004). Maximum a posteriori adaptation of hmm parameters based on speaker space projection. In Speech Communication, 42(1), pages 59–73.

King, S., Hasegawa-Johnson, M. (2013). Accurate speech segmentation by mimicking human auditory processing, Intrenational Conference of Acoustics, Speech, and Signal Processing, pages 8096–8100. DOI: 10.1109/ICASSP.2013.6639242

Kirchhoff, K., Bilmes, J., Das, S., Duta, N., Egan, M., Ji, G., He, F., Henderson, J., Liu, D., Noamany, M., et al. (2003). Novel approaches to arabic speech recognition : report from the 2002 johns-hopkins summer workshop. In Acoustics, Speech, and Signal Processing, 1.

Kohonen, T. (1989). Self-Organization and Associative Memory .3rded.Berlin-Heidelberg-, Germany: Springer.

Kuo, J.-W., Lo, H.-Y. et Wang, H.-M. (2007). Improved hmm/svm methods for automatic phoneme segmentation, Interspeech, pages 2057–2060.

Kvale, K. F. A. K. (1991). Manual segmentation and labelling of continuous speech. Phonetics and Phonology of Speaking Styles, Barcelone, Espagne.

Kvale, K. (1993). Segmentation and labelling of speech. Thèse de Doctorat, Institut de Technologie de Norvège.

Lakshmi, A. et Murthy, H. A. (2006). A syllable based continuous speech recognizer for tamil. Interspeech, ICSLP, Ninth International Conference on Spoken Language Processing, Pittsburgh, PA, USA, September 17-21.

Lamel, L., Messaoudi, A., et Gauvain, J.-L. (2009). Automatic speech to-text transcription in arabic. ACM Transactions on Asian Language Information Processing (TALIP), 8(4), pages 18.

Lamere, P., Wok, K., Walker, W., Gouvea, E., Singh, R., Raj, B. et Wolf, P. (2003). Design of the CMU Sphinx 4 decoder. 8th European Conference on Speech Communication and Technology, pages 1181–1184, Genève, Suisse.

Le Blouch, O. (2009). Décodage acoustico-phonétique et applications à l'indexation audio automatique, Thèse de Doctorat, Université Toulouse III-Paul Sabatier.

Li, X., Wu, X. (2014). Labeling unsegmented sequence data with dnn-hmm and its application for speech recognition. 9th International Symposium on Chinese Spoken Language Processing (ISCSLP), pages 10–14. Chine.

Lichman, M. (2013). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

Ljolje, A. et Riley, M. (1991). Automatic segmentation and labeling of speech. International Conference on Acoustics, Speech, and Signal Processing (ICASSP-91). pages 473–476.

Lonsdale WM., Abrecht DG (1988). Seedling mortality in Mimosa pigra, an invasive tropical shrub. Journal of Ecology 77: 371–385.

Lu, L., Zhang, H.-J. et Jiang, H. (2002). Content analysis for audio classification and segmentation. IEEE Transactions on speech and audio processing, 10(7), pages 504–516.

Maji, S., Berg, A. C. et Malik, J. (2013). Efficient classification for additive kernel svms, IEEE transactions on pattern analysis and machine intelligence, 35 (1), pages 66–77.

Malcangi, M. (2009). Softcomputing approach to segmentation of speech in phonetic units, International Journal of Computers and Communications, 3(3), pages 41– 48.

Malfrère, F., Deroo, O., Dutoit, T., et Ris, C. (2003). Phonetic alignment : speech synthesis-based vs. viterbi-based. Speech Communication, 40(4) :503– 515.

Martín-Iglesias, D., Bernal-Chaves, J., Peláez-Moreno, C., Gallardo-Antolín, A. et Díaz-de María, F. (2005). A speech recognizer based on multiclass svms with hmm-guided segmentation, International Conference on Nonlinear Analyses and Algorithms for Speech Processing, pages 257–266.

Mermelstein, P. (1975). Automatic segmentation of speech into syllabic units. Journal of the Acoustical Society of America, 58(4), pages 880–883.

Mporas, I., Ganchev, T., et Fakotakis, N. (2008). A hybrid archi tecture for automatic segmentation of speech waveforms, International Conference on Acoustics, Speech and Signal Processing, pages 4457–4460. Las Vegas.

Mporas, I., Lazaridis, A., Ganchev, T. et Fakotakis, N. (2009). Using hybrid hmm-based speech segmentation to improve synthetic speech quality, 13th Panhellenic Conference on Informatics, pages 118–122. Corfu.

Nagarajan, T., Murthy, H. A. et Hegde, R. M. (2003). Segmentation of speech into syllable-like units. Interspeech, 1 (2), pages 2893–2896.

Nagarajan, T. et Murthy, H. A. (2004). Language identification using parallel syllable-like unit recognition. International Conference on Acoustics, Speech, and Signal Processing, 1, pages 401–404.

Necibi, K. (2015). Evaluation de la prononciation (thèse de Doctorat) en Informatique, Université d'Annaba

Nguyen, L., Ng, T., Nguyen, K., Zbib, R., et Makhoul, J. (2009). Lexical and phonetic modeling for arabic automatic speech recognition, 10th Annual Conference of the International Speech Communication Association. United Kingdom.

