Détection de mots clés dans un flux de parole

Yassine Ben Ayed

Ecole Nationale Supérieure des Télécommunications
ENST

Détection de mots clés dans un flux de parole

THÈSE

présentée et soutenue publiquement le 23 décembre 2003

pour l'obtention du

Doctorat de l'Ecole Nationale Supérieure des Télécommunications

(Spécialité : Signal et Images)

par

Yassine BEN AYED

Composition du jury

Président :	René Carré Directeur de recherche ENST, LTCI-CNRS
Rapporteurs :	Régine André-Obrecht Professeur Université Paul Sabatier Renato De Mori Professeur Université d'Avignon
Examinateurs :	Gérard Chollet Directeur de recherche ENST, LTCI-CNRS Jean-Paul Haton Professeur Université Henri Poincaré-Nancy 1 Dominique Fohr Chargé de recherche LORIA-CNRS Denis Jouvet Ingénieur France Télécom R&D-DIH-IPS

Laboratoire Lorrain de Recherche en Informatique et ses Applications

Résumé

La reconnaissance automatique de la parole suscite actuellement un grand intérêt. En particulier, la détection de mots clés qui constitue une branche importante de l'interaction homme-machine vu le besoin de communiquer avec nos machines d'une façon naturelle et directe en utilisant la parole spontanée. Cette technique consiste à détecter dans une phrase prononcée, les mots clés caractérisant l'application et de rejeter les mots hors-vocabulaire ainsi que les hésitations, les faux départs etc.

Le travail que nous présentons dans ce manuscrit s'inscrit dans le cadre de la détection de mots clés dans un flux de parole. Tout d'abord, nous proposons de nouveaux modèles "poubelles" fondés sur la modélisation des mots hors-vocabulaire. Puis nous introduisons la reconnaissance à base de boucle de phonèmes, dans laquelle nous appliquons différentes fonctions de récompense favorisant la reconnaissance des mots clés.

Ensuite nous proposons l'utilisation des mesures de confiance afin de pouvoir prendre la décision de rejeter ou d'accepter un mot clé hypothèse. Les différentes mesures de confiance proposées sont basées sur la probabilité d'observation acoustique locale. En premier lieu, nous utilisons les moyennes arithmétique, géométrique et harmonique comme mesures de confiance pour chaque mot clé. En second lieu, nous proposons de calculer la mesure de confiance en se basant sur la méthode à base de boucle de phonèmes.

Enfin nous présentons le problème de détection comme un problème de classification où chaque mot clé peut appartenir à deux classes différentes, à savoir "correct" et "incorrect". Cette classification est réalisée en utilisant des Support Vector Machines (SVM) qui constituent une nouvelle technique d'apprentissage statistique. Chaque mot clé reconnu est représenté par un vecteur caractéristique qui constitue l'entrée du classifieur SVM. Pour déterminer ce vecteur, nous utilisons la probabilité d'observation acoustique locale et nous introduisons ensuite la durée de chaque état. Afin d'améliorer les performances, nous proposons des approches hybrides combinant les modèles poubelles avec mesure de confiance et mesure de confiance avec SVM.

Pour tester les performances de l'ensemble de ces modèles nous utilisons la base de données française SPEECHDAT. L'évaluation de tous les résultats a été réalisée en se basant sur les courbes ROC et les courbes rappel/précision. Les meilleurs résultats ont été obtenus par les méthodes basées sur l'utilisation des SVM. Les méthodes hybrides nous ont permis aussi de réaliser de bonnes performances.

Mots-clés: Reconnaissance de la parole, détection de mots clés, modèle poubelle, mesure de confiance, support vector machines.

Abstract

The automatic speech recognition currently arouses a great interest. In particular, the keyword detection which constitutes a significant branch of the human-machine interaction and which can help us to communicate with our machines in a natural and direct way by using spontaneous speech. This technique consists in detecting in a pronounced sentence, the keywords characterizing the application and in rejecting out-of-vocabulary words as well as hesitations, false starts etc.

The work presented in this thesis report deals with problem of keyword detection in a speech flow. First of all, we propose new garbage models founded on the modelling of the out-of-vocabulary words. Second, we introduce the recognition based on loop of phonemes, in which we apply various reward functions supporting keyword recognition.

Then we propose to use confidence measures in order to make the decision of rejection or acceptance of a given keyword. The various confidence measures used are based on the probability of the local acoustic observation. First, we use these probabilities to calculate the arithmetic, geometric and harmonic means as confidence measures for each keyword. Second, we propose some others confidence measures based on the loop of phonemes recognition method.

Finally we present the problem of detection as a classification problem where each keyword can belong to two different classes, namely "correct" and "incorrect". This classification is carried out by using Support Vector Machines (SVM) which constitute a new technique of statistical training. Each recognized keyword is represented by a characteristic vector which constitutes the entry of the SVM classifier. To determine this vector, we use the probability of the local acoustic observation and then we introduce the duration of each state. In order to improve performances, we propose hybrid approaches combining the garbage models with the confidence measures and the confidence measures with the SVM.

To test our models we use the French data base SPEECHDAT. The evaluation of all results was carried out using the ROC and the recall/precision curves. The best results were obtained by the SVM based methods. The hybrid methods also enabled us to get good performances.

Keywords: Speech recognition, keyword detection, garbage model, confidence measure, support vector machines.

Remerciements

Mes premiers remerciements s'adressent à mes rapporteurs, Madame Régine André-Obrecht et Monsieur Renato De Mori, qui ont bien voulu accepter d'évaluer le présent travail et ce malgré toutes les responsabilités qu'ils assument. Je les remercie pour le temps qu'ils consacreront à la lecture de ce mémoire et je souhaite qu'ils y trouvent entière satisfaction.

Je remercie Monsieur René Carré pour m'avoir fait l'honneur de présider mon jury de thèse et pour l'intérêt qu'il a porté à mon travail.

Je tiens également à remercier Monsieur Denis Jouvet d'avoir accepté de participer au Jury. Je le remercie particulièrement pour sa lecture attentive et ses remarques constructives sur le manuscrit ainsi que pour ses nombreux et précieux conseils.

Mes vifs remerciements s'adressent à Monsieur Gérard Chollet, mon co-directeur de thèse à l'ENST qui m'a fait l'honneur d'encadrer ma thèse pendant ces années. J'ai notamment beaucoup apprécié ses critiques constructives concernant mon travail et le recul dont il a su faire preuve.

J'ai été extrêmement heureux d'avoir Monsieur Jean-Paul Haton comme co-directeur de thèse au LORIA. Je le remercie de m'avoir consacré autant de temps et de m'avoir fait profiter de son expérience et de son savoir. Je n'oublierais guère son soutien aux moments les plus difficiles durant ces années. Merci beaucoup !

Un grand merci à Monsieur Dominique Fohr pour avoir été présent aux instants importants de prise de décision ainsi que pour les discussions riches et vives que nous avons eues durant ces trois dernières anneés. Je le remercie aussi pour ses remarques et critiques qui ont contribué à l'élaboration de ce travail.

Merci à tous les membres de l'équipe parole au sein duquel ce travail fut effectué. Leur compétence et qualité humaine ont grandement facilité mon travail.

Un merci particulier à Monsieur Yann Guermeur pour avoir accepté de lire et corriger ce document. L'aide qu'il m'a apportée fut précieuse.

Mes derniers remerciements mais non les moindres s'adressent à ma femme Salma et à ma famille qui m'ont apporté le soutien dont j'avais besoin pour mener à bien ce travail.

À mon fils Ahmad et à ma femme Salma.
À mes parents, à mes frères et à ma soeur.

Table des matières

Table des figures

Liste des tableaux

Introduction générale

Dialoguer oralement avec les machines était un rêve humain dès l'apparition des premiers ordinateurs. Ce rêve a commencé d'être un sujet de recherche vers les années soixante-dix, notamment avec le premier projet ARPA (Advanced Research Projects Agency) de "compréhension de la parole" qui a été lancé en 1971 [Klatt, 1977]. Le but ultime de ce projet était de pouvoir dialoguer librement en langage naturel avec nos machines. Cependant, c'était très optimiste d'imaginer qu'on puisse réaliser de tels systèmes en une dizaine d'années. En effet, aujourd'hui et après plus de 30 ans, nous n'avons pas encore atteint les performances humaines même au niveau de la première phase de ce projet à savoir la reconnaissance automatique de la parole.

Ce sujet, qui a suscité un grand intérêt des chercheurs du domaine, commence à avoir un impact dans la vie courante. Les progrès réalisés dans le domaine de la RAP nous permettent aujourd'hui de reconnaître la parole, de réaliser des systèmes de dictée vocale et d'élaborer quelques applications interactives guidées très simples à vocabulaire limité. Cependant nous sommes encore loin de réaliser des systèmes de dialogue homme-machine très performants.

À petits pas vers la réalisation d'applications interactives, utilisant la parole spontanée et permettant de comprendre et de répondre aux utilisateurs, un autre domaine a vu le jour depuis une dizaine d'années, il s'agit de la détection de mots clés dans un flux de parole.

La détection de mots clés permet d'éviter en partie la complexité et les défauts des systèmes de reconnaissance automatique de la parole continue, en détectant seulement un petit ensemble de mots clés utiles pour comprendre la phrase prononcée. En effet, dans certaines applications interactives, la prononciation d'un mot clé suffit pour déclencher la réponse appropriée. L'étape de détection s'avère alors plus intéressante qu'une reconnaissance complète de la phrase car elle permet au système de réagir immédiatement à la demande de l'utilisateur [Gorin *et al.*, 1997].

Comme son nom l'indique, cette technique consiste à détecter les mots clés les plus caractéristiques du domaine de l'application considérée, dans l'ensemble des productions vocales, y compris les hésitations, les faux départs etc. Il s'agit donc de traiter la parole spontanée avec toutes ses subtilités, de rejeter les mots hors-vocabulaire, considérés inutiles, et de ne conserver que les mots qui ont une importance pour l'interprétation sémantique de la phrase prononcée, les mots clés de l'application.

Grâce à cette approche, les utilisateurs peu conscients des contraintes des systèmes de reconnaissance ou s'exprimant d'une manière spontanée sans faire attention aux structures grammaticales de leurs phrases, effraient moins les concepteurs des applications interactives. Cette approche a suscité beaucoup d'intérêt et a été utilisée dans de nombreuses applications, entre autres la classification et l'indexation des documents parlés, l'extraction de l'information, l'ac-

cès aux bases de données, le routage et la surveillance des appels téléphoniques [Foote *et al.*, 1995] [Gelin, 1997].

Le travail que nous présentons s'inscrit dans le cadre de la détection de mots clés dans un flux de parole. Tout d'abord, nous étudions plusieurs techniques de détection et nous proposons de nouvelles méthodes basées principalement sur les modèles poubelles et la mesure de confiance, ainsi que des approches hybrides combinant ces deux notions.

En second lieu, nous présentons le système de détection de mots clés, comme un problème de classification, dans lequel chaque mot clé doit être classé comme un mot correct (C) s'il correspond à un mot clé correctement reconnu ou comme un mot incorrect (I) s'il correspond à une fausse acceptation (insertion ou fausse reconnaissance). Ainsi, une mauvaise reconnaissance dans un tel système correspond à une fausse acceptation (I est classé comme C) ou à un faux rejet (C classé comme I). Dans notre travail, nous étudions le classifieur de type SVM (en anglais, Supports Vector Machines) afin d'améliorer les performances de la détection.

Ce mémoire se compose de deux parties : la première partie (chapitres 1, 2 et 3) fait l'état de la recherche dans le domaine. La deuxième partie (chapitres 4, 5, 6 et 7) présente notre contribution à la détection de mots clés.

Dans le premier chapitre, nous décrivons tout d'abord les caractéristiques du signal de la parole, puis nous évoquons les difficultés liées à la reconnaissance vocale. Ensuite nous exposons les différentes étapes utiles à l'extraction des paramètres du signal. Puis nous présentons l'approche phonétique et l'approche probabiliste dans le domaine de la reconnaissance automatique de la parole. Enfin nous détaillons l'application des HMMs pour la reconnaissance automatique de la parole, en résolvant les trois problèmes fondamentaux des HMMs à savoir l'évaluation, le décodage et l'apprentissage.

Nous proposons dans le deuxième chapitre un état de l'art de l'utilisation du modèle poubelle et de la mesure de confiance dans le cadre de notre application. En premier lieu nous exposons les diverses méthodes de construction du modèle poubelle. Ensuite nous détaillons différentes techniques à base de mesure de confiance qui ont été proposées afin de détecter les mots clés. Enfin nous terminons par les applications de détection de mots clés.

Le troisième chapitre est consacré à la classification statistique, dans la première partie, nous décrivons tout d'abord le simple Perceptron et son principe d'apprentissage pour détailler ensuite le cas le plus général du Perceptron multi-couches avec son algorithme de rétro-propagation de l'erreur. Dans la deuxième partie, nous évoquons la théorie d'apprentissage de Vapnik des SVM ainsi que le principe de la minimisation du risque structurel. Ensuite nous introduisons les trois formules mathématiques sur lesquelles sont basés les SVM et nous présentons leurs applications pour la recherche d'un hyperplan optimal permettant de classer des données. Enfin nous terminons par une description des principes des SVM. Dans la troisième partie, nous introduisons les SVM multi-classes et nous présentons les solutions proposées à base des SVM binaires pour décrire à la fin, la résolution du problème multi-classes par une méthode d'optimisation.

Le quatrième chapitre décrit le système de reconnaissance de la parole ESPERE. Ce système se fonde sur les modèles de Markov cachés. Puis nous présentons la base de données utilisée durant toute cette étude. Ensuite nous exposons les différents problèmes de choix des critères

d'évaluation ainsi que quelques mesures d'évaluation utiles pour la détection de mots clés. Nous présentons enfin les courbes ROC et les courbes rappel/précision qui seront utilisées comme moyen d'évaluation des méthodes élaborées tout au long de notre travail.

Le cinquième chapitre permet de présenter différentes approches pour la détection de mots clés se fondant sur la notion de modèle poubelle. Nous évoquons alors différentes techniques novatrices pour mieux résoudre le problème de la détection des mots hors-vocabulaire. En premier lieu, nous commençons par un modèle poubelle sans apprentissage. Ensuite nous proposons une nouvelle méthode à base d'un modèle poubelle appris en utilisant un GMM. Enfin nous abordons une approche hybride combinant les deux premiers modèles. En second lieu, nous proposons d'utiliser une reconnaissance à base de boucle de phonèmes, dans laquelle nous appliquons différentes fonctions de récompense qui favorisent le passage entre les phonèmes constituants un mot clé. Une comparaison de l'ensemble de ces approches est présentée à la fin du chapitre.

Le sixième chapitre s'intéresse aux mesures de confiance. Ces mesures permettent de décider s'il faut rejeter ou accepter un mot clé reconnu. Nous définissons alors différentes mesures de confiance. La base fondamentale de ces mesures est la probabilité d'observation acoustique locale qu'on utilise pour calculer les moyennes arithmétique, géométrique et harmonique comme premières mesures à tester. Ensuite nous proposons la mesure de confiance à base de boucle de phonèmes, dans laquelle nous calculons en premier lieu, un rapport de vraisemblance et en second lieu, une distance de vraisemblance. Ces mesures utilisent le score du mot clé reconnu et son image qui correspond à un ensemble de phonèmes reconnus en se basant sur la méthode de boucle de phonèmes. Enfin nous combinons la notion de modèle poubelle avec la mesure de confiance au sein d'un même système hybride pour la détection de mots clés.

Le septième chapitre introduit les SVM dans le domaine de la détection de mots clés. Nous montrons alors que le problème de détection des mots clés peut être considéré comme un problème de classification, où chaque mot clé peut appartenir à deux classes différentes à savoir correct et incorrect. Le vecteur d'entrée des SVM est un vecteur caractéristique du mot clé considéré, il contient ainsi les caractéristiques obtenues par un alignement de la sortie de notre système de reconnaissance avec les modèles HMM des phonèmes asscociés. Nous proposons différentes techniques pour la représentation vectorielle d'un mot clé. En premier lieu, nous utilisons les mesures de confiance baseés sur les moyennes. En second lieu, nous employons la probabilité d'observation acoustique locale de chaque phonème. Puis nous introduisons un vecteur caractéristique basé sur le nombre de trames dans chaque état. Enfin, nous testons toutes ces différentes méthodes en utilisant un Perceptron multi-couches et un SVM binaire. La meilleure représentation est testée aussi par le SVM multi-classes. Une étude comparative de ces différents modèles termine ce chapitre.

Au terme d'une conclusion, nous esquissons nos perspectives de recherche.

Chapitre 1

Reconnaissance Automatique de la Parole (RAP)

1.1 Introduction

La reconnaissance automatique de la parole est un domaine d'étude actif depuis le début des années 50. Il est clair qu'un outil de reconnaissance de la parole efficace facilitera l'interaction entre les hommes et les machines. Les applications possibles associées à un tel outil sont nombreuses et sont amenées à connaître un grand essor. La plupart des applications en reconnaissance de la parole peuvent être regroupées en quatre catégories : commande et contrôle, accès à des bases de données ou recherche d'informations, dictée vocale et transcription automatique de la parole.

La technologie la plus utilisée depuis plus de 20 ans est basée sur des modèles statistiques : les modèles de Markov cachés (en anglais Hidden Markov Models : HMM) capables de modéliser simultanément les caractéristiques fréquentielles et temporelles du signal de parole. Depuis l'introduction de ces modèles, de nombreux progrès ont été réalisés dans le domaine de la reconnaissance de la parole. Néanmoins, les performances obtenues sont encore largement inférieures à celles des êtres humains, même si les progrès réalisés en moins de 50 ans sont énormes.

Nous présentons dans le paragraphe suivant le signal de parole en analysant ses caractéristiques et ses composantes. Nous exposons dans le paragraphe 1.3, les difficultés liées à la reconnaissance vocale. Ensuite nous détaillons les différentes étapes utiles à l'extraction des paramètres du signal et nous présentons l'approche phonétique. Puis, nous montrons l'approche probabiliste et surtout celle à base des modèles de Markov cachés, qui est la plus utilisée de nos jours dans le domaine de la reconnaissance automatique de la parole. Enfin nous décrivons l'application des HMMs pour la reconnaissance automatique de la parole.

1.2 Le signal de parole

Le signal de parole appartient à la classe des signaux acoustiques produits par des vibrations des couches d'air. Les fluctuations de la pression de l'air produisent des variations de ce signal, en fonction du temps, qui peuvent être enregistrées de façon analogique ou digitale. Ceci constitue une représentation élémentaire du signal de parole [Boite et Kunt, 1987]. Ce signal est le résultat du passage du flux laryngé (l'air qui est passé par nos poumons à travers nos cordes vocales) à

travers le conduit vocal. Le signal de parole est donc une onde acoustique qui se propage dans un milieu donné (en général l'air) et qui est le résultat de la modulation par le conduit vocal d'une onde d'excitation.

Les phonèmes sont les éléments les plus brefs qui permettent de distinguer différents mots [Calliope, 1989]. Un mot peut être considéré comme un ensemble de phonèmes. Plusieurs formes du conduit vocal peuvent produire le même phonème. La forme que le conduit vocal prend pour la production d'un phonème, dans un contexte donné, est assez variable et surtout dépendante de ce contexte. De ce fait, nous remarquons que les formes acoustiques associées à un phonème déterminé sont variables. Cette variabilité est double : d'une part une variabilité du contenu acoustique du phonème (dû à la variété des formes du conduit vocal donnant lieu à ce phonème) et d'autre part une variabilité de la durée du phonème. Cette dernière variabilité résulte essentiellement du fait que le système articulatoire met en jeu des constantes mécaniques qui contrôlent les mouvements musculaires.

L'étude des mécanismes les plus répandus chez l'homme, ont montré que l'analyse des mouvements des différents paramètres articulatoires peut être représentée par un modèle de type source-filtre. Deux filtres placés en cascade constituent le modèle de chaque paramètre articulatoire. Le premier est un filtre intégrateur, il est associé au système mécanique de la chaîne de production de la parole. Le deuxième est un intégrateur d'ordre 2 et il correspond au système physiologique. La source, quant à elle, est représentée par une séquence d'impulsions [George *et al.*, 1994].

La parole est un signal quasi stationnaire. Elle est formée de phonèmes et de transitions entre ces phonèmes (bien que le phonème ne soit pas une entité acoustiquement fixe). Plusieurs types de phonèmes existent : les voyelles, les consonnes fricatives et les consonnes plosives, les nasales et les liquides. Les voyelles sont des phonèmes voisés (l'excitation se fait par la glotte), leur production se fait généralement avec un conduit vocal relativement ouvert et en absence de constrictions et leur prononciation peut être isolée et durable dans le temps. Les consonnes se caractérisent par une constriction (ou occlusion) dans le conduit vocal lors de leur production. Elles peuvent être voisées ou non voisées. Dans le cas des fricatives, la contriction génère un bruit local qui peut persister dans le temps et qui excite une partie du conduit vocal. Contrairement aux voyelles et aux fricatives, les plosives ne durent pas dans le temps, elles sont produites par un relâchement rapide d'une occlusion du conduit vocal, qui produit une perturbation locale se traduisant acoustiquement par un bruit impulsif de faible durée. Des transitions lient les phonèmes adjacents. D'une façon très simplifiée, les transitions acoustiques correspondent à des transitions dans l'appareil de production de l'état correspondant au premier phonème à l'état correspondant au suivant [Koreman *et al.*, 1999].

En conclusion, la parole est un signal quasi stationnaire formé de parties stationnaires et de transitions entre ces différentes parties. C'est un signal non déterministe, dans le sens où deux réalisations d'un même mot auront nécessairement deux formes acoustiques différentes, même si elles sont produites par un même locuteur. Il faut distinguer deux types de variations : la variation acoustique et la variation temporelle non linéaire du signal.

1.3 Les difficultés liées au signal de parole

Le problème de la reconnaissance de la parole réside essentiellement dans la spécifité du signal vocal. Ce signal possède une très grande variabilité. Une même personne ne prononce jamais un mot deux fois de façon identique. La vitesse d'élocution peut varier, la durée du signal est alors modifiée. Toute altération de l'appareil phonatoire peut modifier la qualité de l'émission (exemple : rhume, fatigue,...). De plus, la diction évolue dans le temps. La voix est modifiée au cours des étapes de la vie d'un être humain (enfance, adolescence, âge adulte,...). La variabilité interlocuteur est encore plus évidente. La hauteur de la voix, l'intonation, l'accent diffèrent selon le sexe, l'origine sociale, régionale ou nationale [Haton *et al.*, 1991]. Ainsi la parole est un moyen de communication où de nombreux éléments entrent en jeu, tels que le lieu, l'émotion du locuteur, la relation qui s'établit entre les locuteurs (stressante ou amicale). Ces facteurs influencent la forme et le contenu du message. L'acoustique du milieu (milieu protégé ou environnement bruité), la qualité du microphone ou de la ligne téléphonique, les bruits de la bouche, les hésitations, les mots hors-vocabulaire sont autant d'interférences supplémentaires sur le signal de parole que le système de reconnaissance doit compenser.

L'aspect continu du signal de parole complique encore la tâche de reconnaissance. En effet, lorsqu'on écoute parler une personne, on perçoit une suite de mots, alors que l'analyse du signal vocal ne permet de déceler aucun séparateur. Le même problème de segmentation se retrouve à l'intérieur du mot lui même. Celui-ci est perçu comme une suite de sons élémentaires, les phonèmes. L'analyse du signal ne permet pas aussi de découper en segments distincts le signal acoustique afin d'identifier les différents phonèmes qui le composent.

1.4 Extraction des paramètres

L'objectif de cette phase de reconnaissance est d'extraire des coefficients représentatifs du signal de parole. Ces coefficients sont calculés à intervalles temporels réguliers. En simplifiant les choses, le signal de parole est transformé en une série de vecteurs de coefficients, ces coefficients doivent représenter au mieux ce qu'ils sont censés modéliser et doivent extraire le maximum d'informations utiles pour la reconnaissance.

Parmi les coefficients les plus utilisés et qui représentent au mieux le signal de la parole en reconnaissance de la parole, nous trouvons les coefficients cepstraux, appelés également cepstres. Les deux méthodes les plus connues pour l'extraction de ces cepstres sont : l'analyse spectrale et l'analyse paramétrique. Pour l'analyse spectrale (par exemple, Mel-Scale Frequency Cepstral Coefficients (MFCC)) comme pour l'analyse paramétrique (par exemple, le codage prédictif linéaire (LPC)), le signal de parole est transformé en une série de vecteurs calculés pour chaque trame.

Il existe d'autres types de coefficients qui sont surtout utilisés dans les milieux bruités, nous citons par exemple les coefficients PLP (Perceptual Linear Predictive). Ces coefficients permettent d'estimer les paramètres d'un filtre auto-régressif en modélisant au mieux le spectre auditif [Furui, 1981]. Il existe plusieurs techniques permettant l'amélioration de la qualité de ces coefficients, nous trouvons par exemple : l'analyse discriminante linéaire (LDA), l'analyse discriminante non linéaire (NLDA), etc. Pour plus de détails sur les différentes méthodes d'extraction de ces coefficients, nous invitons le lecteur à consulter par exemple [Aubert *et al.*, 1993] [Fontaine *et al.*, 1997].

Ces coefficients jouent un rôle capital dans les approches utilisées pour la reconnaissance de la parole. En effet, ces paramètres qui modélisent le signal seront fournis au système de reconnaissance pour l'estimation de la probabilité $P(séquence|message)$. Dans notre travail, étant donné que nous ne nous intéressons qu'au milieu non bruité, nous nous sommes limités à l'utilisation des coefficients MFCC. Ces paramètres ont montré une bonne représentation des aspects perceptuels du spectre de parole [Davis et Mermelstein, 1980].

1.5 Les coefficients MFCC

Dans le cadre d'une application de reconnaissance de la parole, seule l'estimation de l'enveloppe spectrale est nécessaire [Deroo, 1998]. L'extraction de coefficients MFCC est basée sur l'analyse par banc de filtres qui consiste à filtrer le signal par un ensemble de filtres passe-bande. L'énergie en sortie de chaque filtre est attribuée à sa fréquence centrale. Pour simuler le fonctionnement du système auditif humain, les fréquences centrales sont réparties uniformément sur une échelle perceptive. Plus la fréquence centrale d'un filtre est élevée, plus sa bande passante est large. Cela permet d'augmenter la résolution dans les basses fréquences, zone qui contient le plus d'information utile dans le signal de parole. Les échelles perceptives les plus utilisées sont l'échelle Mel [1] ou l'échelle Bark [2] . Du point de vue performance des systèmes de reconnaissance de la parole, ces deux échelles sont quasiment identiques. Dans nos expériences, nous avons fait le choix d'utiliser l'échelle Mel.

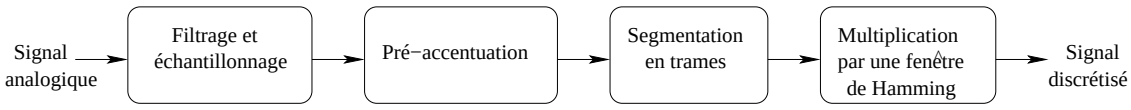

FIG. 1.1 – Mise en forme du signal

Le nombre de filtres utilisés dans une telle analyse est choisi de manière empirique : Zwicker propose 24 filtres [Zwicker et Feldtkeller, 1981]. De la même manière, on choisit empiriquement le type des filtres optimaux pour la reconnaissance de la parole [Benayed, 1999]. Avant tout calcul, il est nécessaire d'effectuer quelques opérations pour mettre en forme le signal de parole. La figure 1.1 illustre l'ensemble de ces opérations. Le signal est tout d'abord filtré puis échantillonné à une fréquence donnée. Une pré-accentuation est effectuée afin de relever les hautes fréquences, ensuite le signal est segmenté en trames. Chaque trame est constituée d'un nombre fixe N d'échantillons de parole. En général N est fixé de telle manière que chaque trame corresponde à environ 30ms de parole (durée pendant laquelle la parole peut être considérée comme stationnaire). Enfin, le fait de traiter un petit morceau de signal amène des problèmes dans le filtrage (effets de bord). Pour éviter cela, nous utilisons des fenêtres de pondération. Ce sont des fonctions que l'on applique à l'ensemble des échantillons prélevés dans la fenêtre du signal original de façon à diminuer les effets de bords. Parmi les fenêtres les plus courantes, nous pouvons citer la fenêtre de Hamming [3]. En général, les fenêtres successives se recouvrent et elles doivent

[1] Échelle Mel : $Mel(f) = \frac{1000}{Log(2)}(1 + \frac{f}{1000})$, f représente la fréquence

[2] Échelle Bark : $Bark(f) = 6.Arcsinh(\frac{f}{1000})$, f représente la fréquence

[3] Fenêtre de Hamming : $w(n) = 0.54 + 0.46.cos(2\pi\frac{n}{N-1})$. Outre ce type de fenêtre, il existe plusieurs autres types tels que rectangulaire, Hann, Blackman,..La fenêtre de Hamming est un cas particulier de la fenêtre de Hann

avoir une longueur suffisante. En pratique, on prend 256 ou 512 échantillons, avec un recouvrement par exemple de la moitié de la taille c'est-à-dire 128 ou 256 échantillons respectivement. Ce traitement implique une hypothèse importante : *Le signal vocal est supposé stationnaire sur une courte période.*

Après cette mise en forme du signal (commune à la plupart des méthodes d'analyse de la parole), une transformée de Fourier discrète (DFT : Discret Fourier Transform), en particulier FFT (Transformée de Fourier Rapide : *Fast Fourier Transform*), est appliquée pour passer dans le domaine fréquentiel et pour extraire le spectre du signal.

Ensuite le filtrage est effectué en multipliant le spectre obtenu par les gabarits des filtres. Ces filtres sont en général, soit triangulaires soit sinusoïdaux. Dans nos expériences, nous avons choisi d'utiliser des filtres triangulaires répartis sur une échelle Mel.

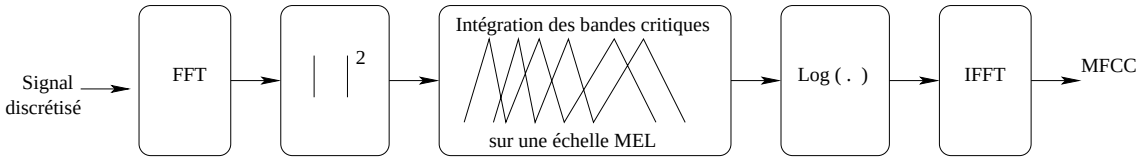

FIG. 1.2 – Calcul des coefficients MFCC (Mel-Scale Frequency Cepstral coefficients)

Le traitement décrit dans le paragraphe précédent permet d'obtenir une estimation de l'enveloppe spectrale (densité spectrale lissée). Il est possible d'utiliser les sorties du banc de filtres comme entrée pour le système de reconnaissance. Cependant, d'autres coefficients dérivés des sorties d'un banc de filtres, sont plus discriminants, plus robustes au bruit ambiant et moins corrélés entre eux. Il s'agit des coefficients cepstraux dérivés des sorties du banc de filtres répartis linéairement sur l'échelle Mel, ce sont les coefficient "MFCC". Le cepstre est défini comme la transformée de Fourier inverse du logarithme de la densité spectrale. Ceci a une interprétation du point de vue de la déconvolution homomorphique : alors que le filtrage linéaire permet de séparer des composantes combinées linéairement, dans le cas de composantes combinées de façon non linéaire (multiplication ou convolution), les méthodes homomorphiques permettent de se ramener au cas linéaire. Pour le signal de parole, la source d'excitation glottique est convoluée avec la réponse impulsionnelle du conduit vocal considéré comme un filtre linéaire :

$$s(t) = e(t) * h(t)$$

où $s(t)$ est le signal de parole, $e(t)$ est la source d'excitation glottique et $h(t)$ est la réponse impulsionnelle du conduit vocal. L'application à l'équation précédente du logarithme du module de la transformée de Fourier donne :

$$Log \mid S(f) \mid = Log \mid E(f) \mid + Log \mid H(f) \mid$$

Par une transformée de Fourier inverse on obtient :

$$s(cef) = e(cef) + h(cef)$$

La dimension du nouveau domaine est compatible avec le temps et s'appelle la *quéfrence* (cef), le nouveau domaine s'appelle le domaine *quéfrentiel*. Un filtrage dans ce domaine s'appelle

liftrage. Ce domaine est intéressant pour faire la séparation du conduit vocal et de la source d'excitation. En effet, si les contributions relevant du conduit vocal et les contributions de la source d'excitation évoluent avec des rapidités différentes dans le temps, alors il est possible de les séparer par application d'une simple fenêtre dans le domaine quéfrentiel (liftrage passe-bas pour le conduit vocal). Le conduit vocal possède une contribution fréquentielle assez lisse qui abouti à un cepstre basse-quéfrence. Réciproquement, la source possède une contribution qui varie très rapidement dans le domaine fréquentiel, son cepstre sera donc dans les hautes quéfrences. Le domaine quéfrentiel est le domaine idéal pour séparer les deux composantes, car non seulement leur contributions sont séparées dans ce domaine, mais aussi elles sont additives [Benayed, 1999]. Les étapes d'une analyse MFCC sont présentées dans la figure 1.2.

1.6 L'approche probabiliste de la RAP

Dans le cadre d'une application de la reconnaissance automatique de la parole, trois facteurs principaux interviennent (figure 1.3) :

- Le locuteur, qui à partir d'un message m (suite de mots) qu'il veut transmettre produit un signal acoustique $s(t)$.
- L'analyseur acoustique, qui à partir du signal $s(t)$ produit une paramétrisation sous forme d' une suite de vecteurs (séquence d'observations o) contenant l'information pertinente pour la reconnaissance.
- Un décodeur dont le rôle consiste à déterminer à partir de la séquence d'observations o, la séquence de mots $\hat{m}$ qui correspond au message m.

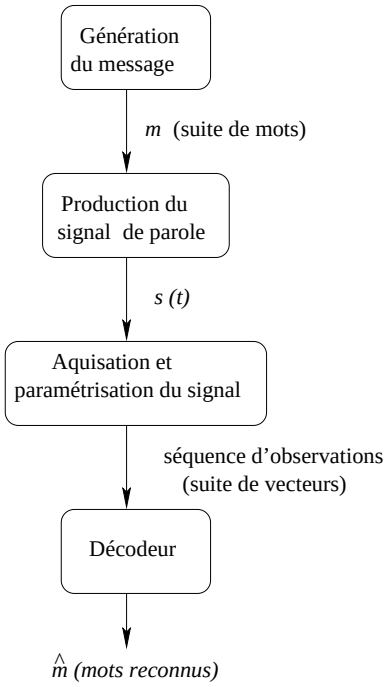

FIG. 1.3 – Les composantes principales du processus de la reconnaissance de la parole.

La reconstitution d'un message m inconnu à partir d'une séquence d'observations o, consiste à retrouver, parmi tous les messages possibles, celui qui selon toute vraisemblance, correspond à o. L'utilisation de la règle de Bayes permet de décomposer la probabilité $P(m|o)$ en deux composantes :

$$\hat{m} = arg_m \, max P(m|o) = arg_m \, max \frac{P(m)P(o|m)}{P(o)}$$

Le dénominateur est constant pour tous les messages possibles, donc on peut l'omettre et $\hat{m}$ sera alors écrit sous la forme suivante :

$$\hat{m} = arg_m \, max P(m)P(o|m)$$

Ainsi, l'étape de reconnaissance consiste à déterminer la suite de mots $\hat{m}$ qui maximise le produit des deux termes $P(m)$ et $P(o|m)$. Le premier terme représente la probabilité *a priori* d'observer la suite de mots m indépendamment du signal. Cette probabilité est déterminée par le modèle de langage. Le deuxième terme indique la probabilité d'observer la séquence de vecteurs acoustiques o sachant une séquence de mots spécifiques m. Cette probabilité est estimée par le modèle acoustique. La qualité d'un tel système de reconnaissance de la parole peut être caractérisée par la précision et la robustesse des deux modèles qui permettent de calculer ces deux termes $P(m)$ et $P(o|m)$.

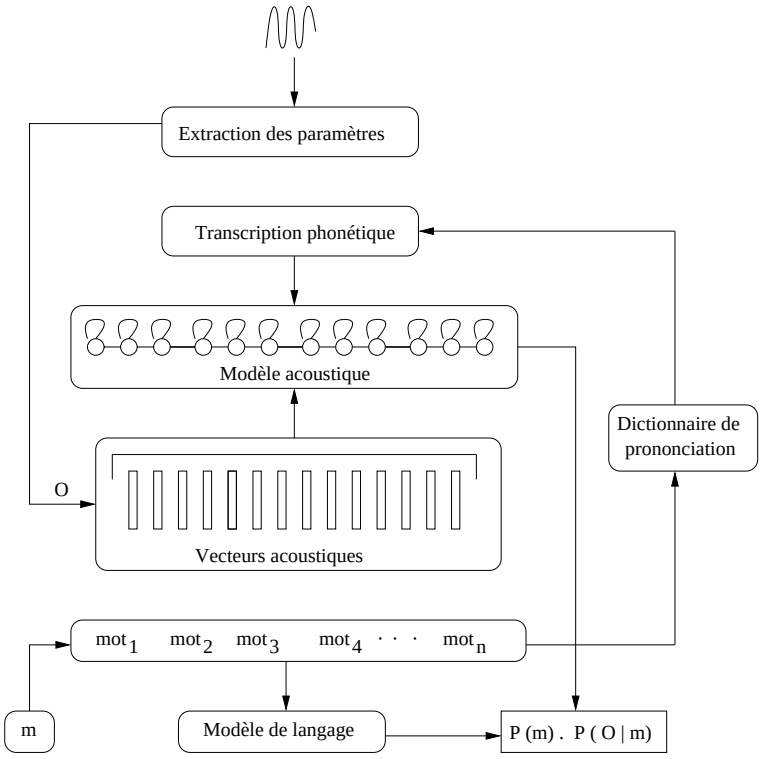

FIG. 1.4 – L'approche probabiliste de la reconnaissance automatique de la parole

L'outil statistique le plus utilisé et le plus performant, de nos jours, pour la modélisation acoustique est fondé sur les modèles de Markov cachés [Bahl *et al.*, 1983] [Rabiner et Juang, 1993]. Les différentes étapes nécessaires à la reconnaissance d'une hypothèse donnée sont illustrées par la figure 1.4. Tout d'abord le signal de parole est subdivisé pour construire une séquence

de vecteurs acoustiques. En utilisant ces vecteurs, le modèle acoustique se charge, à partir des HMMs de phonèmes appris sur un corpus d'apprentissage, de construire la suite des phonèmes hypothèses du signal prononcé. Un seul modèle HMM, représentant l'hypothèse, sera construit par la concaténation de l'ensemble des HMMs de phonèmes qui la compose et génère ainsi la probabilité du signal $s(t)$, ce qui définit la probabilité $P(o|m)$. Ainsi, à partir du dictionnaire des prononciations, la suite des mots hypothèses sera déterminée. Cette suite de mots sera évaluée par le modèle de langage pour estimer la probabilité $P(m)$. En principe, ce processus est répété pour toutes les hypothèses possibles. Le système donne enfin la meilleure hypothèse comme résultat de la reconnaissance.

L'espace de toutes les séquences de mots m augmente très rapidement avec la taille du vocabulaire. Il convient donc de restreindre la recherche à l'espace des séquences de mots les plus plausibles. Les applications récentes en reconnaissance de la parole utilisent souvent des modèles de langage stochastiques. Un modèle de langage est un automate à états finis dont les états représentent les mots du vocabulaire et les arcs les probabilités conditionnelles des transitions. Ces probabilités sont apprises sur des corpus de textes de l'application en question.

Considérons le cas d'une séquence m constituée de la suite des mots m_i avec $i \in \{1, \ldots, L\}$

$$P(m) = P(m_1 m_2 \ldots m_L) = P(m_1) \prod_{i=2}^{L} P(m_i | m_1 m_2 \ldots m_{i-1})$$

Dans la pratique on approxime $P(m_i|m_1 m_2 \ldots m_{i-1})$ par $P(m_i|m_{i-1})$, on parle dans ce cas de modèle de langage bigramme, ou par $P(m_i|m_{i-1}m_{i-2})$ et on parle alors de modèle de langage trigramme. Les modèles bigrammes et trigrammes sont les options les plus courantes, elles impliquent en général peu de restrictions grammaticales, puisque celles-ci portent seulement sur des séquences de 2 ou de 3 mots.

1.7 Application des HMM à la RAP

1.7.1 Introduction

Un problème majeur de la reconnaissance de la parole est de modéliser au mieux des unités représentatives du signal de parole. Il existe en fait deux types de modélisation possibles des propriétés d'un signal donné :

- La modélisation déterministe, qui exploite les propriétés intrinsèques du signal.

- La modélisation statistique, qui caractérise les propriétés statistiques du signal.

Dans ce travail, nous utilisons des modèles statistiques : les modèles de Markov cachés. Un HMM peut être vu comme un ensemble discret de noeuds ou d'états et de transitions ou d'arcs reliant ces états entre eux. Formellement, il peut être défini par l'ensemble des paramètres λ [Rabiner et Juang, 1989] :

$$\lambda = (N, A, B, \pi)$$

- N est le nombre de nœuds ou d'états du modèle.

- $A = \{a_{ij}\} = \{P(q_j|q_i)\}$ est la matrice des probabilités de transition sur l'ensemble des états du modèle. La probabilité de transition est la probabilité de choisir la transition a_{ij} pour accéder à l'état q_j, étant donné un processus à l'état q_i. Pour un HMM d'ordre un, cette probabilité ne dépend que de l'état précédent :

  $$P(q_t = j|q_{t-1} = i, q_{t-2} = k, ...) = P(q_t = j|q_{t-1} = i)$$

  Elle dépend des deux précédents dans le cas d'un HMM d'ordre deux :

  $$P(q_t = j|q_{t-1} = i, q_{t-2} = k, ...) = P(q_t = j|q_{t-1} = i, q_{t-2} = k)$$

  En d'autres termes, l'évolution du système entre deux instants $t-1$ et t ne dépend que de l'état de ce système au temps $t-1$ (ordre 1) ou des deux instants précédents $t-1$ et $t-2$ (ordre deux).

- $B = \{b_j(o_t)\} = \{P(o_t|q_j)\}$ est l'ensemble des probabilités d'émission de l'observation o_t dans l'état q_j. La forme que prend cette distribution détermine le type du HMM. C'est ainsi qu'on parle de HMMs discrets, semi-continus, continus, etc. Pour plus d'informations sur les différents types de HMMs, le lecteur pourra consulter les ouvrages suivants [Rabiner et Juang, 1989] [Roxane, 1995].

- π est la distribution initiale des états, $P(q_0 = j),\ \forall j \in [1, N]$. q_0 représente l'état initial du modèle HMM. Il ne peut émettre de vecteurs acoustiques.

En reconnaissance de la parole, des modèles de Markov gauche-droite d'ordre 1 sont le plus souvent utilisés du fait de l'aspect séquentiel du signal de la parole [Bakis, 1976]. La figure 1.5 illustre un HMM à 3 états typique utilisé en RAP pour la modélisation d'un phonème. Les états d'entrée et de sortie sont fournis pour faciliter la concaténation des modèles entre eux. L'état de sortie d'un modèle de phonème peut être fusionné avec l'état d'entrée d'un autre modèle de Markov caché pour former un modèle composite. Ceci permet aux modèles de phonèmes d'être concaténés ensemble pour former les mots et ainsi les phrases. On remarque que les seules transitions permises sont de type gauche-droite et ceci dans le but de mieux modéliser la contrainte temporelle de la parole. Un HMM est considéré comme un générateur de vecteurs acoustiques, c'est une machine à états finis qui change d'état à chaque unité de temps. Pour chaque unité de temps t, une fois arrivé à l'état q_j, un vecteur acoustique o_t est généré avec une densité de probabilité $b_j(o_t)$. De plus, la transition de l'état q_i à l'état q_j est probabiliste, sa probabilité est généralement notée a_{ij}. En pratique, c'est seulement la séquence d'observations : $O = o_1, o_2, \ldots, o_T$ qui est connue. La séquence d'états est non directement observable, d'où le nom de modèle de Markov "caché".

1.7.2 Les problèmes fondamentaux des HMMs

Soient λ un modèle de Markov caché et O une séquence d'observations acoustiques. La reconnaissance de cette séquence s'effectue en trouvant le modèle λ qui maximise la probabilité $P(\lambda|O)$ (probabilité qu'un modèle λ génère une séquence de vecteurs acoustiques O). Cette probabilité est aussi appelée probabilité *a posteriori*. Malheureusement, il n'est pas possible d'accéder directement à cette probabilité. Mais on peut calculer la probabilité qu'un modèle donné générera une certaine séquence de vecteurs acoustiques $P(O|\lambda)$.