Nofal, M., Abdel-Raheem, E., Henawy, H. E. et Kader, N. S. A. (2003). Arabic automatic segmentation system and its application for arabic speech recognition system, 46th Midwest Symposium on Circuits and Systems, 2, pages 697–700.

Ostendorf, M., Favre, B., Grishman, R., Hakkani-Tur, D., Harper, M., Hillard, D., Hirschberg, J., Ji, H., Kahn, J. G., Liu, Y., et al. (2008). Speech segmentation and spoken document processing. Signal Processing Magazine, 25(3).

Panda, S. P. et Nayak, A. K. (2016). Automatic speech segmentation in syllable centric speech recognition system. International Journal of Speech Technology, 19(1), pages 9–18.

Patil, H. A., Madhavi, M. C., Malde, K. D. et Vachhani, B. B. (2012). Phonetic transcription of fricatives and plosives for gujarati and marathi languages, International Conference on Asian Language Processing, pages 177–180. Hanoi.

Patil, H. A., Patel, T., Talesara, S., Shah, N., Sailor, H., Vachhani, B., Akhani, J., Kanakiya, B., Gaur, Y. et Prajapati, V. (2013). Algorithms for speech segmentation at syllable-level for text-to-speech synthesis system in gujarati, Conference on Asian Spoken Language Research and Evaluation, Gurgaon, India.

Peddinti, V., Povey, D., & Khudanpur, S. (2015). A time delay neural network architecture for efficient modeling of long temporal contexts, 16th Annual Conference of the International Speech Communication Association. Germany.

Pekar, D. et Tsikhanenka, S. (2010). Speech segmentation algorithm based on an analysis of the normalized power spectral density, Journal of Telecommunications and Information Technology, pages 44–49.

Pellegrino, F. (1998). Une approche phonétique en identification automatique des langues : la modélisation acoustique des systèmes vocaliques (Thèse de doctorat) Informatique Université de Paul Sabatier de Toulouse

Pellom, B. L. et Hansen, J. H. L. (1998). Automatic segmentation of speech recorded in unknown noisy channel characteristics, Speech Communication, 25, pages 97–116.

Prasad, V. K., Nagarajan, T. et Murthy, H. A. (2004). Automatic segmentation of continuous speech using minimum phase group delay functions, Speech Communication, 42(3), pages 429–446.

Rabiner, L. R. et Sambur, M. R. (1975). An algorithm for determining the endpoints of isolated utterances. Technical Journal Bell Labs, 54(2), pages 297–315.

Rabiner, L. R., Levinson S. E. et Sondhi, M. M. (1983). On the Application of Vector Quantization and Hidden Markov Models to Speaker Independent, Isolated Word Recognition. Bell System Tech. Journal, 62(4), pages 1075-1105.

Rabiner, L., Juang, B., Levinson, S. et Sondhi, M. (1985). Recognition of isolated digits using hidden markov models with continuous mixture densities, AT&T Technical Journal, 64(6), pages 1211 – 1234.

Rabiner, L. et Juang, B. (1986). An introduction to hidden markov models. IEEE ASSP Magazine, 3(1), pages 4–16.

Rabiner, L. R. (1989). A tutorial on hidden markov models and selected applications in speech recognition. Proceedings of the IEEE, 77(2), pages 257–286.

Radman, A., Zainal, N., Umat, C. et Hamid, B. A. (2015). Effective arabic speech segmentation strategy, Jurnal Teknologi, 77(1), pages 9–13.

Rahman, M. M. et Bhuiyan, M. (2012). Continuous bangla speech segmentation using short-term speech features extraction approaches, International Journal of Advanced Computer Sciences and Applications, 3(11), pages 131–139.

Rahman, M. M. et Bhuiyan, M. A.-A. (2013). Dynamic thresholding on speech segmentation. Int J Res Eng Technol, 2(9), pages 404–411.

Räsänen, O. J., Laine, U. K. et Altosaar, T. (2009). An improved speech segmentation quality measure : the r-value, Interspeech, pages 1851–1854. United Kingdom.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain, Psychological review, 65(6), pages 386.

Rumelhart, D. E., Hinton, G. E. et Williams, R. J. (1986). Learning representations by back-propagating errors, Nature, 323(9), pages, 533-536.

Sá, Â. A., Andrade, A. O. et Soares, A. B. (2008). Estimation of hidden markov models parameters using differential evolution. AISB 2009 Convention Communication, Interaction and Social Intelligence, Aberdeen, Ecosse.

Sá, Â. A., Andrade, A. O., Soares, A. B. et Nasuto, S. J. (2008). A study regarding initialization of hidden markov models parameters using differential evolution, AISB 2008 Convention Communication, Interaction and Social Intelligence, Aberdeen, Ecosse

Saadia Z., Fawad H., Muhammad R., Muhammad H. Y. et Hafiz A. H. (2015). Optimized Audio Classification and Segmentation Algorithm by Using Ensemble Methods, Mathematical Problems in Engineering.

Sakoe, H., Chiba, S. (1978). Dynamic programming algorithm optimization for spoken word recognition. IEEE Transactions on Acoustics, Speech and Signal Processing, 26 (1) pages 43–49.

Salam, M., Mohamad, D. et Salleh, S. (2010). Speech segmentation using divergence algorithm with zero crossing property, 13th International Conference on Computer and Information Technology, pages 488–493. Dhaka.

Samudravijaya, K., Barot, M. (2003). A comparison of public-domain software tools for speech recognition, Workshop on spoken language processing.

Sangeetha, J. et Jothilakshmi, S. (2012). Robust automatic continuous speech segmentation for indian languages to improve speech to speech translation, International Journal of Computer Applications, 53, pages 13–16.

Sarkar, A. et Sreenivas, T. V. (2005). Automatic speech segmentation using average level crossing rate information, International Conference on Acoustics, Speech, and Signal Processing , 1, pages I–397.

Schafer, R. W. (1995). Scientific bases of human-machine communication by voice. Proceedings of the National Academy of Sciences, 92(22) pages 9914–9920.

Shah, N. J., Vachhani, B. B., Sailor, H. B. et Patil, H. A. (2014). Effectiveness of PLP-based phonetic segmentation for speech synthesis, International Conference on Acoustics, Speech and Signal Processing, pages 270-274. Florence.

Shanmugam, S. A. et Murthy, H. A. (2014). A hybrid approach to segmentation of speech using group delay processing and hmm based embedded reestimation. Interspeech, pages 1648–1652. Singapore.

Sharma, M., Mammone, R. (1996). Blind speech segmentation: automatic segmentation of speech without linguistic knowledge, International Conference on Spoken Language, 2, pages 1237-1240.

Shastri, L., Chang, S. et Greenberg, S. (1999). Syllable detection and segmentation using temporal flow neural networks. International Congress of Phonetic Sciences, pages 1721–1724.

Solera-Ureña, R., Padrell-Sendra, J., Martín-Iglesias, D., Gallardo-Antolín, A., Peláez-Moreno, C. et Díaz-de María, F. (2007). Svms for automatic speech recognition : a survey, Progress in nonlinear speech processing, LNCS 4391, pages 190–216.

Spalanzani, A. (1999). Algorithmes évolutionnaires pour l'étude de la robustesse des systèmes de reconnaissance de la parole. Thèse de Doctorat, Université Joseph Fourier-Grenoble I.

Stevens S., Volkman J. (1940). The relation of pitch to frequency. American Journal of Psychology, 53.

Taboada J., Feijoo S., Balsa R., Hernandez C. (1994). Explicit estimation of speech boundaries, IEEE Proc. Sci. Meas. Technol., 141, pages 153-159.

Taylor, P. et Isard, S. (1990). Automatic diphone segmentation using hidden markov models, 3rd International Australian Conference in Speech Science and Technology, pages 250–254.

Thiang. et Suryo W.. (2011). Speech Recognition Using Linear Predictive Coding and Artificial Neural Network for Controlling Movement of Mobile Robot. International Conference on Information and Electronics Engineering, 6, pages 179-186, Singapore.

Ting, C.-M., Salleh, S.-H., Tan, T.S. et Ariff, A. (2007). Automatic phonetic segmentation of malay speech database, 6th International Conference on Information, Communications and Signal Processing, pages 1–4. Singapore.

Tolba, M., Nazmy, T., Abdelhamid, A. et Gadallah, M. (2005). A novel method for arabic consonant/vowel segmentation using wavelet transform, International Journal on Intelligent Cooperative Information Systems, 5(1), pages 353– 364.

Toledano, D. T. et Gómez, L. A. H. (2002). Hmms for automatic phonetic segmentation. LREC Conference on language resources and evaluation Proceedings, pages 1558–1563. Spain.

Toledano, D. T., Gómez, L. A. H. et Grande, L. V. (2003). Automatic phonetic segmentation, IEEE transactions on speech and audio processing, 11(6), pages 617–625.

Umesh, S., Cohen, L. et Nelson, D. (1999). Fitting the mel scale, International Conference on Acoustics, Speech and Signal Processing, 1, pages 217-220, Phoenix, Arizona, USA.

Vachhani, B. B. et Patil, H. A. (2013). Use of plp cepstral features for phonetic segmentation, International Conference on Asian Language Processing (IALP), pages 143–146. Chine

Vapnik, V. (1995). The nature of statistical learning theory, Springer Verlag, New York, USA.

van Hemert, J. P. (1991). Automatic segmentation of speech, IEEE Transactions on Signal Processing, 39, pages 1008–1012.

Vaufreydaz, D. (2002). Modélisation statistique du langage à partir d'Internet pour la reconnaissance automatique de la parole continue, Doctoral dissertation, Université Joseph-Fourier-Grenoble I.

Vidal, E., Marzal, A. (1990). A review and new approaches for automatic segmentation of speech signals, Signal Processing V : Theories and Applications, 1, pages 43–53.

Vorstermans, A., Martens, J.-P. et Coile, B. V. (1996). Automatic segmentation and labelling of multi-lingual speech data., Speech Communication, 19, pages 271–293.