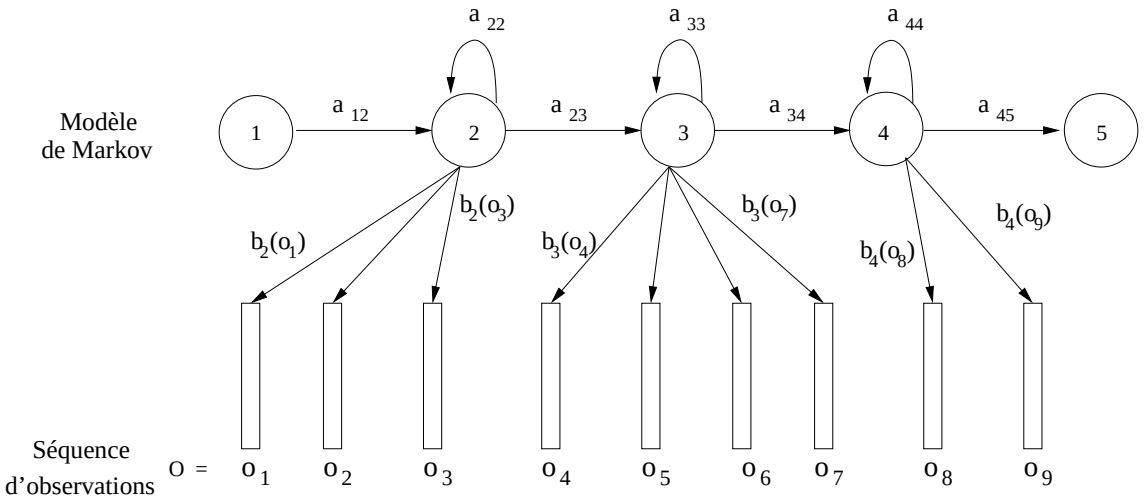

FIG. 1.5 – Exemple d'un HMM à trois états caractérisé par une distribution de probabilités pour chaque état associé à une observation et par des probabilités de transition entre les états.

En utilisant la loi de Bayes, il est possible de lier ces deux probabilités par :

$$P(\lambda|O) = \frac{P(O|\lambda).P(\lambda)}{P(O)}$$

- $P(O|\lambda)$ est la vraisemblance de la séquence d'observations O étant donné le modèle λ.
- $P(\lambda)$ est la probabilité *a priori* du modèle.
- $P(O)$ est la probabilité *a priori* de la séquence des vecteurs acoustiques.

Pour une séquence d'observations connue $O = o_1, o_2, \ldots, o_T.$, $P(O)$ peut être considérée constante, puisqu'elle est indépendante du modèle λ si les paramètres de ce dernier sont fixés. Ainsi maximiser $P(\lambda|O)$ revient à maximiser $P(O|\lambda)P(\lambda)$.

Pour cela, il faut résoudre les trois problèmes fondamentaux des HMMs suivants :

- **Évaluation** :
  Étant donnés une séquence d'observations : $O = o_1, o_2, \ldots, o_T$ et le modèle $\lambda = (N, A, B, \pi)$, comment calculer efficacement $P(O|\lambda)$ la probabilité d'observer la séquence O sachant le modèle λ ?
- **Décodage** :
  Étant donnés une séquence d'observations : $O = o_1, o_2, \ldots, o_T$ et le modèle $\lambda = (N, A, B, \pi)$, comment choisir la séquence d'états $Q = q_1, q_2, \ldots, q_T$ qui a le plus de chance d'émettre la séquence d'observations O ?
- **Apprentissage** :
  Comment déterminer les paramètres du modèle $\lambda = (N, A, B, \pi)$ afin de maximiser $P(O|\lambda)$?

1.7.3 Problème d'évaluation

Soient le modèle $\lambda = (N, A, B, \pi)$, $O = o_1, o_2, \ldots, o_T$ une séquence d'observations et $Q = q_1, q_2, \ldots, q_T$ une séquence d'états. La probabilité d'observer la séquence O pour une séquence d'états Q est :

$$P(O|Q, \lambda) = b_{q_1}(o_1) \,.\, b_{q_2}(o_2) \ldots b_{q_T}(o_T)$$

Or, la probabilité de la séquence Q peut s'écrire sous la forme suivante :

$$P(Q|\lambda) = \pi_{q_1} a_{q_1 q_2} a_{q_2 q_3} \dots a_{q_{T-1} q_T}$$

La probabilité conjointe du chemin Q et des observations O est :

$$P(O, Q|\lambda) = P(Q|\lambda) \; . \; P(O|Q, \lambda)$$

La probabilité de la séquence d'observations O sachant le modèle λ est obtenue par la sommation de $P(O, Q|\lambda)$ sur toutes les séquences d'états Q possibles. Ainsi la probabilité d'émission des observations est :

$$P(O, \lambda) = \sum_{Q} P(O, Q|\lambda)$$

$$P(O, \lambda) = \sum_{q_1, q_2, \dots, q_T} \pi_{q_1} b_{q_1}(o_1) a_{q_1 q_2} b_{q_2}(o_2) \dots a_{q_{T-1} q_T} b_{q_T}(o_T)$$

Pour une machine à N états, ce calcul direct nécessite $(2T-1) * N^T$ multiplications et $N^T - 1$ additions, ce qui le rend trop complexe et impossible à implémenter. Il existe heureusement un algorithme rapide et efficace dit **avant-arrière** (Forward-Backward) qui donne une solution pour mener efficacement ce calcul.

L'algorithme avant-arrière

Soit, la probabilité avant : $\alpha_t(i) = P(o_1, o_2, \dots, o_t, q_t = i|\lambda)$, la probabilité d'observer la séquence $o_1, o_2, \dots, o_t$ et d'être à l'état i à l'instant t sachant le modèle λ. Cette probabilité est calculée d'une manière récursive.

Algorithme 1 Algorithme avant

Initialisation : $\alpha_1(i) = \pi_i b_i(o_1) \quad 1 \leq i \leq N$

Récurrence : $\alpha_{t+1}(j) = [\sum_{i=1}^{N} \alpha_t(i) a_{ij}] b_j(o_{t+1}), \quad t \in \{1, 2, \dots, T-1\} \quad et \quad 1 \leq j \leq N$

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

Cette récursion dépend du fait que la probabilité d'être à l'état j au temps $t+1$ et d'observer o_{t+1} peut être déduite en sommant les probabilités avant pour tous les états prédécesseurs de j pondérées par les probabilités de transition a_{ij}.

De la même manière, soit la probabilité arrière $\beta_t(j)$ définie par :

$$\beta_t(j) = P(o_{t+1}, o_{t+2}, \dots, o_T|q_t = j, \lambda)$$

C'est la probabilité d'observer la séquence $o_{t+1}, o_{t+2}, \dots, o_T$ sachant qu'on est à l'état i au temps t et qu'on a le modèle λ.

De la même façon cette probabilité est calculée d'une manière récursive :

Algorithme 2 Algorithme arrière

Initialisation : $\beta_T(i) = 1 \quad 1 \leq i \leq N$

Récurrence : $\beta_t(i) = \sum_{j=1}^{N} a_{ij}\ b_j(o_{t+1})\ \beta_{t+1}(j), \quad t \in \{T-1, T-2, \ldots, 1\} \quad et \quad 1 \leq i \leq N$

Terminaison : $P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i) = \sum_{i=1}^{N} \pi_i b_i(o_1)\beta_1(i) = \sum_{i=1}^{N} \alpha_t(i)\beta_t(i)$

1.7.4 Problème de décodage

Étant donnés une séquence d'observations O, et un modèle $\lambda = (N, A, B, \pi)$, le problème de décodage revient à la recherche d'une séquence d'états "optimale". Cela peut-être fait de différentes façons. La difficulté réside dans la définition de la séquence d'états optimale. Donc, il faut choisir un critère parmi plusieurs critères d'optimalité. Par exemple, un critère envisageable pour répartir les vecteurs de la séquence d'observations sur les états de la chaîne, consiste à optimiser séparément chaque état q_t. Pour implémenter cette solution, une variable γ est définie par :

$$\gamma_t(i) = P(q_t = i|O, \lambda) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i)\beta_t(i)} = \frac{\alpha_t(i)\beta_t(i)}{P(O|\lambda)}$$

$\gamma_t(i)$ est la probabilité d'être à l'état i au temps t, étant donnée l'observation O et le modèle λ.

L'état optimal à un instant t sera donc :

$$q_t = arg_i\ max[\gamma_t(i)]$$

Ce critère d'optimalité maximise le nombre d'états. Cependant, cette méthode peut aboutir à des erreurs. Par exemple, lorsque le modèle de Markov possède des probabilités de transitions égales à zéro, la séquence optimale obtenue pourrait en fait ne pas être une séquence d'états possibles puisque le critère considéré ne tient pas compte des probabilités des changements d'états. Une solution possible est de modifier le critère d'optimalité. On pourrait par exemple chercher la séquence d'états qui maximise les paires d'états (q_t, q_{t+1}) ou même les triplets d'états (q_t, q_{t+1}, q_{t+2}).

Si ces critères sont tout à fait adaptés à certaines applications, le critère le plus utilisé est celui qui cherche la meilleure séquence d'états globale (le meilleur chemin), c'est-à-dire qui maximise $P(Q|O, \lambda)$ ce qui revient à maximiser $P(Q, O|\lambda)$. Une technique formelle existe pour calculer ce chemin optimal, il s'agit de l'**algorithme de Viterbi**.

L'algorithme de Viterbi

Pour trouver la meilleure séquence d'états $Q = q_1, q_2, \ldots, q_T$, connaissant une séquence d'observations $O = o_1, o_2, \ldots, o_T$, on a besoin de définir la quantité $\delta_t(i)$.

$$\delta_t(i) = max_{q_1, q_2, \ldots, q_{t-1}} P(q_1, q_2, \ldots, q_t = i, o_1, o_2, \ldots, o_t|\lambda)$$

$\delta_t(i)$ est le meilleur résultat (probabilité la plus grande) selon un simple chemin ; ce chemin se compose des t premières observations et se termine dans l'état i. On peut déterminer les $\delta_t(i)$ de façon itérative. On a en effet :

$$\delta_{t+1}(j) = max_{1 \leq i \leq N}[\delta_t(i)a_{ij}]b_j(o_{t+1})$$

Algorithme 3 Algorithme de Viterbi

Initialisation :

$$\delta_1(i) = \pi_i b_i(o_1) \quad 1 \leq i \leq N$$
$$\psi_1(i) = 0$$

Récurrence :

$$\delta_t(j) = max_{1 \leq i \leq N}[\delta_{t-1}(i)a_{ij}]b_j(o_t) \quad 2 \leq t \leq T \quad 1 \leq j \leq N$$
$$\psi_t(j) = arg_{1 \leq i \leq N}\ max[\delta_{t-1}(i)a_{ij}]$$

Terminaison :

$$P^* = max_{1 \leq i \leq N}[\delta_T(i)]$$
$$\psi_T^* = arg_{1 \leq i \leq N}\ max[\delta_T(i)]$$

Recherche :

$$q_t^* = \psi_{t+1}(q_{t+1}^*) \quad t = T-1, T-2, \ldots 1$$

Pour déterminer la séquence d'états, il est donc nécessaire de garder la trace de l'indice i qui a maximisé la formule précédente, et ceci pour tout t et tout j. On réalise ceci par l'intermédiaire d'un tableau $\psi(j)$.

1.7.5 Problème d'apprentissage

Le troisième problème consiste à trouver une méthode pour ajuster les paramètres du modèle $\lambda = (N, A, B, \pi)$ afin de maximiser la probabilité d'une séquence d'observations donnée, sachant le modèle λ. Ce problème n'a pas de solution analytique connue et il n'existe pas de technique optimale pour estimer les paramètres du modèle. On peut cependant choisir $\lambda = (N, A, B, \pi)$ de telle façon que $P(O|\lambda)$ soit localement maximale en utilisant une procédure itérative telle que la méthode de Baum-Welch ou la technique du gradient [Juang, 1985] [Bahl *et al.*, 1986]. Dans ce qui suit nous présentons une procédure itérative basée sur la technique de Baum-Welch.

Pour décrire comment re-estimer les paramètres du HMM, on définit la probabilité $\xi_t(i, j)$ qui représente la probabilité d'être à l'état i au temps t et de faire une transition à l'état j au temps $t + 1$ étant donnée la séquence d'observations O et le modèle λ.

$$\xi_t(i,j) = P(q_t = i, q_{t+1} = j|O, \lambda)$$

D'après les définitions des probabilités avant et arrière, $\xi(i,j)$ peut s'écrire sous la forme suivante :

$$\xi_t(i,j) = \frac{\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)}{P(O|\lambda)}$$

Nous avons défini, précédemment $\gamma_t(i)$ comme étant la probabilité d'être à l'état i au temps t, étant donnée l'observation O et le modèle λ. Ainsi nous pouvons relier $\gamma_t(i)$ à $\xi_t(i,j)$ par une sommation sur j, d'ou la relation suivante :

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j)$$

L'algorithme de Baum-Welch estime les nouveaux paramètres de la chaîne de Markov cachée comme suit :

$$\bar{\pi}_i = \gamma_1(i), \quad 1 \leq i \leq N$$

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \quad 1 \leq i \leq N, \;\; 1 \leq j \leq N$$

$$\bar{b}_j(k) = \frac{\sum_{t=1, o_t=k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)} \quad 1 \leq j \leq N$$

La ré-estimation de π_i est la probabilité d'être à l'état i au temps $t = 1$. La formule de ré-estimation de a_{ij} est le rapport du nombre de transitions de l'état i vers l'état j sur le nombre de transitions partant de l'état i. La ré-estimation de $b_j(k)$ est le rapport du nombre de fois d'être à l'état i en observant k sur le nombre de fois étant dans l'état i.

Nous avons défini le modèle courant $\lambda = (N, A, B, \pi)$, et nous l'avons utilisé pour recalculer ces variables, ainsi nous avons le modèle ré-estimé $\bar{\lambda} = (N, \bar{A}, \bar{B}, \bar{\pi})$. Nous pouvons ainsi affirmer l'une au l'autre de ces propositions :

- le modèle initial λ définit un point critique de la fonction de vraisemblance, dans ce cas $\bar{\lambda} = \lambda$
- le modèle $\bar{\lambda}$ est meilleur que le modèle λ dans le sens où $P(O|\bar{\lambda}) > P(O|\lambda)$, donc la séquence d'observations O est plus probable avec le nouveau modèle $\bar{\lambda}$.

En se basant sur cette procédure, si nous utilisons itérativement le modèle $\bar{\lambda}$ à la place de λ et si nous répétons l'étape de la ré-estimation des paramètres. Nous pouvons alors améliorer la probabilité que O soit observée sachant le modèle jusqu'à atteindre un certain point limite. Le résultat final de la procédure de ré-estimation est appelé : l'estimation au maximum de vraisemblance du HMM (Maximum Likelihood Estimation : MLE). Il existe d'autres critères d'apprentissage, comme les critères MAP (Maximum A Posteriori) [Gauvain et Lee, 1994] ou

MMI (Maximum Mutual Information) [Cardin *et al.*, 1991] [Kapadia *et al.*, 1993], mais leur mise en œuvre est généralement plus difficile.

1.7.6 Densités d'observation continues dans les modèles de Markov cachés

Jusqu'à présent nous n'avons considéré que le cas où les observations prennent des valeurs dans un alphabet fini discret et nous pouvions donc utiliser une loi de probabilité discrète dans chaque état du modèle. Une telle approche n'est pas compatible avec des observations qui sont des signaux continus. Bien sûr, quantifier le signal pourrait permettre de résoudre le problème, mais cela ne pourrait entraîner que des dégradations. Il est donc préférable d'utiliser des modèles de Markov cachés avec des densités d'observation continues. La représentation la plus générale de la fonction des densités de probabilités pour laquelle une procédure de ré-estimation a été effectuée est de la forme :

$$b_j(o_t) = \sum_{m=1}^{M} c_{jm} N(o_t, \mu_{jm}, \Sigma_{jm}) \quad avec \sum_{m=1}^{M} c_{jm} = 1 \quad 1 \leq j \leq N$$

où c_{jm} est le $m^{ième}$ coefficient du mélange dans l'état j et $N(\ )$ est une densité gaussienne de moyenne μ_{jm} et de matrice de covariance Σ_{jm}.

Dans le cas d'une distribution monogaussienne, les formules de ré-estimation de la moyenne et de la matrice de covariance de la densité de probabilité à l'état k sont données par les équations suivantes :

$$\mu_k = \frac{\sum_{t=1}^{T} \gamma_t(k) o_t}{\sum_{t=1}^{T} \gamma_t(k)}$$

$$\Sigma_k = \frac{\sum_{t=1}^{T} \gamma_t(k)(o_t - \mu_k)\overline{(o_t - \mu_k)}}{\sum_{t=1}^{T} \gamma_t(k)}$$

1.8 L'approche phonétique

Les modèles de Markov cachés sont largement employés en RAP ces dernières années. Ces modèles se sont avérés les mieux adaptés aux problèmes de la reconnaissance de la parole [Rabiner et Juang, 1993]. La quasi-totalité des outils de reconnaissance de la parole disponibles actuellement sur le marché sont basés sur cette technologie. Un modèle de Markov caché est un automate stochastique particulier capable, après avoir été entraîné, d'estimer la probabilité qu'une séquence d'observations ait été générée par ce modèle. Idéalement, il faut pouvoir associer à chaque phrase possible un modèle. Il va de soi que ceci est irréalisable en pratique car le nombre de modèles serait très élevé. Des sous-unités lexicales comme le mot, la syllabe, ou le phonème sont utilisées afin de réduire le nombre de paramètres à entraîner. À chacune de ces unités est associé un modèle de Markov caché constitué d'un nombre fini d'états prédéterminés. Ainsi un HMM peut représenter n'importe quel ensemble d'unités acoustiques : mots, phonèmes etc. Mais, dans le cas des grands vocabulaires, le fait d'associer à chaque mot un HMM distinct pose de sérieux problèmes d'apprentissage et de stockage. En effet, pour réaliser cette tâche correctement il faut que le corpus d'apprentissage contienne plusieurs occurrences de chaque mot. Ceci est pratiquement impossible. De plus ajouter un nouveau mot au lexique, demande la collecte de nouvelles occurrences acoustiques et un nouvel apprentissage, ce qui n'est guère facile. Ainsi,

la reconnaissance de grands vocabulaires est toujours effectuée au moyen de HMMs de mots élaborés à partir de HMMs représentant des sous-unités de mots.

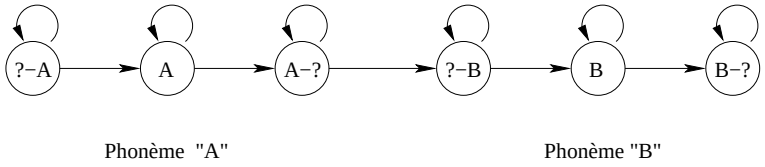

FIG. 1.6 – Modèle de phonème à trois états

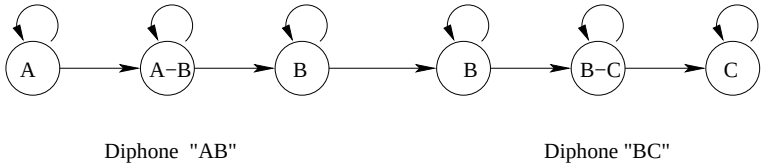

FIG. 1.7 – Modèle de diphone

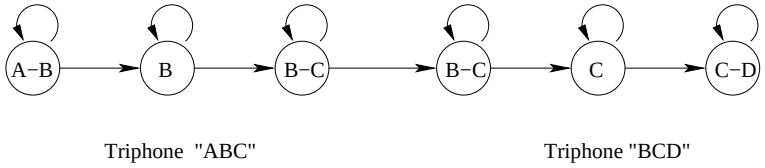

FIG. 1.8 – Modèle de triphone

L'approche fréquemment utilisée consiste à prendre un modèle HMM à trois états par phonème, en faisant l'hypothèse que l'état du milieu modélise la partie stationnaire du phonème et les états extérieurs modélisent la coarticulation avec les phonèmes voisins, figure 1.6. Le nombre de phonèmes est de l'ordre de 35. L'utilisation de diphones en reconnaissance de la parole part du principe que la réalisation acoustique d'un phonème est fortement influencée par les phonèmes qui le précèdent et qui le suivent immédiatement et que les transitions entre phonèmes contiennent plus d'information que les endroits stables à l'intérieur des phonèmes [Ruske et Schotola, 1991] [Marcus, 1992]. Le diphone devient alors une modélisation de la transition entre deux phonèmes avec une partie contextuelle qui correspond à ces phonèmes, figure 1.7. Dans la langue française, le nombre de diphones utilisés est de l'ordre de 1000. Le dernier type utilisé est le triphone, il tient compte des deux phonèmes, le précédent et le suivant pour fixer le contexte, comme c'est illustré par la figure 1.8. D'où le grand nombre de HMMs de triphones qui est de l'ordre de 10000.

La combinaison de plusieurs HMMs en un seul est illustrée par la figure 1.9, où un HMM correspondant au mot **Paris** est obtenu par la concaténation des HMMs correspondant aux phonèmes p, a, r, i. La figure 1.9.a, montre que la construction peut être obtenue par la coïncidence du dernier état du HMM d'un phonème avec le premier état du HMM du phonème suivant. Un HMM plus élaboré est illustré par la figure 1.9.b, où pour le même mot, on ajoute une transition qui permet l'omission d'un phonème. Les HMMs obtenus par la concaténation d'autres HMMs donnent en général des résultats de reconnaissance inférieurs à ceux obtenus directement au moyen de HMMs appris sur les unités qu'ils sont censés représenter. Cependant, l'utilisation

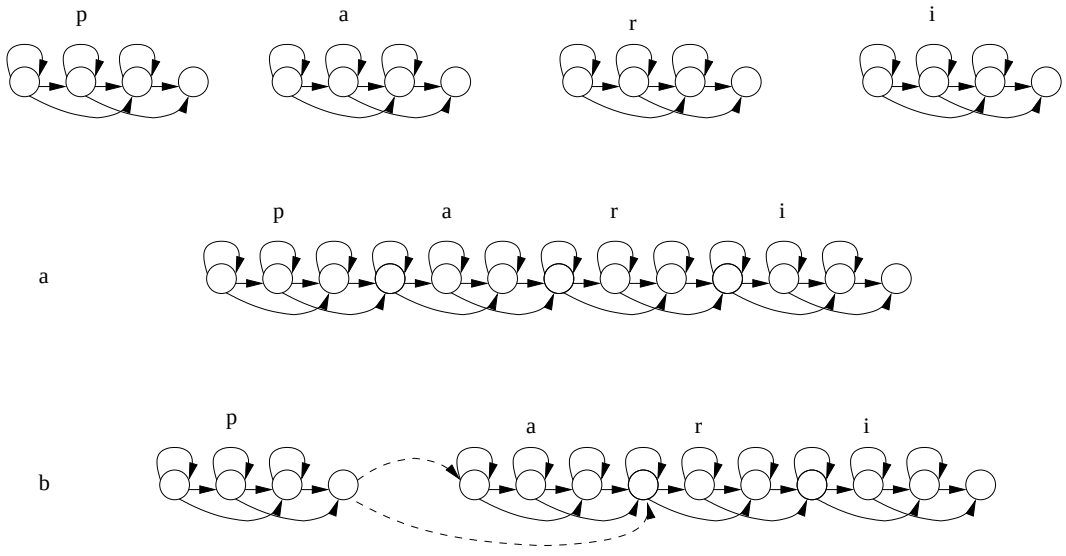

FIG. 1.9 – HMM d'un mot obtenu par concaténation de HMMs de phonèmes

de ce type de construction permet de réduire le nombre de HMMs nécessaires pour une tâche donnée (par exemple, avec une modélisation non contextuelle et pour un vocabulaire de milliers d'entrées, 35 HMMs de phonèmes peuvent suffire plutôt que des milliers de HMMs de mots).

Cette approche phonétique permet d'éviter la collecte d'énormes corpus d'apprentissage qui ont été nécessaires lors de l'utilisation de HMMs de mots. Ceci permet de rendre réalisable la reconnaissance automatique de la parole sur de grands vocabulaires. De plus, l'impact au niveau flexibilité est également très important. En effet, l'ajout de nouveaux mots dans le lexique ne nécessite plus aucune collecte ni apprentissage supplémentaire, il suffit simplement de déterminer la ou les séquences des sous-unités correspondant aux nouveaux mots et de construire selon ces séquences et à partir des HMMs de ces sous-unités, les HMMs modélisant ces nouveaux mots. En plus, cette approche phonétique permet une économie au niveau de l'espace mémoire requis tant lors de la reconnaissance que lors de l'apprentissage. En effet, le nombre de distributions devant être conservées en mémoire est nettement inférieur à celui qu'aurait nécessité l'utilisation d'un HMM par mot de lexique.

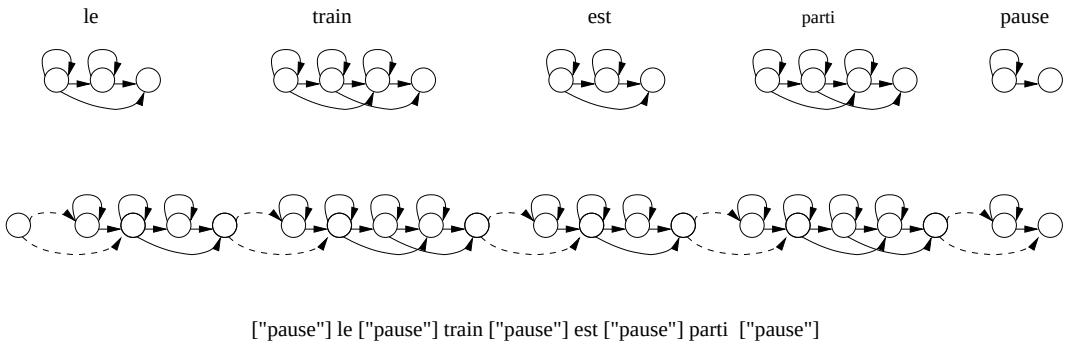

FIG. 1.10 – HMM d'une phrase obtenu par concaténation de HMMs de mots

Finalement, de la même façon qu'on peut construire des HMMs de mots à partir de HMMs de phonèmes, on peut construire des HMMs de phrases à partir de HMMs de mots, figure 1.10.

Chapitre 2

Détection de mots clés

2.1 Introduction

L'évolution de la reconnaissance automatique de la parole a conduit à un développement de systèmes opérant dans des conditions réelles [Zue *et al.*, 1997]. Parmi eux, nous trouvons les systèmes de dialogue et les systèmes à commande vocale [Riccardi *et al.*, 1997] [Garcia *et al.*, 1998]. Les accès automatiques à des services d'information peuvent être considérés parmi les applications les plus importantes de la technologie de la RAP.

Les chercheurs se sont aussi intéressés au développement des systèmes qui ont comme entrée la parole spontanée prononcée par différents locuteurs et dans des environnements divers [Cole *et al.*, 1995]. Ceci pose de nouveaux problèmes, comme l'introduction des mots Hors Vocabulaire (HV), les faux départs, les hésitations, etc. Dans ces situations, le système de reconnaissance doit être capable de détecter et de reconnaître les mots clés de l'application et de rejeter les mots HV. Les systèmes de reconnaissance équipés par des processus de détection de mots clés permettent aux utilisateurs de parler librement et naturellement, sans besoin de s'exprimer avec un format rigide.

Dans un cas particulier d'accès à une base d'informations, l'efficacité du système est conditionnée par deux facteurs principaux : d'une part, ce système doit permettre l'utilisation de la parole spontanée peu contrainte et d'autre part, il doit réaliser de bonnes performances de reconnaissance. Cependant, il y a un compromis entre ces deux aspects. Un mode d'élocution spontané peu contraint (l'acceptation de la parole naturelle et donc la considération d'un grand vocabulaire) rend le processus de reconnaissance très difficile et très lent, ce qui affaiblit le taux de reconnaissance. Un système de reconnaissance idéal doit accepter la parole spontanée et générer une transcription précise de la phrase d'entrée en temps réel.

L'application des systèmes de reconnaissance conventionnels pour la parole spontanée présente quelques problèmes car il faut considérer un grand vocabulaire et le langage doit être modélisé par une grammaire complexe qui considère tous les évènements possibles dans la parole spontanée, comme les phrases tronquées, les phrases grammaticalement incorrectes, toux, début incorrect, etc. Les systèmes de détection de mots clés ont fournit une solution aux processus de la parole spontanée [Wilpon *et al.*, 1990] [Rose, 1995]. En effet le but des systèmes de reconnaissance conventionnels est de trouver une transcription exacte de tous les mots prononcés dans une phrase, alors que les systèmes de détection de mots clés essayent de détecter seulement les

mots qui ont une importance pour l'interprétation sémantique de la phrase et qui sont définis précédemment dans le vocabulaire des mots clés. Dans ces systèmes, les segments sémantiques significatifs sont extraits tandis que le reste est ignoré, le contenu sémantique peut être donc capté sans une reconnaissance détaillée de tous les mots de la phrase. Il suffit donc que les mots clés précédemment définis dans le vocabulaire soient détectés s'ils sont prononcés dans la phrase.

La détection de mots clés a été utilisée dans différentes tâches, tels que la classification des documents parlés, les appels téléphoniques, l'extraction d'information etc. [Wilcox et Bush, 1991] [Foote *et al.*, 1995]. Plusieurs approches ont été proposées dans la littérature pour modéliser les mots hors-vocabulaire ; elles vont de l'utilisation d'un petit nombre de modèles poubelles à l'inclusion de tous les mots possibles dans le contexte de la tâche de détection de mots clés [Rose, 1992] [Rohlicek *et al.*, 1993] [Bourlard *et al.*, 1994]. D'autres auteurs ont proposé l'utilisation des anti-modèles qui modélisent les différents types d'événements incorrects. Par exemple, pour chaque modèle M_φ du phonème φ correspond un anti-modèle $\bar{M}_\varphi$ qui modélise les erreurs de substitutions et de fausse acceptations [Sukkar et Lee, 1996] [Moreau et Jouvet, 2000]. Plusieurs chercheurs se sont intéressés à l'estimation d'une mesure de confiance pour chaque mot clé, et la vérification des mots clés en utilisant ces mesures de confiance [Mathan, 1991] [Chigier, 1992] [Sukkar, 1994] [Cox et Rose, 1996] [Caminero *et al.*, 1997]. Dans la phase de vérification, cette mesure de confiance est utilisée pour accepter ou rejeter chaque mot clé reconnu par le système. De cette manière, la sélection d'un seuil adéquat à cette mesure de confiance, permet de détecter facilement les fausses acceptations, quoi que quelques mots clés correctement reconnus peuvent être aussi rejetés.

Différents systèmes de détection avec différentes architectures ont été proposés. En général ces systèmes appartiennent à trois grandes classes qui sont les HMM, les réseaux de neurones et la programmation dynamique (on retrouve ces classes en reconnaissance classique, en reconnaissance du locuteur etc.). Pour améliorer les performances, différentes architectures hybrides ont été proposées. Le plus souvent, ces architectures sont composées d'une combinaison de modèles HMM avec des méthodes neuronales [Morgan *et al.*, 1990] [Cercadillo et Gomez, 1993] ou avec des méthodes de descente probabiliste généralisée du gradient [Sukkar et Wilpon, 1993]. Le succès des HMMs est principalement dû à deux raisons qui sont, d'une part, leur haute performance en terme de capacité d'apprentissage et de taux de reconnaissance et, d'autre part, leur efficacité en terme de coût de calcul.

Nous présentons dans ce chapitre un état de l'art de l'utilisation du modèle poubelle et de la mesure de confiance dans le cadre de la détection de mots clés. Dans le paragraphe 2.2, nous abordons les diverses méthodes de construction du modèle poubelle. Ensuite nous exposons la mesure de confiance et nous décrivons différentes techniques qui ont été proposées afin de réaliser cette tâche, comme la programmation dynamique, les HMM avec modification de l'algorithme de Viterbi, l'utilisation des traces d'alignement, l'apprentissage discriminant, la méthode à base de seuil sur les scores de reconnaissance, les méthodes d'adaptation, les méthodes basées sur des connaissances acoustiques et linguistiques, les réseaux de neurones et quelques transformations et algorithmes appliqués pour l'extraction des mesures de confiance. Enfin nous terminons en donnant quelques applications réelles de la détection de mots clés.

2.2 Modèle poubelle

En détection de mots clés, le but est de reconnaître les mots clés dans un flux de parole continue, indépendamment du locuteur. La détection de mots clés résoud quelques problèmes liés à la reconnaissance de la parole continue comme les hésitations, les faux départs, les phrases grammaticalement incorrectes, les phrases tronquées, etc. Cependant, elle introduit de nouveaux types de problèmes notamment ceux de fausses acceptations et de faux rejets.

La structure générale d'un système de détection de mots clés peut être décrite comme une combinaison de mots clés et de séquences de parole (ou éventuellement de bruit) constituées de mots non-clés. Un système de détection a pour rôle d'une part la modélisation des mots non-clés afin de réduire les fausses acceptations et d'autre part la modélisation des mots clés pour améliorer leur détection. La performance d'un tel système de détection de mots clés dépend essentiellement de la manière avec laquelle il peut accomplir ces deux tâches.

Dans un système de détection de mots clés, il est souhaitable de réaliser un taux élevé de détection, avec une minimisation du nombre de fausses acceptations. Dans ce cas, il n'est pas suffisant de modéliser seulement les mots clés avec un grand détail acoustique, mais la modélisation des mots HV est aussi nécessaire. Une approche commune pour la tâche de détection de mots clés est l'emploi d'un ensemble de modèles poubelles pour concurrencer les modèles de mots clés. Les modèles poubelles doivent être adaptés aux mots HV (mots non-clés), aux bruits, aux sons non parole, aux faux départs, etc. Le rôle de ces modèles poubelles est d'absorber tous les mots HV.

Plusieurs travaux ont utilisé un modèle poubelle explicite [Rohlicek *et al.*, 1989] [Rose et Paul, 1990] [Lleida *et al.*, 1993] [Rose et Hofstetter, 1993] [Lippmann *et al.*, 1994].

Rohlicek [Rohlicek *et al.*, 1989] décrit l'utilisation de tous les modèles de mots clés et un simple modèle poubelle. Ce dernier est créé par la combinaison des distribution des probabilités d'émission gaussiennes de tous les modèles des mots clés. Ensuite, il propose une deuxième méthode dans laquelle, il représente son modèle poubelle par un réseau de modèles dont les composantes sont des segments des modèles de mots clés. C'est cette dernière méthode qui a donné les meilleures performances.

L'un des premiers systèmes de détection des mots clés utilisant les modèles HMM pour la reconnaissance de la parole continue a été proposé par Rose [Rose et Paul, 1990]. Le principe de ce système repose sur la séparation entre mots clés et mots poubelles. Il a construit un réseau constitué de N mots clés et M mots poubelles comme le montre la figure 2.1. Le point d'opération du système peut être ajusté par la disposition des poids des transitions, $w_1, w_2, ...w_N$ pour les mots clés et $f_1, f_2, ...f_M$ pour les mots poubelles. Dans ce contexte, ce point d'opération réfère à un compromis entre le nombre de faux rejets et celui de fausses acceptations.

Le score associé à un mot clé décodé par l'algorithme de Viterbi est la vraisemblance de ce mot normalisée par sa durée. Le score S_w pour un mot clé w prononcé dans l'intervalle de temps $[T_I, T_F]$ avec un état terminal S_F est donc donné par la formule suivante :

$$S_w = \frac{log\ P(S_F, y_{T_I}, ..., y_{T_F})}{|T_I - T_F|}$$

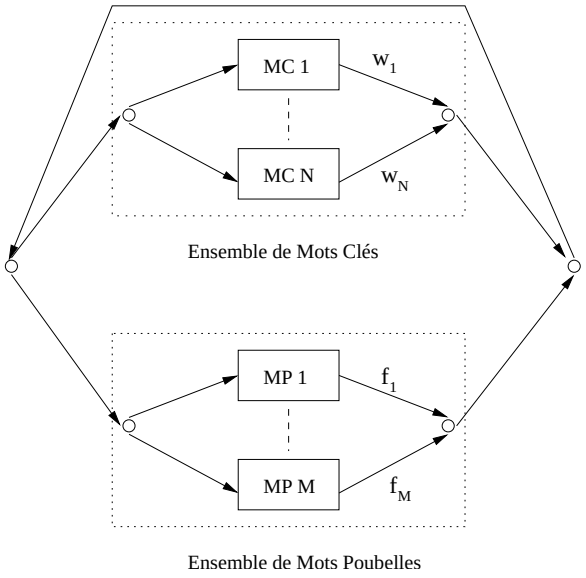

FIG. 2.1 – Description du système de détection de mots clés basé sur l'utilisation d'un réseau de mots clés et de mots poubelles

Où y_{T_i} est un vecteur de paramètres à l'instant T_i.

Pour réduire les fausses acceptations, un réseau parallèle "en arrière plan" de modèles de bruit a été inclus comme le montre la figure 2.2. Le score S_w du mot clé peut être calculé pour une séquence de modèles de bruit qui se chevauchent avec un mot clé. Le score final en terme de probabilité devient alors :

$$S_{LR} = S_w - S_{BA}$$

Où S_{BA} est le score de recouvrement du bruit.

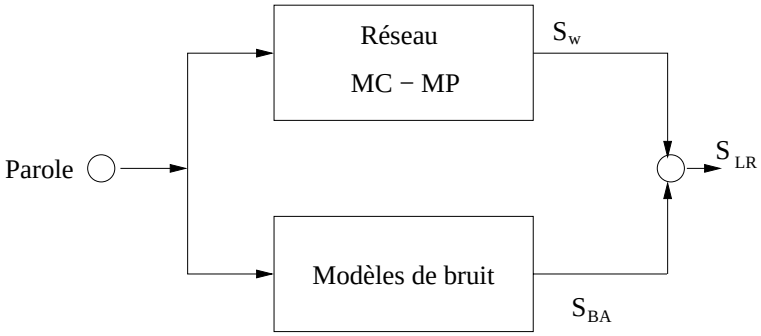

FIG. 2.2 – Réseau parallèle introduisant des modèles de bruit

Pour construire le réseau des modèles poubelles, on commence par utiliser 80 modèles poubelles correspondant aux mots hors-vocabulaire de la base d'apprentissage. Cependant, la majorité de ces mots ne figurent pas dans la base de test, il a donc été proposé d'introduire des modèles de triphones composant les 80 mots hors vocabulaire. On a obtenu ainsi 268 triphones représentés par des HMM linéaires à 3 états chacun. Une amélioration des performances a été donc observée, néanmoins il y a parfois des chevauchements des contextes entre les modèles poubelles et les mots clés, ce qui engendre une dégradation des performances du système de détection. En effet, à chaque chevauchement, le mot clé en question sera représenté par le modèle poubelle superposé. La solution qui a été proposée consiste à utiliser les phonèmes indépendants du contexte au lieu des triphones. Cette solution a donné des performances comparables à la précédente. De plus, elle représente l'avantage d'être beaucoup moins complexe au niveau implémentation.

Le modèle poubelle en ligne [4] est introduit par Boite [Boite *et al.*, 1993] et Bourlard [Bourlard *et al.*, 1994] pour décrire la création d'un modèle poubelle local comme étant la moyenne des n-meilleurs vraisemblances de chaque trame dans la séquence d'observations.

Lleida [Lleida *et al.*, 1993] a mené une étude comparative sur différentes formes de modèles poubelles en introduisant des modèles poubelles aux niveaux phonème, syllabe et mot. En premier lieu, les modèles poubelles sont représentés par des phonèmes indépendants du contexte. En deuxième lieu, il a groupé les syllabes en quatre classes, chaque classe regroupe les syllabes les plus ressemblantes. Ensuite il a fait toutes les combinaisons possibles de ces quatre classes de syllabes afin d'extraire 60 modèles poubelles syllabiques. En troisième lieu il a construit trois modèles poubelles à base de mots. Le premier modèle est pour les mots monosyllabiques, le deuxième est pour les mots bisyllabiques et le troisième est pour les mots qui contiennent trois syllabes ou plus.
Pour la modélisation de ces modèles poubelles, des HMM à trois états ont été utilisés pour les modèles poubelles phonétiques. Pour les modèles à base de syllabes et de mots, il a utilisé un HMM à 10 états (gauche-droite). Les modèles poubelles basés sur les syllabes ont donné les meilleures performances.

Klemm [Klemm *et al.*, 1995] décrit l'utilisation des modèles poubelles basés sur les modèles de mots et de syllabes. Dans le cas des modèles de mots, il a défini deux ensembles, un ensemble pour les modèles des mots clés et un ensemble beaucoup plus grand contenant les modèles des autres mots du vocabulaire considérés comme des modèles poubelles. Ce système de détection est équivalent à un système de reconnaissance grand vocabulaire. Dans le cas des modèles poubelles à base de syllabes, il utilise des concaténations de diphones et de triphones déjà appris par le systèmes de reconnaissance. En tout, il utilise 400 modèles de syllabes pour couvrir 93% du vocabulaire de l'application. La performance des modèles poubelles basés sur les syllabes s'est avérée comparable au système grand vocabulaire.

Meliani [Meliani et O'Shaughnessy, 1998] propose deux types de modèles poubelles lexicaux à base de syllabes. Dans le premier type, chaque syllabe représente un modèle poubelle à part et dans le deuxième type, un modèle poubelle contient toutes les syllabes ayant la même fréquence d'apparition dans le corpus d'apprentissage. En total, 12226 syllabes ont été récupérés de la base d'apprentissage, mais il leur faut une étape de filtrage pour ne laisser que les syllabes les plus fréquentes. Une série d'expériences a été menée afin de décider de la fréquence minimale des

[4] Traduction du mot anglais : on line

syllabes à conserver. Les meilleurs résultats ont été obtenu pour une fréquence de 0.005%.

Zhang [Zhang *et al.*, 2001] utilise cinq modèles poubelles construits de la façon suivante : le premier est appris sur les mots contenant un ou deux phonèmes, le deuxième est appris sur les mots à trois phonèmes et ainsi de suite, le dernier sera appris sur les mots qui ont plus de six phonèmes. Le choix de ces cinq modèles poubelles est basé sur le fait qu'un seul modèle poubelle peut échouer souvent pour couvrir l'éventail des mots hors vocabulaire. En plus, avec ces cinq modèles on essaye de représenter explicitement les entrées poubelles et d'extraire ainsi le plus possible de mots hors-vocabulaire.

La diversité des approches basées sur l'utilisation des modèles poubelles pour la détection de mots clés, montre l'intérêt de cette technique. En effet, l'emploi de ces modèles permet d'absorber les mots hors-vocabulaire prononcés et de diminuer ainsi le nombre de fausses acceptations. Nous proposons alors leur utilisation dans le cadre de notre application dans l'optique d'une amélioration des performances.

2.3 Mesure de confiance

Le système de détection de mots clés risque toujours de produire des insertions, des substitutions et des omissions, ce qui déclenche parfois des fausses acceptations, tout en présentant des non détections. La régulation de ces opérations se fait par la génération des hypothèses de mots et l'association d'un score à chaque mot. Pour éviter les erreurs de détection, il faut être capable de générer suffisamment d'hypothèses et donc de résoudre le problème d'omissions de mots clés. Ensuite, vient l'étape de la vérification qui permet d'accepter seulement les mots clés en se basant sur une mesure de confiance pour rejeter les mots incorrectement reconnus. Dans des situations particulières où la parole prononcée ne contient pas de mots clés ou il y a une grande confusion entre les mots clés, un grand taux de substitution peut être observé. Pour remédier à ce problème, le système de reconnaissance doit être capable de reconnaître les mots clés incorporés dans la parole et de les vérifier ensuite afin de rejeter les mots qui ont un score de confiance faible.

Le meilleur taux de reconnaissance étant étroitement lié au nombre de fausses acceptations, nous devons essayer au cours de l'étape de vérification de réduire le nombre de fausses acceptations sans réduire considérablement le taux de reconnaissance. La sortie du système de reconnaissance doit être modifiée afin d'extraire différents scores qui peuvent être utilisés pour générer des vecteurs caractéristiques qui seront à leurs tour utilisés dans le processus de vérification. Chacun de ces scores représente le niveau de confiance assigné à chacun des mots et il est capable de faire la discrimination entre les mots clés corrects et ceux insérés et donc de réduire le nombre de fausses acceptations.

Plusieurs chercheurs ont travaillé sur la détection des mots clés en utilisant des mesures de confiance [Rivlin *et al.*, 1996] [Weintraub *et al.*, 1997] [Moreau et Jouvet, 2000] [Zhang et Rudnicky, 2001]. Le processus de vérification devient un problème de classification, dans lequel chaque mot clé doit être classé comme correct s'il correspond à un mot clé correctement détecté ou incorrect s'il correspond à une fausse alarme. Le processus de classification des mots clés reconnus se base sur l'utilisation d'un score discriminant associé à chaque mot et la définition d'un seuil d'acceptation pour les mots clés.

2.3.1 Programmation dynamique

L'approche historique de la détection de mots clés est basée sur l'algorithme de recalage temporel Dynamic Time Warping (DTW). Il s'agit d'un algorithme de reconnaissance, mais il convient bien à la détection de mots clés grâce à son estimation de distance. On peut l'appliquer en relâchant les contraintes sur les régions de commencement et de fin de l'échantillon de parole [Myers *et al.*, 1980] [Rabiner et Schmidt, 1980] [Myers et Rabiner, 1981] comme le montre la figure 2.3. La méthode consiste à faire coïncider la séquence de parole à l'entrée du système, avec une séquence de mots de référence concaténés entre eux [Higgins et Wohlford, 1985]. Une alarme se déclenche lorsque la séquence qui correspond le mieux au signal d'entrée contient un mot clé. Pour calculer le meilleur alignement, on utilise généralement l'un de ces trois algorithmes récursifs : one-pass, level-building ou two-level, qui sont presque équivalents dans leur version synchronisée temporellement [Godin et Lockwood, 1989] [Bezie et Lockwood, 1993]. Ces algorithmes récursifs stockent à tout moment les correspondances entre chaque référence et chaque hypothèse de commencement pour les suites de vecteurs d'échantillons. Cette méthode a donné de bons résultats et elle a l'avantage de s'effectuer de manière synchrone. Cependant elle reste peu évolutive et n'offre pas de moyen direct d'intégrer ou de commander des probabilités d'émission.