Vuuren, V., Bosch, L. et Niesler, T. (2015). Unconstrained speech segmentation using deep neural networks, International conference on pattern recognition applications and methods, 1, Lisbon, Portugal.

Waibel, A., Hanazawa, T., Shikano, K., Hinton, G. et Lang, K. (1988). The Journal of the Acoustical Society of America 83, S45.

Wang, J., Wang, J., Chen, T. et Chang, C. (2007). Speech recognition system, Brevet US 7266496 B2.

Wang, H., Lee, T., Leung, C. C., Ma, B. et Li, H. (2015). Acoustic segment modeling with spectral clustering methods, IEEE/ACM Transactions on Audio, Speech and Language Processing, 23(2), pages 264–277.

Yang, H .J., Oehlke, C. et Meinel, C. (2011). German speech recognition: A solution for the analysis and processing of lecture recordings, 10th IEEE/ACIS International Conference on Computer and Information Science, pages 201-206, Sanya, Chine.

Young, S., Evermann, G., Gales, M., Hain, T., Kershaw, D., Liu, X., Moore, G., Odell, J., Ollason, D., Povey, D. et al. (2002). The htk book, Cambridge university engineering department, 3.

Zarrouk, E., Ayed, Y. B., et Gargouri, F. (2014). Hybrid continuous speech recognition systems by hmm, mlp and svm : a comparative study, International Journal of Speech Technology, 17(3), pages 223–233.

Zelinski, R., Class, F. (1983). A segmentation algorithm for connected word recognition based on estimation principles, IEEE Transactions on Acoustics, Speech, and Signal Processing, 31(4), pages 818–827.

Ziółko, B., Manandhar, S. et Wilson, R. C. (2007). Fuzzy recall and precision for speech segmentation evaluation, 3rd Language and Technology Conference, Poznań, Pologne.

Ziółko, B., Manandhar, S., Wilson, R. C. et Ziółko, M. (2006). Wavelet method of speech segmentation, 14th European Signal Processing Conference, pages 1–5. Italy.

Site:

Institut Electronique et Informatique Gaspard Monge (IGM), [Online]: http://www.igm.univ-mlv.fr.

CMUSphinx , Open Source Toolkit For Speech Recognit ion, Project By CMU, Sphinx 4 Application Programmer's Guide, [Online]: http://cmusphinx.sourceforge.net/wiki/tutorialsphinx4

http://www.bedaux.net/ann_cour

http://www.grappa.univ.lille3.fr/polys/apprentissage/sortie005.html#toc1

Annexe

Annexe

1. Introduction

Au travers de cette annexe, nous souhaitons donner en exemple les étapes suivies pour construire un système complet dédié à la reconnaissance automatique de la parole (RAP) en utilisant les librairies de l'outil HTK. Nous commençons à partir de la base de données sous forme de fichiers audio (.wav) qui doit être disponible pour démarrer l'apprentissage des modèles en passant par l'étape d'extraction des caractéristiques et en arrivant à l'étape de la reconnaissance et du calcul du taux de reconnaissance. Nous allons décrire dans ce qui suit les différentes étapes suivies.

2. Enregistrement des fichiers

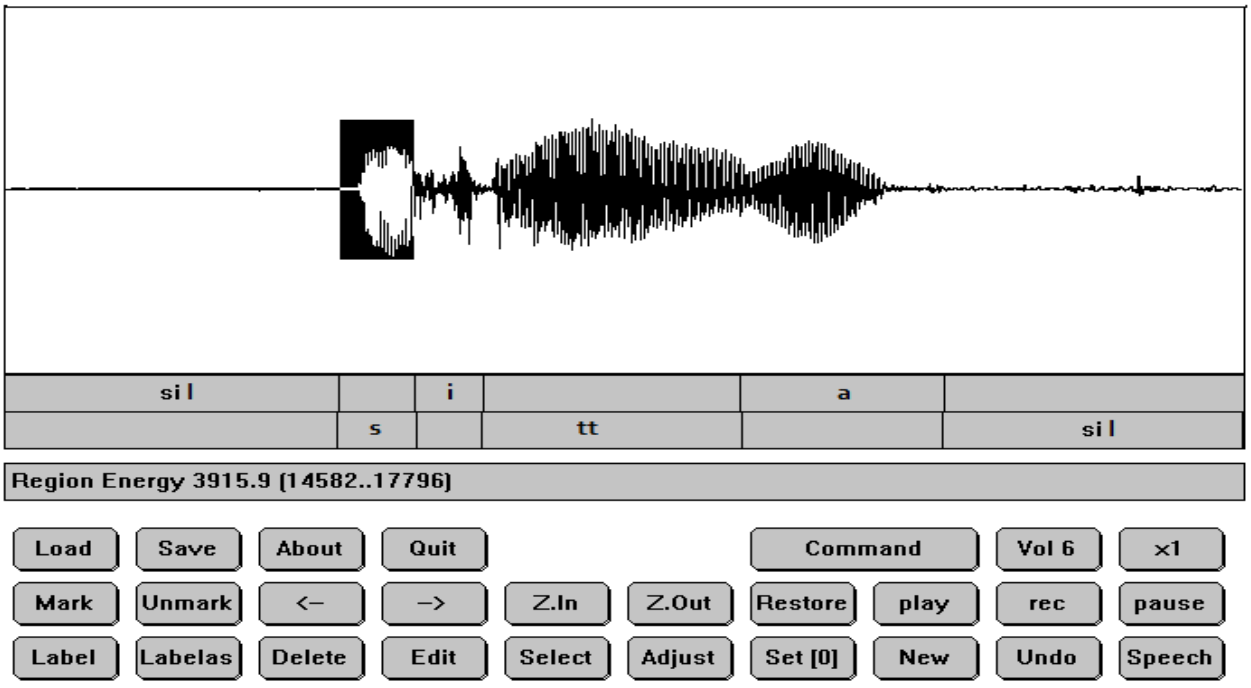
La première opération consiste à enregistrer les mots (phrases) avec le logiciel waveSurfer (que nous avons choisi), les mots utilisés dans notre exemple sont les chiffres Arabe : **wa:hid, iTna:ni …**

Segmentation et étiquetage des entités lexicales :

Cette opération étant manuelle, son but est de définir les entités lexicales, en utilisant la fonction HSLab de HTK

HSlab –F WAVE –Label/Signal/Sitta1.wav

La figure suivante montre ce que donne cette commande.



Le résultat de l'opération est un fichier label dans le répertoire Labels :

Sitta1.lab :

0	560634	sil
560634	746576	s
746576	873378	i
873378	1366439	tt
1366439	1650975	a
1650975	1929932	sil

Sur chaque ligne, il y a le début, la fin et le nom de l'étiquette de chaque segment. On refait l'opération pour tous les autres fichiers de la base.

3. L'extraction des coefficients MFCC

On construit la représentation acoustique du signal (dans notre cas on utilise les coefficients MFCC). Pour cela, On fait appelle à la fonction *HCopy* pour copier les fichiers contenants les cepstraux dans la liste datatrainOneWav.scp. La syntaxe de la commande est comme suite :

Hcopy -T 1 -C confighcopy -S datatrainOneWav.scp

où datatrainOneWav.scp contient la liste des coefficients cepstraux. Avant celà, on crée le fichier confighcopy qui contient les paramètres suivants:

Annexe

```
SOURCEFORMAT = WAV
TARGETRATE = 100000
WINDOWSIZE = 250000
USEHAMMING = T
TARGETKIND = MFCC_D_E_A
NUMCHANS = 8
ENOMALISE = T
CEPLIFTER = 22
NUMCEPS = 3
PREEMCOEF = 0.97
SAVECOMPRESSED = F
SAVEWITHCRC = F
```

Après l'exécution de cette commande, par exemple : pour le fichier sitta2.mfc, on aura, avec la fonction HList :

HList parametres/ Sitta2.mfc

```
C:\htk>HList parametres/Sitta2.mfcc
------------------------------------ Samples: 0->-1 -------------------
--------
0:      -5.154   0.441  -1.249  -0.099  -0.654  -0.095  -0.442  -0.010
1:      -7.431   0.369  -2.714  -0.097  -0.748  -0.163  -0.818  -0.016
```

4. Modélisation des HMMs (Initialisation et Apprentissage)

Pour chaque unité lexicale : s-i-tt-a …etc, on définit le modèle associé. A cet effet, on donne la topologie de chaque modèle, le nombre d'états et les probabilités de transition entre les états. Les valeurs des moyennes et des variances sont initialisées au début à 0 et à 1.

Exemple fichier modèle pour "s" (de sitta)

Annexe

<BeginHMM>

<NumStates> 5 <VecSize> 39 <MFCC_D_E_A>

<State> 2

<Mean> 39

0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0

<Variance> 39

1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0

<State> 3

<Mean> 39

0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0

<Variance> 8

1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0

<State> 4

<Mean> 39

0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0

<Variance> 39

1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0

<TransP> 5

0.000e+0 1.000e+0 0.000e+0 0.000e+0 0.000e+0

0.000e+0 9.000e-1 1.000e-1 0.000e+0 0.000e+0

0.000e+0 0.000e+0 9.000e-1 1.000e-1 0.000e+0

0.000e+0 0.000e+0 0.000e+0 9.000e-1 1.000e-1

0.000e+0 0.000e+0 0.000e+0 0.000e+0 0.000e+0

<EndHMM>

4.1. L'apprentissage

L'apprentissage se fait par les 3 commandes : HInit, HRest, HErest ou (HCompV, HErest dans le cas de données non segmentées)

1) - ***HInit :***

Les moyennes, les variances et les probabilités de transition entre états sont réestimées jusqu'à ce qu'un nombre maximum d'itération soit atteint ou que le critère d'arrêt soit vérifié. C'est l'algorithme de Baum Welch qui s'en charge.