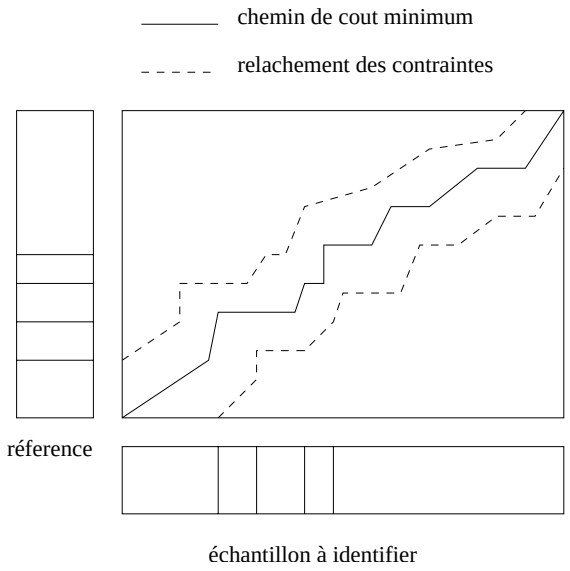

FIG. 2.3 – Programmation dynamique

La programmation dynamique, en tant qu'algorithme d'alignement, sert notamment à la détection de mots clés à base de treillis de phonèmes. [James et Young, 1994] a utilisé un système de reconnaissance HMM basé sur un algorithme de Viterbi modifié afin d'obtenir une décomposition intermédiaire compacte de chaque expression. Cette forme intermédiaire est un treillis de phonèmes dans lequel plusieurs hypothèses de phonèmes sont stockées pour chaque point à travers la parole. L'étape de détection devient une programmation dynamique symétrique qui essaye de faire la correspondance entre les prononciations des mots clés et le treillis avec des pénalités pour les phénomènes d'insertion, d'omission et de substitution.

Silaghi [Silaghi et Bourlard, 2000] propose une nouvelle approche pour la détection de mots clés sans modélisation explicite des mots non clés. Cette approche est basée sur la technique de mesure de confiance en utilisant les probabilités *a posteriori* locales. Elle cherche le segment de

parole qui maximise l'observation moyenne *a posteriori* calculée sur le chemin le plus probable. Ce problème est généralement résolu avec un processus de programmation dynamique très complexe. Les auteurs proposent une nouvelle forme itérative de l'algorithme de décodage de Viterbi (appelé IVD pour "Iterating Viterbi Decoding") afin de résoudre ce problème d'optimisation avec un simple processus de programmation dynamique.

Soient une séquence d'observations acoustiques $X = \{x_1, \dots x_n \dots x_N\}$ dans laquelle on veut détecter un mot clé et M le modèle HMM du mot clé m qui consiste en L états $Q = \{q_1, \dots, q_l \dots q_L\}$. On suppose que dans cette séquence on ne peut détecter qu'un seul mot clé m dont la sous-séquence correspondante est : $X_b^e = \{x_b, \dots x_e\}$ avec $(1 \leq b \leq e \leq N)$. Le problème est vu comme un problème de mise en correspondance entre la séquence X de longueur N et le modèle HMM $\bar{M}$ qui consiste en la séquence d'états $\bar{Q} = \{q_G \dots q_G, q^b, q^{b+1} \dots, q^e, q_G \dots q_G\}$ où on a $b-1$ états poubelles q_G qui précédent q^b et $N-e$ états poubelles qui suivent l'état q^e. Ces états émettent respectivement les séquences des vecteurs X_1^{b-1} et X_{e+1}^N associés aux segments des mots non clés. Étant donnée une estimation de la probabilité $P(q_G|x_n)$ (obtenue par exemple, en utilisant les fonctions de densités de probabilités apprises sur les mots non clés), le chemin optimal $\bar{Q}^*$ (et par conséquent e^* et b^*) est donc donné par :

$$\bar{Q}^* = argmin_{\bar{Q} \in \bar{M}} \{-logP(Q|X_b^e) - \sum_{n=1}^{b-1} logP(q_G|x_n) - \sum_{n=e+1}^{N} logP(q_G|x_n)\}$$

Cette équation peut être résolue par une application directe de la programmation dynamique. Le problème principal est de trouver la meilleure estimation de la probabilité $P(q_G|x_n)$ afin de minimiser l'erreur introduite par les différentes approximations. L'algorithme IVD estime la probabilité $P(q_G|x_n)$ d'une façon itérative (voir algorithme 4).

Algorithme 4 Algorithme IVD

1. Soit $\epsilon = -log(P(q_G|x_n))$

2. Initialisation de ϵ à $-log$ du maximum des probabilités *a posteriori* locales $P(q_k|x_n)$ pour chaque trame x_n.

3. À l'itération t, on cherche le chemin optimal $(\bar{Q}_t, b_t, e_t)$ selon l'équation précédente étant donnée la valeur estimée ϵ de $P(q_G|x_n)$.

4. Pour $t = t+1$, la valeur estimée de ϵ_t est définie comme étant la moyenne des probabilités *a posteriori* locales le long du chemin optimal Q_t.

$$\epsilon_{t+1} = \frac{1}{e_t - b_t + 1} logP(Q_t|X_{b_t}^{e_t})$$

5. Retourner à l'étape (2) et itérer jusqu'à convergence. Si nous ne sommes pas intéressés à une segmentation optimale, ce processus peut être arrêté dès que ϵ atteint un seuil minimum qui nous permet de déclarer que le mot clé en question est détecté.

La preuve de la convergence de cet algorithme IVD est donnée dans [Silaghi et Bourlard, 1999]. Il est caractérisé par une convergence rapide et il est plus efficace, en terme de temps de calcul et d'espace mémoire demandés, qu'un algorithme de programmation dynamique prenant en compte toutes les combinaisons possibles des états de début et de fin.

2.3.2 Algorithme de Viterbi et de Baum-Welch

Le modèle HMM peut être utilisé pour réaliser la reconnaissance ou la détection de mots clés. La seule différence revient dans le choix du critère de décision. Dans le cas de la reconnaissance de la parole, le but est de trouver la séquence d'observations la plus proche d'un modèle donné. Ceci est approximé par la recherche de la meilleure séquence d'états pour trouver le mot correspondant en utilisant l'algorithme de Viterbi. Pour la détection de mots clés, le but est de déterminer la probabilité qu'un mot clé soit présent à un instant donné.

Rohlicek [Rohlicek *et al.*, 1989] présente une méthode pour estimer la probabilité de détecter un modèle de mot clé à l'instant t. Le score $S_f(w,t)$ du mot w à l'instant t utilise seulement le score $\alpha(s,t)$ de l'algorithme de Baum-welch à l'état s, tel que :

$$S_f(w,t) = \frac{\alpha(e_w,t)}{\sum_s \alpha(s,t)}$$

Où e_w est le dernier état du mot w.

La sommation se fait sur l'ensemble de tous les états s. Le maximum local est recherché dans ces scores pour déterminer l'éventuelle apparition d'un mot clé.

La probabilité *forward-backward* mesure la probabilité qu'un mot clé se termine à un instant t, étant donné l'ensemble des observations et le modèle du mot. Cette estimation est calculée, une seconde fois, par l'algorithme de Baum-Welch, cette fois ci en utilisant une recherche en arrière. Pendant l'étape "forward", les scores $\alpha(s,t)$ ont été calculés. De manière similaire, une fois la séquence est achevée, le calcul des $\beta(s,t)$ se fait en sens inverse. Les scores avant et arrière sont alors combinés pour donner le score total $S_{fb}(w,t)$ du mot, qui n'est autre que la probabilité que le mot clé w se termine à l'instant t, étant données toutes les observations :

$$S_{fb}(w,t) = \frac{\alpha(e_w,t)\beta(e_w,t)}{\sum_s \alpha(s,t)\beta(s,t)}$$

Ces scores combinent implicitement les scores acoustiques et lexicaux.

Jouvet [Jouvet et Monné, 1999], combine deux approches pour la reconnaissance des épellations prononcées à travers une ligne téléphonique. La première approche est basée sur un algorithme *forward-backward* et la deuxième utilise une procédure d'extraction à base d'un HMM discret. Dans cette dernière approche, tout d'abord un décodage du signal de la parole est réalisé afin de fournir la séquence de lettres contenant les erreurs de substitution, d'insertion et de suppression. Ensuite, sachant la séquence de lettres reconnue, la procédure d'extraction recherche le nom le plus probable dans le lexique. Diverses procédures d'extraction ont été proposées dans

la littérature, la meilleure performance a été obtenue en utilisant un modèle de Markov discret dans lequel les lettres reconnues représentent ses sorties. Chaque lettre est modélisée par un modèle HMM à deux états et trois transitions. De même, les erreurs d'insertion, de substitution et de suppression sont modélisées par des fonctions de densités de probabilités. Ce formalisme permet d'utiliser des procédures d'apprentissage de HMM pour estimer les valeurs optimales des paramètres de chaque modèle. Après le processus d'apprentissage, ces modèles représentent les erreurs de reconnaissance observées au niveau du résultat obtenu.

2.3.3 Utilisation des traces d'alignements

D'autres travaux ont été réalisés en se basant sur l'utilisation des traces d'alignements. La trace du modèle sur le signal est constituée des informations qui proviennent des statistiques sur l'utilisation des fonctions de densité de probabilité le long du chemin optimal de Viterbi. Parmi ces travaux citons Mathan [Mathan, 1991] qui a étudié une technique de rejet basée sur la classification des traces d'alignement (post-traitement). Les traces utilisées contiennent le nombre et la valeur moyenne des trames observées par gaussienne le long du chemin optimal, le score obtenu par le meilleur modèle et la différence de score entre les deux meilleurs modèles. La décision d'accepter ou de rejeter la trace est réalisée par un réseau connexionniste classifieur binaire (PMC : Perceptron Multi-Couches) à deux sorties, l'une pour l'acceptation, l'autre pour le rejet.

Mauuary [Mauuary, 1994], avec une modélisation à base d'allophones, a également testé l'utilisation de traces, limitées aux durées de réalisation de chaque phonème (c'est-à-dire le nombre de trames mises en correspondance avec les gaussiennes de chacun des phonèmes). La technique s'est avérée moins efficace sur les données d'exploitation, cependant elle a réalisé de bonnes performances de rejet en effectuant une classification des traces par l'algorithme des plus proches voisins.

Chigier [Chigier, 1992] propose l'utilisation de classifieurs gaussiens sur des données constituées du score de reconnaissance des mots clés et des modèles poubelles, de la différence des scores, des scores normalisés et de la durée. Il a testé trois types de classificateurs : une monogaussienne par mot clé, avec une matrice de covariance diagonale et un autre avec une matrice pleine et un classifieur à mélange de gaussiennes avec une matrice de covariance diagonale. Les paramètres des monogaussiennes étaient estimés au maximum de vraisemblance et les paramètres des mélanges de gaussiennes étaient estimés par un apprentissage correctif avec descente de gradient afin de maximiser le pourcentage de détections correctes. Les résultats obtenus par ces trois types de classificateurs sont très proches.

2.3.4 Apprentissage discriminant

Pour améliorer la discrimination entre les mots clés et les mots HV, plusieurs études ont introduit la technique de l'apprentissage discriminant. Rose [Rose, 1992], maximise la différence entre le *log* de la probabilité du mot clé et celle du modèle poubelle, grâce à une modification des paramètres des modèles par une procédure corrective visant à maximiser l'information mutuelle entre les modèles et les séquences acoustiques [Bahl *et al.*, 1986]. Torre [Torre et Acero, 1994] utilise une procédure d'apprentissage discriminant visant à minimiser une fonction de coût pondérant différemment les erreurs de substitution, de faux rejet et de fausse acceptation. Cette procédure assimile chaque probabilité d'erreur à une transformation sigmoïdale de la différence de probabilité d'émission entre les deux meilleurs modèles considérés (c'est-à-dire deux modèles

de mots clés dans le cas de la probabilité de substitution, le modèle du mot clé et le modèle poubelle dans le cas de la probabilité de fausse acceptation ou de faux rejet). Les paramètres des modèles sont optimisés par descente de gradient.

Sukkar [Sukkar et Wilpon, 1993] introduit un système hybride qui consiste en une analyse discriminante à deux étapes. La première étape utilise un apprentissage basé sur la descente probabiliste généralisée (GPD). La seconde étape réalise une combinaison linéaire, de la sortie de la première étape avec les scores obtenus par les HMM. Ainsi, le pouvoir de discrimination du HMM est combiné avec celui de la GPD, ceci sera utilisé pour réaliser la classification mot clé/mot non-clé.

Dans la formalisation originale, la GPD a été développée pour minimiser les erreurs de reconnaissance en optimisant les paramètres du HMM [Chou *et al.*, 1992]. La GPD définit en premier lieu une fonction de distance et en second lieu, une fonction de perte (fonction de la fonction distance) qui doit être minimisée. L'intérêt de cette fonction est d'accentuer la séparation de classes dans le cas où ces dernières sont fortement corrélées. La fonction de distance dépend elle même d'une fonction de discrimination définie par :

$$R_j(x_j,\ a_{i,j}) = x_j^t\ .\ a_{i,j} \quad Avec \quad j \in \{1, 2, ...N, g\}$$

où x_j est le vecteur de discrimination du modèle du mot clé numéro j obtenu par une concaténation des vecteurs moyens de tous les états du HMM, N le nombre de mots clés, g l'indice du modèle poubelle, i l'index du HMM du mot clé dont le score du nœud terminal est le plus grand et enfin $a_{i,j}$ est le vecteur poids de la GPD. Le but est de déterminer $a_{i,j}$, pour $j \in \{1, 2, ...N, g\}$, de manière à ce que, si l'expression ne contient pas de mot clé, alors nous obtenons la formule suivante :

$$max_j R_j(x_j,\ a_{i,j}) = R_g(x_g,\ a_{i,g}) \quad Avec \quad j \in \{1, 2, ...N, g\}$$

De la même manière, si l'expression contient un mot clé et si ce mot est correctement reconnu, l'équation devient :

$$max_j R_j(x_j, a_{i,j}) = R_i(x_i, a_{i,i}) \quad Avec \quad j \in \{1, 2, ...N, g\}$$

La fonction distance d'un mot non clé est donnée par la formule suivante :

$$d_i(X, A_i) = -R_g(x_g, a_{i,g}) + log[e^{\eta R_i(x_i, a_{i,i})} + e^{\eta R_k(x_k, a_{i,k})}]^{\frac{1}{\eta}}$$

où

$$X = [x_1, x_2, ..., x_N, x_g]$$

$$A_i = [a_{i,1}, a_{i,2}, a_{i,3}, ...a_{i,N}, a_{i,g}]$$

η est un nombre positif et k est l'index du modèle du mot clé avec le second meilleur score au niveau du nœud terminal.

Dans le cas d'un mot clé cette distance s'écrit de la façon suivante :

$$d_i(X, A_i) = -R_i(x_i, a_{i,i}) + log[e^{\eta R_g(x_g, a_{i,g})} + e^{\eta R_k(x_k, a_{i,k})}]^{\frac{1}{\eta}}$$

La fonction de perte est donnée par la relation suivante :

$$L_i(X, A_i) = \frac{1}{1+e^{-\gamma\, d_i(X, A_i)}} \qquad \text{où } \gamma \text{ est une constante}$$

Une procédure itérative basée sur le gradient est utilisée pour minimiser la fonction de perte.

$$(A_i)_{n+1} = (A_i)_n - \varepsilon \nabla L_i(X_n, (A_i)_n)$$

où $\nabla L_i(X_n, (A_i)_n)$ est le gradient de la fonction perte relative à A_i évalué au n^{ieme} échantillon d'apprentissage X_n et ε la taille adaptée du pas d'apprentissage.

De bonnes performances ont été réalisées avec la même approche pour la vérification de chiffres enchaînés [Sukkar, 1994].

2.3.5 Seuil sur les scores de reconnaissance

Parmi les travaux de détection de mots clés, nous trouvons, ceux qui sont basés sur un seuil et dans lesquels le rejet ou l'acceptation se fait en comparant le score d'un mot à ce seuil. Mazor [Mazor et Feng, 1993], décrit l'application d'un seuil aux diverses quantités, en incluant la vraisemblance des hypothèses des mots clés et la différence entre la vraisemblance de la meilleure hypothèse de mots clés et de celle qui la suit dans la tâche de vérification.

Rivlin [Rivlin *et al.*, 1996] est le premier à utiliser une approche basée sur l'estimation de la probabilité *a posteriori* du phonème. Il utilise le logarithme de ces probabilités *a posteriori* sur un intervalle d'hypothèses de phonèmes afin de calculer une mesure de confiance au niveau phonème. Une mesure de confiance au niveau du mot est créée en utilisant les mesures des phonèmes qu'ils constituent.

Bayya [Bayya, 2000] propose d'effectuer le rejet des mots HV en utilisant une mesure de confiance qui est une fonction de la distance entre le meilleur score obtenu et les K-meilleurs scores suivants.

Moreau [Moreau *et al.*, 2000] propose une méthode qui consiste en un post-traitement des hypothèses de reconnaissance par le calcul d'une mesure de confiance pour chaque hypothèse. Cette mesure est basée sur le rapport de vraisemblance au niveau le plus élémentaire qui est le niveau des trames acoustiques.

Soit w, le résultat du décodage du signal d'entrée X, le rapport de vraisemblance s'écrit sous la forme suivante :

$$LR(X|w) = \frac{p(X \mid w\ correct)}{p(X \mid w\ incorrect)}$$

En fixant un seuil w_o, nous avons les décisions suivantes :

$$LR(X \mid w) \geq w_o \Rightarrow w\ \ est\ accepté$$
$$LR(X \mid w) < w_o \Rightarrow w\ \ est\ rejeté$$

Pour calculer ce rapport de vraisemblance, il a estimé que ce dernier sera la combinaison des rapports de vraisemblances calculés au niveau des trames acoustiques. Pour cela, il a effectué l'alignement de l'entrée X sur le modèle de Markov associé à l'hypothèse w, afin d'associer à chaque trame x_i du signal $X = (x_1, ..., x_T)$ un état acoustique q_i de la séquence d'états $Q = (q_1, ..., q_T)$.

On définit le rapport de vraisemblance au niveau de la trame x_i par l'équation suivante :

$$LR(x_i \mid q_i) = \frac{p(x_i \mid Mq_i)}{p(x_i \mid M_{\bar{q}_i})}$$

où $p(x_i \mid Mq_i)$ et $p(x_i \mid M\bar{q}_i)$ sont respectivement les scores de x_i sachant le modèle des événements corrects associés à l'état q_i et le modèle des événements incorrects associés au même état q_i.

Ainsi le rapport de vraisemblance global s'écrit de la façon suivante :

$$LR(X|w) = \frac{\prod_{i=1}^{T} p(x_i \mid Mq_i)}{\prod_{i=1}^{T} p(x_i \mid M_{\bar{q}_i})}$$

La mesure de confiance $CM(w)$ de l'hypothèse w sera le log du rapport de vraisemblance global normalisé par le nombre de trames acoustiques T :

$$CM(w) = \frac{1}{T}\ \ Log[LR(X|w)]$$

Afin de calculer cette mesure de confiance, il est nécessaire de faire l'apprentissage du modèle Mq_i et celui du modèle des événements incorrects (ou anti-modèle) M$\bar{q}_i$ pour chaque état q_i. La principale difficulté réside dans la détermination de l'anti-modèle qui doit modéliser les différents types d'événements incorrects. Pour cela, trois densités ont été apprises pour chaque état q_i : $\bar{q}_{i(sub)}$ pour les erreurs de substitution, $\bar{q}_{i(hv)}$ pour les erreurs de fausses acceptations sur les mots hors-vocabulaire et $\bar{q}_{i(br)}$ pour les erreurs de fausse acceptations sur les bruits. Ces densités sont estimées à partir des trames acoustiques associées à l'état q_i au sein d'alignements incorrects (fausses acceptations sur les mots hors-vocabulaire ou sur bruit, les substitutions). Plusieurs combinaisons de ces trois densités ont été testées dans [Moreau et Jouvet, 1999]. Dans le cas présenté, on a choisi la moyenne arithmétique des vraisemblances :

$$P(x|M_{\bar{q}_i}) = \frac{1}{3}[M_{\bar{q}_{i(sub)}}(x) + M_{\bar{q}_{i(hv)}}(x) + M_{\bar{q}_{i(br)}}(x)]$$

Le modèle Mq_i est représenté par une densité de probabilité apprise de la même manière que les densités précédentes, à partir des trames associées à l'état q_i dans un corpus d'alignements corrects.

Toutes ces densités sont estimées dans le même espace de paramètres acoustiques que celui de la modélisation markovienne, c'est-à-dire les coefficients cepstraux et leurs dérivées premières et secondes. Ceci représente un avantage de cette méthode puisqu'elle ne nécessite pas l'extraction d'autres paramètres de post-traitement supplémentaire.

2.3.6 Méthodes d'adaptation

Pour réaliser la tâche de la détection de mots clés, quelques chercheurs ont travaillé sur la stratégie d'adaptation. Parmi eux, citons Gupta [Gupta et Soong, 1998] qui montre qu'un seuil adaptatif, basé sur la longueur des expressions, fournit une amélioration par rapport au seuil statique pour une tâche de reconnaissance de chiffres.

Moreau [Moreau *et al.*, 2000] résoud le problème de rejet des données incorrectes dans une tâche de grand vocabulaire par l'utilisation d'un algorithme d'adaptation incrémentale pour adapter les modèles des mots et ceux des poubelles. L'approche qu'il a proposée utilise un modèle poubelle pour capturer les mots hors-vocabulaire, il étudie l'impact de l'adaptation des modèles des mots et des modèles poubelles aux données du domaine qui sont collectées durant la phase d'exploitation du système. Il suppose que l'utilisation des données spécifiques à la tâche améliore les performances du système de reconnaissance et particulièrement sa capacité de rejeter les mots incorrects. La technique proposée est basée sur l'algorithme EM de segmentation incrémentale. L'adaptation incrémentale est utilisée dans ce cas afin d'adapter un modèle HMM aux données du domaine. Cette adaptation consiste en l'estimation de nouveaux paramètres des fonctions de densités de probabilités par une procédure itérative à deux étapes. La première consiste à aligner les trames d'apprentissage sur le modèle donnant la séquence optimale. La deuxième calcule les nouveaux paramètres du modèle en utilisant la base complète (la base initiale $X^{(Init)}$ et la base d'apprentissage $X^{(Adapt)}$).

L'objectif de cette adaptation est l'estimation de l'ensemble des paramètres λ qui maximise conjointement les paramètres du modèle, les séquences d'états S_{Adapt} (pour les données d'adaptation) et la séquence des états S_{Init} (pour la base initiale qui reste inchangée). Ainsi, il maximise la quantité suivante :

$$\lambda^{opt} = argmax_{\lambda}\ [max_{S_{Adapt}}\ \ P(\lambda,\ S_{Init}^{opt(\lambda_{init})},\ S_{Adapt}|X^{(Init)},\ X^{(Adapt)})]$$

À chaque itération, les formules de ré-estimation des paramètres (la moyenne : μ et l'écart type : σ^2) pour chaque fonction de densité gaussienne sont données par :

$$\mu = \frac{\sum_{i \in Init} X_i + \sum_{j \in Adapt} X_j}{N_{Init} + N_{Adapt}}$$

$$\sigma^2 = \frac{\sum_{i \in init} X_i^2 + \sum_{j \in Adapt} X_j^2}{N_{Init} + N_{Adapt}} - \mu^2$$

$\{X_i\ ,\ i \in Init\}$ est l'ensemble des trames N_{Init} de la base initiale aligné avec la fonction de densité de probabilité (fdp) (cet ensemble de trames est fixé) et $\{X_j\ ,\ j \in Adapt\}$ est l'ensemble des trames N_{Adapt} de la base d'adaptation aligné avec cette fdp (cet ensemble de trames est déterminé à chaque itération).

Cette approche a permis une amélioration des performances du système de reconnaissance puisque les modèles des mots et le modèle de poubelle qu'il utilise sont plus spécifiques au domaine grâce à l'algorithme d'adaptation incrémentale.

2.3.7 Connaissances acoustiques et linguistiques

En plus de l'utilisation des méthodes basées uniquement sur des connaissances acoustiques, nous trouvons des travaux qui combinent les deux types de connaissances, acoustiques et linguistiques. En effet, la partie linguistique présente aussi une information significative et il arrive parfois qu'un mot ayant un score acoustique faible soit correctement reconnu grâce au modèle de langage utilisé. Les mesures de confiance acoustiques s'avèrent donc, dans de tels cas, insuffisantes. Rose [Rose *et al.*, 1998] propose une nouvelle méthode qui combine les deux types de connaissances : linguistiques et acoustiques. Dans cette approche, on incorpore la notion de mesure de confiance acoustique dans l'automate stochastique utilisé pour décrire le modèle de langage n-gram du système de reconnaissance [Riccardi *et al.*, 1996]. Dans un cas simple, un état de cet automate peut correspondre au contexte du mot w_i et le poids d'un arc peut correspondre à la probabilité de produire w_i sachant le mot précédent. La méthode proposée étend la notion d'état pour inclure non seulement le contexte, mais aussi une représentation discrète de la confiance acoustique c_i correspondante à l'histoire du mot w_i. On ajoute donc un état qui correspond à la confiance acoustique étendant ainsi l'espace des états de l'automate stochastique considéré. Par exemple, si on considère un modèle de langage tri-gram où la probabilité d'un mot

w_i sachant son histoire h est approximée à $P(w_i|w_{i-1}, w_{i-2})$, alors ce modèle sera étendu et la même probabilité sera représentée par $P(w_i|w_{i-1}, c_{i-1}, w_{i-2}, c_{i-2})$ où c_i est une variable discrète, $c_i \in \{0, 1\}$ qui exprime la confiance acoustique de l'histoire du mot w_i. Si $c_i = 0$, la confiance acoustique accordée est faible sinon cette confiance est forte.

Hernandez [Hernandez-Abrego et Marino, 2000], explore l'influence des informations contextuelles sur les mesures de confiance pour les résultats de la reconnaissance de la parole continue. Il a proposé une approche à trois étapes. Tout d'abord, il effectue l'extraction de trois mesures de confiance acoustiques à la sortie des résultats de reconnaissance. Ensuite, ces mesures sont compilées à l'aide d'un système d'inférence flou qui prend en entrée ces trois types de mesures de confiance et fournit en sortie une seule mesure de confiance acoustique floue comprise entre 0 et 1. Les paramètres de ce moteur sont estimés directement à partir des exemples avec une stratégie d'évolution. Enfin, au niveau du modèle de post-traitement, on intègre l'information linguistique qui va être utilisée pour ré-estimer la mesure de confiance de chaque mot w_i. Cette nouvelle mesure de confiance est calculée comme étant le produit de la mesure acoustique fournie par le moteur d'inférence et un coefficient de proportionnalité $S(w_i)$ comme c'est indiqué par l'équation suivante :

$$C_f(w_i) = C_i(w_i) * S(w_i)$$

$C_f(w_i)$ est la nouvelle mesure de confiance ré-estimée et $C_i(w_i)$ est la mesure de confiance acoustique déjà calculée. $S(w_i)$ est un coefficient de proportionnalité au niveau duquel on fait intervenir l'information contextuelle puisqu'il dépend des probabilités contextuelles $P(w_i|w_{i-1})$ et $P(w_{i+1}|w_i)$. Ce coefficient est calculé à l'aide d'un autre moteur flou qui prend en entrée ces deux probabilités ainsi que les mesures de confiance acoustiques des mots w_{i+1} et w_{i-1}. Cette méthode nous fournit alors une nouvelle mesure de confiance intégrant deux types de connaissances : linguistiques et acoustiques.

Wessel [Wessel *et al.*, 2001] [Wessel *et al.*, 1999], présente plusieurs mesures de confiance pour la reconnaissance de la parole continue grand vocabulaire. Il propose d'estimer la mesure de confiance du mot hypothèse directement en tant que sa probabilité *a posteriori*, connaissant toutes ses observations acoustiques. Ces probabilités sont calculées en utilisant un graphe de mots et en appliquant un algorithme $forward - backward$.

Pour calculer la probabilité *a posteriori* d'un mot hypothèse, il est utile d'introduire les limites explicites entre les mots dans une séquence de mots $w_1^M = w_1, ..., w_M$ pour une séquence d'observations acoustiques $x_1^T = x_1, ..., x_T$. On note τ le temps de début et t le temps final du mot w. Avec ces définitions $[w; \tau, t]$ est une hypothèse spécifique pour le mot w . Une séquence de M mots d'hypothèses peut être alors donnée par :

$$[w, \tau, t]_1^M = [w_1, \tau_1, t_1], ..., [w_M, \tau_M, t_M]$$

où $\tau_1 = 1, t_M = T$ et $t_{n-1} = \tau_n$ pour $n = 2, ..., M$.

On suppose que la génération des observations acoustiques : $x_{\tau_m}^{t_m} = x_{\tau_m}, ..., x_{t_m}$ dépend uniquement du mot w_m. La probabilité *a posteriori* $p([w;\tau,t] \mid x_1^T)$ pour une hypothèse de mot $[w;\tau,t]$ peut être calculée par la sommation des probabilités *a posteriori* de toutes les phrases qui contiennent le mot hypothèse $[w;\tau,t]$ comme le montre l'équation suivante :

$$p([w;\tau,t] \mid x_1^T) = \sum_{\substack{[w,\tau,t]_1^M \\ \ni\, n \in \{1,\ldots,M\} \\ [w_n,\tau_n,t_n] = [w,\tau,t]}} \frac{\prod_{m=1}^{M} [p(x_{\tau_m}^{t_m} \mid w_m).p(w_m \mid w_1^{m-1}]}{p(x_1^T)}$$

Dans l'algorithme de *forward-backward*, on calcule la probabilité *forward* et la probabilité *backward* du mot hypothèse et on combine les deux probabilités dans la probabilité *a posteriori* du mot hypothèse. Contrairement à l'algorithme de *forward-backward* classique qui travaille au niveau d'état HMM, l'algorithme de *forward-backward* utilisé procéde directement au niveau du mot hypothèse. Si on suppose qu'on a un graphe de mots et qu'on utilise un modèle de langage *m-gram* et que $h_2^{m-1} = h_2, ..., h_{m-1}$ sont les mots prédécesseurs du mot w, on peut calculer la probabilité $forward\ \Phi(h_2^{m-1}; [w;\tau,t])$ sachant que la dernière hypothèse d'une séquence de n mots d'hypothèses est $[w;\tau,t]$ et que leur histoire est h_2^{m-1}.

$$\Phi(h_2^{m-1}; [w;\tau,t]) = p(x_\tau^t \mid w). \sum_{h_1} \sum_{\acute{\tau}} \Phi(h_1^{m-2}; [h_{m-1}; \acute{\tau}, \tau - 1]).p(w \mid h_1^{m-1})$$

Puisque τ est le temps de début du mot w, $\tau - 1$ indique le temps de fin du mot précédent h^{m-1}.

De la même manière, supposons que f_1^{m-2} représente les m-2 mots successeurs immédiats du mot w. Avec cette définition, on peut calculer la probabilité *backward* $\Psi([w,\tau,t]; f_1^{m-2})$ sachant que la première hypothèse de la séquence des n mots d'hypothèses est $[w;\tau,t]$ et que leur futur est f_1^{m-2}.

$$\Psi([w;\tau,t], f_1^{m-2}) = p(x_\tau^t \mid w). \sum_{f_{m-1}} \sum_{\acute{t}} \Psi([f_1; t+1, \acute{t}]; f_2^{m-1}).p(f_{m-1} \mid w f_1^{m-2})$$

D'après les équations précédentes, la probabilité *a posteriori* d'une hypothèse peut être calculée en sommant toutes les probabilités histoires et futurs du mot hypothèse $[w;\tau,t]$:

$$p([w;\tau,t] \mid x_1^T) = \sum_{h_2^{m-1}} \sum_{f_1^{m-2}} \frac{\Phi(h_2^{m-1}; [w;\tau,t]).\Psi([w;\tau,t], f_1^{m-2})}{p(x_1^T).p(x_\tau^t \mid w)}. \prod_{i=1}^{m-2} p(f_i \mid h_{i+1}^{m-1} w f_1^{i-1})$$

La probabilité *a posteriori* définie par cette équation peut être utilisée maintenant comme une mesure de confiance pour chaque mot hypothèse :

$$C([w;\tau,t]) = p([w;\tau,t] \mid x_1^T)$$

2.3.8 Réseaux de neurones

La deuxième grande classe de méthodes en reconnaissance automatique de la parole est basée sur les réseaux de neurones. Nous présentons dans ce que suit quelques travaux réalisés dans la détection de mots clés en se basant sur les Réseaux de Neurones Artificiels (RNA). Morgan [Morgan *et al.*, 1991], introduit l'utilisation des RNA dans la recherche de mots clés. Tout d'abord, il utilise un système standard pour détecter les mots clés et obtenir les régions susceptibles de contenir un mot clé. Ensuite, il utilise un réseau de neurones pour la décision. Clary [Clary et Hansen, 1992], utilise un RNA pour combiner les vecteurs acoustiques successifs afin de créer un nouveau vecteur, qui sera utilisé par la suite dans un modèle de Markov semi-continu pour la détection des mots clés. Zeppenfeld [Zeppenfeld et Waibel, 1992], utilise un RNA temporel (TDNN) avec les méthodes classiques de programmation dynamique. Il emploie pour chaque mot clé un RNA dont les sorties sont fournies à un algorithme de programmation dynamique pour faire la détection de mots clés. La quantification vectorielle et le mélange de gaussiennes peuvent très bien s'appliquer à la détection de mots clés en les utilisant comme facteurs de complément et d'optimisation [Tadj, 1995].

Bernardis [Bernardis et Bourlard, 1998], utilise un système hybride HMM/RNA. Il montre que l'utilisation de la probabilité *a posteriori* locale accumulée (obtenue à partir de la sortie d'un Perceptron multi-couches) normalisée par le nombre de trames contenus dans le segment du mot a réalisé de bonnes mesures de confiance et de bons scores pour la ré-estimation des N-meilleurs hypothèses. Ceci est confirmé par Williams [Williams et Renals, 1997], qui utilise le même type de mesures de confiance en les comparant avec plusieurs autres approches.

Charlet [Charlet *et al.*, 2001] présente une technique de combinaison de mesures de confiance utilisant un Percepton Multi-Couches (PMC). Dans cette approche, on suppose que chaque segment de parole composant une hypothèse de reconnaissance w (un mot ou une séquence de mots) est décrit par un ensemble de caractéristiques phonétiques comme voisé\non-voisé, voyelle\consonne etc. On calcule alors un score représentant chacune de ces caractéristiques qu'on combine après en utilisant un PMC afin d'extraire un score global pour cette hypothèse de reconnaissance.

En premier lieu on utilise un RNA flou pour calculer la probabilité discrète d'une caractéristique phonétique spécifique j du segment i notée x_i^j où chaque segment phonétique φ_i est donné par l'alignement de Viterbi associé à une hypothèse. Donc, pour une hypothèse X, reconnue comme w, on calcule le rapport statique de vraisemblance logarithmique suivant :

$$LLR_j(X \mid w) = \sum_{i}^{nbseg} Log(\frac{P(x_i^j \mid M_{\varphi_i})}{P(x_i^j \mid M_{\bar{\varphi}_i})})$$

où M_{φ_i} et $M_{\bar{\varphi}_i}$ sont respectivement le modèle et l'anti-modèle associés au segment phonétique φ_i et $nbseg$ le nombre de segments composants l'hypothèse X.

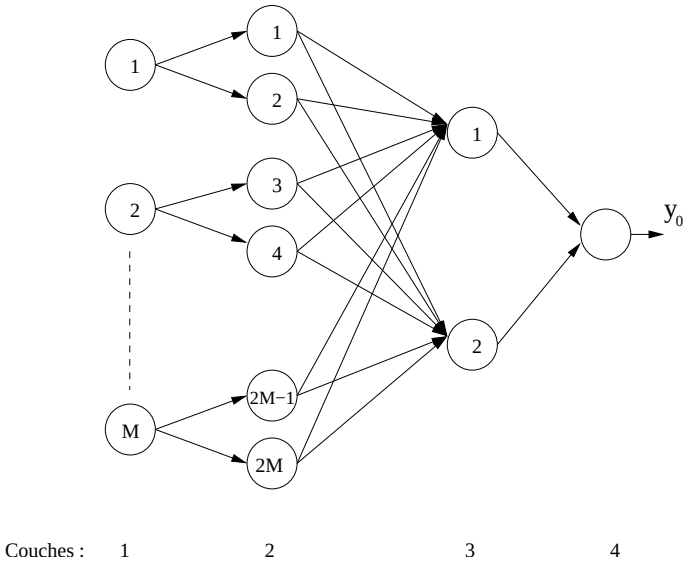

FIG. 2.4 – Architecture du réseau de neurone flou

Dans la figure 2.4, nous donnons l'architecture du RNA utilisé pour décider si une caractéristique phonétique donnée est présente ou non. Ce réseau est dérivé du RNA flou de Glorennec [Glorennec, 1993].

La première couche du réseau de neurone est composée de 27 neurones qui correspondent à 8 coefficients MFCC, le logarithme de l'énergie de la trame et leurs dérivées premières et secondes. Le rôle de la deuxième couche est de calculer les valeurs d'adhésion associées à chaque entrée et à chaque sous-ensemble flou. La troisième couche est composée seulement de deux neurones. Chaque neurone est connecté à tous les neurones de la couche précédente. La dernière couche est composée d'un seul neurone calculant la sortie finale y_0 par la méthode du centre de gravité.

$$y_0 = \frac{\sum_i \alpha_i p_i}{\sum_i \alpha_i}$$

où y_0 représente la probabilité recherchée, les p_i et les α_i représentent respectivement les poids des connexions et les sorties des neurones de la couche précédente. Les valeurs de sortie sont normalisées par le nombre de trames composants le segment de parole afin de trouver un score final pour ce segment.

Les paramètres du RNA sont appris en utilisant l'algorithme de rétro-propagation afin de minimiser l'erreur quadratique moyenne. Durant l'apprentissage, la sortie est soit 0 soit 1, selon

la présence ou l'absence théoriques de la caractéristique phonétique pour le phonème associé au segment phonétique.

En deuxième lieu, on combine le score obtenu avec le Log de la vraisemblance du HMM : $LL_{HMM}(X \mid w)$, afin d'obtenir le score global $Sc_j(X \mid w)$ associé à la caractéristique phonétique j :

$$Sc_j(X \mid W) = \alpha \;\; LL_{HMM}(X \mid w) + (\alpha - 1) \;\; LLR_j(X \mid w)$$

où α est un coefficient d'interpolation optimisé sur l'ensemble d'apprentissage pour minimiser le taux d'erreur de la reconnaissance.

Enfin, les scores globaux $Sc_j(X|w)$ sont combinés pour en dériver une seule mesure de confiance :

$$Sc_{NN}(X \mid w) = f(Sc_1(X \mid w), ..., Sc_j(X \mid w), ..., Sc_n(X \mid w))$$

où n est le nombre des mesures de confiance.

La fonction f est approximée par l'utilisation du PMC. La couche d'entrée du réseau est composée de $n+1$ neurones, avec les n premières entrées correspondent aux scores $Sc_j(X \mid w)$ et la dernière à la constante 1. Cette couche est entièrement connectée à la deuxième couche qui est la couche cachée composée de Q neurones (dans cette application Q=6). La couche de sortie est composée d'un seul neurone dont la sortie est le score final $Sc_{NN}(X \mid w)$. Les paramètres du RNA sont estimés durant la phase d'apprentissage en utilisant l'algorithme classique de rétro-propagation du gradient d'erreur.

2.3.9 Transformations et algorithmes

D'autres études incluent l'exécution de quelques transformations linéaires et de quelques algorithmes pour l'extraction du vecteur caractéristique et pour la détection de mots clés. Voici une liste de quelques travaux.

Kamppari [Kamppari et Hazen, 2000], présente une technique de calcul de la confiance au niveau du mot fondée sur une combinaison de plusieurs caractéristiques, elles-mêmes basées seulement sur les informations acoustiques extraites d'un classifieur phonétique. Il utilise l'analyse discriminante linéaire de Fisher afin de fusionner l'ensemble des caractéristiques acoustiques en un simple score de confiance en utilisant une projection linéaire.

Vergyri [Vergyri, 2000] décrit un processus de post-traitement qui traite les caractéristiques au niveau des mots comme sources de connaissances indépendantes et les combine dans un seul modèle logarithmique linéaire pour calculer la probabilité *a posteriori* de la séquence de mots. Ce modèle est utilisé pour calculer le score de l'hypothèse. Les paramètres de ce modèle sont optimisés à l'aide d'une approche de combinaison de modèles discriminants. Cette méthode utilise elle même une méthode d'optimisation simplex afin de minimiser la fonction du taux d'erreur empirique sur la base d'apprentissage.

Maison [Maison et Gopinath, 2001] montre que la normalisation du maximum d'entropie convient très bien au rejet des mots inutiles. Il l'utilise comme fonction objective pour choisir les paramètres des mesures de confiance basées sur le graphe des mots et pour optimiser les combinaisons des différentes mesures de confiance. Il a montré que la combinaison linéaire de la technique basée sur le graphe des mots et le score acoustique donne de bonnes performances. Hazen [Hazen et Bazzi, 2001] combine deux méthodes, la première basée sur un modèle explicite pour la détection des mots HV et la deuxième identifie les mots insérés en se basant sur une mesure de confiance extraite par les systèmes de reconnaissance. Une projection discriminante linéaire simple du vecteur des caractéristiques est employée afin d'extraire une seule mesure de confiance pour chaque mot. L'apprentissage du vecteur de projection est effectué en utilisant l'erreur de classification minimale.

Zhang [Zhang *et al.*, 2001] fait une étude sur l'estimation d'une mesure de confiance pour une application de reconnaissance de la parole continue grand vocabulaire indépendamment du locuteur. Il a proposé 10 paramètres générés à partir de différents niveaux du processus de reconnaissance. Un algorithme d'analyse de fiabilité des paramètres a été développé afin d'extraire la mesure de confiance finale. Moreno [Moreno *et al.*, 2001] présente une application de l'algorithme de classification "Boosting" pour le calcul des mesures de confiance. Il dérive un vecteur de caractéristiques à partir du treillis de reconnaissance de la parole et l'introduit dans le classifieur. Ce classifieur combine des centaines de systèmes d'apprentissage simples et dérive des règles de classification pour réduire le taux d'erreur de confiance. Palmer [Palmer et Ostendorf, 2001] a étudié trois méthodes différentes pour améliorer les scores de confiance : arbre de décision, modèle linéaire généralisé et interpolation linéaire utilisée pour les sorties de la première et de la deuxième méthode.

Charlet [Charlet *et al.*, 2001], présente une technique de combinaison de mesures de confiance basée sur la fonction de régression logistique. Les mesures de confiance utilisées sont calculées au niveau de chaque segment de parole composant une hypothèse de reconnaissance w (un mot ou une séquence de mots). Elles décrivent un ensemble de caractéristiques phonétiques comme voisé\non-voisé, voyelle\consonne etc.

La technique de régression logistique utilisée permet de fusionner des mesures de confiance pour donner à la fin une réponse sous forme d'une probabilité. Elle est basée sur l'hypothèse que le rapport logarithmique de vraisemblance d'un ensemble de mesures de confiance cm_i avec $0 < i < N + 1$ peut être estimé par une combinaison linéaire comme suit :

$$log\frac{p(cm_1, ...cm_N \mid correcte)}{p(cm_1, ...cm_N \mid incorrecte)} = b_0 + b_1.cm_1 + ... + b_N.cm_N$$

Par conséquent,

$$p(correcte \mid cm_1, ...cm_N) = \frac{1}{1 + exp - (a_0 + a_1.cm_1 + ... + a_N.cm_N)}$$

$$= \frac{1}{1 + exp - (A \ . \ CM)}$$

où $A = \{a_0, a_1, ..., a_N\}$ est un vecteur de coefficients et $CM = \{1, cm_0, cm_1, ..., cm_N\}$ le vecteur des mesures de confiance.

Ainsi la régression logistique permet l'estimation de la probabilité *a posteriori* que la réponse du classifieur soit correcte en donnant toutes les mesures de confiance.

Les coefficients du vecteur A de la fonction de régression logistique sont estimés afin de maximiser l'ensemble des vraisemblances de développement :

$$L = \sum_{i=1..K} c_i \; log(p_i) + (1 - c_i) \; log(1 - p_i)$$

Où pour chaque exemple i :
- $c_i = 1$ si le test d'indice i est correct, si non $c_i = 0$,
- p_i est la probabilité *a posteriori*, que le test i soit correct, en donnant le vecteur des mesures de confiance CM_i associé au test i.

2.4 Les applications

Les recherches réalisées dans le domaine de la détection de mots clés dans un flux de parole visent généralement à faciliter l'interaction entre l'homme et la machine en détectant les mots les plus intéressant pour l'interprétation sémantique de ce qui a été prononcé. Les applications possibles dans ce domaine de recherche sont nombreuses, nous pouvons citer par exemple, les opérateurs de service automatique [Sukkar et Wilpon, 1993], où l'utilisateur demande le type de service qu'il souhaite (par exemple : apprendre les prix des actions du marché ou interroger une base de donnée téléphonique : adresse et numéro de téléphone d'une personne, hôtel etc.), les systèmes de routage téléphonique ou de classification des documents parlés etc. [Wilpon *et al.*, 1990] [Gorin *et al.*, 1997].