Le fichier train.*scp* contient la liste de tous les fichiers « .mfcc ». Après, on exécute la ligne de commande suivante (exemple pour l'étiquette : « s »)

HInit -I words.mlf -S train.scp -H hmm/macros -M hmm/hmm0 -T 1 -C config -l s hmm/s

On obtient l'affichage suivant :

```
C:\htk>HInit -C config/hinit.conf -A -o Modeles/HinitSurApp/wa -l wa -L Lab
els/ -i 90 -T 1 -m 2 Modeles/Gabarit/wa -S Listes/mfcc.app.lst
HInit -C config/hinit.conf -A -o Modeles/HinitSurApp/wa -l wa -L Labels/ -i
T 1 -m 2 Modeles/Gabarit/wa -S Listes/mfcc.app.lst
Initialising  HMM Modeles/Gabarit/wa . . .
 States   :   2  3  4 (width)
 Mixes  s1:   1  1  1 (  8  )
 Num Using:   0  0  0
 Parm Kind:  MFCC_E_D
 Number of owners = 1
 SegLab   :  wa
 maxIter  :  90
 epsilon  :  0.000100
 minSeg   :  2
 Updating :  Means Variances MixWeights/DProbs TransProbs

 - system is PLAIN
3 Observation Sequences Loaded
Starting Estimation Process
Iteration 1: Average LogP =  -206.84230
Iteration 2: Average LogP =  -204.30554  Change =     2.53676
Iteration 3: Average LogP =  -202.36621  Change =     1.93933
Iteration 4: Average LogP =  -201.46590  Change =     0.90031
Iteration 5: Average LogP =  -201.46590  Change =    -0.00001
Estimation converged at iteration 6
Output written to directory current
```

Le fichier s, ainsi calculé sera déposé dans le répertoire : hmm0 .

2) - On fait ensuite l'apprentissage avec la commande ***HRest :***

HRest -I words.mlf -i 100 -S train.scp -H hmm/hmm0/macros -T 1 -M hmm/hmm1 -C config -l s hmm/hmm0/s

On obtient alors le résultat suivant :

```
C:\htk>HRest -C config/hrest.conf -A -l wa -L Labels -M Modeles/HrestSurApp
 -e 0.005 -i 90 -T 1 -m 2 Modeles/HinitSurApp/wa -S Listes/mfcc.app.lst
HRest -C config/hrest.conf -A -l wa -L Labels -M Modeles/HrestSurApp -e 0.00
 90 -T 1 -m 2 Modeles/HinitSurApp/wa -S Listes/mfcc.app.lst
Reestimating HMM Modeles/HinitSurApp/wa . . .
 States   :   2  3  4 (width)
 Mixes  s1:   1  1  1 (  8  )
 Num Using:   0  0  0
 Parm Kind:  MFCC_E_D
 Number of owners = 1
 SegLab   :  wa
 MaxIter  :  90
 Epsilon  :  0.005000
 Updating :  Transitions Means Variances

 - system is PLAIN
3 Examples loaded, Max length = 28, Min length = 27
Ave LogProb at iter 1 = -200.57607 using 3 examples
Ave LogProb at iter 2 = -165.94873 using 3 examples  change =   34.62733
Ave LogProb at iter 3 = -163.15111 using 3 examples  change =    2.79762
Ave LogProb at iter 4 = -161.99404 using 3 examples  change =    1.15707
Ave LogProb at iter 5 = -161.29968 using 3 examples  change =    0.69435
Ave LogProb at iter 6 = -160.68065 using 3 examples  change =    0.61903
Ave LogProb at iter 7 = -160.32179 using 3 examples  change =    0.35886
Ave LogProb at iter 8 = -160.27058 using 3 examples  change =    0.05121
Ave LogProb at iter 9 = -160.26553 using 3 examples  change =    0.00505
Ave LogProb at iter 10 = -160.26434 using 3 examples  change =    0.00119
Estimation converged at iteration 10
```

On répète, après, l'opération pour toutes les unités lexicales étudiées.

3) –La commande : ***HErest :***

HERest -C config -I phones0.mlf -t 250.0 150.0 1000.0 -S train.scp -H hmm/hmm0/macros -H hmm/hmm0/hmmdefs -M hmm/hmm1 monophones0

On crée à cet effet un fichier vide Config et un fichier texte Monophone0 contenant toutes les étiquettes :

Monophone0 :

w

a

h

i

d

T

N

…

On obtient ainsi le résultat :

```
C:\htk>HERest -C config/herest.conf -A -T 1 -d Modeles/HrestSurApp/ -m 2 -M
 Modeles/HerestSurApp -S Listes/mfcc.app.lst -L Labels/ Listes/ListeModeles
HERest -C config/herest.conf -A -T 1 -d Modeles/HrestSurApp/ -m 2 -M Modeles
estSurApp -S Listes/mfcc.app.lst -L Labels/ Listes/ListeModeles
HERest  ML Updating: Transitions Means Variances

 System is PLAIN
21 Logical/21 Physical Models Loaded, VecSize=8
Pruning-Off
 Processing Data: ithnaniTest.mfcc; Label ithnaniTest.lab
 Utterance prob per frame = -1.687774e+000
 Processing Data: aalaa1.mfcc; Label aalaa1.lab
```

Le fichier NewMacros est alors créé dans le répertoire hmm/hmm1

A partir de celui-ci, on crée des fichiers pour chaque étiquette :