Une autre nouvelle application est l'indexation audio. Dans cette tâche, le système doit classer les vidéos, les enregistrements audio et les événements vidéo par leurs contenus [Jones *et al.*, 1995] [Gelin et Wellekens, 1996]. La classification est réalisée en se basant sur des occurrences suffisantes de mots clés qui caractérisent correctement le domaine de l'enregistrement audio. Ainsi, l'utilisateur est capable de faire un balayage sur une très grande base audio et d'extraire l'information appropriée sans des connaissances explicites de tout le contenu de la base de données.

Une troisième application est la surveillance des conversations téléphoniques. Dans ce cas, la tâche principale consiste à chercher des mots clés importants pour comprendre le contenu de la conversation [Yapanel, 1997].

Une importante application des systèmes de détection de mots clés, est l'initialisation de l'interaction homme-machine. Par exemple, pour des gens handicapés on peut fournir quelques

appareils électroménagers qui peuvent répondre à des mots spécifiques, donc le système détecte ces mots clés en négligeant tous les autres entrées acoustiques et répond à la demande [Pols, 1997].

2.5 Conclusion

Les recherches réalisées en matière de rejet des entrées incorrectes ont démontré l'efficacité des modèles poubelles. Cette technique s'avère d'autant plus intéressante qu'elle ne nécessite pas la modification des algorithmes de reconnaissance et qu'elle est applicable en temps réel. Ces arguments nous ont mené à choisir cette technique dans nos travaux de rejet des entrées incorrectes puisque le problème de la modélisation de la parole hors-vocabulaire s'est imposé comme problème fondamental. Dans la suite, nous proposons cependant d'autres approches modélisant autrement les entrées incorrectes et nous combinons ces différents modèles afin d'améliorer les résultats.

Le problème de détection de mots clés peut être aussi traité d'une autre manière qui consiste à ajouter un processus de vérification après la phase de reconnaissance en utilisant une mesure de confiance. La notion de mesure de confiance a été employée par plusieurs chercheurs afin de réaliser des systèmes de détection de mots clés. Cette mesure peut être calculée en utilisant différents types d'informations telles que les informations acoustiques et linguistiques. Dans la littérature, plusieurs méthodes ont été proposées pour calculer différentes mesures de confiance. Dans ce chapitre, nous avons mis l'accent sur les techniques les plus utilisées dans ce domaine, comme par exemple, la programmation dynamique, l'utilisation des algorithmes de Viterbi et de Baum-Welch, l'utilisation des traces d'alignements, l'apprentissage discriminant, les RNA et quelques autres méthodes d'adaptation. Cette grande diversité de méthodes prouve l'importance de la technique de la mesure de confiance permettant de prendre la décision d'accepter ou de rejeter un mot reconnu. Dans cette thèse, nous nous sommes intéressés à cette technique afin de réaliser un système de détection de mots clés.

Les mesures de confiance que nous proposons utilisent l'information acoustique au niveau phonème. Ce choix est justifié par le fait que ce type d'information représente les propriétés les plus caractéristiques des mots. En plus, l'utilisation des modèles acoustiques a fait l'objet de plusieurs autres travaux et a montré son efficacité vis à vis du problème de détection de mots clés. Nous avons adopté plusieurs méthodes pour calculer la mesure de confiance d'un mot donné et nous avons ensuite, combiné ces méthodes avec des modèles poubelles. L'idée est d'utiliser un modèle poubelle pour effectuer un premier filtrage des entrées, puis nous appliquons la mesure de confiance afin d'extraire seulement les mots clés réellement prononcés. La combinaison de ces deux techniques nous a permis d'améliorer les performances du système de détection.

Chapitre 3

Classification statistique

3.1 Introduction

Le problème de détection de mots clés dans un flux de parole est traité dans la littérature par deux techniques principales, qui sont les modèles poubelles et la mesure de confiance. Cette dernière consiste en un processus de vérification des mots reconnus, il s'agit donc d'une classification de ces mots en deux classes : correct et incorrect. C'est pour cette raison que nous avons pensé à utiliser une méthode de classification plus efficace que le simple seuil. Cette approche présente aussi l'avantage de pouvoir utiliser un ensemble de caractéristiques relatives à chaque mot reconnu et de les combiner au sein du classifieur afin de décider de l'acceptation ou du rejet du mot considéré. Tout d'abord nous commençons par l'utilisation des Réseaux de Neurones Artificiels (RNA) en particulier le Perceptron Multi-Couches (PMC). Ensuite nous présentons les SVM comme une technique de classification innovante et efficace et qui a fait ses preuves dans plusieurs domaines de recherche. Son utilisation dans le domaine de la détection de mots clés nous a paru très intéressante. C'est pourquoi, nous avons choisi d'employer les SVM pour classer les mots clés corrects et incorrects et pour ainsi résoudre le problème de la détection de mots clés.

Les bases des SVM ont été développées par [Vapnik, 1995]. Les SVM ont gagné de la popularité grâce à plusieurs caractéristiques attractives et des prometteuses [Pontil et Verri, 1997] [Mangasarian et Musicant, 2001] [Chapelle *et al.*, 2002]. Leur formulation incarne le principe de minimisation du risque structurel qui a été montrée meilleure que la minimisation du risque empirique traditionnellement employée par les RNA conventionnels [Gunn *et al.*, 1997]. La minimisation du risque structurel minimise la limite supérieure du risque prévu, ce qui s'oppose à la minimisation du risque empirique qui minimise l'erreur sur la base d'apprentissage. C'est cette différence qui donne aux SVM une plus grande capacité de généralisation, but de l'apprentissage statistique. Les SVM ont été développés afin de résoudre le problème de classification, mais récemment ils ont été étendus aux problèmes de régression [Mukherjee *et al.*, 1997] [Schölkopf, 2000].

Dans la première partie, nous détaillons les principes de fonctionnement du PMC comme moyen de classification très connu et bien adapté à notre problématique. Nous commençons alors par décrire le simple Perceptron et son principe d'apprentissage pour détailler ensuite le cas le plus général du PMC avec son algorithme de rétro-propagation de l'erreur.

Dans la deuxième partie de ce chapitre, nous donnons un aperçu sur la théorie d'apprentissage et nous décrivons la minimisation et la consistance du risque empirique. Ensuite, nous abordons

les problèmes de convergence et de contrôle de la capacité de généralisation des processus d'apprentissage. Enfin, nous définissons le principe de la minimisation du risque structurel sur lequel se basent les SVM.

Dans la troisième partie, nous introduisons les trois formules mathématiques sur lesquelles sont basés les SVM. Ensuite nous montrons comment ces formules peuvent être appliquées à la recherche d'un hyperplan optimal, dans le cadre de la classification. Nous détaillons cette applicabilité dans le cas des données linéairement séparables et des données non linéairement séparables. Enfin nous décrivons les principes des SVM.

La dernière partie du chapitre est consacrée aux SVM multi-classes. Nous commençons par une introduction, ensuite nous abordons les diverses méthodes proposées pour résoudre le problème de la classification multi-classes par des SVM binaires. Enfin nous présentons la résolution du problème multi-classes par une méthode d'optimisation.

3.2 Le Perceptron Multi-Couches

3.2.1 Introduction

Les Réseaux de Neurones Artificiels (RNA) constituent un sujet de recherche couvrant de nombreuses disciplines parmi lesquelles l'intelligence artificielle, les statistiques, la neurobiologie, la psychologie et les sciences cognitives. Ils réalisent un traitement d'information distribué et ils sont composés d'unités de calcul primitives (les neurones) fonctionnant en parallèle et reliées entre elles par des connexions (les synapses). Le principe général de fonctionnement d'un réseau de neurones est la transmission de l'activité d'un groupe de neurones à un autre via les synapses.

Dans un réseau, la connaissance est généralement répartie et elle est stockée dans la topologie des neurones et dans les poids des connexions. Les réseaux de neurones s'organisent par des méthodes d'apprentissage automatique et on n'a pas besoin de connaître les règles logiques qui modélisent ce processus, ce qui permet leur utilisation d'une manière simple et rapide. Plusieurs autres avantages peuvent être attribués aux réseaux de neurones telles que la flexibilité, la tolérance au bruit et aux détériorations et la grande capacité de généralisation. Durant ces dernières années, les réseaux de neurones ont été appliqués à des problèmes aussi variés que complexes : reconnaissance de visages, contrôle de robots, identification de locuteur, reconnaissance des langages, reconnaissance de phonèmes, reconnaissance de mots isolés, etc.

Les Perceptrons multi-couches sont les plus connus des réseaux de neurones artificiels. Ils utilisent un paradigme d'apprentissage supervisé pour mettre au point une fonction d'entrée-sortie non linéaire à partir des exemples d'apprentissage. Ils présentent aussi de bonnes méthodes d'interpolation du fait que leurs fonctions d'activation et les fonctions associées aux sorties sont continues. Ce type de réseaux présente plus particulièrement une bonne aptitude à la généralisation et il a déjà fait ses preuves dans plusieurs applications notamment dans le domaine de la classification et de la reconnaissance des formes. Il a été largement utilisé aussi dans le domaine de la reconnaissance automatique de la parole dans lequel s'inscrivent les travaux présentés au niveau de ce manuscrit.

3.2.2 Perceptron

Le Perceptron, historiquement le premier RNA, est apparu en 1958 grâce aux travaux de Rosenblatt. Il est inspiré du système visuel et possède deux couches de neurones formels, une couche d'entrée dite "perceptive" et une couche de sortie dite "décisionnelle". Ce réseau parvient à apprendre, à identifier des formes simples et à calculer certaines fonctions logiques. Il constitue donc le premier système artificiel présentant une faculté jusque là réservée aux êtres vivants : la capacité d'apprendre par l'expérience.

L'élément de base du Perceptron est le neurone formel proposé par McCulloch et Pitts en 1943 et qui constitue une modélisation mathématique très simplifiée du neurone biologique et dont le fonctionnement est celui d'un automate à seuil.

Le modèle de McCulloch et Pitts s'appuie sur une formalisation simple d'un neurone possédant des entrées binaires ou réelles et une sortie binaire. Il reçoit un nombre variable d'entrées en provenance de neurones amonts. À chacune de ces entrées est associé un poids synaptique représentant la force de la connexion. Il est aussi doté d'une sortie unique qui se ramifie ensuite pour alimenter un nombre variable de neurones avals. Son principe de fonctionnement est simple : il calcule la somme de ses entrées pondérée par leurs poids synaptiques et la compare à un seuil fixé θ. Le neurone est considéré actif en renvoyant 1 comme sortie dans le cas où le résultat est supérieur au seuil θ. Sinon il est considéré comme inactif et il renvoie une sortie nulle. Cette sortie est calculée selon la formule suivante :

$$y = F\,(\sum_{i=1}^{N} w_i x_i - \theta)$$

$F()$ est une fonction à seuil, au niveau du neurone cette fonction est considérée comme fonction de décision qui détermine si un neurone est actif ou non, $x = (x_1, x_2, ...x_N)$ est le vecteur d'entrée et $w = (w_1, w_2, ...w_N)$ est le vecteur des poids. Schématiquement, on peut représenter le neurone formel de McCulloch et Pitts par la figure 3.1.

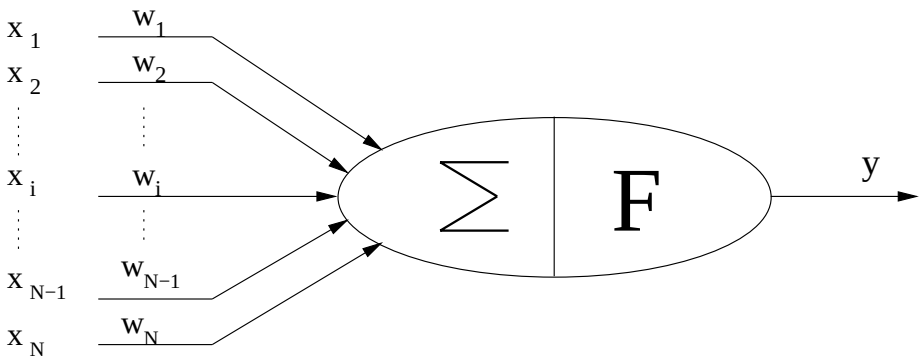

FIG. 3.1 – Architecture d'un neurone formel

Dans sa première version, le neurone formel était donc implémenté avec une fonction à seuil (0 ou 1), mais de nombreuses versions existent. Ainsi le neurone de McCulloch et Pitts a été généralisé de différentes manières, en choisissant d'autres fonctions d'activation, comme les fonctions linéaires par morceaux, les sigmoïdes ou les gaussiennes par exemples. Cependant, les fonctions

sigmoïdes sont de loin les plus utilisées comme fonctions d'activation dans les RNAs.

Malgré tout l'enthousiasme que soulève le travail de Rosenblatt dans le début des années 60, la fin de cette décennie sera marquée en 1969, par une critique violente du Perceptron par Minsky et Papert. Ils montrent dans un livre ("Perceptrons") toutes les limites de ce modèle et soulèvent particulièrement l'incapacité du Perceptron à résoudre les problèmes non linéairement séparables, tels que le célèbre problème du *XOR*. Il s'en suivra alors, face à la déception, une période noire d'une quinzaine d'années dans le domaine des réseaux de neurones artificiels.

3.2.3 Perceptron Multi-Couches

Ces 20 dernières années, les réseaux de neurones ont pris une part de plus en plus importante dans le monde de la recherche. En effet, au début des années quatre-vingt Hopfield a montré que certaines règles d'apprentissage comme par exemple la technique de rétro-propagation de l'erreur peut permettre à des réseaux de neurones d'apprendre des fonctions que le Percepton ne peut pas apprendre. Pendant les années quatre vingt dix, les réseaux de neurones étaient devenus un domaine attractif pour plusieurs chercheurs.

Les réseaux de neurones les plus utilisés en reconnaissance automatique de la parole sont les PMC. Un PMC est un réseau dont les neurones sont disposés en plusieurs couches successives et où chaque neurone d'une couche est connecté à tous les neurones de la couche précédente et à chaque neurone de la couche suivante ; il n'y a en revanche aucune connexion entre les neurones d'une même couche. De plus, un PMC est un réseau en passe-avant : les informations, ou activations, circulent dans un seul sens, c'est-à-dire des neurones d'une couche aux neurones de la couche suivante. Une couche cachée dans un PMC est une couche de neurones qui n'est ni la couche d'entrée ni la couche de sortie. Un PMC peut contenir autant de couches cachées que l'on désire mais il a été montré que quel qu'en soit le nombre il existe un PMC avec une seule couche cachée équivalente [Hornik *et al.*, 1989]. L'addition de cette couche cachée permet au réseau de modéliser des fonctions de décision complexes et non linéaires entre n'importe quels espaces d'entrée et de sortie.

La figure 3.2 montre un exemple d'un réseau PMC à une seule couche cachée. Dans ce réseau, il y a N entrées, L unités sur la couche cachée et P unités sur la couche de sortie. Chaque neurone est relié uniquement à tous les neurones de la couche suivante.

- x_k $k^{ième}$ entrée du réseau, $k \in \{1, \ldots, N\}$.
- w_{jk} poids affecté à la connexion reliant le neurone k de la couche d'entrée au neurone j de la couche cachée, $j \in \{1, \ldots, L\}$.
- w_{ij} poids affecté à la connexion reliant le neurone j de la couche cachée au neurone i de la couche de sortie, $i \in \{1, \ldots, P\}$.

L'activation z_j de la $j^{ième}$ unité de la couche cachée est donnée par l'équation suivante :

$$z_j = F(a_j)$$

Avec :

$$a_j = \sum_{k=1}^{N} w_{jk}.x_k = w_j^t.x$$

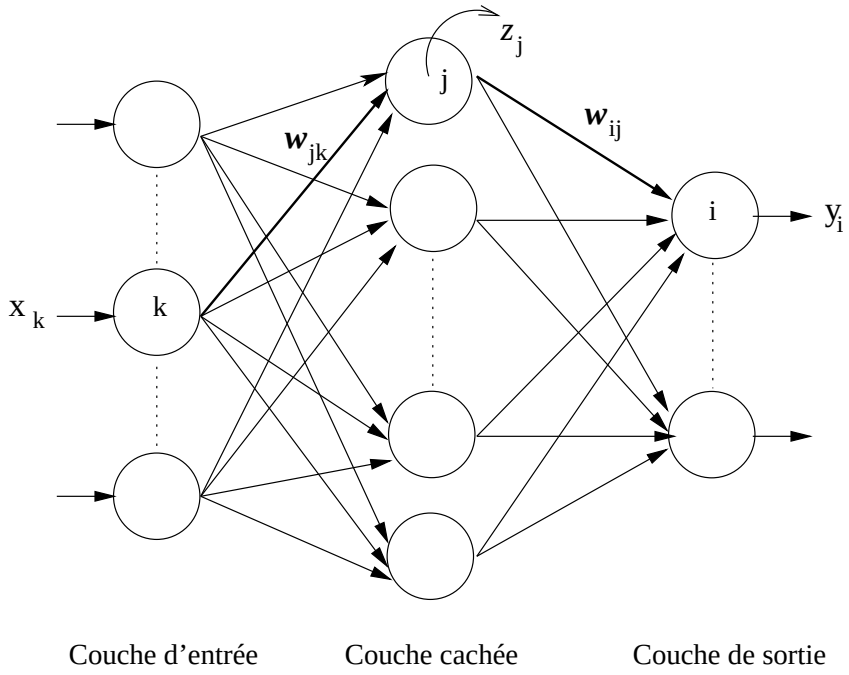

FIG. 3.2 – Architecture d'un PMC

$F()$ est une fonction d'activation, $x = (x_1, x_2, ..., x_N)$ est le vecteur d'entrée et w^t est le transposé du vecteur des poids $w = (w_1, w_2, ..., w_N)$.

De la même façon, l'activation de la i^{ime} unité de la couche de sortie est obtenue en transformant le produit scalaire a_i avec la fonction d'activation F :

$$y_i = F(a_i) = F(\sum_{j=1}^{L} w_{ij}.F(\sum_{k=1}^{N} w_{jk}.x_k))$$

La fonction d'activation la plus utilisée est la fonction sigmoïde donnée par l'équation suivante :

$$F(x) = \frac{1}{1 + e^{-x}}$$

Les différents termes de la somme pondérée des noeuds du réseau constituent les paramètres de celui-ci. Ils sont estimés lors d'un entraînement dit *supervisé*. Un entraînement est dit *supervisé* lorsque les entrées sont présentées au réseau les unes après les autres avec la sortie imposée correspondant à chaque entrée. Le problème de l'entraînement d'un réseau revient à trouver un ensemble de paramètres (poids) w qui minimise une fonction d'erreur E et il est réalisé en deux étapes. Dans un premier temps, une propagation "en avant" permet de calculer la valeur en sortie, puis dans un second temps, une propagation "en arrière" ajuste les pondérations afin de minimiser l'erreur entre la sortie calculée et la sortie désirée.

Les techniques du type descente du gradient sont les plus adaptées pour estimer ces paramètres (les poids du réseau w). Elles sont basées sur le calcul du gradient de l'erreur par rapport aux paramètres du réseau ($\frac{\partial E}{\partial w_{ij}}$) et l'ajustement de ces paramètres jusqu'à ce qu'un minimum de l'erreur soit atteint.

La méthode de "back-propagation" (rétro-propagation) est une adaptation de l'algorithme de *Widrow-Hoff* aux réseaux à couches, elle vise à minimiser un des critères d'erreur (erreur quadratique ou entropie) entre les sorties réelles et désirées pour chacun des vecteurs d'entrée. Cette méthode permet aussi de modifier les valeurs des poids du réseau jusqu'à atteindre ses valeurs optimales représentant le réseau appris. La fonction d'erreur (pour un réseau à plus d'une couche cachée) utilisée pour tester si le réseau a bien appris ou non est une fonction non linéaire des poids avec plusieurs minima possibles. Par exemple, le critère quadratique proposé dans l'algorithme de *Widrow-Hoff* original est :

$$E_x = \frac{1}{2} \times ||y - d||^2 = \frac{1}{2} \times \sum_{i=1}^{P} (y_i - d_i)^2$$

Afin de corriger les poids du réseau on utilise donc l'algorithme de *Widrow-Hoff* qui s'écrit comme suit :

$$\Delta w_{jk} = -\eta . \frac{\partial E_x}{\partial w_{jk}} = -\eta . \frac{\partial E_x}{\partial a_j} \times \frac{\partial a_j}{\partial w_{jk}}$$

où η est appelé le pas d'apprentissage et doit être assez faible pour garantir la convergence du processus et assez grand pour éviter une convergence trop lente.

Pour la correction des poids entre la couche cachée et la sortie on a :

$$\Delta w_{ij} = -\eta . \frac{\partial E_x}{\partial a_i} \times \frac{\partial a_i}{\partial w_{ij}}$$

où $\frac{\partial a_i}{\partial w_{ij}} = z_j$, et $\frac{\partial E_x}{\partial a_i}$ peut se calculer en appliquant la règle de dérivation des fonctions composées :

$$\frac{\partial E_x}{\partial a_i} = \frac{\partial y_i}{\partial a_i} \times \frac{\partial E_x}{\partial y_i} = F'(a_i) . \frac{\partial E_x}{\partial y_i} = F'(a_i) . (y_i - d_i)$$

Soit en posant $\delta_i = F'(a_i) . (y_i - d_i)$, il vient :

$$\Delta w_{ij} = -\eta . \delta_i . z_j$$

Pour la correction des poids entre la couche d'entrée et la couche cachée on a :

$$\Delta w_{jk} = -\eta . \frac{\partial E_x}{\partial a_j} \times \frac{\partial a_j}{\partial w_{jk}}$$

où, $\frac{\partial a_j}{\partial w_{jk}} = x_k$, et $\frac{\partial E_x}{\partial a_j}$ s'écrit en appliquant la même règle de dérivation que précédemment :

$$\frac{\partial E_x}{\partial a_j} = \frac{\partial z_j}{\partial a_j} \times \frac{\partial E_x}{\partial z_j} = F'(a_j) . \sum_{i=1}^{P} \delta_i . w_{ij}$$

Soit en posant $\delta_j = F'(a_j) . \sum_{i=1}^{P} \delta_i . w_{ij}$, il vient :

$$\Delta w_{jk} = -\eta . \delta_j . x_k$$

Ce calcul peut se généraliser dans la cas des réseaux à N couches cachées. L'algorithme ci-dessous résume l'expression de la mise à jour des poids dans l'algorithme de rétro-propagation du gradient utilisé pour l'apprentissage d'un PMC.

Algorithme 5 Algorithme de rétro-propagation du gradient.

0- Choix de la valeur du pas d'apprentissage η, du seuil minimal d'erreur et du nombre maximal d'itérations.
1- Initialisation aléatoire des poids à des petites valeurs.
2- Choix d'un vecteur d'entrée x.
3- Propagation en avant dans le réseau pour obtenir le vecteur de sortie y.
4- Calculer δ_i^λ entre la couche de sortie et la couche qui la précède. (λ est le nombre de couches dans le réseau)

$$\delta_i = F'(a_i).(y_i - d_i)$$

5- Calculer les deltas des couches précédentes par propagation arrière de l'erreur.

Pour chaque couche cachée c allant de $\lambda - 1$ *à 1 faire*

$$\delta_j^c = F'(a_j^c).\sum_{i=1}^{P} \delta_i^c.w_{ij}^c$$

Faire.

6- Mettre à jour les poids en utilisant :

$$\Delta w_{ij}^c = -\eta.\delta_i^c.z_j^c$$

7- Retourner en 2. et répéter pour l'entrée suivante, jusqu'à ce que l'erreur en sortie soit inférieure à la limite fixée ou que le nombre maximum d'itérations soit atteint.

3.3 Théorie de l'apprentissage de Vapnik

3.3.1 Introduction

D'après Vapnik [Vapnik, 1995], le modèle général de l'apprentissage statistique est constitué des trois composantes de la figure 3.3 :

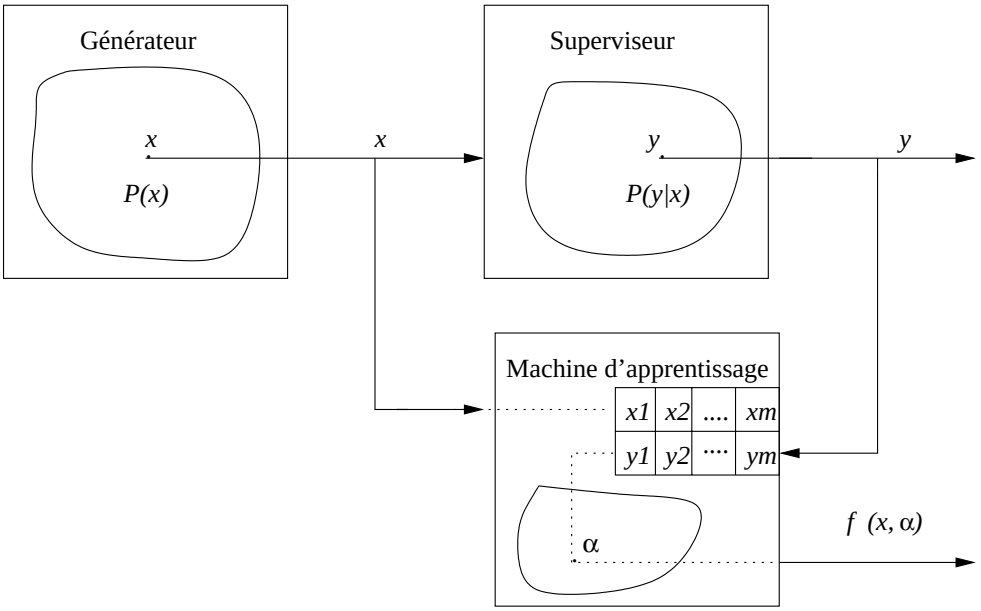

FIG. 3.3 – Modèle général de l'apprentissage statistique selon Vapnik

- Un générateur de vecteurs aléatoires $x \in \Re^p$, tirés indépendamment les uns des autres. Ils sont identiquement distribués suivant une distribution de probabilité $P(x)$, fixe mais inconnue.
- Un superviseur qui retourne une valeur de sortie désirée y pour chaque vecteur d'entrée x, suivant une distribution conditionnelle de probabilité $P(y \mid x)$, qui est aussi fixe mais inconnue.
- Une machine (ou un modèle) d'apprentissage (M) capable de réaliser l'ensemble des fonctions $f(x, \alpha), \alpha \in \Lambda$ où Λ est un ensemble de paramètres.

Le problème de l'apprentissage statistique à partir d'exemples étiquetés, appelé *apprentissage supervisé*, consiste à rechercher la fonction f qui approche au mieux les réponses du superviseur, dans l'ensemble des fonctions $F = \{f(x, \alpha), \alpha \in \Lambda\}$ réalisables par le modèle choisi M. La seule connaissance sur le problème dont on dispose, *a priori*, pour faire le choix de cette fonction est l'ensemble :

$$D = \{(x_1, y_1), ..., (x_m, y_m)\}$$

appelé *ensemble d'apprentissage*, composé de m observations indépendantes et identiquement distribuées suivant $P(x, y)$.

La *fonction perte* formalise la notion de la qualité d'apprentissage. Notons $L(y, f(x, \alpha))$ la fonction coût qui correspond à la réponse $f(x, \alpha)$ associée à l'entrée x et à la réponse y donnée par le superviseur. Cette fonction peut prendre des formes différentes, selon le type du problème

auquel on est confronté. Par exemple, pour la reconnaissance de formes, la fonction coût s'écrit sous la forme suivante :

$$L(y, f(x,\alpha)) = \begin{cases} 0 & \text{si } y = f(x,\alpha) \\ 1 & \text{si } y \neq f(x,\alpha) \end{cases}$$

L'espérance mathématique de la fonction coût définit le *risque fonctionnel* $R(\alpha)$:

$$R(\alpha) = \int L(y, f(x,\alpha))\, dP(x,y) \tag{3.1}$$

Le but de l'apprentissage est donc de trouver la fonction $f(x,\alpha_0)$ qui minimise le risque fonctionnel $R(\alpha)$, parmi la famille des fonctions $f(x,\alpha)$ réalisables par notre machine M, dans le cas où $P(x,y)$ n'est pas connue et où toute l'information dont on dispose est l'ensemble d'apprentissage D.

3.3.2 Minimisation du risque empirique

On ne peut pas minimiser directement le risque fonctionnel $R(\alpha)$ à partir de l'ensemble d'apprentissage. Le principe inductif de la Minimisation du Risque Empirique (MRE) consiste à remplacer le risque $R(\alpha)$ par le risque empirique [Osuna *et al.*, 1997a] :

$$R_{emp}(\alpha) = \frac{1}{m} \sum_{i=1}^{m} L(y, f(x,\alpha)) \tag{3.2}$$

qui est l'espérance empirique de la fonction coût sur l'ensemble d'apprentissage. Ainsi nous approchons la fonction $f(x,\alpha_0)$ qui minimise le risque fonctionnel $R(\alpha)$ par la fonction $f(x,\hat{\alpha})$ qui minimise le risque empirique $R_{emp}(\alpha)$.

Le problème de l'apprentissage à partir d'exemples (principe d'induction) est qu'il ne permet pas pour un modèle M donné, le contrôle de la généralisation qui est la première qualité qu'on attend d'un modèle. De façon intuitive, un modèle d'apprentissage a une *bonne capacité de généralisation* lorsqu'il produit, en moyenne sur tous les exemples, des réponses proches de la valeur désirée, ce qui revient à dire que son risque est faible. Un faible risque empirique, par contre, n'est pas une garantie d'une bonne généralisation.

3.3.3 Consistance du principe de minimisation du risque empirique

Définition 1
Soit $H = \{L(y, f(x,\alpha)), \alpha \in \Lambda\}$ un ensemble de fonctions et $P(x,y)$ une distribution de probabilité.

Le principe est **consistant** pour H et P si la méthode de minimisation du risque empirique, appliquée à chaque échantillon de taille m, produit une suite de fonctions $\{L(y, f(x,\alpha_m)),\ m \in N\}$ tel que :

$$\begin{gathered} R(\alpha_m) \longrightarrow_{m\to\infty} Inf_{\alpha\in\Lambda} R(\alpha) \\ R_{emp}(\alpha_m) \longrightarrow_{m\to\infty} Inf_{\alpha\in\Lambda} R(\alpha) \end{gathered} \tag{3.3}$$

C'est-à-dire que la suite des espérances du risque fonctionnel et celles du risque empirique convergent en probabilité vers la même limite qui est la valeur la plus petite du risque fonctionnel $R(\alpha)$.

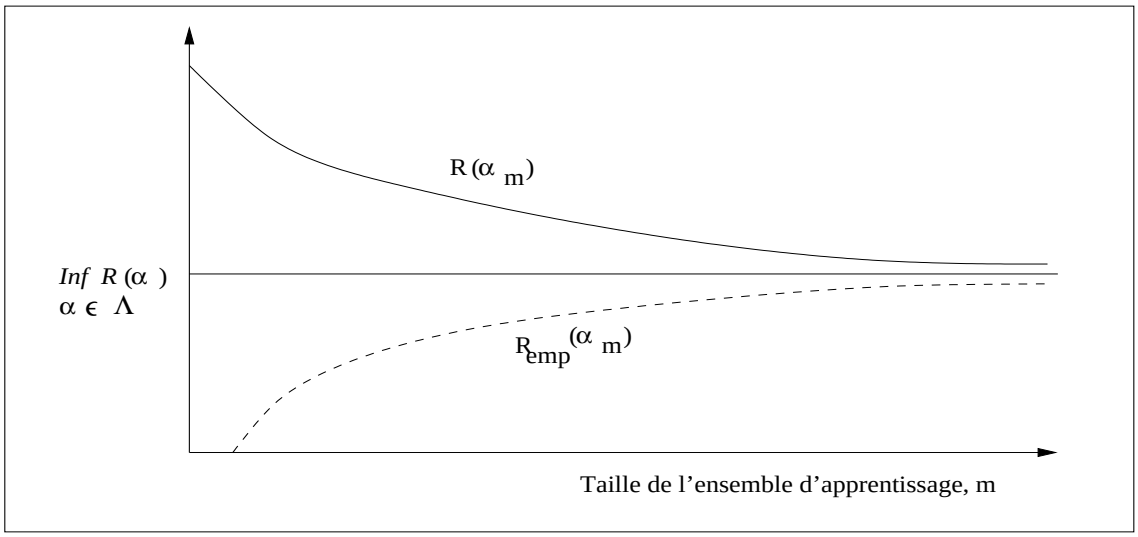

FIG. 3.4 – Consistance du principe de minimisation du risque empirique. Risque $(R(\alpha_m))$ et risque empirique $(R_{emp}(\alpha_m))$ en fonction du nombre de points d'apprentissage, $(InfR(\alpha))$ est le risque minimal.

La figure 3.4 représente une interprétation du double problème de la consistance du principe de la MRE. Ce double problème consiste à faire tendre les deux risques $R(\alpha)$ et $R_{emp}(\alpha)$ vers la même limite lorsque l'échantillon d'apprentissage devient suffisamment grand.

Le résultat clé de la théorie d'apprentissage de Vapnik est donné par le théorème suivant :

Théorème 1
Soit $\{L(y, f(x, \alpha)), \alpha \in \Lambda\}$ un ensemble de fonctions qui vérifient la condition :

$$A \leq \int L(y, f(x, \alpha)) \ dP(x, y) \leq B$$

$$A \leq risque \ \ fonctionnel \leq B$$

où $\{A, B\} \in \Re$

Une condition nécessaire et suffisante de consistance non triviale du principe de minimisation du risque empirique est que le risque empirique $R_{emp}(\alpha)$ converge uniformément vers le vrai risque $R(\alpha)$ sur l'ensemble des fonctions $\{L(y, f(x, \alpha)), \alpha \in \Lambda\}$.

$$lim_{m \to \infty} \ \ P\{sup_{\alpha \in \Lambda}(R(\alpha) - R_{emp}(\alpha)) > \epsilon\} = 0, \ \ \forall \ \ \epsilon > 0 \tag{3.4}$$

Il est important de remarquer que la présence de la borne *sup* dans l'expression (3.4) implique que la consistance du principe de minimisation du risque empirique est nécessairement déterminée par la pire des fonctions de l'ensemble $\{L(y, f(x, \alpha)), \ \ \alpha \in \Lambda\}$. D'où, l'approche dans

le *pire des cas* qui est la seule analyse possible pour la consistance du principe de MRE.

Ce théorème est un résultat théorique, mais il n'est pas directement applicable, puisque le risque $R(\alpha)$ n'est pas calculable et la distribution de probabilités est inconnue. Pour cette raison, Vapnik a utilisé plusieurs notions intermédiaires afin de déduire d'autres conditions suffisantes pour la consistance du principe de minimisation du risque empirique.

3.3.4 Bornes sur la vitesse de convergence des processus d'apprentissage

Vapnik a introduit les notions de **VC-entropie**, **VC-entropie recuite** et **fonction de croissance** pour aboutir à d'autres conditions suffisantes pour la consistance du principe de minimisation du risque empirique.

Soit $N^{\Lambda}((x_1, y_1), \ldots, (x_m, y_m))$ la quantité qui compte le nombre de vecteurs distincts suffisant à "représenter" l'ensemble des fonctions $\{L(y, f(x, \alpha)),\ \alpha \in \Lambda\}$ sur l'ensemble $D = \{(x_1, y_1), \ldots, (x_m, y_m)\}$. Pour les fonctions à valeurs binaires, nous pouvons donner à partir de cette quantité, les définitions suivantes :

Définition 2
La VC-entropie d'un modèle notée $H^{\Lambda}(m)$ est l'espérance du logarithme de la diversité de l'ensemble des fonctions que le modèle peut réaliser (c-à-d le nombre de séparations différentes possibles), sur un échantillon de taille donnée :

$$H^{\Lambda}(m) = E\ ln[N^{\Lambda}((x_1, y_1), \ldots, (x_m, y_m))]$$

Définition 3
La VC-entropie recuite d'un modèle notée $H_r^{\Lambda}(m)$ est le logarithme de l'espérance de la diversité de l'ensemble des fonctions que le modèle peut réaliser sur un échantillon de taille donnée :

$$H_r^{\Lambda}(m) = ln\ E[N^{\Lambda}((x_1, y_1), \ldots, (x_m, y_m))]$$

Définition 4
La fonction de croissance notée $G^{\Lambda}(m)$ est la fonction supérieure sur l'ensemble des échantillons D de la diversité de l'ensemble des fonctions que le modèle peut réaliser sur un échantillon de taille donnée :

$$G^{\Lambda}(m) = Sup_D N^{\Lambda}((x_1, y_1), \ldots, (x_m, y_m))$$

Cette fonction de croissance est majorée par une fonction notée $\Phi(m, d)$ qui correspond à la taille de la plus grande trace d'une famille F de dimension de Vapnik-Chervonenkis d sur un ensemble S de taille m (voir définition 5).

$$\Phi(m, d) = sup\{card(T_F(S)) | card(S) = m\ et\ D_F = d\}$$

$$Si \quad m \leq d \quad alors \quad G^{\Lambda}(m) \leq \Phi(m, d) = 2^m$$

Théorème 2
Si $m > d$ alors la fonction de croissance est bornée de la manière suivante :

$$G^{\Lambda}(m) \leq \Phi(m,d) \leq (\frac{e^1 \times m}{d})^d$$

Vapnik et Chervonenkis ont étendu les définitions précédentes à des fonctions à valeurs réelles, et ils ont montré l'existence d'un grand nombre de bornes supérieures et de quelques bornes inférieures concernant d'une part la différence entre le risque fonctionnel $R(\alpha)$ et le risque empirique $R_{emp}(\alpha)$ et d'autre part la différence entre les risques $R(\alpha_m)$ obtenus sur des échantillons de taille m et le plus petit risque possible $inf_{\alpha \in \Lambda} R(\alpha)$.

À partir de cette théorie des bornes, nous obtenons le résultat principal, qui s'appuie sur la VC-dimension du modèle d'apprentissage et qui est le suivant :

Définition 5
Soit X un ensemble et F une partie de l'ensemble des parties de X ($F \subseteq 2^X$). Pour tout sous ensemble S de X, on appelle **trace** de F sur S l'ensemble suivant :

$$T_F(S) = \{S \cap H \mid H \in F\}$$

Lorsque $T_F(S)$ est égal à l'ensemble des parties de S, on dit que S est *pulvérisé* par F.

Un ensemble fini S est pulvérisé par F si tout étiquetage de S peut être décrit par au moins une fonction f de F.

La dimension de Vapnik-Chervonenkis (**VC-dimension**) D_F de F est le cardinal du plus grand ensemble pulvérisé par F.

$$D_F = max_S\{k \mid card(S) = k \ \ et \ \ T_F(S) = 2^S\}$$

Le résultat clé de la théorie de la consistance est exprimé par le théorème suivant :

Théorème 3
Pour que le principe de minimisation du risque empirique soit consistant, indépendamment de la distribution de probabilité des exemples, il suffit que l'ensemble des fonctions que le système d'apprentissage est capable d'implémenter possède une **VC-dimension finie**.

Soit h la VC-dimension de l'ensemble des fonctions réalisables par le système d'apprentissage. L'inégalité suivante sera vérifiée avec une probabilité au moins égale à $1 - \eta$ avec $\eta \ \in]0,1]$:

$$R(\alpha) \leq R_{emp}(\alpha) + \Phi(\frac{m}{h}, \frac{-Log(\eta)}{m}) \tag{3.5}$$

Par exemple, dans le cas de la reconnaissance des formes, l'inégalité précédente sera équivalente à :

$$R(\alpha) \leq R_{emp}(\alpha) + \sqrt{\frac{h[Log(\frac{2m}{h}) + 1] - Log(\frac{\eta}{4})}{m}} \tag{3.6}$$

3.3.5 Contrôle de la capacité de généralisation des processus d'apprentissage

À partir de l'inégalité (3.6), le second membre appelé **le risque garanti** est composé de deux termes, le risque empirique et une quantité qui dépend du quotient $\frac{m}{h}$, appelée **intervalle de confiance**, puisqu'elle représente la différence entre le risque empirique $R_{emp}(\alpha)$ et le risque $R(\alpha)$.

Si le rapport $\frac{m}{h}$ est suffisamment grand, la minimisation du risque empirique suffit pour contrôler la capacité de généralisation du modèle. En fait une faible valeur du risque empirique permet de garantir une faible valeur du risque fonctionnel $R(\alpha)$. En revanche, lorsque la taille m de l'échantillon est petite, le rapport $\frac{m}{h}$ est petit et l'intervalle de confiance peut prendre une valeur importante. Ceci implique que la minimisation du risque empirique n'est pas suffisante pour garantir une valeur minimale du risque fonctionnel. Ainsi pour minimiser le second membre de l'inégalité (3.6), il faut minimiser simultanément les deux termes qui le composent ; pour cela, il faut pouvoir contrôler le VC-dimension h du système.

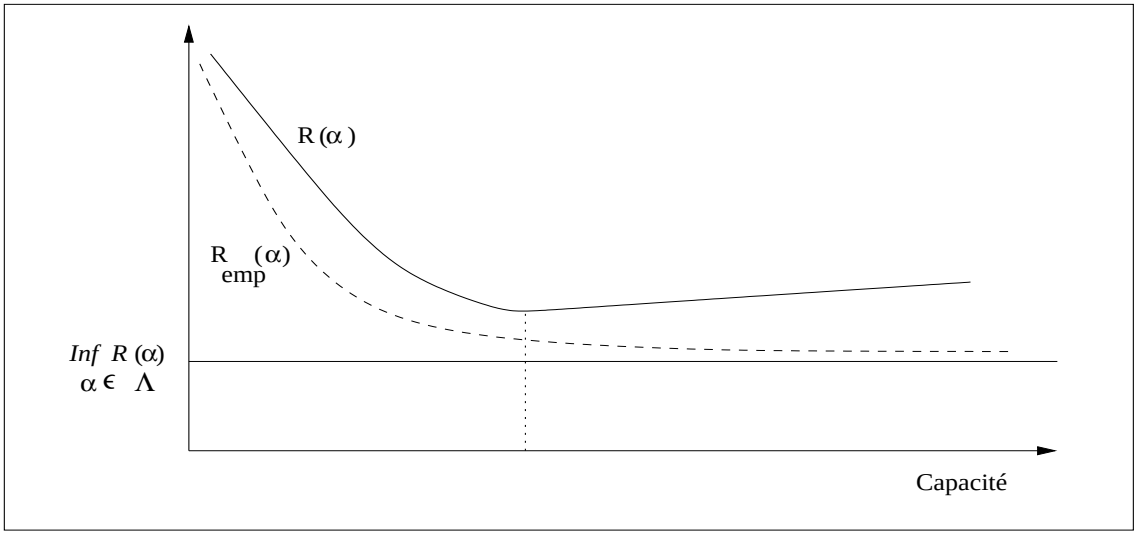

FIG. 3.5 – Consistance du principe de minimisation du risque empirique. Risque ($R(\alpha)$), risque minimal ($InfR(\alpha)$) et risque empirique ($R_{emp}(\alpha)$) en fonction de la capacité du modèle pour un ensemble d'apprentissage fixe.

Dans la pratique, le rapport entre le nombre d'exemples et la *capacité* de la machine d'apprentissage, qui caractérise sa complexité, est de grande importance. De façon sommaire nous pouvons dire que **plus** cette capacité est grande :

i) **plus** le risque fonctionnel et le risque empirique (respectivement $R(\alpha)$ et $R_{emp}(\alpha)$) sont faibles.

ii) mais **plus** le nombre de données d'apprentissage doit être grand pour assurer que $R(\alpha)$ converge vers $InfR_{\alpha\in\Lambda}(\alpha)$.

Quand la taille de l'ensemble d'apprentissage est fixe, il existe une capacité optimale, au-dessus de laquelle le risque augmente même si le risque empirique continue à décroître (figure 3.5).

3.3.6 Minimisation du risque structurel

Afin de contrôler la VC-dimension h, Vapnik propose d'appliquer un nouveau principe inductif qui est le principe de **Minimisation du Risque Structurel** (MRS). Ce principe est basé sur la minimisation conjointe des deux causes d'erreur : le risque empirique et l'intervalle de confiance.

Définition 6
Soit $F = \{f(x,\alpha), \alpha \in \Lambda\}$ une famille de fonctions.
Une **structure** sur F est une suite de sous-ensembles emboîtés $F_i = \{f^i(x,\alpha), \alpha \in \Lambda_i\}$:

$$F_1 \subset F_2 \subset ... \subset F_i \subset ...$$

dont les VC-dimensions h_i sont finies. Ces VC-dimensions forment alors une suite croissante.

$$h_1 \leq h_2 \leq ... \leq h_i \leq ...$$

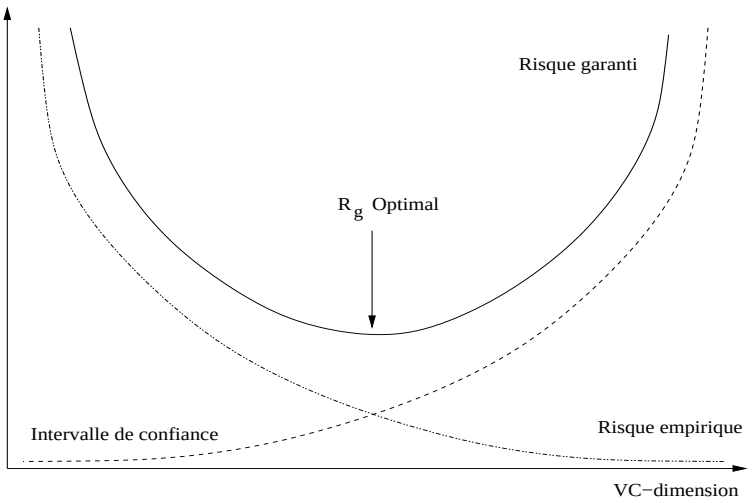

FIG. 3.6 – Comportement du risque empirique, de l'intervalle de confiance et du risque garanti en fonction de la VC-dimension.

En pratique, lorsque la VC-dimension augmente, le risque empirique décroît tandis que l'intervalle de confiance croît (un classifieur à large capacité a un faible risque empirique c'est-à-dire

qu'il apprend bien mais généralise mal).

L'objectif du principe de minimisation du risque structurel consiste à chercher un compromis entre la qualité de l'approximation sur l'échantillon considéré et la complexité de la fonction qui réalise l'approximation. Ceci est illustré par la figure 3.6 qui montre le comportement du risque empirique, l'intervalle de confiance et le risque garanti en fonction de la VC-dimension. On remarque que le minimum du risque garanti est atteint pour une valeur optimale de h [Muller *et al.*, 2001].

3.4 Support Vector Machines

3.4.1 Introduction

Les SVM constituent un nouveau type d'algorithmes d'apprentissage, originalement introduits par Vapnik et Boser [Boser *et al.*, 1992] [Vapnik, 1998], et successivement étendu par plusieurs autres chercheurs [Burges, 1998] [Furey *et al.*, 2000] [Ganapathiraju, 2002]. Ses performances robustes remarquables obtenues avec des bases dispersées et bruitées ont fait des SVM des systèmes de choix pour de nombreuses applications. Par exemple, dans le domaine de la reconnaissance des formes, les SVM ont été utilisés pour la reconnaissance de l'écriture des chiffres isolés [Cortes et Vapnik, 1995] [Schölkopf *et al.*, 1996], l'identification d'objets [Blanz *et al.*, 1996], l'identification du locuteur [Kharroubi *et al.*, 2001], la détection de visages en images [Osuna *et al.*, 1997b] et la catégorisation de texte [Joachims, 1998] [Sebastiani, 2002].

Pour la reconnaissance des formes (cas de deux classes d'échantillons), les SVM recherchent une surface de décision optimale, déterminée par certains points de l'ensemble d'apprentissage appelés *vecteurs support*, en projetant les données d'entrée non-linéairement séparables dans un espace de plus grande dimension appelé *espace de caractéristiques* (figure 3.7). Cette surface, qui est dans l'espace des caractéristiques, peut être considérée comme un hyperplan optimal de décision. Elle est obtenue par la résolution d'un problème de programmation quadratique dépendant des paramètres de régularisation.

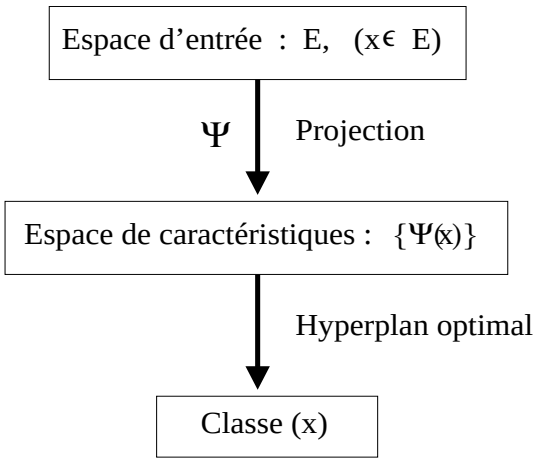

FIG. 3.7 – Principe des SVM

Les systèmes d'apprentissage SVM, sont des algorithmes basés sur les trois principes mathématiques suivants [Ciarlet, 1994] :

- **Le principe de Fermat (1638)** : Les points qui minimisent ou maximisent une fonction dérivable annulent sa dérivée. Ils sont appelés points stationnaires.

- **Le principe de Lagrange (1788)** : Pour résoudre un problème d'optimisation sous contraintes, il suffit de rechercher un point stationnaire x^0 du Lagrangien L de la fonction f à optimiser.

$$L(x, \lambda) = f(x) + \sum_{i=0}^{K} \lambda_i f_i(x) \tag{3.7}$$

  Où les *fi* expriment les contraintes et les λ_i sont des constantes appelées coefficients de Lagrange.

- **Le principe de Kuhn-Tucker (1951)** : Les relations de Kuhn-Tucker peuvent s'appliquer en utilisant les fonctions convexes f et *fi* . Il est toujours possible de trouver un point-selle (x^0 , λ^*) qui vérifie l'équation suivante :

$$\min_x L(x, \lambda^*) = L(x^0, \lambda^*) = \max_{\lambda \geq 0} L(x^0, \lambda) \tag{3.8}$$

3.4.2 Cas des données linéairement séparables

Soit une base d'exemples, S = $\{(x_i,y_i)$, $x_i \in \Re^n$; $y_i \in \{$ -1, 1$\}$; $i = 1, \ldots, m$ $\}$, linéairement séparables.

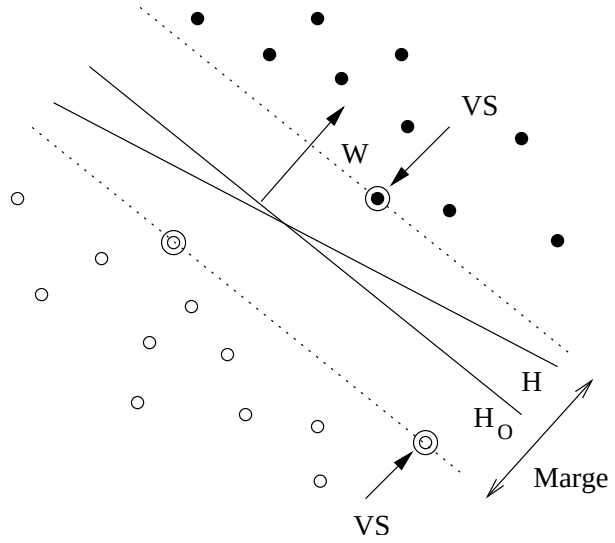

FIG. 3.8 – Hyperplans séparateurs : H est un hyperplan quelconque, H_O est l' hyperplan optimal, VS : sont les Vecteurs Support.

Soit H : (w, b) un hyperplan séparateur qui est l'un des hyperplans de la figure 3.8 et qui vérifie les conditions suivantes :

$$w.x_i + b \geq 1 \quad si \quad y_i = 1 \tag{3.9}$$

$$w.x_i + b \leq -1 \quad si \quad y_i = -1 \tag{3.10}$$

Ce qui est équivalent à :

$$y_i \ (w.x_i + b) \geq 1 \quad \forall i \in \{1, ..., m\} \tag{3.11}$$

La distance $d(w, b, x)$ entre un point x et l'hyperplan (w, b) est donnée par :

$$d(w, b, x) = \frac{|w.x+b|}{||w||}$$

Définition 7
On appelle **hyperplan optimal**, l'hyperplan séparateur qui se situe à la distance maximale des vecteurs x les plus proches parmi l'ensemble des exemples. Cet hyperplan **maximise la marge**.

L'hyperplan optimal est donné par la maximisation de la marge $M(w, b)$, sous les contraintes de l'équation 3.11. La marge $M(w, b)$ s'écrit sous la forme suivante :

$$\begin{aligned} M(w, b) &= \min_{x_i:y_i=-1} d(w, b, x_i) + \min_{x_i:y_i=1} d(w, b, x_i) \\ &= \min_{x_i:y_i=-1} \frac{|w.x_i+b|}{||w||} + \min_{x_i:y_i=1} \frac{|w.x_i+b|}{||w||} \\ &= \frac{1}{||w||}[\min_{x_i:y_i=-1} |w.x_i + b| + \min_{x_i:y_i=1} |w.x_i + b|] \\ &= \frac{1}{||w||}[1 + 1] = \frac{2}{||w||} \end{aligned}$$

Plus la marge est grande, plus l'erreur attendue est faible. Maximiser la marge $M(w, b)$, revient à minimiser le carré de la norme $||w||^2 = (w.w)$ du vecteur w, sous les contraintes de l'équation 3.11.

Ainsi l'hyperplan optimal qui vérifie les conditions de l'équation 3.11 est celui qui minimise la fonction définie par :

$$\phi(w) = \frac{1}{2} w.w \tag{3.12}$$

En appliquant le principe de Kuhn-Tucker, nous sommes menés à la recherche d'un point-selle w^0,b^0, λ^0 du Lagrangien :

$$L(w, b, \lambda) = \frac{1}{2} w.w - \sum_{i=1}^{m} \lambda_i [y_i(w.x_i + b) - 1] \tag{3.13}$$

où λ= (λ_1,...,λ_m).

Ce point-selle vérifie les conditions :

$$\lambda_i^0 [y_i(w^0.x_i + b^0) - 1] = 0, \quad pour \ \ tout \ \ i \in \{1, \ldots, m\} \tag{3.14}$$

Définition 8
Les vecteurs support sont les vecteurs x_i pour lesquels l'égalité $y_i \ (w^0 \ . \ x_i + b^0 \ ) = 1$ est vérifiée. Concrètement, ce sont les vecteurs les plus proches de l'hyperplan optimal. Pour tous les autres exemples qui vérifient l'équation 3.14, nous avons donc $\lambda_i^0 = 0$.

Les conditions d'annulation des dérivées partielles du Lagrangien permettent d'écrire les relations que vérifie l'hyperplan optimal, avec λ_i^0 non nuls seulement pour les vecteurs support.

$L(w, b, \lambda)$ est minimal par rapport à b :

$$\frac{\delta L(w,b,\lambda)}{\delta b} = 0 \Longleftrightarrow \sum_{i=1}^{m} \lambda_i^0 y_i = 0 \tag{3.15}$$

$L(w, b, \lambda)$ est minimal par rapport à w :

$$\frac{\delta L(w,b,\lambda)}{\delta w} = 0 \Longleftrightarrow w^0 = \sum_{i=1}^{m} \lambda_i^0 x_i y_i \tag{3.16}$$

Ces contraintes d'égalité permettent d'exprimer la forme duale du Lagrangien :

$$F(\lambda) = \sum_{i=1}^{m} \lambda_i - \frac{1}{2} \sum_{i,j=1}^{m} \lambda_i \lambda_j y_i y_j (x_i.x_j) \tag{3.17}$$

Pour trouver le point-selle, il suffit alors de maximiser (et non plus de minimiser) *F(λ)* dans le quadrant positif $\lambda_i \geq 0$, pour tout $i \in \{1, \ldots, m\}$ sous la contrainte :

$$\sum_{i=1}^{m} \lambda_i^0 y_i = 0 \tag{3.18}$$

La résolution de l'équation 3.14, nous donne les solutions suivantes :

$$\lambda_i^0 = 0 \quad ou \quad y_i \, (w^0.x_i + b^0) = 1 \quad avec \quad \lambda_i^0 > 0 \tag{3.19}$$

Pour calculer le biais b^0 nous avons utilisé l'équation :

$$y_i \, (w^0.x_i + b^0) = 1 \quad avec \quad \lambda_i^0 > 0$$

Pour un vecteur support de la classe $+1$:

$$w^0.vs_{(classe+1)} + b^0 = 1$$

Pour un vecteur support de la classe -1 :

$$w^0.vs_{(classe-1)} + b^0 = -1$$

D'après ces deux équations nous obtenons la valeur du biais b_0.

$$b^0 = \frac{1}{2}([w^0.vs_{(classe+1)}] + [w^0.vs_{(classe-1)}]) \tag{3.20}$$

Ainsi, la fonction de classification $classe(x)$ s'écrit sous la forme suivante :

$$classe(x) = sign(w^0.x + b^0)$$

$$classe(x) = sign[(\sum_{x_i \in VS} \lambda_i^0 y_i x_i).x + b^0]$$

$$classe(x) = sign[\sum_{x_i \in VS} \lambda_i^0 y_i (x_i.x) + b^0] \tag{3.21}$$

Si la fonction $classe(x)$ est négative alors x appartient à la classe -1 sinon x appartient à la classe $+1$.

3.4.3 Cas des données non-linéairement séparables

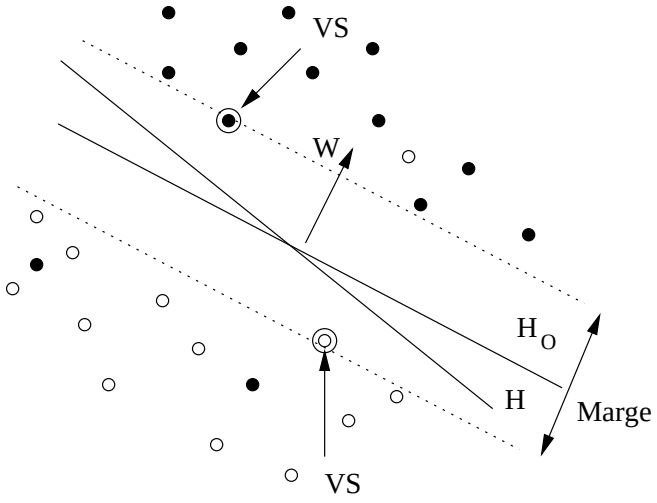

FIG. 3.9 – Hyperplans séparateurs dans le cas de données non-linéairement séparables : H est un hyperplan quelconque, H_O est l'hyperplan optimal, VS : sont les Vecteurs Support.

Dans ce cas les exemples ne sont pas linéairement séparables (figure 3.9) et l'hyperplan optimal sera celui qui vérifie les contraintes suivantes :

- La distance entre l'hyperplan et les vecteurs bien classés doit être maximale.
- La distance entre l'hyperplan et les vecteurs mal classés doit être minimale.

Pour résoudre ce problème, on peut reformuler les conditions des équations 3.9 et 3.10, en introduisant des variables positives ξ_i, pour $i = 1, \ldots, m$. Nous obtenons les inégalités suivantes :

$$(w.x_i) + b \geq +1 - \xi_i, \quad si \quad y_i = +1 \tag{3.22}$$

$$(w.x_i) + b \leq -1 + \xi_i, \quad si \quad y_i = -1 \tag{3.23}$$

Ainsi, au lieu de chercher le vecteur de poids w^0 qui minimise le carré de la norme $(w.w)$ comme dans le cas des données séparables, on cherche à minimiser la quantité suivante :

$$\phi(w, \xi) = \frac{1}{2} w.w + C \sum_{i=1}^{m} \xi_i \tag{3.24}$$

Où $\xi = (\xi_i,\ldots,\xi_m)$ et C est une constante choisie par l'utilisateur.

La solution λ_i^0 s'obtient de la même manière, par la maximisation du Lagrangien dual.

$$\max_\lambda L(w, b, \lambda)$$

Sous des contraintes, un peu différentes du cas d'une classification de données linéairement séparables :

$$\sum_{i=1}^{m} \lambda_i^0 y_i = 0 \quad 0 \leq \lambda_i \leq C \quad \forall i \in \{1, ..., m\}$$

Ainsi, nous obtenons l'équation de l'hyperplan optimal au sens des nouvelles contraintes tenant compte des ξ_i. Les vecteurs support sont toujours les vecteurs exemples les plus proches de cet hyperplan.

3.4.4 Principes des SVM

Nous venons de voir que l'idée fondamentale des SVM est de rendre possible la projection des vecteurs d'entrée dans un espace de représentation interne de plus grande dimension afin que l'hyperplan optimal construit sur cet espace soit général, indépendamment de la dimension de ce dernier.

Soit Ψ une fonction non linéaire qui transforme l'espace d'entrée E à un espace interne H de plus grande dimension.

$$\begin{aligned} \Psi : \mathrm{E} &\rightarrow \mathrm{H} \\ \mathrm{x} &\rightarrow \mathrm{h} \end{aligned}$$

où $x \in E$ et $h \in H$.

Ainsi l'ensemble d'exemples $S^{'}$ sera transformé sous la forme suivante :

$$S^{'} = \{(\Psi(x_i), y_i), x_i \in \Re^n; y_i \in \{-1, 1\}; \quad i = 1, ..., m\}$$

.

Pour construire l'hyperplan optimal dans l'espace des représentations internes sans considérer explicitement cet espace, il suffit de savoir calculer les produits scalaires entre les vecteurs support et les vecteurs de l'espace de représentation interne. Pour cela, on s'est appuyé sur le résultat mathématique suivant : dans un espace de Hilbert, un produit scalaire peut s'exprimer sous la forme :

$$K(x, x^*) = \textstyle\sum_{i=1}^{\infty} \Psi_i(x)\ \Psi_i(x^*)$$

où $K(x, x^*)$ est une fonction symétrique vérifiant les **conditions de Mercer** :

$$\int K(x, x^*)\ \varphi(x)\ \varphi(x^*)\ dx\ dx^* \geq 0$$

pour toute fonction $\varphi(x)$ telle que $\int\ \varphi(x)^2\ dx\ < \infty$.

Ainsi, il suffit de reprendre toutes les équations énoncées dans les paragraphes précédents, pour la recherche de l'hyperplan optimal, et de remplacer les produits scalaires des vecteurs d'entrée $\{(x_i.x_j)\}$ par leurs images dans l'espace de représentation interne $\{\Psi(x_i)\ .\ \Psi(x_j)\}$ ou tout simplement par des fonctions noyaux $K(x_i,x_j)$:

$$K(x_i,\ x_j) = \Psi(x_i).\Psi(x_j)$$

Par exemple, le problème d'optimisation à résoudre revient à la maximisation de la forme quadratique suivante (Lagrangien dual) :

$$F(\lambda) = \sum_{i=1}^{m} \lambda_i - \frac{1}{2} \sum_{i,j=1}^{m} \lambda_i \lambda_j y_i y_j K(x_i, x_j) \tag{3.25}$$

sous les contraintes :

$$\sum_{i=1}^{m} \lambda_i y_i = 0 \ ; \ 0 \leq \lambda_i \leq C \quad pour \ \ tout \quad i \in \{0, \ldots, m\}$$

Après avoir calculé les λ_i^0, on peut définir la fonction de classification par la formule suivante :

$$classe(x) = sign[\sum_{x_i \in VS} \lambda_i^0 y_i K(x_i, x) + b^0] \tag{3.26}$$

Les VS sont les vecteurs x_i associés aux $\lambda_i > 0$. Géométriquement, ce sont les vecteurs les plus proches de l'hyperplan optimal qui sépare les deux classes dans l'espace des représentations internes.

Plusieurs fonctions noyaux ont été utilisées dans les SVM, dont :

- **Le produit scalaire** :

$$K(x, y) = x \times y$$

- **Le noyau polynomial** :

$$K(x, y) = [(x \times y) + 1]^d$$

  où d est le degré du polynôme.

- **Le noyau RBF** (Radial Basis Function) :

$$K(x, y) = \exp(-\gamma |x - y|^2)$$

  où γ est à déterminer de façon empirique.

3.5 Support vector machines multi-classes

3.5.1 Introduction

Initialement les SVM ont été conçus principalement pour la classification binaire [Cortes et Vapnik, 1995]. La question de leur extension au problème de la classification multi-classes reste un thème de recherche. Ce problème muti-classes est typiquement résolu par la combinaison de plusieurs SVM binaires. Actuellement, il y a deux types d'approches pour les SVM multi-classes : une par décomposition en combinant plusieurs classifieurs binaires, l'autre par considération de toutes les classes dans une seule formule d'optimisation [Weston et Watkins, 1998] [Guermeur *et al.*, 1999] [Hsu et Lin, 2001].

Le problème multi-classes consiste à construire une fonction de décision, étant donnée la base d'apprentissage : $S = \{(x_i, y_i), x_i \in \Re^n; i = 1, \ldots, m, y_i \in \{1, ..., k\}\}$, où k représente le nombre de classes des exemples.

3.5.2 Résolution du problème multi-classes par des SVM binaires

Le premier type de SVM multi-classes est basé sur la combinaison de plusieurs classifieurs. Nous trouvons trois méthodes : un-contre-tous, un-contre-un, et la méthode DAGSVM (Directed Acyclic Graph Support Vector Machines).

La première méthode qui a été implémentée est l'approche un-contre-tous [Bottou *et al.*, 1994]. Cette méthode construit k modèles SVM (un SVM pour chaque classe). Le $i^{ème}$ SVM est appris avec tous les exemples. La $i^{ème}$ classe est indexée avec des étiquettes positives et toutes les autres avec des étiquettes négatives. Ce $i^{ème}$ classifieur construit un hyperplan entre la $i^{ème}$ classe et les $k-1$ autres classes.

La deuxième méthode, appelée un-contre-un, a été introduite par [Knerr *et al.*, 1990]. La première utilisation de cette stratégie avec les SVM a été réalisée par [Krebel, 1998]et [Friedman, 1996]. Cette méthode construit $k(k-1)/2$ classifieurs où chacun est appris sur les données de deux classes. Pendant la phase de test et après la construction de tous les classifieurs, on utilise la stratégie de vote proposée par [Friedman, 1996]. Si le signe de la fonction de décision des deux classes i et j indique que x appartient à la $i^{ème}$ classe, et le vote pour la $i^{ème}$ est incrémenté par $+1$. alors que, le vote pour la $j^{ème}$ classe est décrémenté de 1. À la fin, on prédit que x appartient à la classe qui a le vote le plus grand.

La troisième méthode, DAGSVM (Directed Acyclic Graph Support Vector Machines) a été proposée par [Platt *et al.*, 2000]. Sa phase d'apprentissage est la même que celle de la méthode un-contre-un, elle utilise la résolution proposée par les $k(k-1)/2$ classifieurs binaires. Cependant, dans la phase de test, elle utilise un graphe acyclique dirigé binaire avec une seule racine. Ce graphe possède $k(k-1)/2$ nœuds internes et k sorties. Chaque nœud est un SVM binaire pour des $i^{ème}$ et $j^{ème}$ classes. Pour un exemple x, en commençant par le nœud racine, la fonction de décision binaire est évaluée. Ensuite, elle se déplace à droite ou à gauche selon sa valeur de sortie. Enfin, on obtient un chemin de parcours de la racine jusqu'au nœud de sortie qui indique la classe recherchée. Les figures 3.10 et 3.11 donnent un exemple de décision à quatre classes.

3.5.3 Résolution du problème SVM multi-classes par une seule optimisation

Vapnik [Vapnik, 1998] et Weston [Weston et Watkins, 1999] ont proposé une autre approche pour la résolution du problème multi-classes. Cette approche se base sur la considération d'un seul et simple problème d'optimisation. L'idée est similaire à celle de la méthode un-contre-tous. On construit k règles de deux classes, où la $j^{ème}$ fonction $w_j.x + b$ sépare les vecteurs d'apprentissage qui correspondent à la classe j de tous les vecteurs des autres classes. Nous avons alors k fonctions de décision mais toutes s'obtiennent par la résolution d'un seul problème. Le problème d'optimisation du SVM binaire [Vapnik, 1995] est généralisé et il devient un problème de minimisation de la fonction suivante :

$$\phi(w, \xi) = \frac{1}{2}\sum_{j=1}^{k}(w_j \,.\, w_j) + C\sum_{i=1}^{m}\sum_{j \neq y_i} \xi_i^j$$

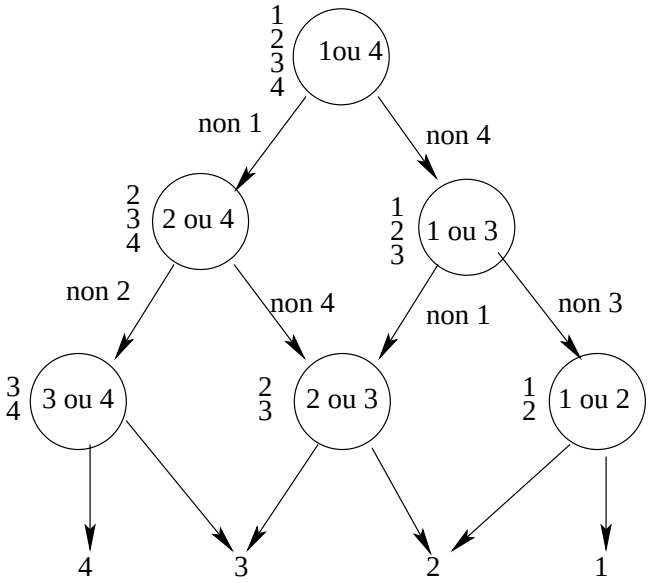

FIG. 3.10 – La décision DAG pour la recherche de la meilleure classe parmi 4 classes, la liste d'états équivalente pour chaque nœud est indiquée à côté du nœud.

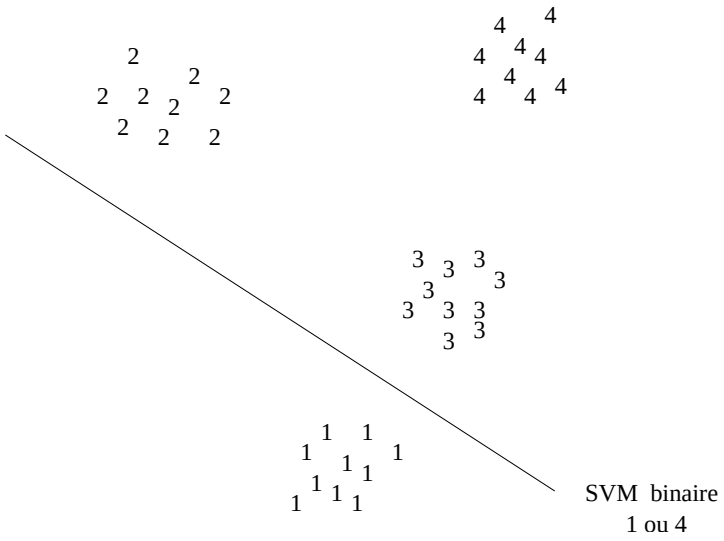

FIG. 3.11 – Diagramme de l'espace d'entrée pour un problème de 4 classes, un SVM binaire (un-contre-un) ne peut exclure qu'un seule classe.

$$avec \qquad (w_{y_i} \,.\, x_i) + b_{y_i} \;\geq\; (w_j \,.\, x_i) + b_j + 2 - \xi_i^j$$

$$et \qquad \xi_i^j \geq 0, \quad j = 1, ..., m, \quad j \in \{1, \ldots, k\} \backslash y_i$$

La solution du problème, revient à la recherche du point-selle du Lagrangien :

$$L(w, b, \xi, \alpha, \beta) = \frac{1}{2} \sum_{j=1}^{k} (w_j w_j) + C \sum_{i=1}^{m} \sum_{j=1}^{k} \xi_i^j$$

$$- \sum_{i=1}^{m} \sum_{j=1}^{k} \alpha_i^j [((w_{y_i} - w_j) x_i + b_{y_i} - b_j - 2 + \xi_i^j] - \sum_{i=1}^{m} \sum_{j=1}^{k} \beta_i^j \xi_i^j \tag{3.27}$$

avec les conditions :

$$\alpha_i^{y_i} = 0, \quad \xi_i^{y_i} = 2, \quad \beta_i^{y_i} = 0, \quad i = 1 \ldots, m$$

et les contraintes :

$$\alpha_i^j \geq 0, \quad \beta_i^j \geq , \quad \xi_i^j \geq 0, \quad i = 1 \ldots, m, \quad j \in \{1 \ldots, k\} \backslash y_i$$

où α et β sont les vecteurs multiplicateurs de Lagrange.

Le Lagrangien doit être maximisé par rapport à α et β et minimisé par rapport à w, b et ξ.

On introduit la notation :

$$A_i = \sum_{j=1}^{k} \alpha_i^j \qquad et \qquad c_i^n = \begin{cases} 1 & \text{if } y_i = n, \\ 0 & \text{if } y_i \neq n, \end{cases}$$

$L(w, b, \alpha, \beta, \xi)$ est minimal par rapport à w_n :

$$\frac{\delta L(w, b, \alpha, \beta, \xi)}{\delta w_n} = 0 \qquad \Rightarrow \qquad w_n = \sum_{i}^{m} (c_i^n A_i - \alpha_i^n) x_i \tag{3.28}$$

$L(w, b, \alpha, \beta, \xi)$ est minimal par rapport à b_n :

$$\frac{\delta L(w, b, \alpha, \beta, \xi)}{\delta b_n} = 0 \qquad \Rightarrow \qquad \sum_{i}^{m} \alpha_i^n = \sum_{i}^{m} c_i^n A_i \tag{3.29}$$

$L(w, b, \alpha, \beta, \xi)$ est minimal par rapport à ξ_n :

$$\frac{\delta L(w, b, \alpha, \beta, \xi)}{\delta \xi_n} = 0 \qquad \Rightarrow \qquad \beta_j^n + \alpha_j^n = C \qquad et \qquad 0 \leq \alpha_j^n \leq C \tag{3.30}$$

En remplaçant w_n de l'équation 3.28, par sa valeur dans l'équation 3.27, on obtient :

$$W(w, b, \xi, \alpha, \beta) = \frac{1}{2} \sum_{r=1}^{k} \sum_{i=1}^{m} \sum_{j=1}^{m} (c_i^r A_i - \alpha_i^r)(c_j^r A_j - \alpha_j^r)(x_i \,.\, x_j)$$

$$-\sum_{r=1}^{k}\sum_{i=1}^{m}\alpha_i^r[\sum_{j=1}^{m}(c_j^{y_i}A_j-\alpha_j^{y_i})(x_i\,.\,x_j)-\sum_{j=1}^{m}(c_j^rA_j-\alpha_j^r)(x_i\,.\,x_j)+b_{y_i}-b_r-2]$$

$$-\sum_{r=1}^{k}\sum_{i=1}^{m}\alpha_i^r\xi_i^r+C\sum_{r=1}^{k}\sum_{i=1}^{m}\xi_i^r-\sum_{i=1}^{m}\sum_{r=1}^{k}\beta_i^r\xi_i^r \tag{3.31}$$

En tenant compte de la contrainte de l'équation 3.30, la troisième partie de l'équation 3.31 qui est en fonction de ξ s'annule.

Soit B_1 et B_2 deux paramètres définit par :

$$B_1=\sum_{i,r}\alpha_i^rb_{y_i}=\sum_r b_r(\sum_i c_i^rA_i)$$

$$B_2=-\sum_{i,r}\alpha_i^rb_r=-\sum_r b_r(\sum_i\alpha_i^r)$$

d'après l'équation 3.29, on a :

$$\sum_i\alpha_i^r=\sum_i c_i^rA_i$$

donc $B_1=-B_2$, l'équation 3.31 devient :

$$W(\alpha)=2\sum_{i,r}\alpha_i^r+\sum_{i,j,r}(\frac{1}{2}c_i^rc_j^rA_iA_j-\frac{1}{2}c_i^rA_i\alpha_j^r-\frac{1}{2}c_j^rA_j\alpha_i^r+\frac{1}{2}\alpha_i^r\alpha_j^r-$$

$$c_j^{y_i}A_j\alpha_i^r+\alpha_i^r\alpha_i^{y_i}+c_j^rA_j\alpha_i^r-\alpha_i^r\alpha_j^r)(x_i\,.\,x_j)$$

or

$$\sum_r c_i^rA_i\alpha_j^r=\sum_r c_j^rA_j\alpha_i^r$$

donc

$$W(\alpha)=2\sum_{i,r}\alpha_i^r+\sum_{i,j,r}(\frac{1}{2}c_i^rc_j^rA_iA_j-\frac{1}{2}c_i^rA_i\alpha_j^r-\frac{1}{2}c_j^rA_j\alpha_i^r+\frac{1}{2}\alpha_i^r\alpha_j^r-$$

$$c_j^{y_i}A_j\alpha_i^r+\alpha_i^r\alpha_i^{y_i}+c_j^rA_j\alpha_i^r-\alpha_i^r\alpha_j^r)(x_i\,.\,x_j)$$

mais

$$\sum_r c_i^rc_j^r=c_i^{y_i}=c_j^{y_i}$$

donc, nous obtenons la formule suivante :

$$W(\alpha)=2\sum_{i,j}\alpha_i^j\;+\;\sum_{i,r,j}[-\frac{1}{2}\;c_r^{y_i}A_iA_r\;+\;\alpha_i^j\alpha_r^{y_i}-\frac{1}{2}\;\alpha_i^j\;\alpha_r^j]\;\;(x_i\,.\,x_r)$$

C'est une fonction quadratique à variable α et avec les contraintes suivantes :

$$\sum_{i=1}^{m}\alpha_i^n=\sum_{i=1}^{m}c_i^nA_i,\quad n=1,...,k$$

$$0 \leq \alpha_i^j \leq C, \quad \alpha_i^{y_i} = 0, \quad i = 1, ..., m \quad j \in \{1, ..., k\} \backslash y_i$$

Ainsi on obtient la fonction de décision suivante :

$$f(x) = arg\ max_n [\sum_{i=1}^{m} (c_i^n A_i - \alpha_i^n)(x_i.x) + b_n] \quad n = 1, ..., k \tag{3.32}$$

Comme précédemment, il suffit de remplacer le produit scalaire $(x_i.x_r)$ par son image dans l'espace de représentation interne en utilisant la fonction de projection Ψ ou par la fonction noyau K :

$$(x_i.x_r) \Rightarrow (\Psi(x_i).\Psi(x_r)) = K(x_i.x_r)$$

Ainsi, nous avons résolu le problème de classification multi-classes, la fonction de classification est $f(x)$ donnée par l'équation 3.32.

En conclusion, les SVM multi-classes sont employés dans plusieurs domaines et ils ont prouvé leur efficacité à identifier les différentes classes des données qui lui sont présentées. Pour notre tâche de détection et afin de classer les mots clés de notre application, nous envisageons d'employer le dernier type des SVM multi-classes où le problème de classification multi-classes est résolu en considérant une seule optimisation.

3.6 Conclusion

La détection de mots clés dans un flux de parole est un processus de vérification des mots reconnus, il s'agit donc d'une classification de ces mots. Parmi les méthodes de classification, nous avons pensé en premier lieu à employer les PMC et en second lieu à proposer l'utilisation des SVM pour la détection de mots clés.

Les PMC sont les réseaux de neurones les plus connus, ils utilisent un paradigme d'apprentissage supervisé. Ils ont été largement utilisé dans le domaine de la reconnaissance automatique de la parole et ils ont fait leurs preuves dans plusieurs problèmes de classification.

L'application des SVM consiste à projeter les données d'entrée dans un espace de plus grande dimension (espace de caractéristiques) dans l'optique que les données non linéairement séparables dans l'espace d'entrée deviennent linéairement séparables (dans l'espace de caractéristiques). Ceci permet de construire un hyperplan optimal séparant les deux classes. Les SVM ont montré leur efficacité dans un grand nombre d'applications. Leur utilisation comme moyen de classification des entrées correctes et incorrectes dans le domaine de la détection de mots clés, peut donner de très bons résultats. De plus, les SVM nous permettront d'exploiter un grand nombre de caractéristiques et de les combiner afin d'améliorer les performances de notre système.

Les SVM multi-classes, qui présentent les mêmes avantages que les SVM biclasses, sont aussi considérés comme une autre alternative pour la résolution du problème de détection des mots clés. En effet, classer les entrées en plusieurs classes présente deux intérêts supplémentaires.

D'une part on peut classer les mots non seulement en correct et incorrect mais aussi donner la vraie classe de ce mot (même en présence d'erreurs de reconnaissance). D'autre part, cette classification a un plus grand pouvoir de discrimination puisqu'on répartit plus nos données et donc le risque d'ambiguïté est atténué.

Il s'avère donc intéressant d'essayer ces techniques dans nos travaux et de voir quelle amélioration ils peuvent apporter. L'application des SVM pour la détection de mots clés est présentée au chapitre 7.

Deuxième partie

Chapitre 4

Système de reconnaissance et méthodes d'évaluation

4.1 Introduction

Dans cette deuxième partie du manuscrit, nous présentons nos réalisations et nos contributions à la détection de mots clés. Afin de souligner les apports des méthodes proposées, nous nous sommes trouvés face à un problème de comparaison délicat et non évident. En effet, si nous voulons comparer plusieurs systèmes de détection, il nous faut satisfaire un certain nombre de conditions. Premièrement, nous avons besoin de définir les mêmes critères d'évaluation pour toutes les méthodes. Deuxièmement, nous devons utiliser le même vocabulaire et donc la même base d'apprentissage et la même base de test pour une évaluation égale. Troisièmement et pour que notre comparaison soit plus intéressante et plus correcte, il est préférable d'employer le même système de reconnaissance.

Dans ce chapitre qui nous sert de base pour comparer et évaluer toutes les méthodes réalisées, nous décrivons les trois outils de base de comparaison évoqués ci-dessus. Tout d'abord, dans la section 4.2, nous donnons une description détaillée du système de reconnaissance utilisé. Ensuite dans la section 4.3, nous présentons la base de données que nous avons employée durant tout le travail de thèse. Enfin dans la section 4.4, nous montrons le problème du choix des critères d'évaluation et nous présentons les courbes ROC, les courbes de rappel/précision et la notion d'intervalle de confiance.

4.2 Description du système de reconnaissance

Dans cette étude, nous utilisons le système de reconnaissance ESPERE (Engine for SPEech REcognition), qui est un système à base de HMM développé au LORIA [Fohr *et al.*, 2000]. Ce système est composé principalement de trois modules : un module de traitement acoustique, un module d'apprentissage et un moteur de reconnaissance.

- **le module de traitement acoustique :**
  La paramétrisation du signal est basée sur les coefficients MFCC. Cependant, l'utilisateur peut changer cette paramétrisation. Par exemple, la taille de la fenêtre d'analyse et son recouvrement, le nombre de filtres triangulaires, les fréquences de coupure des filtres et

le nombre de coefficients cepstraux. L'utilisateur peut aussi enlever le premier coefficient MFCC C_0 et le remplacer par le logarithme de l'énergie du signal. Enfin les dérivées et l'accélération peuvent être ajoutées.

- **le module d'apprentissage :**
  Le noyau de ce module utilise l'algorithme de ré-estimation de Baum Welch avec densités continues. L'utilisateur peut définir la topologie du modèle de Markov caché c'est-à-dire le nombre d'états et les transitions permises. L'unité modélisée peut être un mot, un triphone ou un phonème. Ces unités sont apprises soit de façon isolée soit de façon continue.

  Le nombre de fonctions de densité de probabilité par état est déterminé durant la phase d'apprentissage. En effet, le nombre des composantes du mélange est incrémenté de manière itérative jusqu'à atteindre la valeur désirée par un mécanisme de dédoublement.

- **le moteur de reconnaissance :**
  Le moteur de reconnaissance met en application un algorithme synchrone à un seul passage. Il utilise un HMM appris comme décrit précédemment, un lexique et une grammaire. La structure du lexique permet à l'utilisateur plusieurs prononciations possibles pour un mot donné. La grammaire peut présenter les couples de mots qui se suivent appelés "bigrammes". Pour accélérer la reconnaissance, un seuil de pruning (qui représente le pourcentage des états actifs prolongés à la trame suivante) peut être appliqué.

En conclusion l'architecture modulaire d'ESPERE et ses facilités d'adaptation fournissent un toolkit de recherche qui permet le test de nouveaux algorithmes, modèles et paramètres acoustiques. L'implémentation du logiciel ESPERE contient plus de 20000 lignes de code en C^{++} et il fonctionne sur un PC-Linux ou PC-Windows.

Pour faire l'apprentissage de notre système de reconnaissance, nous utilisons une modélisation contextuelle des phonèmes. Ces modèles de phonèmes dépendants du contexte permettent de tenir compte du phénomène de coarticulation. Il s'agit des triphones, un triphone tient compte de deux phonèmes, le phonème précédent et le phonème suivant afin de fixer le contexte. Chaque triphone est modélisé par un HMM à 3 états (gauche-droite) à densités continues, avec 4 gaussiennes par état.

L'extraction des paramètres a été faite en se basant sur les coefficients Mel-Cepstre (MFCC). Nous avons utilisé une fenêtre d'analyse de 256 échantillons de signal et 24 filtres triangulaires. À chaque trame, correspond un vecteur de 35 coefficients : 11 MFCC (le premier coefficient C_0 est enlevé), 12 dérivées premières et 12 dérivées secondes.

4.3 Base de données

Pour l'apprentissage du système de reconnaissance, nous avons utilisé un corpus extrait de la base de données française SPEECHDAT 1000 (voir annexe). Le corpus utilisé est composé de 8836 phrases phonétiquement riches, prononcées par 1000 locuteurs et enregistrées à travers le réseau téléphonique. Cette base est utilisée pour faire l'apprentissage de 9562 modèles de triphones.

La base de test utilisée contient 2245 phrases lues représentant une durée de 3 heures et 23 minutes. Ces phrases appartiennent à la base de données française SPEECHDAT 5000 (voir annexe). Ces phrases, phonétiquement riches sont prononcées par 900 locuteurs, différents de ceux qui ont participé à l'enregistrement de la base d'apprentissage. Nous avons choisi la base SPEECHDAT car d'une part, elle constitue une base connue et de taille importante et, d'autre part, elle représente une richesse tant au niveau phonétique qu'au niveau des locuteurs. Cette base nous servira ensuite pour l'évaluation des différentes méthodes proposées.

Afin de fixer les paramètres de nos différentes méthodes, nous avons utilisé une base de développement composée de 250 phrases extraites de la base de données française SPEECHDAT 1000 (voir annexe). Ces phrases, sont différentes de celles utilisées dans la phase d'apprentissage.

Pour notre tâche de détection, nous nous sommes inspirés des travaux de Mathan [Mathan et Miclet, 1991] et de Lai [Lai et Shi, 2001] pour fixer un ensemble de 20 mots clés qui sont les suivants : *lundi, mardi, mercredi, jeudi, vendredi, samedi, dimanche, janvier, février, mars, avril, mai, juin, juillet, août, septembre, octobre, novembre, décembre, premier*. La liste de ces mots et le nombre de leurs occurrences sont fournis dans le tableau 4.1.

Dans le corpus de test utilisé, nous disposons de 15820 mots dont 3220 sont des occurrences de mots clés et 12600 sont des mots hors-vocabulaire. Notre tâche consiste à détecter les occurrences des 20 mots clés choisis dans cette base de test.

Mot clé	**Nombre d'occurrences**
Lundi	160
Mardi	184
Mercredi	136
Jeudi	219
Vendredi	150
Samedi	188
Dimanche	182
Janvier	178
Février	145
Mars	147
Avril	143
Mai	161
Juin	152
Juillet	171
Août	164
Septembre	157
Octobre	181
Novembre	156
Décembre	132
Premier	116

TAB. 4.1 – La liste des mots clés choisis et leurs nombres d'occurrences dans la base de test.

4.4 Les systèmes d'évaluation

4.4.1 Introduction

Dans le domaine de la reconnaissance automatique de la parole, il est très difficile de faire une comparaison entre deux systèmes de reconnaissance mais nous pouvons les classifier suivant des critères, comme par exemple la rapidité de reconnaissance, la taille de vocabulaire, la grammaire utilisée, la robustesse au bruit, l'indépendance vis-à-vis des locuteurs etc. Néanmoins, le critère définitif de comparaison entre deux systèmes reste le taux de reconnaissance, mais cette comparaison n'est valide que si les deux systèmes sont évalués au moins dans les mêmes conditions d'application.

4.4.2 Estimation du taux d'erreur

La reconnaissance automatique de la parole implique la bonne gestion des erreurs de reconnaissance. Ces erreurs d'origines diverses doivent être estimées d'une façon identique pour qu'on puisse faire la comparaison entre les différents types de systèmes de reconnaissance. Par exemple dans une application de reconnaissance de la parole continue, nous pouvons trouver trois types d'erreurs :

- Erreur d'insertion : le système reconnaît un mot alors qu'il n'est pas prononcé,
- Erreur de substitution : le système confond deux mots différents c'est-à-dire pour un mot prononcé le système reconnaît un autre mot,
- Erreur d'omission : le système ne reconnaît pas un mot prononcé.

L'existence de ces trois types d'erreurs va compliquer le problème d'évaluation surtout qu'une erreur de substitution peut être considérée comme étant la combinaison simultanée d'une erreur d'omission et d'une erreur d'insertion ce qui revient alors à considérer cette erreur comme double.

Dans la littérature, plusieurs définitions ont été introduites afin de calculer et de comparer les performances des systèmes de reconnaissance. Parmi ces définitions, nous présentons le Taux de Mots Corrects (TMC) (Word Correct Rate) qui est la mesure la plus intuitive pour mesurer les performances d'un système de reconnaissance de la parole. Elle est basée sur le nombre de mots correctement reconnus et le nombre de mots total à reconnaître. La formule de TMC s'écrit sous la forme suivante :

$$TMC = 100 \times \frac{Nombre\ \ de\ \ mots\ \ correctement\ \ reconnus}{Nombre\ \ total\ \ des\ \ mots\ \ à\ \ reconnaitre}$$

Le problème majeur de cette mesure c'est qu'elle ne tient pas compte de toutes les erreurs commises, car elle ne dépend pas du nombre d'insertions. C'est pour cette raison que dans la majorité des évaluations une autre mesure est utilisée, le Taux d'Erreur (TE) calculé au niveau mot, appelé aussi WER (Word Error Rate). Il est calculé de la façon suivante :

$$TE = 100 \times \frac{Nombre\ \ d'insertions + Nombre\ \ de\ \ substitutions + Nombre\ \ d'omissions}{Nombre\ \ total\ \ des\ \ mots\ \ à\ \ reconnaitre}$$

La mesure du taux d'erreur TE est plus fidèle que celle du TMC, puisqu'elle permet la mise en évidence de toutes les erreurs du système de reconnaissance. Malgré que cette mesure soit plus juste, elle possède aussi un revers. En effet, dans des tâches telle que la détection de mots clés dans un flux de parole continue, il est évident que nous avons beaucoup d'erreurs d'insertion et dans les cas extrêmes nous trouvons plus d'insertions que de mots à reconnaître. Dans ces conditions, le taux d'erreur (TE) sera donc supérieur à 1 et le taux de reconnaissance (TMC) sera inférieur à zéro d'où la nécessité de trouver un autre système d'évaluation en plus de cette simple mesure.