Annexe

~o
<STREAMINFO> 1 39
<VECSIZE> 8<NULLD><MFCC_E_D_A><DIAGC>
~h "s"
<BEGINHMM>
<NUMSTATES> 5
<STATE> 2
<MEAN> 39
4.534622e+000 -4.984165e-001 -4.560236e+000 9.344954e-001 -4.169609e-001 -7.946982e 001 -2.169581e-001 5.237545e-003 4.534622e+000 -4.984165e-001 -4.560236e+000 9.344954e-001 -4.169609e-001 -7.946982e-001 -2.169581e-001 5.237545e-003 4.534622e+000 -4.984165e-001 -4.560236e+000 9.344954e-001 -4.169609e-001 -7.946982e 001 -2.169581e-001 5.237545e-003 4.534622e+000 -4.984165e-001 -4.560236e+000 9.344954e-001 -4.169609e-001 -7.946982e-001 -2.169581e-001 5.237545e-003 4.534622e+000 -4.984165e-001 -4.560236e+000 9.344954e-001 -4.169609e-001 -7.946982e-001 -2.169581e-001
<VARIANCE> 39
4.475359e+000 1.892094e+000 2.782526e+000 1.088175e-002 8.059898e-001 2.108334e-001 5.063429e-001 5.233153e-004 4.475359e+000 1.892094e+000 2.782526e+000 1.088175e-002 8.059898e-001 2.108334e-001 5.063429e-001 5.233153e-004 4.475359e+000 1.892094e+000 2.782526e+000 1.088175e-002 8.059898e-001 2.108334e-001 5.063429e-001 5.233153e-004 4.475359e+000 1.892094e+000 2.782526e+000 1.088175e-002 8.059898e-001 2.108334e-001 5.063429e-001 5.233153e-004 4.475359e+000 1.892094e+000 2.782526e+000 1.088175e-002 8.059898e-001 2.108334e-001 5.063429e-001
<GCONST> 9.061142e+000
<STATE> 3
<MEAN> 39
4.534622e+000 -4.984165e-001 -4.560236e+000 9.344954e-001 -4.169609e-001 -7.946982e 001 -2.169581e-001 5.237545e-003 4.534622e+000 -4.984165e-001 -4.560236e+000 9.344954e-001 -4.169609e-001 -7.946982e-001 -2.169581e-001 5.237545e-003 4.534622e+000 -4.984165e-001 -4.560236e+000 9.344954e-001 -4.169609e-001 -7.946982e 001 -2.169581e-001 5.237545e-003 4.534622e+000 -4.984165e-001 -4.560236e+000 9.344954e-001 -4.169609e-001 -7.946982e-001 -2.169581e-001 5.237545e-003 4.534622e+000 -4.984165e-001 -4.560236e+000 9.344954e-001 -4.169609e-001 -7.946982e-001 -2.169581e-001
<VARIANCE> 39
4.475359e+000 1.892094e+000 2.782526e+000 1.088175e-002 8.059898e-001 2.108334e-001 5.063429e-001 5.233153e-004 4.475359e+000 1.892094e+000 2.782526e+000 1.088175e-002 8.059898e-001 2.108334e-001 5.063429e-001 5.233153e-004 4.475359e+000 1.892094e+000 2.782526e+000 1.088175e-002 8.059898e-001

```
   2.108334e-001 5.063429e-001 5.233153e-004 4.475359e+000 1.892094e+000 2.782526e+000 1.088175e-002
   8.059898e-001 2.108334e-001 5.063429e-001 5.233153e-004 4.475359e+000 1.892094e+000 2.782526e+000
   1.088175e-002 8.059898e-001 2.108334e-001 5.063429e-001
<GCONST> -5.940096e-001
<STATE> 4
<MEAN> 39
   4.534622e+000 -4.984165e-001 -4.560236e+000 9.344954e-001 -4.169609e-001 -7.946982e 001 -2.169581e-
   001 5.237545e-003  4.534622e+000 -4.984165e-001 -4.560236e+000 9.344954e-001 -4.169609e-001 -
   7.946982e-001 -2.169581e-001 5.237545e-003 4.534622e+000 -4.984165e-001 -4.560236e+000 9.344954e-001
   -4.169609e-001 -7.946982e 001 -2.169581e-001 5.237545e-003 4.534622e+000 -4.984165e-001 -
   4.560236e+000 9.344954e-001 -4.169609e-001 -7.946982e-001 -2.169581e-001 5.237545e-003 4.534622e+000
   -4.984165e-001 -4.560236e+000 9.344954e-001 -4.169609e-001 -7.946982e-001 -2.169581e-001
<VARIANCE> 39
   4.475359e+000 1.892094e+000 2.782526e+000 1.088175e-002 8.059898e-001 2.108334e-001 5.063429e-001
   5.233153e-004  4.475359e+000 1.892094e+000 2.782526e+000 1.088175e-002 8.059898e-001 2.108334e-001
   5.063429e-001 5.233153e-004 4.475359e+000 1.892094e+000 2.782526e+000 1.088175e-002 8.059898e-001
   2.108334e-001 5.063429e-001 5.233153e-004 4.475359e+000 1.892094e+000 2.782526e+000 1.088175e-002
   8.059898e-001 2.108334e-001 5.063429e-001 5.233153e-004 4.475359e+000 1.892094e+000 2.782526e+000
   1.088175e-002 8.059898e-001 2.108334e-001 5.063429e-001
<GCONST> 3.333740e+000
<TRANSP> 5
   0.000000e+000 1.000000e+000 0.000000e+000 0.000000e+000 0.000000e+000
   0.000000e+000 8.198466e-001 1.801535e-001 0.000000e+000 0.000000e+000
   0.000000e+000 0.000000e+000 9.210444e-001 7.895560e-002 0.000000e+000
   0.000000e+000 0.000000e+000 0.000000e+000 8.722638e-001 1.277362e-001
   0.000000e+000 0.000000e+000 0.000000e+000 0.000000e+000 0.000000e+000
<ENDHMM>
```