4.4.3 Les Courbes ROC : Receiver Operating Characteristic

Dans une tâche de détection de mots clés dans un flux de parole, nous disposons d'un ensemble de mots prononcés et dans lequel nous nous intéressons uniquement à la détection d'un sous-ensemble plus restreint contenant des mots clés fixés a priori par l'utilisateur et qui dépendent de l'application considérée. Pour l'évaluation d'un tel système, les définitions précédentes du paragraphe 4.4.2 ne sont plus valides car la nature des erreurs a changé. Nous distinguons alors deux nouveaux types d'erreurs possibles qui sont :

- Erreur de type I : se produit quand le système de reconnaissance ne détecte pas un mot clé alors qu'il est prononcé, appelée encore Faux Rejet (FR),
- Erreur de type II : se produit quand le système de reconnaissance détecte un mot clé alors qu'il n'est pas prononcé, appelée encore Fausse Acceptation ou Fausse Alarme (FA).

Nous définissons alors deux mesures pour ces deux types d'erreurs qui sont le Taux de Faux Rejet (TFR) et le Taux de Fausse Acceptation (TFA) données respectivement par les deux formules suivantes :

$$TFR = \frac{Nombre\ \ de\ \ faux\ \ rejets}{Nombre\ \ total\ \ de\ \ mots\ \ clés\ \ à\ \ détecter} \tag{4.1}$$

$$TFA = \frac{Nombre\ \ de\ \ fausses\ \ acceptations}{Nombre\ \ total\ \ de\ \ mots\ \ non-clés} \tag{4.2}$$

Pour un système de détection, chaque couple de valeurs des taux TFR et TFA représente un point de fonctionnement. Afin de représenter toutes les possibilités du système, il est préférable de considérer plusieurs points de fonctionnement. Il s'agit de représenter un taux en fonction de l'autre obtenant ainsi une courbe de performance pour le système. La courbe obtenue est une courbe ROC (Receiver Operating Characteristic) appelée aussi *courbe caractéristique d'opération du récepteur.*

Une courbe ROC est un graphe qui peut être utilisé aussi pour représenter le Taux de Détection (TD) de mots clés (avec $TD = TMC$) en fonction du taux de fausses acceptations (TFA) [Egan, 1975] [Bradley, 1997]ou bien pour représenter le taux de détection de mots non-clés, c'est-à-dire Taux de Rejet Correct (TRC) en fonction du taux de faux rejet (TFR) [Williams,

1999] [Hazen et Bazzi, 2001]. Nous remarquons que les informations contenues dans ces deux types de courbes sont équivalentes puisque nous avons :

$$TD = 1 - TFR$$

$$TRC = 1 - TFA$$

La figure 4.1 illustre trois courbes ROC possibles de type TD en fonction de TFA. La première correspond à un test idéal, avec un taux de détection TD égal à 1 (TFR =0) pour tous les TFA possibles (le meilleur cas correspond à un taux de fausse acceptation TFA égal à zéro). La deuxième représente un test d'un cas limite avec un taux de détection TD égal au taux de fausse acceptation TFA et elle est représentée par la droite $TFA = TD$. La troisième courbe est un courbe ROC typique qui se trouve entre ces deux courbes extrêmes.

Une autre disposition possible des courbes ROC consiste à représenter la probabilité d'erreur de type I en fonction de la probabilité d'erreur de type II, c'est-à-dire le taux de faux rejet TFR en fonction du taux de fausses acceptations TFA [Verlinde *et al.*, 2000]. La figure 4.2 fournit une illustration schématique de deux courbes ROC possibles correspondant à deux tests : un cas limite et un cas typique. Le cas limite est représenté par une droite d'équation $TFA+TFR = 1$ et donc $TFR = 1 - TFA$. Nous savons bien que pour un système parfait, nous avons $TFA + TFR = 0$, ce qui correspond au point d'origine $TFA = TFR = 0$, un cas typique est représenté donc par une courbe qui se trouve entre ces deux extrêmes.

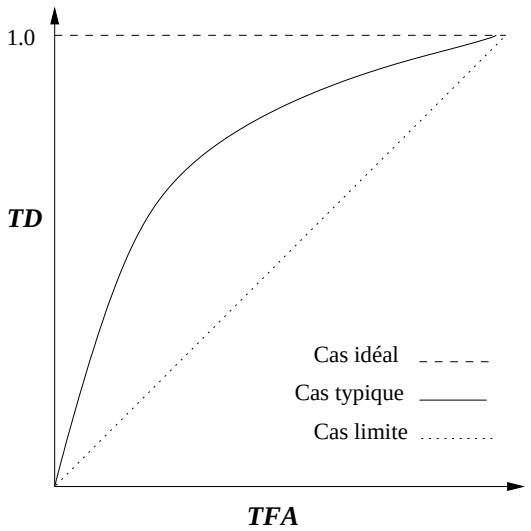

FIG. 4.1 – Illustration schématique de courbes ROC pour un test idéal, estimé et typique.

Soit T un seuil de classification, ce seuil est utilisé pour accepter ou rejeter la réponse du système de détection. Autrement dit, si le score de la réponse du système de détection est supérieur à T alors nous acceptons cette réponse, sinon elle sera rejetée. La variation du seuil T le long de l'axe des abscisses donne différentes valeurs de TFR et de TFA. Quand elles sont tracées, elles donnent le graphe théorique illustré par la figure 4.3.

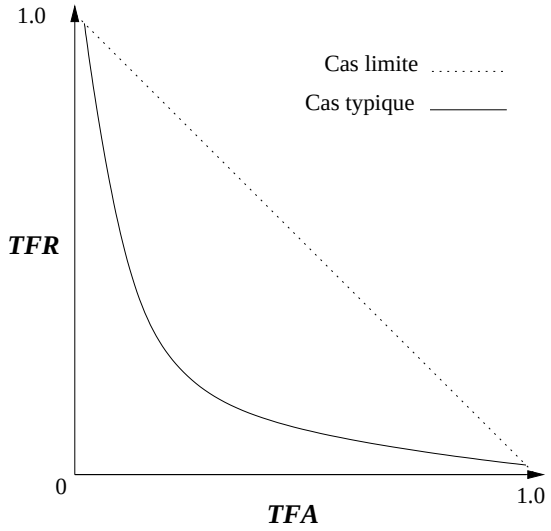

FIG. 4.2 – Illustration schématique de courbes *ROC* de type P(erreur de type II) en fonction de P(erreur de type I).

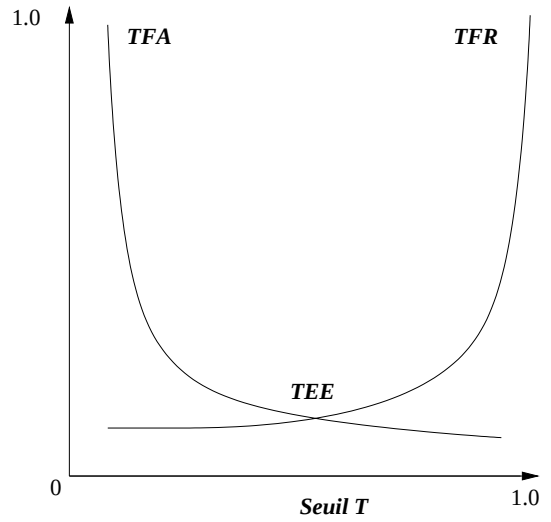

FIG. 4.3 – TFA et TFR en fonction du seuil T.

Quand on fait augmenter la valeur du seuil T alors le TFA diminue et le TFR augmente. Un grand TFA signifie qu'un mot non-clé a une grande tendance d'être accepté comme étant un mot clé, alors qu'un grand TFR signifie qu'un mot clé a une grande tendance d'être rejeté. Ainsi un grand TFA rend le système moins efficace, puisque par exemple, dans une application d'accès par mot de passe, n'importe quel utilisateur peut accéder au système. De même un grand TFR va compliquer l'utilisation du système.

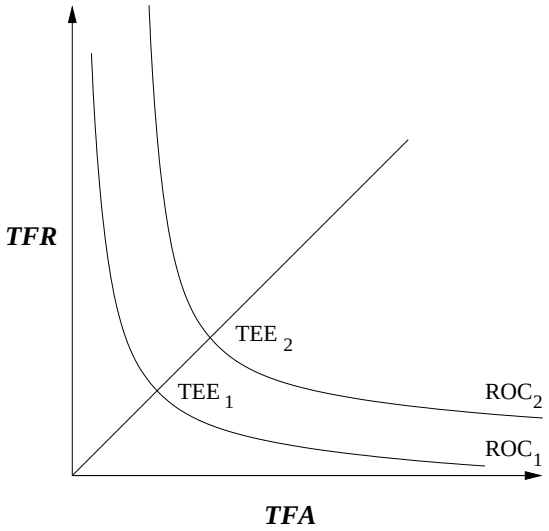

FIG. 4.4 – TFR en fonction TFA.

Il est impossible de minimiser les deux taux TFR et TFA en même temps comme le montre la figure 4.4. Cependant un compromis peut être atteint quand $TFR = TFA$, ce point s'appelle le Taux d'Erreur Égale (TEE). La valeur du TEE peut être utilisée pour comparer les résultats de deux systèmes de détection. Le meilleur système c'est celui qui a le plus petit TEE. En effet, si la valeur du TEE est faible alors les valeurs TFA et TFR le sont aussi et donc le système commet peu de fautes. Dans la figure 4.4, nous remarquons que le TEE_1 est meilleur que le TEE_2 et donc la courbe ROC_1 montre une meilleure qualité du système que la courbe ROC_2. Plus la courbe est proche de l'origine plus le système est performant.

Un autre type de courbes ROC bien connu dans le domaine de la détection de mots clés consiste à représenter le taux de détection (TD) en fonction du nombre de Fausses Acceptations par Mot Clé et par Heure (FA/MC/H). Cette disposition de courbes ROC prend mieux en compte la notion du temps puisqu'au niveau de l'axe des abscisses le nombre de fausses acceptations est normalisé par le nombre d'heures. Ce type de courbes est considéré comme étant la plus efficace pour la comparaison de différents systèmes de détection. Dans la figure 4.5, nous illustrons un exemple de deux courbes ROC de ce type où la courbe ROC_1 montre une meilleure performance du système que la courbe ROC_2.

Pour obtenir une valeur unique décrivant cette courbe, nous utilisons la moyenne des probabilités de détection pour un taux de fausses acceptations par mot clé et par heure variant entre 0 et 10. Cette valeur est appelée *Valeur de mérite* ou FOM pour "Figure Of Merit" et elle est utilisée pour mesurer les performances d'un système de détection de mots clés pour de faibles taux de fausses acceptations (FA/Mc/H entre 0 et 10) [Yapanel, 1997]. La formule du FOM

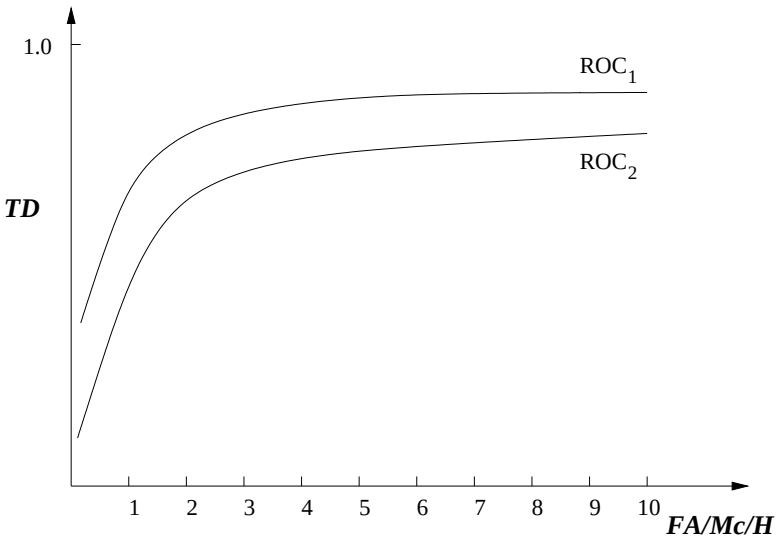

FIG. 4.5 – Courbe ROC, indicateur FOM

préconisée par le NIST (National Institute of Standards and Technology, USA) est donnée par :

$$FOM = \frac{(p_1 + p_2 + \ldots + p_N + a p_{N+1})}{10 \times T}$$

où :
p_i est le pourcentage de détections correctes avant la $i^{ème}$ fausse acceptation.
T est la durée du signal de parole.
N est le premier entier $\geq 10T - 1/2$.
$a = 10T - N$ (facteur d'interpolation à 10 fausses acceptations par heure).

Dans le cas où notre base de test contient exactement une heure de parole, la formule du FOM se simplifie à :

$$FOM = \frac{(p_1 + p_2 + \ldots + p_{10})}{10}$$

Les probabilités de détection p_i sont calculées de deux manières différentes selon la valeur de la durée T de la base de test. En effet, si $T \leq 1$ alors la probabilité p_i est placée sur la courbe ROC correspondante au niveau de l'abscisse $x = (i - 1/2)$. Sinon, cette valeur est reportée au point d'abscisse $x = (i - 1/2)/T$ fausses acceptations par heure et par mot clé.

4.4.4 Les mesures de rappel et de précision

Parmi les objectifs d'un système de détection figure la diminution du nombre de fausses acceptations et de faux rejets. Afin de mesurer la qualité d'un système par rapport à cet objectif on peut utiliser deux critères bien connus qui se nomment "précision" et "rappel". Ces deux mesures sont fréquemment utilisées dans plusieurs domaines faisant appel à des techniques proches de la statistique et surtout dans ceux qui utilisent des filtres binaires (soit un objet est sélectionné par le filtre, soit il ne l'est pas). En particulier, on les rencontre en permanence en traitement

automatique de la langue, aussi bien en analyse ou en compréhension de texte qu'en fouille de documents, etc. Dans notre cas, ces deux mesures vont nous servir pour évaluer les performances de notre système de détection.

Définitions

La précision d'un ensemble de mots clés détectés par le système est la proportion des mots clés corrects dans cet ensemble. Elle est définie par :

$$précision = \frac{Nombre\ de\ mots\ clés\ correctement\ détectés}{Nombre\ total\ des\ mots\ clés\ détectés}$$

Le rappel d'un ensemble de mots clés détectés rend compte de la quantité de bonnes réponses par rapport au nombre de mots clés réel. Autrement dit, le rappel est le taux de mots clés corrects détectés par rapport au nombre des mots clés à détecter. Il est défini par :

$$rappel = \frac{Nombre\ de\ mots\ clés\ correctement\ détectés}{Nombre\ de\ mots\ clés\ \ à\ détecter}$$

On peut également définir les notions de "bruit" et de "silence" qui sont respectivement les notions complémentaires de la "précision" et du "rappel" :

$$bruit = 1 - précision \quad et \quad silence = 1 - rappel$$

Les courbes Rappel/Précision

Idéalement, on voudrait qu'un système de détection donne de bons taux de précision et de rappel en même temps. Cependant, un système qui donne 100% de rappel et 100% de précision est non atteignable en pratique car cela signifie que ce système détecte tous les mots clés corrects et rien que les mots clés corrects chose parfaite mais très difficilement réalisable. En effet, les deux mesures de rappel et de précision sont intimement reliées et il n'est pas possible d'augmenter l'une sans diminuer l'autre. Par exemple, en tentant d'améliorer la précision, c'est-à-dire d'augmenter la performance du système à ne détecter que les mots clés corrects, on réduit nécessairement sa fenêtre de possibilités et dans ce cas on réduit aussi sa capacité de rappel car on va provoquer beaucoup de silence. De même on peut facilement avoir un très bon taux de rappel en relâchant les contraintes de détection et on aura dans ce cas un grand nombre de mots clés corrects mais aussi un grand nombre de mots clés non corrects c'est-à-dire beaucoup de bruit et donc un faible taux de précision.

Les mesures de précision et de rappel ne sont pas statiques, c'est-à-dire qu'un système n'a pas qu'une seule valeur de mesure de précision ou de rappel. Le comportement d'un système peut varier en faveur de l'une au détriment de l'autre et vice-versa. Ainsi pour un système donné, on peut tracer l'évolution de la précision en fonction du rappel par une courbe rappel/précision.

L'allure générale d'une courbe "rappel/précision" (pour un cas typique) est donnée dans la figure 4.6. Cette courbe montre qu'il est toujours possible d'obtenir une précision élevée au prix

d'un rappel faible ou un rappel élevé au prix d'une précision faible. Dans la pratique, on essaye de trouver un compromis entre ces deux exigences. La courbe optimale, représentée par une droite horizontale à un niveau de précision égale à 1, correspond à un cas idéal où le système arrive à détecter tous les mots clés corrects et rien que ceux-ci.

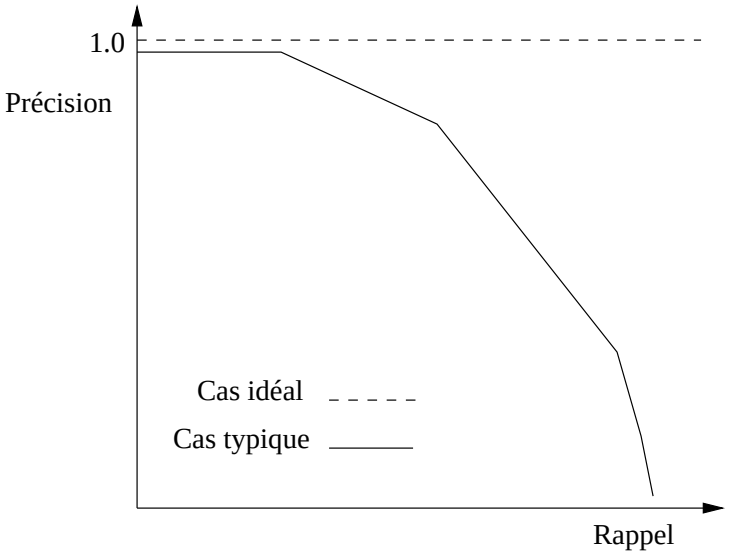

FIG. 4.6 – Courbes rappel/précision

Dans la littérature, on parle aussi de courbe rappel/précision interpolée où on essaye de calculer la précision pour des valeurs prédéfinies du rappel, de 0% à 100% par pas de 10%. En pratique, les valeurs du rappel peuvent ne pas être atteintes exactement : les valeurs de la précision doivent donc être interpolées. La règle d'interpolation généralement utilisée définit la valeur de la précision pour un niveau de rappel i comme étant la valeur maximale de précision pour tout niveau de rappel supérieur ou égal à i. Cette définition permet d'obtenir une valeur de précision pour un niveau de rappel nul, alors qu'une telle valeur n'existe pas en réalité, et ce en adoptant la valeur de précision la plus importante. Par exemple, s'il existe 200 mots clés, la valeur interpolée de la précision pour un rappel de 10% correspond à la meilleure précision obtenue avec au moins 20 mots clés corrects.

L'avantage de cette interpolation est qu'elle permet de connaître la précision pour des valeurs standardisées. Ainsi, si on étudie les performances du système pour chaque mot séparément, on peut facilement obtenir la courbe moyenne du système en moyennant simplement toutes les précisions obtenues aux différents niveaux du rappel pour tous les mots. Ainsi, les performances d'un système de détection sur un ensemble de mots peuvent être caractérisées par une seule courbe.

La mesure F

Une autre manière de tenir compte de ces deux mesures est de les combiner en une seule mesure. La F-measure de Van Risbergen [Rijsbergen, 1979], ou encore efficacité globale, combine précision et rappel en une mesure unique donnée par :

$$F = \frac{2 \times précision \times rappel}{précision + rappel}$$

Cette mesure représente la moyenne harmonique du rappel et de la précision et elle a quelques caractéristiques intéressantes. Par exemple, si $précision = rappel = x$ alors $F = x$. En plus, cette mesure décline dès que l'une des deux valeurs est faible et elle est maximale quand les deux quantités le sont. Il s'agit donc d'une mesure à interprétation facile, fidèle à la variation des deux quantités qui la composent. Cependant, il ne faut pas oublier qu'une partie de l'information est perdue, puisque cette mesure rend compte d'un seul point de la courbe dans la zone des bonnes performances (où généralement le rappel et la précision sont du même ordre de grandeur).

4.4.5 Intervalle de confiance

Niveau de confiance

Dans la pratique, il est fréquent d'introduire la notion d'incertitude des mesures en estimant à chaque fois un intervalle de confiance. Nous cherchons à déterminer l'intervalle de confiance $[a, b]$ centré sur la valeur x qui est une valeur estimée d'un paramètre inconnu θ. Nous supposons que cet intervalle contient la vraie valeur de θ avec une probabilité de $1 - \alpha$ fixée a priori.

$$P[a < \theta < b] = 1 - \alpha$$

Cette probabilité est appelée **niveau de confiance** de l'estimation, on la désigne par $1 - \alpha$. Le α (**coefficient de confiance**) représente le risque que nous prenons de se tromper en affirmant que θ est bien dans l'intervalle proposé. Une estimation par intervalle de confiance sera d'autant meilleure que l'intervalle sera petit pour un coefficient de confiance grand.

Pour déterminer l'intervalle de confiance, nous avons besoin de connaître outre la taille de l'échantillon, la loi de probabilité du paramètre à estimer. Comme il n'existe pas de résolution générale pour ce dernier problème, différentes solutions ont été proposées. Nous nous intéressons dans ce manuscrit à la solution la plus adoptée qui est l'estimation d'une proportion.

Estimation d'une proportion

Soit une population dont les individus possèdent un caractère A avec une probabilité p. On cherche à déterminer cette probabilité inconnue en prélevant un échantillon de taille N dans cette population. Nous constatons que n parmi les N individus possèdent le caractère A. La proportion $f_N = \frac{n}{N}$ approxime la vraie valeur de p mais avec quelle confiance ?

Soit $F_N = \frac{n}{N}$; F_N une variable aléatoire construite comme étant la somme de N variables aléatoires $O_1, ..., O_N$ et de même paramètre p. Il s'agit donc, d'après le théorème central limite, d'une variable aléatoire dont la loi de probabilité tend vers une loi normale de moyenne p et d'écart type $\sqrt{\frac{p(1-p)}{N}}$. Cette approximation est valable uniquement si la taille de l'échantillon est suffisamment grande (en pratique $N > 50$).

Nous définissons alors l'intervalle de confiance autour de p de la façon suivante :

$$P(|f_N - p| < t) = 1 - \alpha$$

$$P(f_N - t < p < f_N + t) = 1 - \alpha$$

Avec α est le risque , f_N est une réalisation d'une variable aléatoire $N(p, \sqrt{\frac{p(1-p)}{N}})$.

Par une normalisation et un centrage nous obtenons une nouvelle variable aléatoire U définie par :

$$U = \frac{f_N - p}{\sqrt{\frac{p(1-p)}{N}}} : N(0,1)$$

Nous avons

$$P(|f_N - p| < t) = 1 - \alpha$$

donc

$$P(|U| < u) = 1 - \alpha$$

avec

$$u = \frac{t}{\sqrt{\frac{p(1-p)}{N}}}$$

Ainsi l'intervalle de confiance de niveau $1 - \alpha$ s'écrit sous la forme suivante :

$$P[a < \theta < b] = P[f_N - u\sqrt{\frac{p(1-p)}{N}} < p < f_N + u\sqrt{\frac{p(1-p)}{N}}] = 1 - \alpha$$

La valeur de u sera lue sur une table de loi normale $N(0,1)$.

Pour $\alpha = 0.05$, nous avons $u = 1.96$. Par conséquent, l'intervalle de confiance de niveau 0.95 d'une proportion p est égale à $\pm 1.96\sqrt{\frac{p(1-p)}{N}}$ où N est la taille de l'échantillon. Pour tous les résultats obtenus dans notre travail, nous adoptons un niveau de confiance similaire de 0.95.

Chapitre 5

Modèle poubelle

5.1 Introduction

Les systèmes de reconnaissance par mots clés doivent être capables de rejeter les entrées incorrectes, les mots hors-vocabulaire et les diverses perturbations accidentelles (hésitations, bruit, etc.) afin de ne garder que l'ensemble des mots clés [Sukkar, 1994] [Rose, 1995].

Plusieurs recherches dans le domaine de la détection de mots clés dans un flux de parole se basent sur l'introduction de modèles représentant les mots hors-vocabulaire et appelés modèles poubelles [Ramalingam *et al.*, 1999] [Maison et Gopinath, 2001]. L'idée est d'utiliser dans les systèmes de reconnaissance, des modèles poubelles dont le rôle est d'absorber les mots hors-vocabulaire. La question qui se pose dans ce cas, est : comment construire de tels modèles poubelles ?

Ce chapitre est consacré à l'utilisation des modèles poubelles comme première solution envisagée pour absorber les mots hors-vocabulaire. Nous commençons par tester un modèle poubelle sans apprentissage. Ensuite, nous proposons une nouvelle méthode à base d'un modèle poubelle appris en utilisant un GMM. Enfin, nous abordons une approche hybride combinant les deux premières méthodes. Dans une deuxième étude, nous proposons une technique de détection de mots clés à base de boucle de phonèmes, c'est-à-dire que notre grammaire sera composée seulement d'un ensemble de phonèmes tout en favorisant le passage entre les phonèmes qui constituent un mot clé donné. En premier lieu, nous proposons une récompense qui a la même valeur pour tous les mots d'où le nom de "récompense constante", ensuite nous introduisons la fonction affine qui dépend du nombre de phonèmes dans chaque mot clé pour calculer la valeur de la récompense de passage. Enfin nous utilisons la fonction sigmoïde.

5.2 Modélisation des mots

Dans la première étape de notre travail et comme première approche pour détecter les mots clés, nous avons réalisé un système qui consiste à reconnaître uniquement les mots clés. Nous avons donc utilisé une grammaire qui ne modélise que ces mots clés et qui ne tient pas compte de tous les autres mots. Par conséquent, nous avons obtenu un grand nombre d'insertions car chaque suite de mots hors-vocabulaire est confondu avec la suite de mots clés le plus ressemblant. Ensuite, nous avons pensé qu'il serait intéressant de réaliser une reconnaissance à grand vocabulaire où il s'agit de modéliser explicitement tous les mots, c'est-à-dire les mots clés et les mots

hors-vocabulaire. Pour ce faire, nous avons utilisé un vocabulaire de 20.000 mots et nous avons adopté une modélisation contextuelle des phonèmes en utilisant les triphones. Dans la phase de reconnaissance, nous conservons seulement l'ensemble des mots clés de l'application considérée. Les résultats obtenus par cette méthode sont illustrés dans le tableau 5.1.

	Insertions	Omissions	Substitutions	Mots acceptés	FA/Mc/H	TD
Grand vocabulaire	80	1473	124	1625	3.1	50.5%

TAB. 5.1 – Grand vocabulaire.

Cette modélisation grand vocabulaire présente plusieurs inconvénients. D'une part, elle nécessite la modélisation de 20.000 mots et la construction de la grammaire correspondante. D'autre part, elle ne résoud pas le problème des mots hors-vocabulaire puisqu'il est toujours possible d'avoir de nouveaux mots hors-vocabulaire (hésitations, formules de politesse, etc.) pendant la phase d'exploitation du système. En plus, le temps de reconnaissance est important et nous obtenons un faible taux de détection de mots clés qui est de l'ordre de 50% pour un nombre de fausses acceptations par mot clé et par heure ($FA/Mc/H$) égal à 3.1.

Pour améliorer les performances de notre système et afin d'obtenir un compromis entre la modélisation explicite des mots clés et celle de tous les mots (mots clés et mots hors-vocabulaire), nous avons choisi de modéliser explicitement les mots clés et de représenter les mots hors-vocabulaire par un modèle "poubelle" qui permet de les absorber. Dans ce qui suit, nous proposons différentes approches basées sur la notion "poubelle".

5.2.1 État poubelle sans apprentissage

Pour détecter les mots clés et absorber les mots hors-vocabulaire, qui varient d'un locuteur à un autre selon l'âge, le sexe, etc., nous nous sommes inspirés des travaux de Boite [Boite *et al.*, 1993] et de Bourlard [Bourlard *et al.*, 1994] afin d'élaborer un état poubelle au fur et à mesure de la phase de reconnaissance.

Cet état poubelle est représenté par un HMM à un seul état et ne comportant pas de gaussiennes apprises. La probabilité d'observation de cet état à un instant t [5] lors de la phase de reconnaissance (en ligne) est appelée "score local" de l'état poubelle. Ce score est calculé comme étant la moyenne des N meilleurs scores locaux (les probabilités d'observations) des états des modèles phonétiques utilisés pour décrire les mots clés, comme c'est indiqué par l'équation suivante :

$$SE_p = \frac{\sum_{i=1}^{N} Sc_i}{N}$$

où SE_p est le score de l'état poubelle, Sc_i le score local d'indice i qui représente la probabilité d'observation de l'état i et N est le nombre des meilleurs scores utilisés. Les modèles phonétiques

[5] Cette probabilité est calculée pour chaque trame de la séquence d'observations de l'expression à reconnaître.

sont représentés par des triphones. Chaque triphone est modélisé par un HMM à 3 états (gauche-droite) à densités continues, avec 4 gaussiennes par état. Les scores Sc_i vérifient la condition suivante :

$$Sc_1 > Sc_2 > \ldots > Sc_N$$

Pour déterminer la valeur de N nous avons mené une série d'expériences sur notre base de développement. Les résultats obtenus sont présentés dans le tableau 5.2. Nous remarquons que le meilleur résultat est obtenu pour $N = 40$, nous avons donc retenu cette valeur pour la suite de nos expériences.

N	5	10	15	20	25	30	35	**40**	45	50
TE	29.9%	45.8 %	60.2%	70.1%	78.2%	81.4 %	84.4 %	**86.7**%	79.3 %	62.2 %

TAB. 5.2 – Taux de reconnaissance en fonction de la valeur de N.

Le score obtenu SE_p, représentant la probabilité d'observation de l'état poubelle en ligne, est faible. Ceci explique la grande tendance du système à insérer des mots clés. En effet, les résultats obtenus avec ce score ont donné un grand nombre d'insertions.

Pour remédier à cette augmentation du nombre d'insertions, il est nécessaire d'introduire une variable de pénalisation pour le mot en entrée (P_{me}). Cette pénalité améliorera le taux de rejet et permettra en la faisant varier, de trouver un compromis entre le nombre de fausses acceptations et le taux de détection.

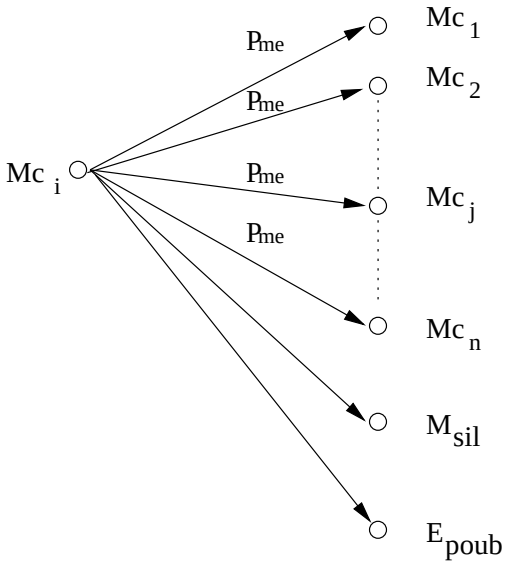

FIG. 5.1 – Pénalisation des passages du mot clé Mc_i vers les autres mots Mc_j pour $j \in \{1, \ldots, n\}$.

Afin d'appliquer cette technique de pénalisation, nous avons procédé de la manière suivante :

supposons que nous avons reconnu le mot clé Mc_i et que ce mot peut être suivi par l'état poubelle (E_{poub}), ou le modèle silence (M_{sil}) ou encore par un mot clé Mc_j, $j \in \{1, 2, \dots n\}$ comme indiqué sur la figure 5.1. Alors, pour affaiblir le passage vers les mots clés et favoriser le passage vers l'état poubelle, nous avons fait diminuer le logarithme de la probabilité du passage du mot Mc_i au mot Mc_j en utilisant une variable de pénalisation :

$$log[P(Mc_j|Mc_i)] - P_{me} \tag{5.1}$$

$$P(Mc_j|Mc_i) = \frac{1}{N_{succ(i)}}$$

où $Mc_j \in \{Mc_1 ..... Mc_n\}$ et $N_{succ(i)}$ est le nombre de modèles successeurs au modèle Mc_i : $N_{succ(i)} = n + 2$. P_{me} est une variable de pénalisation dont les valeurs ont été déterminées en utilisant la base de développement.

Pendant la phase de reconnaissance, nous disposons donc des modèles HMM des phonèmes, d'un modèle HMM correspondant au silence et d'un état HMM qui a pour rôle l'absorption de tous les mots hors-vocabulaire. Chaque mot clé est modélisé par la concaténation des modèles HMM des phonèmes qui le composent (cf. figure 5.2). Ainsi, nous disposons des modèles HMM des mots clés, du modèle de silence et de l'état poubelle. Par simplification, nous représentons dans la suite un mot clé par un modèle HMM à trois états (gauche-droite) seulement.

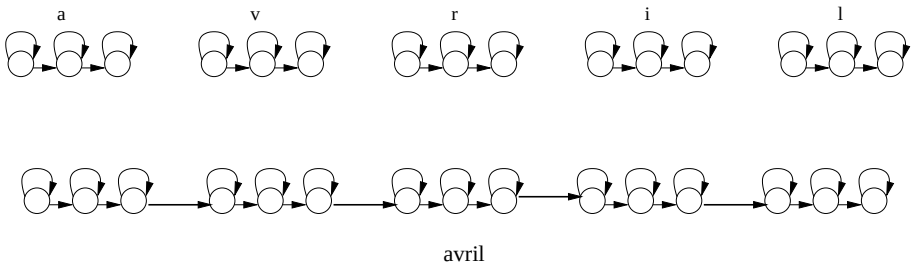

FIG. 5.2 – Modélisation du mot clé "avril" par la concaténation des phonèmes a, v, r, i et l.

En plus des modèles de mots clés, du modèle silence et de l'état poubelle, nous disposons aussi d'une grammaire présentée par la figure 5.3. La figure 5.4 montre l'exemple de reconnaissance de la phrase : "ok je réserve pour **jeudi** après midi voilà" en utilisant l'état poubelle. Le résultat obtenu est le suivant : E_{poub} **jeudi** E_{poub}. Durant cette phase de reconnaissance, nous utilisons l'algorithme de Viterbi qui nous fournit la meilleure séquence d'états, et donc de modèles HMM, représentant l'expression à reconnaître.

Nous avons étudié les performances de notre approche en faisant varier la valeur de la pénalisation afin de voir l'évolution du nombre d'insertions et d'acceptations des mots clés. Nos expériences ont été menées sur la base de test qui contient 2245 phrases. La courbe ROC (TD en fonction du nombre de FA/MC/H) et la courbe "rappel/précision" données respectivement par les figures 5.5 et 5.6 illustrent les résultats obtenus par la méthode état poubelle sans apprentissage avec une modélisation contextuelle des phonèmes. Cette méthode a donné un FOM de l'ordre de 60.1% et une efficacité de 64.5% avec un intervalle de confiance égal à 1.6%.

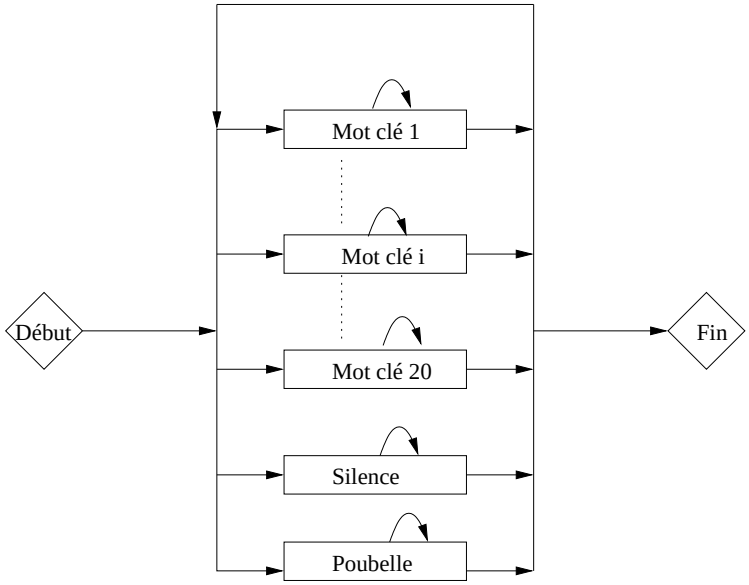

FIG. 5.3 – Grammaire de l'ensemble des mots clés avec un seul modèle poubelle et un modèle silence.

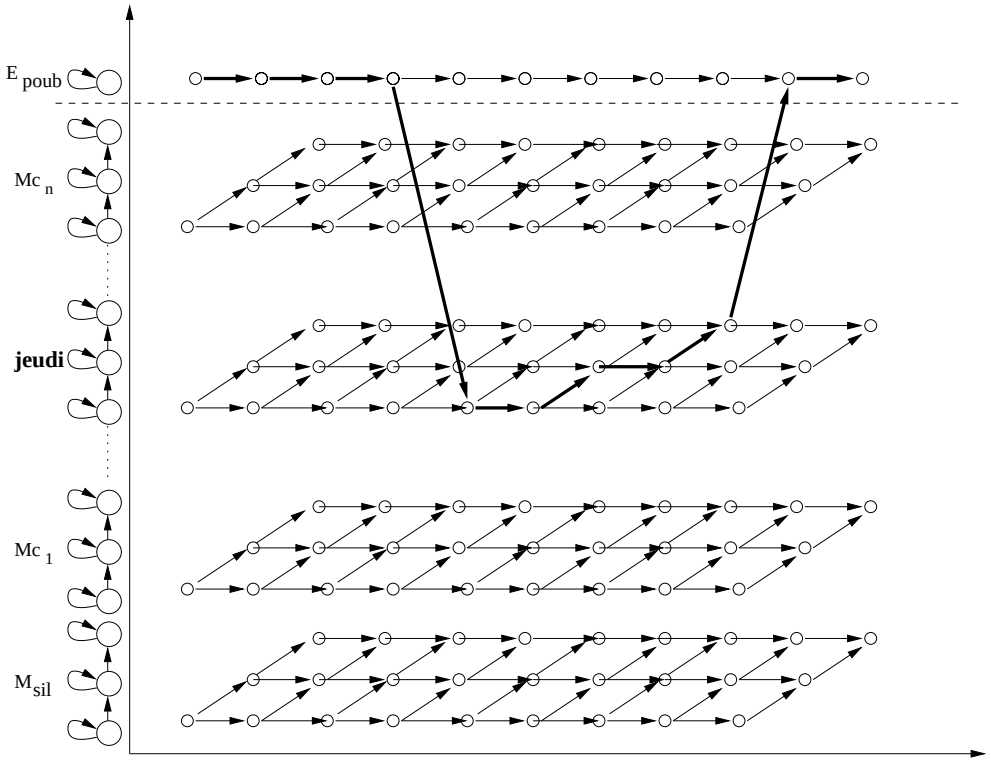

FIG. 5.4 – Reconnaissance de la phrase "ok je réserve pour **jeudi** après midi voilà" en utilisant un état poubelle sans apprentissage.

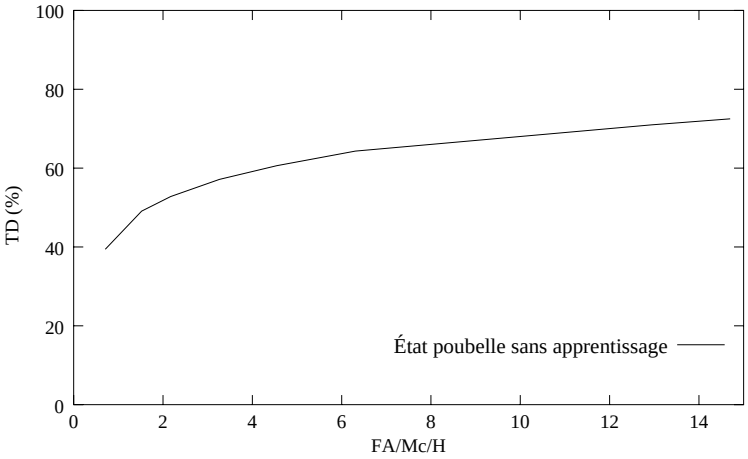

FIG. 5.5 – Courbe ROC de l'état poubelle sans apprentissage.

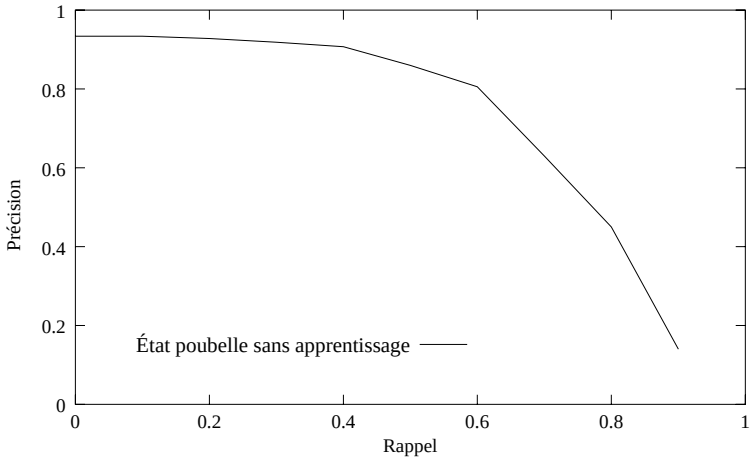

FIG. 5.6 – Courbe rappel/précision de l'état poubelle sans apprentissage.

5.2.2 Modèle poubelle avec apprentissage

Plusieurs travaux ont été réalisés en utilisant un modèle poubelle avec apprentissage [Rose et Hofstetter, 1993] [James et Young, 1994], représenté par un HMM à plusieurs états. Yapanel [Yapanel, 1997] a montré qu'un modèle poubelle représenté par un HMM à 6 états est meilleur que celui représenté par un HMM à 9 états et qu'un modèle poubelle modélisé par un HMM à 3 états est meilleur qu'un modèle à 6 états c'est-à-dire que si le modèle HMM est représenté par moins d'états alors il donne de meilleurs résultats.

Les modèles de mélange de gaussiennes *(Gaussian Mixture Models : GMM)* sont largement utilisés. Ils peuvent être considérés comme des modèles HMM à un seul état où la fonction de densité est composée d'un mélange de gaussiennes. Une approche basée sur les GMM consiste à produire un modèle sous forme d'une somme pondérée de M gaussiennes. Chaque gaussienne g_i est caractérisée par un poids p_i, un vecteur moyen μ_i de dimension n et une matrice de covariance Σ_i de dimension $(n \times n)$. L'apprentissage des paramètres du modèle GMM (p_i, u_i, Σ_i) $i \in \{1, \ldots, M\}$, est réalisé généralement à l'aide de l'algorithme EM (Expectation-Maximisation) qui représente une technique générale pour l'estimation par maximum de vraisemblance [Li *et al.*, 1995]. Les modèles à base de mélanges de gaussiennes ont fait leurs preuves dans plusieurs domaines notamment, en identification automatique du locuteur [Reynolds, 1994] [Schmidt *et al.*, 1997] [Kharroubi, 2002] où ils fournissent les meilleurs résultats actuels.

Soient λ un modèle de mélange de gaussiennes et $O = (o_1, o_2, \ldots, o_T)$ une séquence d'observations relative à un segment de parole. Pendant la phase de reconnaissance du segment de parole, la vraisemblance qu'un vecteur o_t soit produit par le modèle λ est donnée par le mélange de gaussiennes suivant :

$$f(o_t|\lambda) = \sum_{i=1}^{M} p_i f_i(o_t)$$

où f_i est une gaussienne donnée par la formule suivante :

$$f_i(o_t) = \frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma_i|^{\frac{1}{2}}} exp[-\frac{1}{2}(o_t - \mu_i)^T (\Sigma)^{-1} (o_t - \mu_i)]$$

Ainsi, la vraisemblance pour que la totalité du segment de parole O soit produite par le modèle λ s'écrit sous la forme suivante :

$$f(O|\lambda) = \prod_{t=1}^{T} f(o_t|\lambda)$$

Du fait des bonnes performances obtenues avec les GMM, nous avons décidé de modéliser notre modèle poubelle par un mélange de gaussiennes. Pour fixer le nombre de gaussiennes nécessaires, nous avons mené sur notre base de développement une série d'expériences avec différents modèles poubelles, où chacun est modélisé par un mélange de M gaussiennes avec $M \in \{8, 16, 32, 64, 128, 256, 512\}$.

Les résultats obtenus montrent qu'en augmentant le nombre de gaussiennes les performances s'améliorent. Cependant, en passant de 256 à 512, les résultats restent comparables, mais le temps de calcul est beaucoup plus important. Pour ces raisons, nous avons fixé M à 256 et nous avons donc utilisé un mélange de 256 gaussiennes pour la modélisation de notre modèle poubelle.

Dans la phase de reconnaissance nous disposons des modèles des triphones, du modèle de silence, (où chacun est modélisé par un HMM à 3 états (gauche-droite) à densités continues et avec 4 gaussiennes par état) et du modèle poubelle modélisé par un mélange de 256 gaussiennes.

La probabilité d'observation du modèle poubelle est faible et ceci est une conséquence naturelle de la manière avec laquelle nous l'avons réalisé. En effet ce modèle ainsi défini, est très général puisqu'il ne représente pas un phonème ou un mot précis. Par conséquent, le système de reconnaissance a une grande tendance à insérer les mots clés, et ceci va faire augmenter le nombre d'insertions.

Pour remédier à cette augmentation du nombre d'insertions, nous introduisons, de la même façon que dans l'approche précédente, (état poubelle sans apprentissage), une variable de pénalisation du mot en entrée (P_{me}) comme dans l'équation 5.1. Les différentes valeurs de pénalisation ont été choisies en se basant sur une série d'expériences menées sur notre base de développement. La variation de cette pénalisation permet de trouver un compromis entre le nombre de fausses acceptations et le taux de détection.

Pendant la phase d'apprentissage, sont appris les modèles des phonèmes, le modèle silence et le modèle poubelle GMM. En phase de reconnaissance, nous disposons donc de tous les modèles de mots clés, du modèle silence, du modèle GMM et de la grammaire (figure 5.3). De la même façon illustrée par la figure 5.4, et en remplaçant l'état poubelle $''E''_{poub}$ par le modèle poubelle $''M''_{poub}$, le résultat obtenu par notre système pour la reconnaissance de la phrase "ok je réserve pour **jeudi** après midi voilà " est le suivant : M_{poub} **jeudi** M_{poub}. Durant cette phase de reconnaissance, nous utilisons l'algorithme de Viterbi afin d'avoir la meilleure séquence d'états et donc de modèles HMM représentant l'expression à reconnaître.

Nous avons étudié les performances de notre approche en faisant varier la valeur de la pénalisation du mot en entrée pour trouver un compromis entre le nombre de fausses acceptations et le taux de détection ainsi qu'entre le taux de rappel et le taux de précision. Les résultats obtenus, en utilisant la base de test, sont présentés sur les figures 5.7 et 5.8.

Cette approche a donné un FOM de l'ordre de 66.8% et une efficacité de 70.4% avec un intervalle de confiance de 1.5%. Nous remarquons donc une importante amélioration des performances par rapport à celles obtenues par la méthode précédente (état poubelle sans apprentissage).

5.2.3 Modèle combiné

Les deux méthodes précédentes utilisent deux techniques différentes pour représenter les mots hors-vocabulaire. L'une utilise le corpus de test (méthode adaptative), l'autre le corpus d'apprentissage. Afin de profiter des avantages de ces deux méthodes, nous avons proposé de faire leur combinaison, d'où le nom de notre nouvelle approche "modèle combiné". Dans cette méthode, nous utilisons les deux représentations des mots hors-vocabulaire définies précédemment. La première

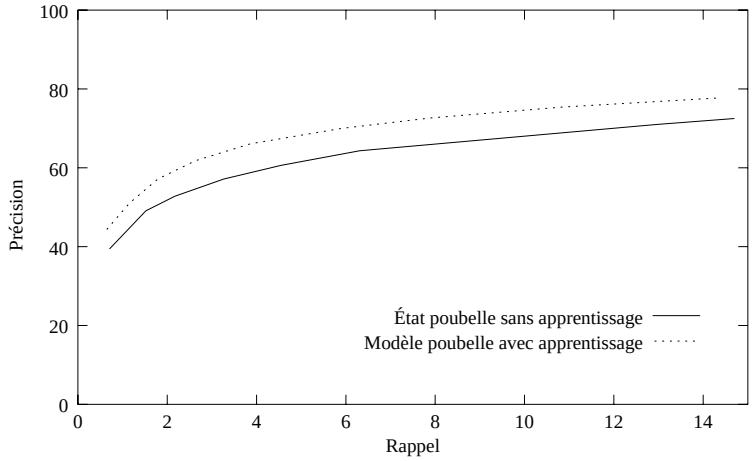

FIG. 5.7 – Courbe ROC du modèle poubelle avec apprentissage.

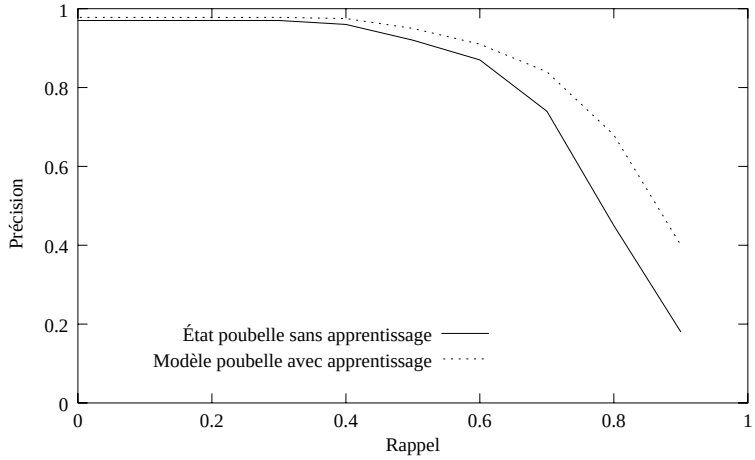

FIG. 5.8 – Courbe rappel/précision du modèle poubelle avec apprentissage.

utilise un modèle poubelle avec apprentissage représenté par un GMM à 256 gaussiennes déterminées au cours de la phase d'apprentissage et la deuxième un état poubelle sans apprentissage représenté par un HMM à un seul état dont le score local est calculé au cours de la phrase de test.

Nous apprenons d'abord le premier modèle poubelle en utilisant toujours la même base d'apprentissage. À l'issue de cette étape, nous disposons des modèles HMM des mots clés, d'un modèle HMM correspondant au silence, d'un modèle poubelle GMM et de la grammaire représentée figure 5.9. Ensuite, pendant la phase de test, nous utilisons la notion d'état poubelle sans apprentissage en calculant pour chaque trame émise le score (SE_p) du modèle. Ce score est toujours la moyenne des N meilleurs scores locaux des différents états des phonèmes des mots clés, sans tenir compte du score du modèle poubelle déjà appris. Enfin, nous utilisons l'algorithme de Viterbi pour trouver la meilleure séquence d'états et donc des modèles HMM représentant la séquence d'observations relative à l'expression prononcée.

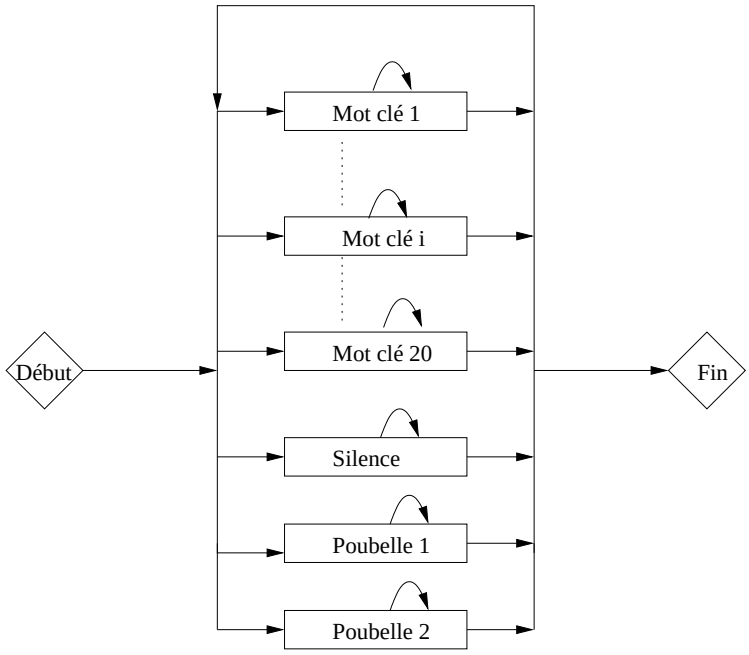

FIG. 5.9 – Grammaire de l'ensemble des mots clés avec deux modèles poubelles et un modèle silence

La décision prise par le système de détection est basée, non seulement sur le score de l'état poubelle sans apprentissage, mais aussi sur celui du modèle poubelle appris. Cela donne au système plus de chance de tomber sur un modèle poubelle que de détecter un mot clé. Ceci nous a permis de faire baisser les taux d'insertion des mots clés obtenus avec les deux méthodes précédentes. La figure 5.10 illustre la reconnaissance de la phrase : "ok je réserve pour **jeudi** après midi voilà", en utilisant un modèle combiné. Le résultat obtenu est le suivant : M_{poub} E_{poub} **jeudi** E_{poub} M_{poub}.

L'aide fournie par le modèle poubelle appris consiste à reconnaître les mots hors-vocabulaire déjà rencontrés lors de la phase d'apprentissage et donc de fournir au système une réponse plus précise. C'est aussi un moyen pour faire un deuxième passage sur les décisions prises par l'état

poubelle en ligne afin de fournir des résultats plus exacts. Ceci a été confirmé par les expériences que nous avons menées sur la même base de test et qui nous ont montré l'efficacité de cette nouvelle approche.

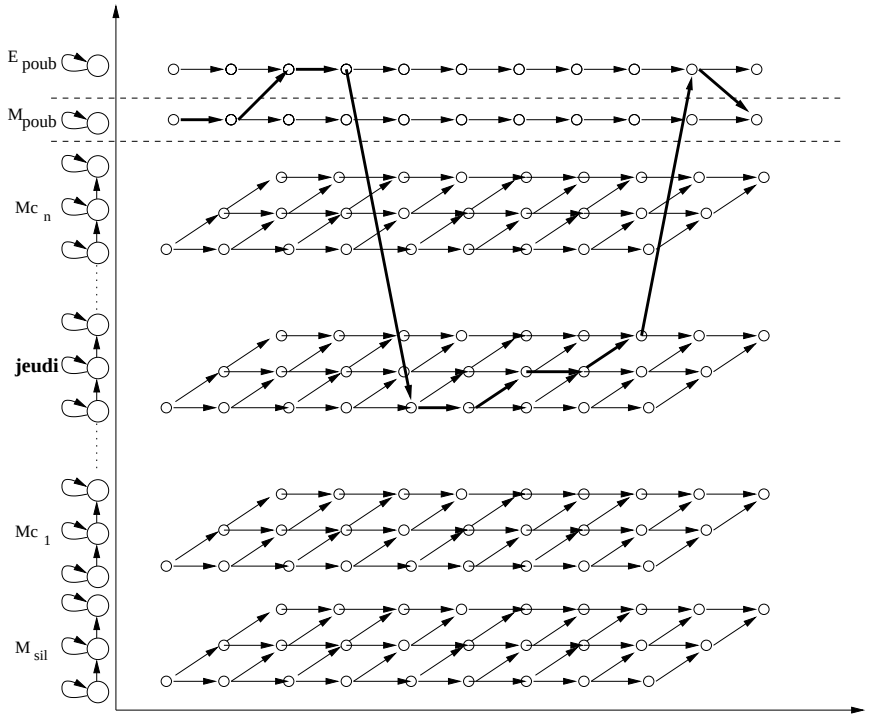

FIG. 5.10 – Reconnaissance de la phrase "ok je réserve pour jeudi après midi voilà" en utilisant un modèle poubelle combiné.

Les figures 5.11 et 5.12 donnent respectivement la courbe ROC et la courbe rappel/précision obtenues par la méthode combinée en se basant sur une modélisation contextuelle des phonèmes. En fixant N (N est le nombre des meilleurs scores à utiliser pour calculer SE_p) à la valeur 40 et en utilisant la même base de test, nous obtenons un FOM égal à 70.6% et une efficacité de 75% avec un intervalle de confiance de 1.4%. Les résultats obtenus avec cette nouvelle méthode soulignent l'amélioration réalisée par rapport aux deux méthodes précédentes.

5.3 Modèle à base de boucles de phonèmes

5.3.1 Introduction

Dans un système utilisant une grammaire avec des mots clés et des modèles poubelles, nous remarquons une grande tendance à insérer des mots clés dès que leurs premiers phonèmes sont reconnus, même si les autres phonèmes constituant le mot clé ont de très faibles valeurs de vraisemblance. Nous pouvons expliquer ce phénomène par le fait que le système n'a pas beaucoup de choix : il reconnaît un mot clé et non pas un modèle poubelle puisque il a déjà trouvé des phonèmes de ce mot, et il s'agit du mot le plus probable. Ceci engendre un grand nombre d'insertions et donc une dégradation des résultats. Afin de remédier à ce problème, nous avons proposé de faire la reconnaissance en se basant sur une boucle de phonèmes. Le système fournit

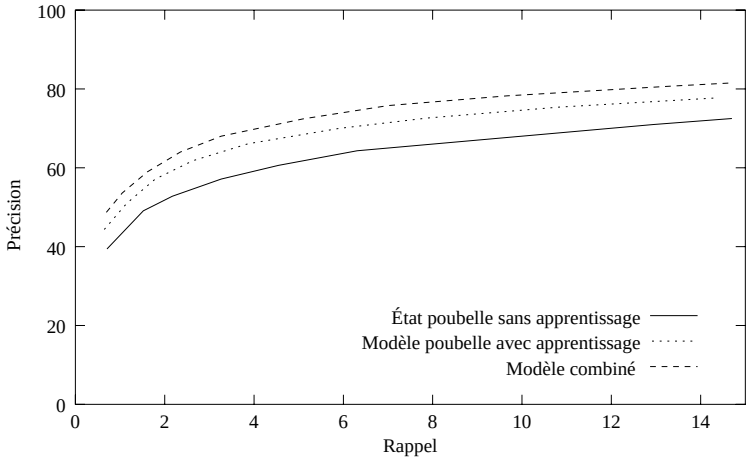

FIG. 5.11 – Courbe ROC du modèle poubelle combiné.

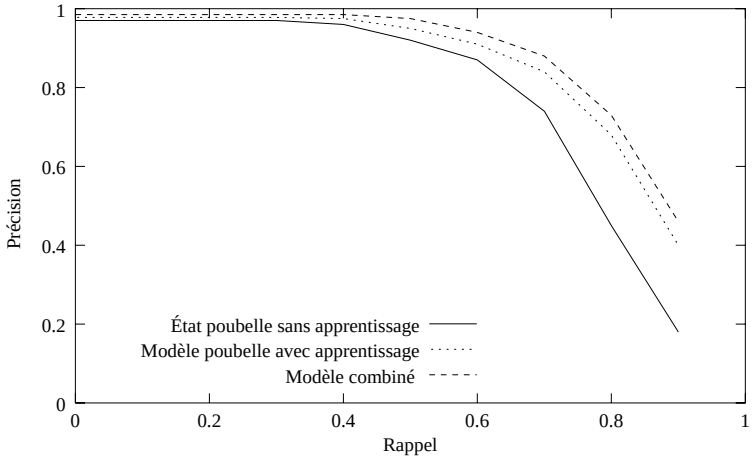

FIG. 5.12 – Courbe rappel/précision du modèle poubelle combiné.

dans ce cas la suite des phonèmes les plus probables pour une séquence d'observations donnée. Il ne reste alors qu'à introduire au niveau de la grammaire utilisée les transcriptions phonétiques des différents mots clés. Enfin, nous disposons d'une grammaire composée de 20 mots clés avec leurs transcriptions phonétiques, de 31 phonèmes indépendants du contexte et du modèle silence comme illustré par la figure 5.13.

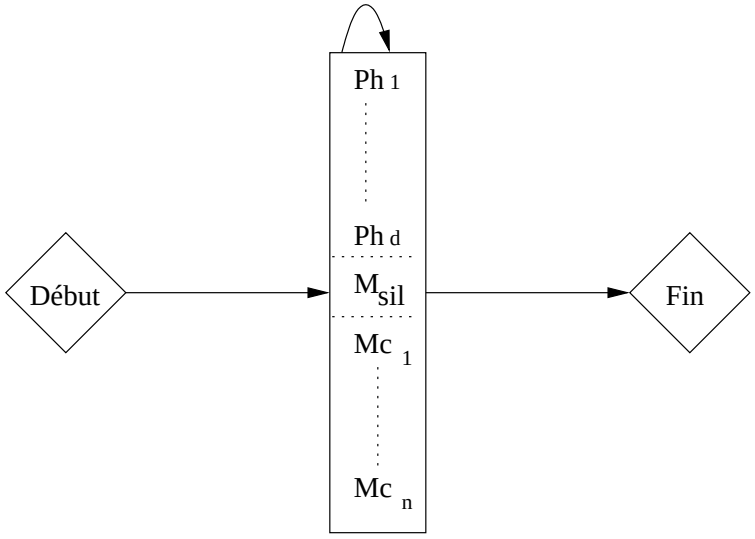

FIG. 5.13 – grammaire à base de boucle de phonèmes.

Ayant la suite des phonèmes reconnus, le système effectue la mise en correspondance entre les transcriptions phonétiques des mots clés fournis par la grammaire et les groupements de phonèmes reconnus. À une mise en correspondance réussie, le système détecte alors le mot clé concerné. Les autres phonèmes (ne correspondant à aucun mot clé) sont laissés à l'état. Ainsi, pour une séquence d'observations correspondant à un message donné, nous obtenons une séquence constituée de phonèmes et de mots clés, comme illustré par la figure 5.14.

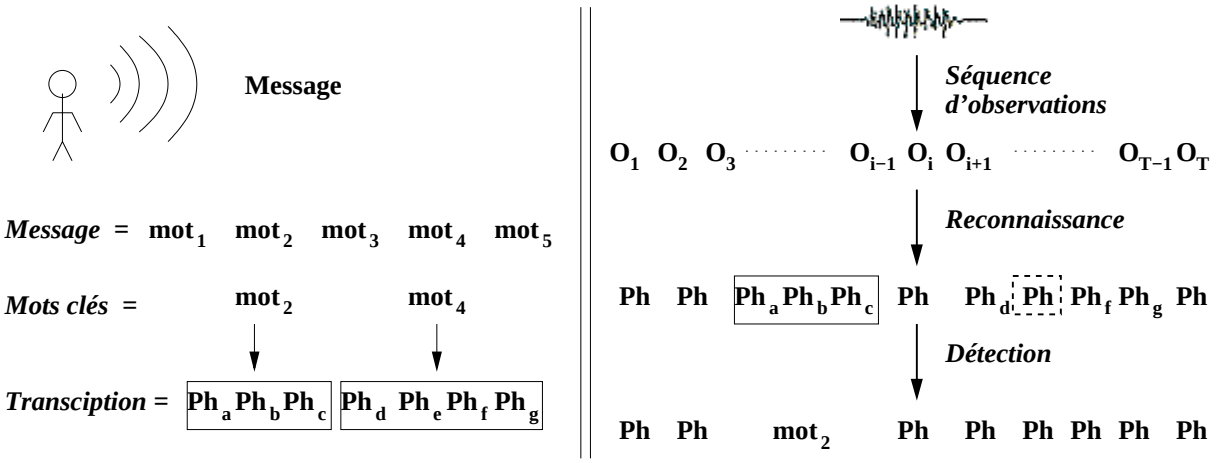

FIG. 5.14 – Reconnaissance à base de boucle de phonèmes

Avec cette méthode, le système de reconnaissance a tendance de reconnaître plus de phonèmes que de mots clés. En effet, il suffit qu'un seul phonème dans un mot clé soit mal reconnu pour que le mot clé sera déclaré comme une séquence de phonèmes. Ceci est illustré par la figure 5.14 où le phonème Ph_e a été mal reconnu dans le deuxième mot clé dont la transcription phonétique est $[Ph_dPh_ePh_fPh_g]$. À cause de cette erreur, ce mot clé a été remplacé par une séquence de phonèmes. Avec ce genre d'erreurs, nous avons obtenu en résultat beaucoup plus de phonèmes que de mots clés (un grand nombre d'omissions de mot clés). Pour cette raison, nous avons pensé à favoriser la reconnaissance des mots clés en augmentant leurs probabilités ou d'une autre façon, en pénalisant les passages aux phonèmes isolés.

Dans ce travail, nous avons choisi trois méthodes à base de boucles de phonèmes qui favorisent la reconnaissance des mots clés et pénalisent le passage aux phonèmes isolés. Dans la suite, nous décrivons ces méthodes et nous comparons leurs performances.

5.3.2 Méthode à récompense constante

Afin de favoriser la reconnaissance des mots clés, nous avons proposé comme première approche, l'ajout d'une récompense pour aider le système à détecter les mots clés. Cette récompense a la même valeur pour tous les mots clés (méthode à récompense constante), indépendamment de la longueur du mot clé, c'est-à-dire du nombre de phonèmes qui le constituent.

Supposons que nous ayons reconnu le phonème Ph_i qui peut être suivi par un mot clé M_c d'indice k avec $k \in \{1, 2, \ldots n\}$, par un phonème d'indice j tel que $j \in \{1, 2, \ldots d\}$ ou par le modèle silence M_{sil} comme c'est indiqué par la figure 5.15. Alors, le logarithme de la probabilité de passage du phonème Ph_i au mot clé d'indice k (Mc_k) est incrémenté d'une constante de récompense (positive), comme c'est illustré par l'expression suivante :

$$log[P(Mc_k|Ph_i)] + C$$

$$P(Mc_k|Ph_i) = \frac{1}{N_{succ(i)}}$$

où $Mc_k \in \{Mc_1.....Mc_n\}$ un mot clé donné, $Ph_i \in \{Ph_1.....Ph_d\}$ un phonème reconnu et $N_{succ(i)}$ est le nombre de modèles successeurs au phonème Ph_i, il correspond à la somme du nombre de mots clés n, le nombre de phonèmes d plus le modèle silence, nous avons donc : $N_{succ(i)} = n + d + 1$, toutes les transitions entre les modèles sont initialement équiprobables. De cette manière, le système peut passer plus facilement à un mot clé au lieu de passer à un phonème isolé. Ainsi, nous pouvons remédier au problème des phonèmes reconnus avec une faible vraisemblance dans un mot clé donné.

Afin de trouver un compromis entre le taux de détection et le nombre de fausses acceptations, nous avons réalisé une série d'expériences sur la base de développement permettant de fixer les différentes valeurs de récompense. En utilisant notre base de test qui contient 2245 phrases et en faisant varier la valeur de la récompense (pour tous les mots), nous avons tracé la courbe ROC et la courbe rappel/précision correspondantes comme illustré par les figures 5.16 et 5.17. Nous obtenons un FOM égal à 58.6% et une efficacité de l'ordre de 62% avec un intervalle de confiance de 1.6%.

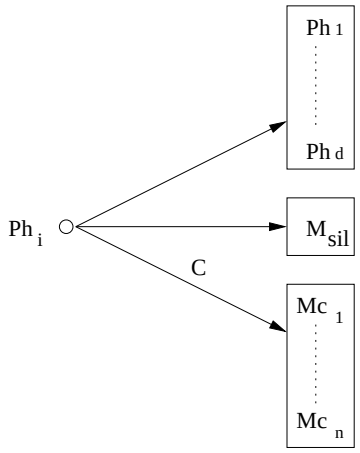

FIG. 5.15 – Favorisation du passage aux mots clés.

Bien que cette méthode nous ait permis d'améliorer les résultats en diminuant le nombre d'omissions des mots clés, elle présente encore quelques inconvénients, notamment le problème de la non prise en compte de la longueur du mot à reconnaître. En effet, dans un mot clé long, la probabilité d'avoir un nombre de phonèmes mal reconnus est supérieure à la probabilité d'en avoir dans un mot court. Le système aura donc tendance à découper le mot long afin de reconnaître des mots plus courts, le reste des phonèmes étant considérés comme des phonèmes isolés. Cette méthode consiste à attribuer la même récompense pour tous les mots sans tenir compte du nombre de phonèmes qui les constituent et c'est pourquoi le système nous a donné un grand nombre d'insertions de mots clés courts.

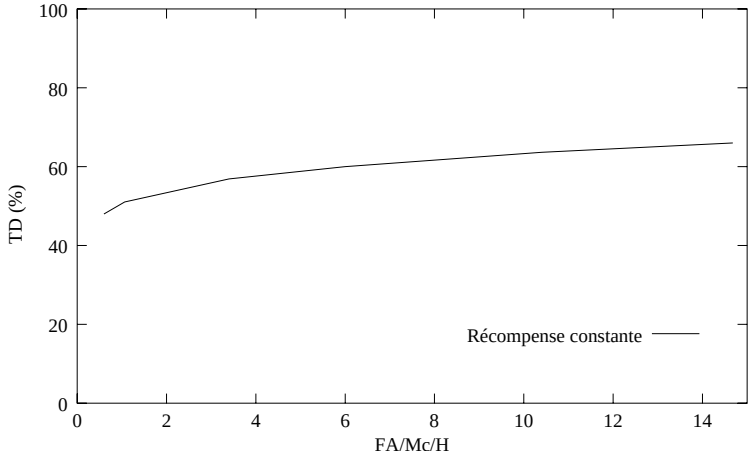

FIG. 5.16 – Courbe ROC de la méthode à récompense constante.

5.3.3 Méthode à récompense affine

Pour remédier à l'inconvénient de la méthode précédente, nous avons conçu une autre formule de récompense qui, au contraire de la récompense constante, différencie les récompenses attribuées aux mots longs et celles attribuées aux mots courts. Afin d'atteindre ce but, cette nouvelle méthode utilise une fonction affine pour calculer la valeur de la récompense à accorder à chaque

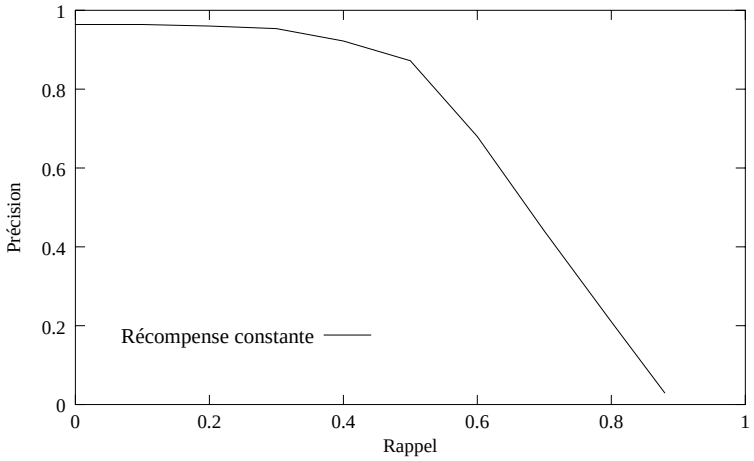

FIG. 5.17 – Courbe rappel/précision de la méthode à récompense constante.

mot clé. Cette fonction f tient compte du nombre de phonèmes dans le mot en question :

$$f(x_m) = a * x_m + b$$

où x_m est le nombre de phonèmes composants le mot m, a et b sont des coefficients choisis par l'utilisateur.

Ainsi, à chaque mot clé correspond une valeur de la récompense qui dépend du nombre des phonèmes qui le constituent.

De la même manière que dans l'approche précédente, le logarithme de la probabilité de passage du phonème Ph_i au mot clé d'indice k est incrémenté d'une récompense variable, comme illustré par l'expression suivante :

$$log[P(Mc_k|Ph_i)] + a \ \times \ x_m + b$$

$$P(Mc_k|Ph_i) = \frac{1}{N_{succ(i)}}$$

où $Mc_k \in \{Mc_1.....Mc_n\}$, $Ph_i \in \{Ph_1.....Ph_d\}$ et $N_{succ(i)}$ est le nombre de modèles successeurs au phonème Ph_i : $N_{succ(i)} = n + d + 1$. Toutes les transitions entre les modèles sont initialement équiprobables.

Le principal intérêt de cette méthode c'est qu'elle distribue ses récompenses proportionnellement à la longueur des mots clés considérés. De ce fait, le problème de favorisation des mots clés courts ne se pose plus car les récompenses sont attribuées d'une manière plus égale.

Les résultats obtenus par cette méthode sont données sur les figures 5.18 et 5.19. Les courbes sont tracées en faisant varier le couple $\{a, b\}$, pour calculer à chaque fois respectivement le taux de détection en fonction du nombre de fausses acceptations par mot clé et par heure et la précision en fonction du rappel. Les différentes valeurs du couple $\{a, b\}$ ont été fixées en utilisant notre base de développement.

Avec cette nouvelle méthode nous passons d'un FOM de 58.6% et d'une efficacité de 62% (avec la méthode précédente) à un FOM de 61.7% et à une efficacité de 65.4% avec un intervalle de confiance de 1.6%. Ces améliorations ne sont pas négligeables et elles prouvent l'intérêt de cette méthode adaptative au nombre de phonèmes dans un mot clé.

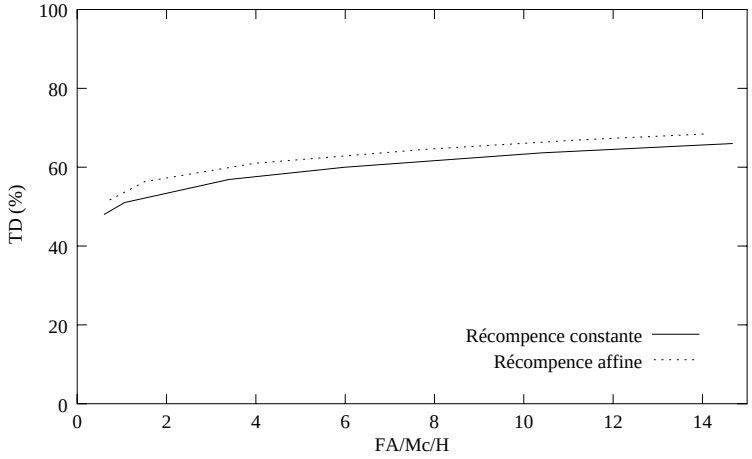

FIG. 5.18 – Courbe ROC de la méthode à récompense affine.

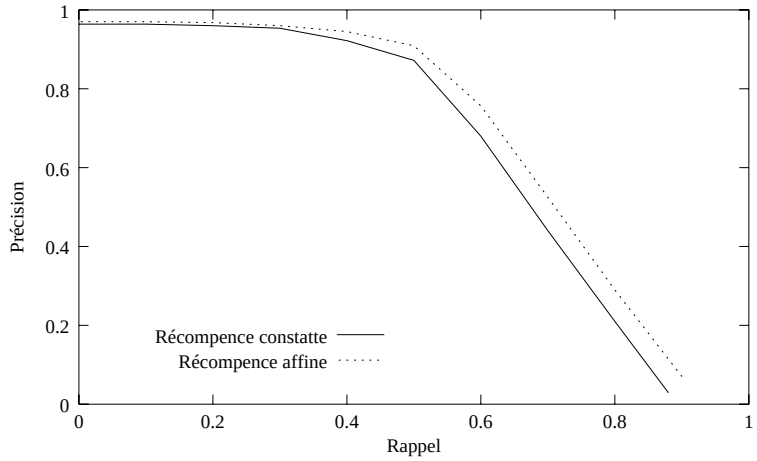

FIG. 5.19 – Courbe rappel/précision de la méthode à récompense affine.

5.3.4 Méthode à récompense sigmoïdale

Dans la méthode à récompense affine, nous avons remarqué que le principe de s'adapter au nombre de phonèmes est avantageux. Nous avons donc décidé de conserver ce même principe en essayant de trouver une fonction plus intéressante que la fonction affine simple. En plus dans la méthode précédente, nous avons remarqué qu'à une certaine longueur des mots clés la récompense attribuée devient importante ce qui augmente le nombre de fausses acceptations. Il nous faut donc chercher une fonction qui garde les avantages de la fonction affine en atténuant les effets des cas extrêmes. La solution que nous proposons consiste à utiliser une fonction de seuillage.

La fonction sigmoïde est la fonction de seuillage bien connue pour ses propriétés intéressantes. En effet, elle constitue la forme analytique la plus courante pour la décision. Cette fonction s'écrit dans notre cas sous la forme suivante :

$$f(x_m) = \frac{E}{1 + exp(-a \times x_m + b)}$$

où x_m est le nombre de phonèmes composants le mot m, E, a et b sont des constantes positives choisies par l'utilisateur.

De même que précédemment, le logarithme de la probabilité de passage du phonème Ph_i au mot clé d'indice k est incrémenté d'une récompense en fonction du nombre de phonèmes, comme l'indique l'expression suivante :

$$log[P(Mc_k|Ph_i)] + \frac{E}{1 + exp(-a \times x_m + b)}$$

$$P(Mc_k|Ph_i) = \frac{1}{N_{succ(i)}}$$

où $Mc_k \in \{Mc_1.....Mc_n\}$, $Ph_i \in \{Ph_1.....Ph_d\}$ et $N_{succ(i)}$ est le nombre de modèles successeurs au phonème Ph_i : $N_{succ(i)} = n + d + 1$. Toutes les transitions entre les modèles sont initialement équiprobables.

Ainsi, nous faisons correspondre à chaque mot clé, une récompense qui se calcule en fonction du nombre de phonèmes qui le constituent en utilisant la fonction sigmoïde. Ceci nous permettra de favoriser équitablement le passage aux mots clés au lieu des phonèmes isolés. Les différents paramètres de la fonction sigmoïde ont été choisis en se basant sur les résultats obtenus par une série d'expériences menées sur la base de développement.

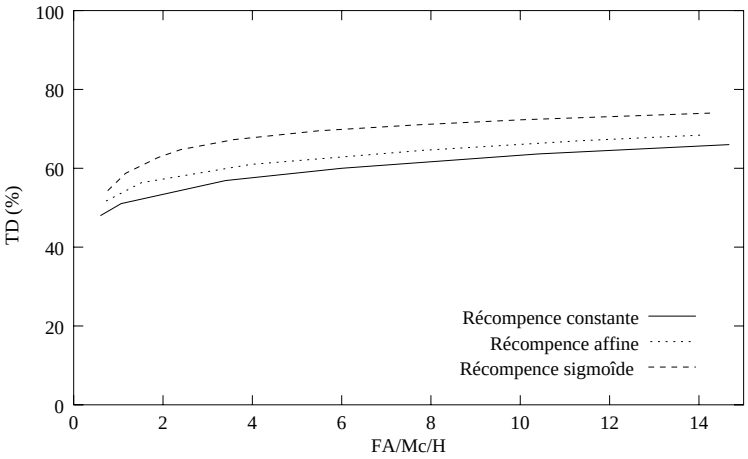

FIG. 5.20 – Courbe ROC de la méthode à récompense sigmoïdale.

Une série d'expériences a été menée sur la base de test tout en faisant varier les paramètres a et b de la fonction sigmoïde. Les résultats obtenus nous ont permis de tracer la courbe ROC et la courbe rappel/précision présentées respectivement par les figures 5.20 et 5.21. Ces résultats montrent bien l'amélioration engendrée par cette nouvelle méthode. En effet, nous obtenons un

FOM de l'ordre de 67.9% et une efficacité égale à 71.3% avec un intervalle de confiance de 1.5%. Soit une amélioration de l'ordre de 6% environ par rapport à la méthode à récompense affine.

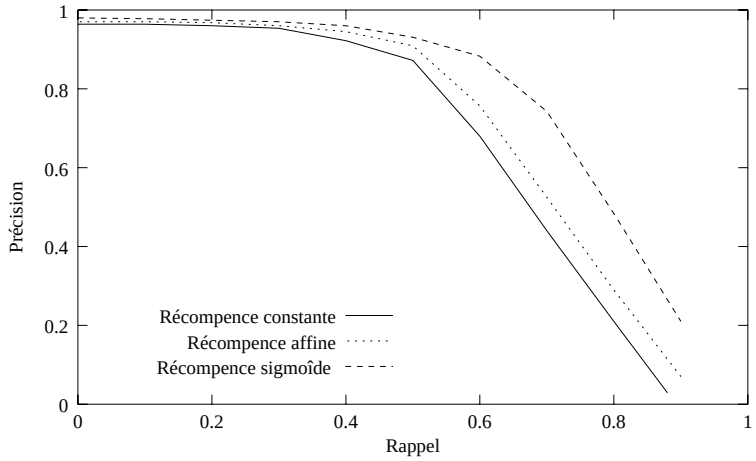

FIG. 5.21 – Courbe rappel/précision de la méthode à récompense sigmoïdale.

Le gain réalisé par cette méthode peut être expliqué principalement par le fait que la fonction sigmoïde limite à un certain seuil les récompenses qu'elle attribue aux mots très longs. En effet, ces mots sont généralement peu probables, mais avec la méthode affine ils ont acquis des récompenses très importantes favorisant leurs insertions inutiles.

5.4 Conclusion

Nous venons de présenter dans ce chapitre deux manières différentes pour détecter les mots clés. Pour la première, la modélisation des mots hors-vocabulaire est basée sur un modèle poubelle et dans la deuxième, la reconnaissance est faite en se basant sur des phonèmes bouclés. Pour la modélisation des mots hors-vocabulaire, nous avons proposé deux nouvelles approches : un modèle poubelle avec apprentissage à base d'un GMM et un modèle combiné utilisant les modèles poubelles avec et sans apprentissage. Dans la seconde étude, nous avons proposé trois approches basées sur l'attribution d'une récompense pour favoriser la reconnaissance des mots clés considérés. Afin de calculer la valeur de cette récompense, la première approche utilise une fonction constante, la seconde applique la fonction affine et la troisième introduit la fonction sigmoïde. Pour évaluer nos différentes méthodes, un jeu de test de 2245 phrases extraites de la base française SPEECHDAT 5000 a été utilisé. Cette évaluation nous a permis de mener une étude comparative entre les différentes approches.

L'analyse des résultats des différentes propositions montre que chacune des approches présentées dans ce document a contribué, à sa façon, à l'amélioration des performances. L'utilisation d'un modèle GMM a augmenté la valeur de FOM d'environ de 6.7% et l'efficacité de 5.9%. Nous avons aussi constaté que l'approche combinée prend mieux en compte le rejet des mots hors-vocabulaire en utilisant deux modèles poubelles, l'un appris sur le corpus d'apprentissage et l'autre s'adapte automatiquement (en ligne) aux données de test. Elle a fait progresser la valeur de FOM de 3.8% et l'efficacité de 4.6%.

Type du modèle	**FOM**	**Efficacité**
État poubelle sans apprentissage	60.1%	64.5%
Modèle poubelle avec apprentissage	66.8%	70.4%
Modèle combiné	70.6%	75.0%
Modèle à récompense constante	58.6%	62.0%
Modèle à récompense affine	61.7%	65.4%
Modèle à récompense sigmoïde	67.9%	71.3%

TAB. 5.3 – Tableau récapitulatif des résultats obtenus en utilisant les différents modèles poubelles.

L'utilisation d'une grammaire à base de boucle de phonèmes qui introduit la notion de récompense nous a permis de réaliser une détection de mots clés sans modélisation des mots hors-vocabulaire par des modèles poubelles. En premier lieu, nous avons choisi la même valeur de récompense pour tous les mots clés sans prise en compte de la longueur du mot à reconnaître. Cette méthode nous a donné un FOM de l'ordre de 58.6% et une efficacité de 62%. Pour améliorer ces résultats, nous avons proposé l'utilisation d'une récompense qui varie en fonction du nombre de phonèmes en suivant une loi affine. L'objectif de l'utilisation d'une récompense affine est de mieux tenir compte de la longueur de chaque mot clé à reconnaître. Cette méthode a permis d'améliorer la valeur de FOM de 3.1% et l'efficacité de 3.4%. Pour augmenter encore les performances, nous avons pensé à utiliser une autre fonction de récompense plus intéressante. Nous avons donc essayé d'appliquer la fonction sigmoïde pour favoriser la reconnaissance des mots clés. Cette méthode nous a permis d'obtenir un FOM de l'ordre de 67.9% et une efficacité égale à 71.3% c'est-à-dire une amélioration de 6% environ par rapport à la méthode à base de récompense affine. Nous avons aussi obtenu une amélioration par rapport au deuxième meilleur résultat atteint par l'utilisation du modèle poubelle avec apprentissage. En effet, il est à noter que les valeurs de FOM et d'efficacité du modèle à récompense sigmoïdale sont supérieures à celles du modèle poubelle avec apprentissage de 1% environ.

Enfin, en comparant le modèle combiné et le modèle à récompense sigmoïdale, nous pouvons remarquer que les résultats obtenus par le premier modèle sont plus intéressants. Cependant, il ne faut pas oublier que la dernière méthode présente l'avantage d'être utilisable dans n'importe quelle application puisqu'elle n'emploie pas un modèle poubelle spécifique à une application donnée, ce qui n'est pas le cas du modèle combiné.

Le modèle poubelle avec apprentissage, le modèle combiné et le modèle à récompense sigmoïde ont montré une amélioration des performances par rapport aux travaux de Rose [Rose et Paul, 1990] décrits dans le paragraphe 2.2. Ces travaux constituent une référence dans l'état de l'art des systèmes de détection de mots clés en utilisant les modèles poubelles. Leur évaluation a été faite sur un corpus de test contenant 20 mots clés pour une durée de conversation de 22 minutes et un total de 353 occurrences de mots clés. En terme de FOM, les meilleures performances qu'ils ont réalisées sont de l'ordre de 62.7%.

Chapitre 6

Mesure de confiance

6.1 Introduction

Avec l'augmentation du nombre des applications de la technologique de la reconnaissance automatique de la parole, les exigences de bonnes performances de détection des mots clés se sont multipliées. En effet, dans les applications pratiques de la reconnaissance automatique de la parole, comme par exemple, des services vocaux interactifs, nous sommes souvent confrontés à des utilisateurs employant la parole spontanée pour interagir avec le système. Dans ce contexte, l'utilisation de la mesure de confiance peut se révéler très utile pour détecter les mots clés de l'application.

L'efficacité d'un système de reconnaissance peut, en effet, être améliorée si nous sommes capables de prédire si une hypothèse proposée par le système de reconnaissance est correcte ou incorrecte. Parmi les applications concernées par cette amélioration, nous pouvons citer : les systèmes de gestion de dialogue qui peuvent être améliorés de manière significative si nous avons une bonne idée de la précision de la reconnaissance [Bouwman et Hulstijn, 1998] , la détection de mots hors-vocabulaire qui est aussi indispensable pour un bon système de détection de mots clés [Rose, 1995] [Bayya, 2000], la sélection de portions de signal utiles pour l'adaptation non supervisée de modèles acoustiques [Kemp et Waibel, 1998], certaines méthodes alternatives d'élagage dans les algorithmes de programmation dynamique efficace [Neti *et al.*, 1997] et l'utilisation des mesures de confiance dans les outils de diagnostic des systèmes de reconnaissance de la parole [Eide *et al.*, 1995].

Ainsi, il est souhaitable de calculer une mesure de confiance pour chaque unité reconnue par le système (phonème, mot, etc.), c'est-à-dire d'associer à chacune de ces hypothèses de reconnaissance une mesure qui correspond à sa fiabilité, ce qui permet de rejeter les hypothèses les moins fiables. Dans le cas de la reconnaissance de mots clés, il peut être intéressant d'utiliser un score de confiance pour rejeter les fausses acceptations. Afin de calculer ce score, plusieurs travaux ont utilisé les probabilités *a posteriori* des mots. D'autres travaux ont exploré la combinaison des informations acoustiques et linguistiques des mots pour déterminer leurs scores [Jiang et Lee, 2003].

Ce chapitre est consacré à l'utilisation des mesures de confiance comme méthode de détection de mots clés. Tout d'abord, nous commençons par l'emploi des moyennes des probabilités *a posteriori* des phonèmes pour calculer une mesure de confiance par mot clé. Ensuite, nous introduisons la notion de mesure de confiance à base de boucle de phonèmes, dans laquelle nous proposons

de calculer en premier lieu, un rapport de vraisemblance et en second lieu, une distance de vraisemblance. Pour calculer ces deux paramètres, nous utilisons les scores du mot clé reconnu et de son image (un ensemble de phonèmes reconnus) trouvé par la méthode de boucle de phonèmes. Enfin, nous combinons la notion de modèle poubelle avec celle de mesure de confiance.

6.2 Mesures de confiance basées sur les moyennes des probabilités *a posteriori*

Comme première intuition, pour calculer une mesure de confiance d'un mot clé reconnu, nous avons pensé à l'utilisation des probabilités *a posteriori* des phonèmes qui composent le mot en question. Ces probabilités représentent bien la meilleure source d'informations dont nous pouvons nous servir afin d'accepter ou de rejeter le mot reconnu. Diverses méthodes ont été proposées pour calculer une mesure de confiance d'une hypothèse de reconnaissance, comme décrit au paragraphe 2.3. Parmi elles, nous trouvons des techniques qui se basent sur des combinaisons de mesures de confiance calculées au niveau des phonèmes et qui permettent d'obtenir à la fin une mesure de confiance pour tout le mot en question.

Nous avons utilisé ce type de techniques et nous avons calculé une mesure de confiance pour chaque mot clé comme étant la moyenne des probabilités *a posteriori* des phonèmes. En premier lieu, nous employons trois types de moyennes : arithmétique, géométrique et harmonique. En second lieu, nous introduisons la notion de normalisation que nous appliquons à ces différentes moyennes.

6.2.1 Pré-traitement

Pour calculer la mesure de confiance d'un mot clé reconnu, il faut connaître une mesure élémentaire qui est la probabilité *a posteriori* de chaque phonème. En pratique, pour avoir cette valeur, tout d'abord nous commençons par faire la reconnaissance en utilisant une grammaire basée sur les modèles des mots clés, le modèle silence et une boucle de phonèmes indépendants du contexte (figure 5.13). Durant cette étape, nous essayons d'avoir le minimum d'omissions puisque dans la prochaine étape, nous aurons besoin de l'ensemble des mots clés reconnus afin de détecter les mots clés qui sont réellement prononcés. Le fait d'avoir des omissions pendant la phase de reconnaissance, nous empêchera de réagir ensuite dans la phase de vérification (ou de décision) et nous ne pourrons plus combler ce manque.

Ensuite, nous réalisons un alignement de la séquence des phonèmes de la phrase reconnue avec leurs modèles HMM. Ainsi, nous pouvons mémoriser pour chaque état de chaque modèle de phonème, le nombre de trames associés et leurs probabilités d'observations.