4.2. La reconnaissance

Pour le reconnaissance, on aura besoin de la fonction *HParse* et du fichier texte gram. Le fichier gram contient :

```
$chiffre= wahid| iTna:ni| arbaHa| xamsa|sil;
(SENT-START < $chiffre > SENT-END)
```

On exécute la ligne suivante :

HParse gram wdnet

Le résultat est mis dans un fichier wdnet, qui contiendra les infos suivantes :

Annexe

```
VERSION=1.0
N=14   L=20
I=0    W=SENT-END
I=1    W=sil
I=2    W=!NULL
I=3    W=khamsa
I=4    W=arbaaa
I=5    W=iTnani
I=6    W=wahid
I=7   W=sil
I=8   W=SENT-START
I=9   W=!NULL
I=10   W=!NULL
J=0    S=1    E=0
J=1    S=3    E=1
J=2    S=10   E=2
J=3    S=2    E=3
J=4    S=4    E=3
J=5    S=5    E=3
J=6    S=6    E=3
J=7    S=7    E=3
J=8    S=8    E=3
J=9    S=9    E=3
J=10   S=10   E=4
J=11   S=10   E=5
J=12   S=10   E=6
J=13   S=10   E=7
J=14   S=10   E=8
J=15   S=10   E=9
J=16   S=3    E=10
J=17   S=11   E=10
J=18   S=13   E=11
J=19   S=0    E=12
```

La reconnaissance est ensuite effectuée sur tous les fichiers. Pour cela, on appelle la commande *Hvite* qui utilise un modèle de mots et un dictionnaire (correspondance entre les labels et les mots à trouver au niveau de la reconnaissance).

Le fichier dictionnaire :

```
SENT-START []
SENT-END []
wahid wa hi d
iTnani i T na ni
arbaaa ar ba aah
khamsa kha m sah
sil sil
```

Ensuite, on fait le test de reconnaissance. On considère un enregistrement test appelé Test.wav sur lequel on applique toutes les étapes précédentes.

Après, on exécute la commande *HVite* :

HVite -H hmm/hmm15/macros -H hmm/hmm15/hmmdefs -S train.scp -l '*' -i recout.mlf -w wdnet -p 0.0 -s 5.0 dict monophone1

Un fichier est alors généré, avec le mot iTna:ni reconnu, le départ et la fin du mot dans le fichier, ainsi qu'un score.

Test2.rec :

```
0 6500000 sil 226.876945
6500000 18500000 iTnani -432.831009
18500000 22000000 sil -324.636047
```

5. L'évaluation

L'évaluation des résultats se fait avec la fonction *HResult*.

HResults -I words.mlf tiedlist recout.mlf

Annexe

```
D:\DOCTORAT\hmm\HTK\essaiArabe\BenO-Bahi-PourJeudi-16Avr\2etape>HRes
====================== HTK Results Analysis =======================
  Date: Tue Apr 21 09:49:55 2015
  Ref : word0.mlf
  Rec : recout0.mlf
------------------------ Overall Results --------------------------
SENT: %Correct=16.00 [H=4, S=21, N=25]
WORD: %Corr=88.08, Acc=84.77 [H=133, D=9, S=9, I=5, N=151]
------------------------ Confusion Matrix -------------------------
       b   f   i   m   n   r   t   a   ʂ   s   s
                                   :   `   y   i
                                               l
                                                   Del [ %c / %e]
   f   0  26   0   0   0   0   0   0   0   0   0    0
   i   0   0  10   1   1   1   2   2   0   1   0    7 [55.6/5.3]
   r   0   1   0   0   0  22   0   0   0   0   0    2 [95.7/0.7]
  ʂ`   0   0   0   0   0   0   0   0  25   0   0    0
 sil   0   0   0   0   0   0   0   0   0   0  50    0
Ins    2   0   0   0   0   0   2   1   0   0   0
=====================================================================
```

%Correct Taux de reconnaissance = **16.00%**

Acc Accuracy (précision) = **84.77%**

H Nombre d'éléments bien classés	H	= 133
D Nombre d'éléments supprimés	D	= 9
S Nombre d'éléments substitués	S	= 9
I Nombre d'éléments insérés	I	= 5
N Nombre total d'éléments	N	= 151