Enfin, nous appliquons l'algorithme de Viterbi pour les trois états du modèle HMM du phonème Ph_i étant donné la séquence d'observations O_t correspondante, afin de calculer la probabilité d'observation acoustique locale $P(O_t|Ph_i)$. Ainsi, nous obtenons la probabilité *a posteriori* du phonème Ph_i : $P(Ph_i|O_t)$ donnée par l'équation suivante :

$$P(Ph_i|O_t) = \frac{P(O_t|Ph_i)P(Ph_i)}{\sum_j P(O_t|Ph_j)P(Ph_j)}$$

où :

$P(O_t|Ph_i)$ est la probabilité d'observation acoustique locale du phonème Ph_i.

$P(Ph_i)$ est la probabilité *a priori* du phonème Ph_i, tous les phonèmes étant équiprobables.

$m = \{Ph_1, \ldots, Ph_N\}$: est la séquence des phonèmes du mot clé prononcé.

$O = \{O_1, \ldots, O_T\}$: est la séquence des observations acoustiques, qui est équivalente à :

$O = \{O_{d[1]}, \ldots, O_{f[1]}; \ldots; O_{d[i]}, \ldots, O_{f[i]}; \ldots; O_{d[N]}, \ldots, O_{f[N]}\}$, où $d[i]$ et $f[i]$ représentent respectivement, les trames de début et de fin du phonème numéro i, avec $d[1] = 1$ et $f[N] = T$.

6.2.2 Mesures de confiance à base de moyennes

Dans le paragraphe précédent, nous avons calculé la probabilité *a posteriori* de chaque phonème. Un mot clé est composé d'un ensemble de phonèmes, la question qui se pose est comment peut-on combiner ces différentes valeurs de probabilités de phonèmes afin de calculer une seule mesure de confiance pour chaque mot clé. Les combinaisons les plus populaires sont bien sûr les moyennes, dont on peut trouver différents types : moyenne arithmétique, géométrique et harmonique.

Sans conteste, la moyenne arithmétique est la plus courante car elle est la plus intuitive et la plus naturelle. La moyenne géométrique applique le même principe que la moyenne arithmétique, mais en utilisant la notion de multiplication au lieu de l'addition. La moyenne géométrique de deux nombres a et b est la racine carrée de leur produit $m_g = \sqrt{ab}$, c'est une valeur moyenne entre ces deux nombres. En effet, nous pouvons démontrer facilement que si $a < b$ alors $a < m_g < b$. Cette moyenne n'est pas très courante, mais elle a prouvé son efficacité dans plusieurs problèmes mathématiques. Le dernier type de moyenne est la moyenne harmonique, elle représente la moyenne arithmétique des inverses. C'est la moins intuitive, mais elle présente un intérêt surtout au niveau de la physique des ondes, qui traite de périodes et de fréquences qui sont des notions inverses. Cette moyenne a aussi fait ses preuves dans d'autres domaines de la physique où les moyennes géométrique et arithmétique ne sont pas absentes non plus. Cependant, nous remarquons, que chaque moyenne essaie d'extraire des informations à partir des données (suite de nombres) d'une façon différente des autres. Ces trois manières différentes de traiter les choses, nous ont incité à les tester toutes.

La mesure de confiance est générée pour chaque mot en utilisant les mesures de confiance des phonèmes qui le composent. La mesure de confiance au niveau phonème est calculée comme étant la probabilité *a posteriori* causée par l'observation acoustique. Les formules des trois mesures de confiance résultantes sont les suivantes :

$$MC_a = \frac{1}{N}[\sum_{i=1}^{N} MC_i] \tag{6.1}$$

$$MC_g = exp[\frac{1}{N}[\sum_{i=1}^{N} Log(MC_i)] \tag{6.2}$$

$$MC_h = \frac{N}{\sum_{i=1}^{N} \frac{1}{MC_i}} \tag{6.3}$$

où :

$MC_i = P(Ph_i|O_t)$, N est le nombre total de phonèmes dans le mot clé m. MC_a, MC_g et MC_h sont des mesures de confiance calculées respectivement, comme moyenne arithmétique, géométrique et harmonique.

Pour chaque mesure de confiance $MC_f \in \{MC_a, MC_g, MC_h\}$ nous fixons un seuil γ. Si le score de confiance d'un mot m est inférieur au seuil fixé, ce mot sera rejeté. Les différentes valeurs du seuil γ ont été choisies en se basant sur notre base de développement.

$$m = \begin{cases} Accepté & \text{si } MC_f(m) > \gamma \\ Rejeté & \text{sinon} \end{cases}$$

Nous avons étudié les performances de notre approche en faisant varier le seuil γ de chaque mesure de confiance et nous avons pu visualiser ainsi, l'évolution du taux de détection et du nombre de fausses acceptations des mots clés. Les résultats obtenus sont représentés par la courbe ROC et la courbe rappel/précision correspondantes comme illustré par les figures 6.1 et 6.2.

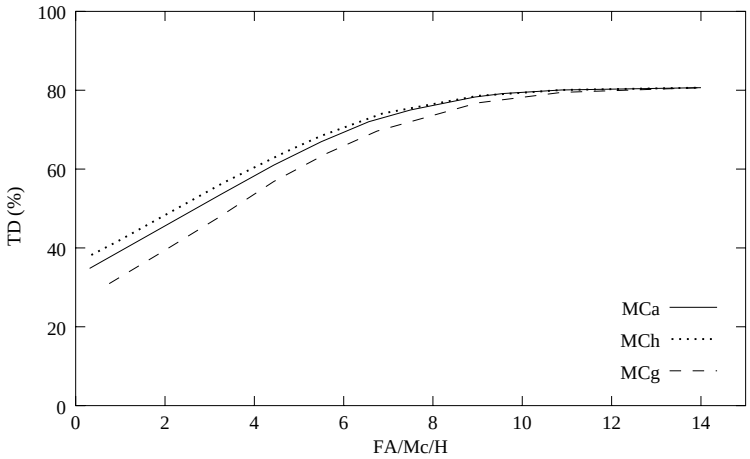

FIG. 6.1 – Courbes ROC des mesures de confiance calculées comme moyenne arithmétique, moyenne géométrique et moyenne harmonique des probabilités d'observations acoustiques locales des phonèmes.

Nous obtenons, pour la moyenne géométrique un FOM de l'ordre de 59.8% et une efficacité égale à 64.1% avec un intervalle de confiance de 1.6%. Concernant la moyenne arithmétique, nous atteignons la valeur de 63.6% pour la FOM et la valeur de 67.3% pour l'efficacité avec le même

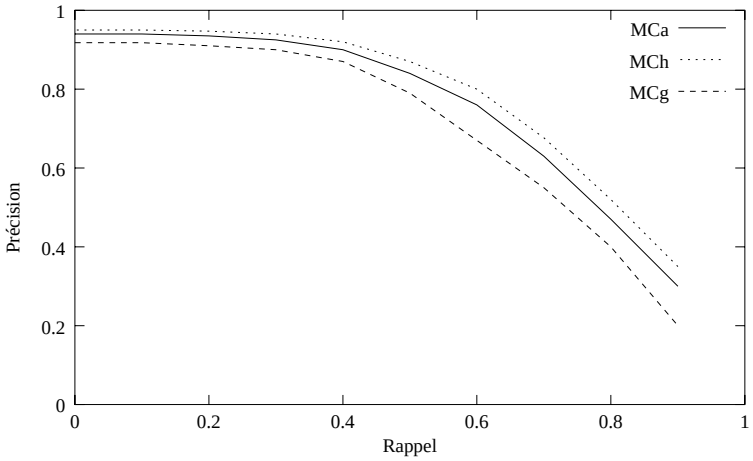

FIG. 6.2 – Courbes rappel/précision des mesures de confiance calculées comme moyenne arithmétique, moyenne géométrique et moyenne harmonique des probabilités d'observations acoustiques locales des phonèmes.

intervalle de confiance. Enfin, l'utilisation de la moyenne harmonique nous donne un FOM de l'ordre de 65.2% et une efficacité de 69% toujours avec la même valeur de l'intervalle de confiance.

Les meilleures performances sont obtenues par la moyenne harmonique suivies par celles obtenues par la moyenne arithmétique. Ces deux moyennes sont basées sur la même loi, qui est la sommation. Les faibles performances obtenues par la moyenne géométrique peuvent être expliquées par le fait que cette dernière est basée sur la loi de multiplication. En effet, nous pouvons remarquer que cette moyenne chute rapidement dès l'introduction d'une valeur faible par rapport aux autres.

6.2.3 Mesures de confiance à base de moyennes normalisées

La durée d'un phonème est une information importante. Nous avons pensé à incorporer cette information dans le calcul des mesures de confiance. Nous avons donc appliqué à chaque mesure de confiance une pondération en normalisant tous les phonèmes par leurs durées respectives. Ainsi, la mesure de confiance de chaque mot clé sera basée sur une combinaison de mesures de confiance calculées au niveau des phonèmes et normalisées par les durées respectives de ces derniers. Nous définissons alors le paramètre T_i comme étant la durée du phonème Ph_i. T_i est donné par la formule suivante :

$$T_i = f[i] - d[i] + 1$$

Ainsi, nos trois mesures de confiance calculées comme étant les moyennes arithmétique, géométrique et harmonique normalisées s'écrivent maintenant sous les formes suivantes :

$$MC_{an} = \frac{1}{N}[\sum_{i=1}^{N} \frac{MC_i}{T_i}] \tag{6.4}$$

$$MC_{gn} = exp[\frac{1}{N}[\sum_{i=1}^{N} Log(\frac{MC_i)}{T_i}] \tag{6.5}$$

$$MC_{hn} = \frac{N}{\sum_{i=1}^{N} \frac{T_i}{MC_i}} \tag{6.6}$$

Les scores de confiance MC_{an}, MC_{gn} et MC_{hn} représentent respectivement la moyenne arithmétique normalisée, la moyenne géométrique normalisée et la moyenne harmonique normalisée. Ils seront utilisés pour fournir une décision finale pour accepter ou rejeter chaque mot reconnu.

Nous procédons de la même manière que dans la méthode précédente, dans le sens où chaque mot ayant un score de confiance inférieur à un seuil fixé sera rejeté. Les valeurs du seuil ont été fixées par l'utilisation de la base de développement. Les performances réalisées par ces nouvelles méthodes basées sur la notion de normalisation sont illustrées par les figures 6.3 et 6.4.

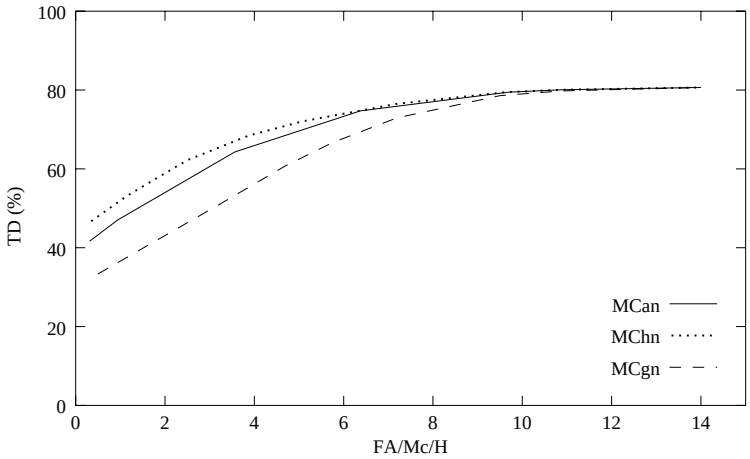

FIG. 6.3 – Courbes ROC des mesures de confiance calculées comme étant les moyennes arithmétique, géométrique et harmonique normalisées.

En normalisant les probabilités des phonèmes par leurs durées, nous avons amélioré les résultats des trois types de moyennes. En effet, nous obtenons un FOM égale à 62% et une efficacité de l'ordre de 65.8% pour la moyenne géométrique, donc une amélioration de 2.2% au niveau de la valeur FOM et 1.7% au niveau de l'efficacité. Concernant la moyenne arithmétique normalisée, nous avons réalisé une amélioration par rapport à la moyenne arithmétique normale de 4.9% au niveau FOM et de 5.1% au niveau efficacité. Enfin, pour la moyenne harmonique, nous avons obtenu les meilleurs valeurs de FOM et d'efficacité, en effet nous atteignons la valeur de 70.3% pour la mesure de FOM et la valeur de 74.2% pour l'efficacité avec un intervalle de confiance de 1.5%.

En conclusion, nous remarquons qu'avec l'utilisation des moyennes normalisées les performances obtenues gardent le même ordre d'importance que précédemment. C'est-à-dire que les meilleures performances sont toujours réalisées par la moyenne harmonique ensuite par la moyenne arithmétique et enfin par la moyenne géométrique.

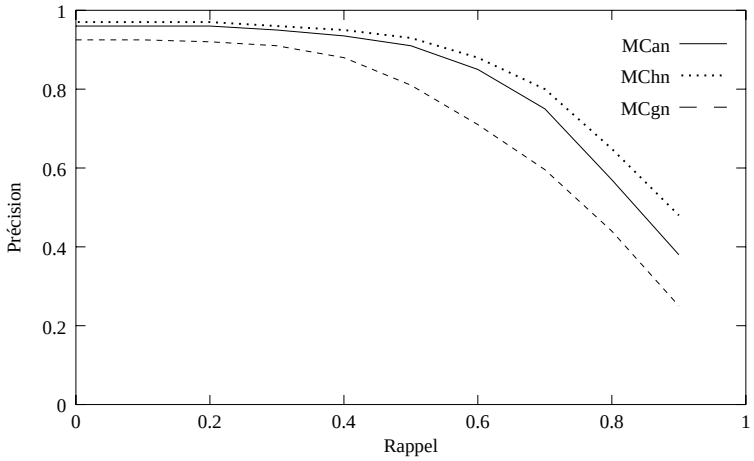

FIG. 6.4 – Courbes rappel/précision des mesures de confiance calculées comme étant les moyennes arithmétique, géométrique et harmonique normalisées.

6.3 Mesures de confiance à base de boucle de phonèmes

Dans un système de reconnaissance automatique de la parole utilisant une grammaire composée d'un ensemble de mots clés et de modèles poubelles, le risque d'insertion de mots clés est important. En effet, comme nous l'avons déjà expliqué au niveau du paragraphe 5.2, il suffit qu'un mot commence ou se termine par les mêmes phonèmes qu'un mot clés donné pour qu'il soit remplacé par ce dernier. Ce problème est encore plus sérieux dans le cas où on a des mots qui ressemblent beaucoup à des mots clés. En conclusion, nous pouvons affirmer qu'avec une grammaire composée seulement de mots clés et de modèles poubelles, nous obtenons un grand nombre d'insertions de mots clés.

Effectuer la reconnaissance de la parole en se basant sur une boucle de phonèmes est une solution envisageable. En effet, une telle méthode nous permet de faire la reconnaissance phonème par phonème et de trouver à chaque fois le phonème le plus proche de celui prononcé. Cependant, la question qui reste posée dans ce cas est, de trouver le mot clé prononcé, car nous savons très bien qu'il suffit d'un phonème non correctement reconnu pour que le mot en entier ne soit pas détecté (nous ne pouvons pas trouver une mise en correspondance entre la transcription phonétique du mot et l'ensemble des phonèmes qui lui est associé).

Afin de remédier à ce problème, nous proposons de prêter plus d'attention aux phonèmes reconnus avec des vraisemblances importantes et d'en profiter pour décider de l'acceptation ou du rejet du mot en question.

Dans cette section, nous proposons de calculer nos mesures de confiance en se basant sur le niveau le plus élémentaire qui est le niveau de la trame acoustique. En premier lieu, nous calculons un rapport de vraisemblance entre le mot reconnu et les phonèmes qui lui sont associés par la méthode à base de boucle de phonèmes. En second lieu, nous introduisons le calcul d'une distance de vraisemblance entre ces deux entités.

6.3.1 Pré-traitement

Pour calculer nos différentes mesures de confiance, nous avons besoin en premier lieu de la probabilité d'observation de chaque trame. Afin d'extraire cette information, il est nécessaire de passer par trois étapes.

Tout d'abord, nous utilisons un système de reconnaissance à base de boucle de phonèmes pour trouver la liste des phonèmes reconnus associés à une séquence d'observations O correspondante à une suite de mots prononcés m. Puis, nous réalisons un alignement de la séquence de phonèmes reconnus sur leurs modèles HMM afin de sauvegarder pour chaque trame sa probabilité d'observation.

Ensuite, nous réalisons la reconnaissance en se basant sur une grammaire composée de l'ensemble des mots clés et du modèle de silence. Ceci nous permet d'avoir un grand nombre d'insertions de mots clés dont en aura besoin par la suite. De la même manière, nous réalisons un alignement de la séquence des phonèmes reconnus associés à la même séquence d'observations O sur leurs modèles HMM afin de sauvegarder pour chaque trame sa probabilité d'observation.

Enfin, nous associons à chaque trame o_t de O un état q_t, nous obtenons alors, pour le décodage de $O = (o_1, o_2, \ldots, o_T)$, la séquence acoustique des états $Q = (q_1, q_2, \ldots, q_T)$.

Ainsi, nous pouvons accorder à chaque mot clé reconnu deux ensembles de probabilités. Le premier ensemble est composé des probabilités d'observations des états qui constituent ce mot notées $P(o_t|q_i)$. Le deuxième ensemble correspond aux probabilités $P(o_t|q_{bi})$ qui sont trouvées en utilisant la méthode à base de boucle de phonèmes. Nous proposons de calculer en premier lieu, un rapport de vraisemblance entre ces deux ensembles de probabilités et en second lieu, nous cherchons à mesurer leur distance de vraisemblance.

6.3.2 Rapport de vraisemblance

Dans cette approche, la décision d'accepter ou de rejeter un mot clé reconnu est prise en se basant sur un test de rapport de vraisemblance. Ce dernier est calculé comme étant un rapport entre le mot reconnu et la séquence de phonèmes qui lui correspond obtenue par la méthode à base de boucle de phonèmes. Il s'obtient en combinant les rapports de vraisemblance calculés pour chacune des trames o_t de l'observation acoustique O et il s'écrit donc, sous la forme suivante :

$$Rv(O|m) = \frac{\Pi_{i=1}^{T} P(o_t|q_i)}{\Pi_{i=1}^{T} P(o_t|q_{bi})} \tag{6.7}$$

où : $Rv(O|m)$ est le rapport de vraisemblance, $P(o_t|q_i)$ est la probabilité d'observation correspondante à la trame o_t sachant l'état q_i du mot clé reconnu. $P(o_t|q_{bi})$ est la probabilité d'observation correspondante à la trame o_t sachant l'état q_{bi}, cette probabilité est trouvée par la méthode de boucle de phonèmes.

Afin de tenir compte de la durée de la séquence d'observations, nous proposons une autre mesure de confiance $Rv_n(O|m)$ qui s'obtient en appliquant, sur $Rv(O|m)$, une simple normalisation par le nombre des trames acoustiques T.

$$Rv_n(O|m) = \frac{Rv(O|m)}{T} \tag{6.8}$$

Chacun des scores $Rv(O|m)$ et $Rv_n(O|m)$ sera utilisé ensuite, afin de fournir une décision finale permettant d'accepter ou de rejeter le mot reconnu. Pour ce faire, nous fixons pour chaque score un seuil γ, le mot en question sera donc rejeté si son score est inférieur à ce seuil. Ces valeurs de seuil ont été choisies en se basant sur la base de développement. Par exemple pour le rapport $Rv(O|m)$ nous avons :

$$m = \begin{cases} Accepté & \text{si } Rv(O|m) > \gamma \\ Rejeté & \text{sinon} \end{cases}$$

Pour valider ces deux nouvelles approches, nous avons mené une série d'expériences sur notre base de test. En faisant varier le seuil γ, nous avons tracé les courbes ROC et les courbes rappel/précision correspondantes aux deux mesures $Rv(O|m)$ et $Rv_n(O|m)$. Ces courbes sont données au niveau des figures 6.5 et 6.6.

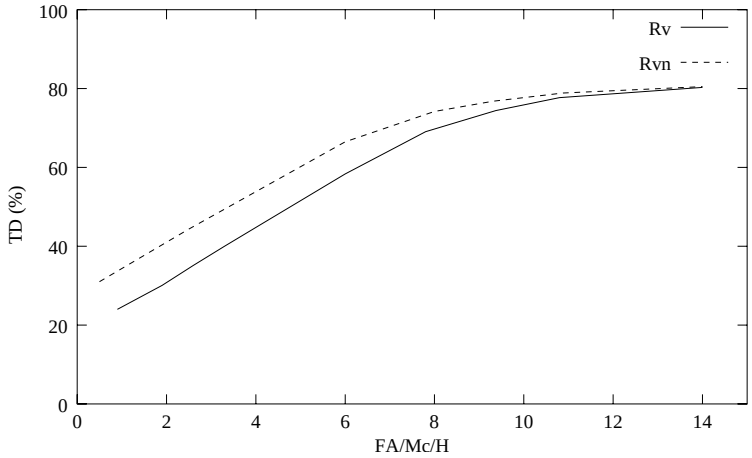

FIG. 6.5 – Courbes ROC de l'approche basée sur le rapport de vraisemblance avec et sans normalisation.

En premier lieu, nous obtenons un FOM de l'ordre de 61.6% et une efficacité de 65.2% avec un intervalle de confiance de 1.6%. En second lieu, c'est-à-dire après normalisation, la valeur du FOM a augmenté de 6.6% et nous atteignons donc 68.2%. Pour la mesure d'efficacité nous obtenons 72% avec un intervalle de confiance de 1.5%.

6.3.3 Distance de vraisemblance

Dans cette méthode, nous proposons une autre mesure de confiance basée sur une distance de vraisemblance entre le mot reconnu et son image (obtenu par la méthode à base de boucle de phonèmes). Cette distance est calculée en utilisant les probabilités d'observations définies au niveau le plus élémentaire : celui des trames acoustiques. Cette distance s'écrit alors :

$$D(O|m) = \frac{1}{N}[\sum_{i=1}^{N} \sum_{j=d[i]}^{f[i]} |P(o_t|q_j) - P(o_t|qb_j)|] \tag{6.9}$$

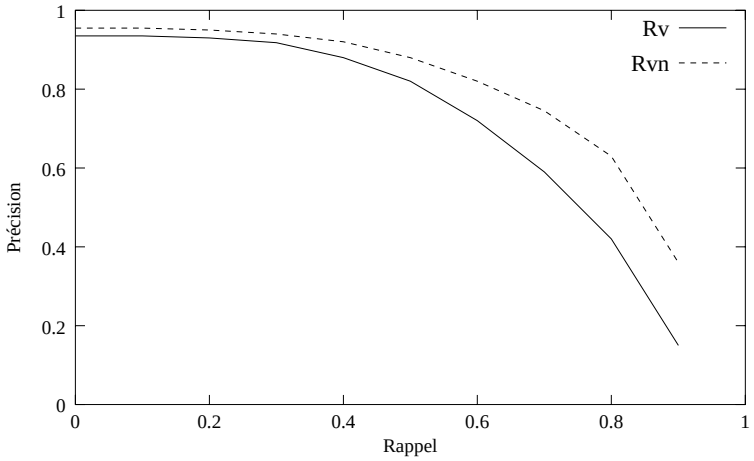

FIG. 6.6 – Courbes rappel/précision de l'approche basée sur le rapport de vraisemblance avec et sans normalisation.

où N est le nombre total des phonèmes du mot clé m et $d[i]$ et $f[i]$ représentent respectivement les trames de début et de fin du phonème Ph_i.

De la même manière qu'avec le rapport de vraisemblance, nous avons introduit la durée des phonèmes pour calculer une autre mesure de confiance. Nous définissons alors une nouvelle mesure de confiance, "la distance de vraisemblance normalisée", qui s'obtient en appliquant à la distance précédente une normalisation par la durée de chaque phonème. Cette nouvelle distance est donnée donc par :

$$D_n(O|m) = \frac{1}{N}[\sum_{i=1}^{N} \frac{1}{f[i] - d[i] + 1} \sum_{j=d[i]}^{f[i]} |P(o_t|q_j) - P(o_t|qb_j)|] \tag{6.10}$$

Les scores $D(O|m)$ et $D_n(O|m)$ appliqués à un mot hypothèse m seront utilisés ensuite pour décider de l'acceptation ou du rejet de ce mot. Il suffit pour cela, de fixer un seuil d'acceptation pour chacune de ces deux mesures. Ce seuil appartient à l'ensemble des valeurs qui ont été choisies en se basant sur la base de développement. Tout mot ayant un score plus faible que le seuil correspondant sera rejeté.

Les deux figure 6.7 et 6.8 représentent respectivement les courbes ROC et les courbes rappel/précision des deux distances $D(O|m)$ et $D_n(O|m)$. Ces courbes sont obtenues en faisant varier les seuils d'acceptation et en utilisant notre base de test habituelle.

L'utilisation de la distance de vraisemblance comme mesure de confiance permettant d'accepter les mots clés réellement prononcés et de rejeter les mots clés insérés, nous a donné les résultats suivants : une valeur de FOM de l'ordre de 65.4% et une efficacité de 69.1% avec un intervalle de confiance de 1.5% pour l'approche basée sur la distance de vraisemblance normale. Ces résultats sont améliorés en appliquant la notion de normalisation. En effet, la valeur de la FOM a augmenté de 6.1% et nous obtenons 71.5%, de la même façon, la valeur de l'efficacité a progressé de 6.4% et nous atteignons 75.5% avec un intervalle de confiance de 1.4%.

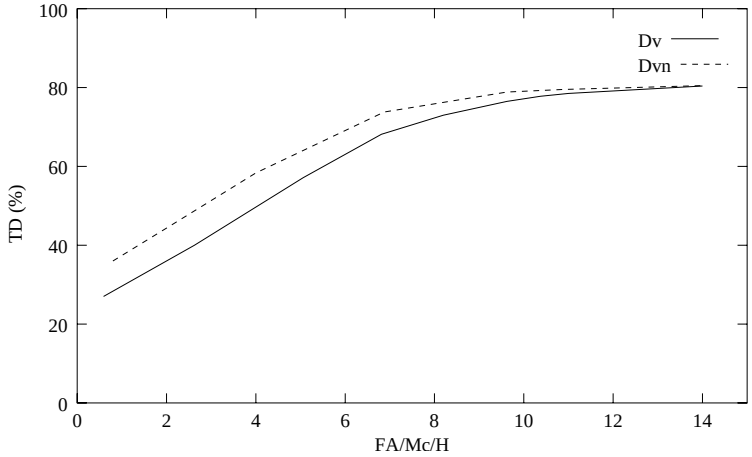

FIG. 6.7 – Courbes ROC de la méthode basée sur la distance de vraisemblance avec et sans normalisation.

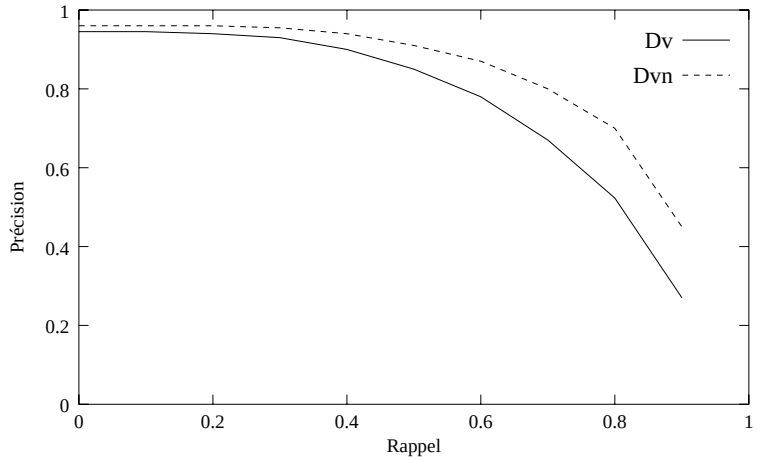

FIG. 6.8 – Courbes rappel/précision de la méthode basée sur la distance de vraisemblance avec et sans normalisation.

6.4 Modèle hybride

La combinaison des sorties de différents classifieurs ou systèmes de reconnaissance peut souvent améliorer le taux de reconnaissance et la robustesse d'un système [Halberstadt et Glass, 1998]. Les résultats obtenus par de telles combinaisons sont plus complets surtout quand les différents classifieurs utilisent différentes approches de modélisation pour réaliser les mêmes objectifs. C'est pourquoi nous avons combiné nos deux méthodes de détection de mots clés, à savoir la mesure de confiance et le modèle poubelle. Une manière simple pour combiner ces deux méthodes consiste à détecter les mots hors-vocabulaire durant la phase de reconnaissance en utilisant le modèle poubelle, puis à appliquer une mesure de confiance sur les hypothèses des mots reconnus pour raffiner notre détection.

Il s'agit donc d'appliquer séquentiellement la méthode du modèle poubelle et une mesure de confiance. En premier lieu, nous avons utilisé un état poubelle sans apprentissage et un modèle poubelle avec apprentissage, il s'agit donc du modèle combiné, puisque c'est le modèle qui nous a donné les meilleurs résultats. En second lieu, nous vérifions si les mots clés reconnus correspondent bien à des mots clés corrects. Ceci est réalisé par l'intermédiaire d'une mesure de confiance calculée pour chaque mot clé. Dans notre travail, nous avons utilisé la moyenne harmonique normalisée qui a donné aussi les meilleurs résultats par rapport aux autres moyennes introduites dans le paragraphe 6.2.3.

Au niveau de ce modèle hybride, nous commençons par l'utilisation du modèle poubelle avec apprentissage représenté par un GMM à 256 gaussiennes déterminées au cours de la phase d'apprentissage. Ensuite, nous introduisons l'état poubelle sans apprentissage représenté par un HMM à un seul état et dont le score est calculé au cours de la phase de test. Puis, nous effectuons la reconnaissance en utilisant une grammaire basée sur les modèles des mots clés, le modèle de silence et les deux modèles poubelles. Durant cette étape, nous essayons d'absorber le maximum de mots hors-vocabulaire en faisant varier le coefficient Pme comme indiqué dans la figure 5.1. Ensuite, nous réalisons un alignement de la séquence de phonèmes reconnus sur leurs modèles HMM afin de mémoriser pour chaque état de chaque phonème, le nombre de trames qui lui sont affectées et la probabilité d'observation de chaque trame. Enfin, nous appliquons l'algorithme de Viterbi sur la séquence d'observations correspondante à chaque phonème Ph_i pour calculer la probabilité d'observation acoustique locale $P(O_t|Ph_i)$. Ces probabilités seront utilisées ensuite pour calculer la mesure de confiance de chaque mot reconnu en appliquant la moyenne harmonique normalisée. De la même manière, nous fixons les valeurs de seuil en se basant sur la base de développement pour pouvoir prendre la décision finale d'accepter ou de rejeter les mots reconnus.

En faisant varier le seuil de décision, nous avons tracé la courbe ROC (figure 6.9) et la courbe rappel/précision (figure 6.10) correspondantes. La valeur obtenue du FOM est de l'ordre de 72.5% et la valeur de l'efficacité est égale à 76.8% à un intervalle de confiance de 1.4% près. Donc il s'agit d'une amélioration de 1.9% au niveau FOM et de 1.8% au niveau efficacité par rapport à la méthode combinée du chapitre précédent.

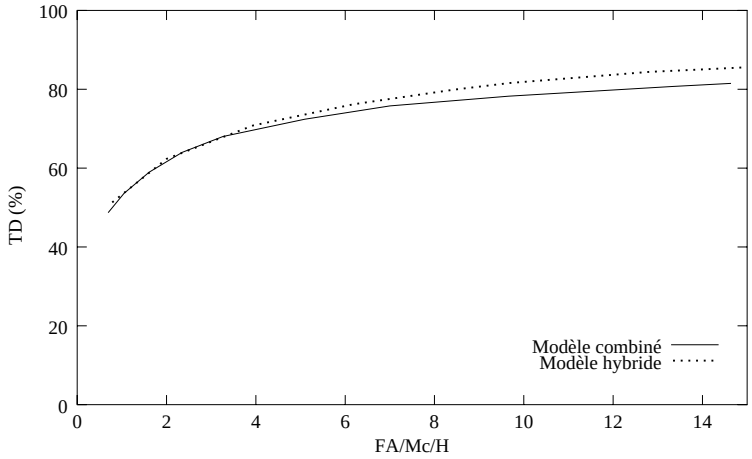

FIG. 6.9 – Courbes ROC du modèle hybride et du modèle combiné.

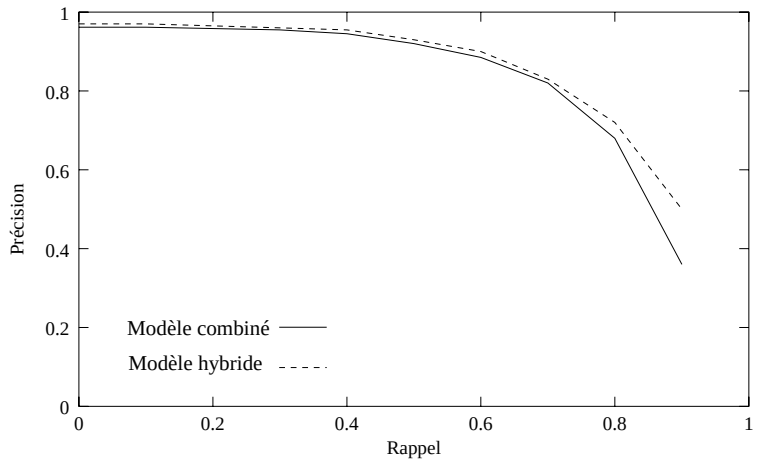

FIG. 6.10 – Courbes rappel/précision du modèle hybride et du modèle combiné .

6.5 Conclusion

Dans ce chapitre nous avons présenté un ensemble de méthodes de détection de mots clés basées sur l'utilisation des mesures de confiance qui permettent de rejeter ou d'accepter un mot clé reconnu.

Tout d'abord, nous avons introduit trois mesures de confiance basées sur trois types de moyennes, arithmétique, géométrique et harmonique. Ensuite, nous avons introduit la notion de normalisation pour raffiner nos mesures. À ce stade, nous avons utilisé les mêmes types de moyennes mais en les normalisant. Puis, nous avons adopté une double reconnaissance pour chaque suite de mots. La première est une reconnaissance standard utilisant une grammaire composée de mots clés et d'un modèle poubelle. La deuxième est une reconnaissance à base de boucle de phonèmes. Ces deux méthodes nous ont permis de calculer une mesure confiance basée sur les résultats qu'elles nous ont fournies en calculant un rapport et une distance de vraisemblance. Enfin, nous avons combiné les notions de mesure de confiance et de modèle poubelle, en proposant un modèle hybride. Ce dernier est basé en premier lieu, sur le modèle combiné qui utilise un modèle poubelle et un état poubelle. En second lieu, il se sert d'une mesure de confiance pour décider de l'acceptation ou non de chaque mot reconnu par le système. La mesure de confiance utilisée est celle basée sur la moyenne harmonique normalisée des probabilités acoustiques locales des phonèmes.

Pour évaluer les performances de ces différentes méthodes nous avons utilisé une base de test composée de 2245 phrases, phonétiquement riche, extraites de la base française SPEECHDAT 5000 (voir annexe).

Type du modèle	**FOM**	**Efficacité**
Moyenne géométrique	59.8%	64.1%
Moyenne géométrique normalisée	62.0%	65.8%
Moyenne arithmétique	63.6%	67.3%
Moyenne arithmétique normalisée	68.5%	72.4%
Moyenne harmonique	65.2%	69.0%
Moyenne harmonique normalisée	70.3%	74.2%
Rapport de vraisemblance	61.6%	65.2%
Rapport de vraisemblance normalisé	68.2%	72.0%
Distance de vraisemblance	65.4%	69.1%
Distance de vraisemblance normalisée	71.5%	75.5%
Modèle hybride	72.5%	76.8%

TAB. 6.1 – Tableau récapitulatif des résultats obtenus en utilisant les différentes méthodes de mesures de confiance.

En analysant les résultats obtenus, nous remarquons que les performances de nos méthodes ont été améliorées en introduisant la notion de normalisation. En effet, pour la première série d'expériences utilisant les trois types de moyennes, la moyenne géométrique normalisée a atteint une valeur de FOM de l'ordre de 62% et une efficacité de 65.8%, c'est-à-dire une amélioration de 2.2% au niveau FOM et de 1.7% au niveau efficacité. De la même façon pour la moyenne

arithmétique normalisée, nous atteignons une valeur de FOM égale à 68.5% et une efficacité de 72.4% alors qu'avec la moyenne arithmétique normale nous avons une valeur de FOM de seulement 63.6% et une efficacité de 67.3%. Les meilleurs performances obtenues par ces trois types de moyennes sont celles obtenues par la moyenne harmonique normalisée qui nous révèle une valeur de FOM de l'ordre de 70.3% et une efficacité de 74.2% avec une amélioration de 5.1% au niveau FOM et de 5.2% au niveau efficacité par rapport à la moyenne harmonique normale.

Concernant la deuxième série d'expériences basée sur l'utilisation d'une double reconnaissance et dans laquelle nous avons proposé deux types de mesures de confiance, un rapport et une distance de vraisemblance, nous remarquons de la même façon une amélioration des performances grâce à la notion de normalisation. Les performances les plus élevées sont obtenues avec la mesure de confiance à base de distance de vraisemblance normalisée qui nous a donné une valeur de FOM de l'ordre de 71.5% et une efficacité égale à 75.5%. Ces deux valeurs sont nettement meilleures que celles obtenues par la distance de vraisemblance normale puisque, pour cette dernière nous avons eu une valeur de FOM de 65.4% et une efficacité de 69.1%.

Pour le rapport de vraisemblance normalisé, nous avons atteint 68.2% comme valeur de FOM et 72% pour l'efficacité, c'est-à-dire une amélioration de l'ordre de 6.6% au niveau FOM et de l'ordre de 6.8% au niveau efficacité par rapport à la mesure de confiance à base de rapport de vraisemblance sans normalisation.

Enfin, la notion de mesure de confiance a été combinée avec celle de modèle poubelle. Dans une première étape, nous avons essayé d'absorber les mots hors-vocabulaire à l'aide des modèles poubelles. Dans une deuxième étape, nous avons raffiné nos résultats par l'utilisation d'une mesure de confiance afin de maintenir ou de rejeter les mots reconnus par le système de reconnaissance. Cette méthode, nous a permis de réaliser les performances suivantes : 72.5% comme valeur de FOM et 76.8% pour l'efficacité. Ainsi, nous obtenons les meilleures performances accomplies jusque là par les différentes méthodes que nous avons proposées.

Chapitre 7

La classification pour la détection de mots clés

7.1 Introduction

Les techniques d'apprentissage numérique et de classification statistique ont permis de réaliser des progrès dans plusieurs domaines de recherche notamment l'intelligence artificielle et la reconnaissance des formes. Il existe une multitude de classifieurs dont les techniques d'apprentissage et de classification diffèrent d'un classifeur à un autre, leur but étant d'atteindre des performances maximales. Choisir un classifieur dépend d'une part du type du problème posé et de l'autre part de la nature des données présentées.

Le problème de détection de mots clés est traditionnellement résolu par deux approches principales, les modèles "poubelles" et la mesure de confiance. Dans ce chapitre, nous montrons que la détection des mots clés peut être aussi assimilée à un problème de classification où il s'agit de classer des mots reconnus par le système de reconnaissance en deux classes "correct" et "incorrect". Notre approche est validée par l'utilisation et la comparaison de deux types de classifieurs le Perceptron Multi-Couches (PMC) et les SVM afin de détecter les 20 mots clés de notre application.

Dans la deuxième section de ce chapitre, nous exposons notre vision du problème de détection et nous expliquons comment il peut être formulé comme un problème de classification. Dans la troisième section, nous justifions notre choix d'adopter les SVM comme une méthode alternative de classification.

Dans la deuxième partie de ce chapitre qui commence à partir de la quatrième section, nous entamons l'étape d'expérimentation du PMC et des SVM avec plusieurs représentations vectorielles des mots. Nous commençons alors par employer les PMC et les SVM comme moyens de combinaison des différentes mesures de confiance. Ensuite nous proposons l'utilisation directe des informations qui nous ont servis pour le calcul des mesures de confiance, comme entrées à nos classifieurs. Enfin, nous testons le PMC et les SVM multi-classes visant à mieux classer les mots reconnus. Une conclusion générale de cette étude de classification est donnée en dernier lieu de ce chapitre.

7.2 La détection des mots clés comme problème de classification

Le problème de la détection de mots clés dans un flux de parole a été résolu dans la littérature en utilisant deux techniques différentes comme nous l'avons déjà décrit au chapitre 2. Ces deux techniques ne sont certes pas les seules, mais les plus utilisées et les plus efficaces. La première approche, basée sur l'utilisation du modèle "poubelle", consiste à modéliser tous les mots clés de l'application par des modèles HMM et à modéliser tous les mots hors-vocabulaire par des modèles "poubelles" qui ont pour rôle l'absorption de ces mots.

La seconde approche utilise la mesure de confiance. Cette approche est composée principalement de deux étapes. Dans la première étape qui est l'étape de reconnaissance, nous essayons de reconnaître tous les mots clés prononcés de façon à minimiser les omissions. La deuxième étape est une étape de post-traitement dans laquelle nous calculons une mesure de confiance pour chaque mot clé. Cette mesure permet de décider si le mot reconnu correspond bien à un mot clé prononcé ou bien au contraire il s'agit d'un mot clé inséré.

Pour notre part, nous remarquons que la mesure de confiance n'est qu'une technique de classification où nous essayons de classer les mots clés reconnus en deux classes : la classe des mots clés correctement reconnus et la classe des mots clés insérés. Nous proposons d'utiliser un moyen de classification plus performant que le simple seuil de la technique de la mesure de confiance. Un classifieur nécessite souvent une représentation vectorielle de son entrée, qui est dans notre cas le mot à classer. Cette représentation nous permet de bien caractériser le mot en question en utilisant plusieurs informations le concernant, ce qui n'est pas permis avec l'approche de mesure de confiance à base d'un simple seuil. L'utilisation d'un classifieur tels que les SVM ou le PMC nous permet de résoudre le problème des données non linéairement séparables tandis que la mesure de confiance n'est qu'un simple classifieur linéaire.

7.3 Les SVM pour la classification

Les SVM constituent une méthode de classification à apprentissage statistique. Cette méthode récente a vu le jour grâce à la théorie d'apprentissage de Vapnik [Vapnik, 1995] qui a montré l'intérêt de la minimisation du risque structurel par rapport à la minimisation du risque empirique utilisée dans plusieurs autres méthodes de classification notamment les RNAs. Le principe du minimisation du risque structurel a montré une grande capacité de généralisation chose qu'on cherche toujours à acquérir dans les méthodes de classification. En plus les SVM sont aussi capables de traiter le cas des données non linéairement séparables. Cette propriété nous intéresse particulièrement car elle nous permet de combler les défauts de la simple méthode de mesure de confiance qui n'est qu'une technique de séparation linéaire.

Les SVM ont montré leur efficacité dans plusieurs applications, et ils constituent une méthode innovante dans le domaine de classification à apprentissage statistique. Comme le PMC, l'apprentissage des SVM est de type supervisé et le réglage des différents paramètres de ce classifieur se fait d'une manière semi-manuelle.

Dans notre travail de détection de mots clés, nous souhaitons classer l'ensemble des mots clés obtenus par le système de reconnaissance en deux classes. Les mots correctement reconnus sont associés à la classe "correct" des mots clés corrects et les mots clés insérés et qui provoquent des

fausses acceptations sont associés à la classe "incorrect". Comme première intuition nous avons pensé à utiliser les SVM biclasses pour détecter les mots clés prononcés. En classifiant ainsi les résultats du système de reconnaissance, nous avons l'ensemble des mots clés correctement reconnus est représenté par une seule classe qui est la classe "correct". Dans le même esprit d'utilisation des SVM et afin de bien représenter chaque mot clé par ses propres caractéristiques, nous avons pensé à utiliser les SVM multi-classes. Dans ce cas, à chaque mot clé correspond une classe et tous les autres mots insérés sont représentés par la classe "incorrect". L'application des SVM multi-classes peut aussi améliorer les performances parce qu'on répartit plus nos données et donc le risque d'ambiguïté est atténué.

7.4 Combinaison des mesures de confiance

Dans le chapitre précédent, nous avons proposé en premier lieu l'utilisation de trois types de mesures de confiance, qui sont les moyennes arithmétique, géométrique et harmonique. Comme nous l'avons déjà précisé ces moyennes extraient l'information de trois manières différentes. Nous avons remarqué que la moyenne harmonique normalisée nous a fournit les meilleurs résultats. Dans l'objectif d'améliorer encore les performances, nous avons eu l'idée de combiner ces différentes mesures de confiance afin de profiter des avantages de chacune d'elles. Parmi les méthodes de combinaison les plus utilisées, nous pouvons citer, la combinaison par vote majoritaire et la combinaison linéaire des scores. Dans notre travail et afin de mieux fusionner nos six mesures de confiance (les moyennes arithmétique, géométrique, harmonique et leurs moyennes normalisées correspondantes), nous avons décidé d'utiliser le PMC comme moyen de combinaison. En effet, le PMC connu comme la méthode de classification supervisée la plus utilisé, réalise une combinaison de ses entrées, en permettant leur classification même dans le cas ou elles sont linéairement non séparables. Il est donc plus performant en fusion de données que la simple combinaison linéaire ou encore que le vote majoritaire (qui n'est qu'une combinaison linéaire à coefficients égaux).

Pour l'apprentissage du PMC, nous nous sommes servis d'une base composée de 2710 phrases prononcées par 1000 locuteurs et extraites de la base française SPEECHDAT 1000 (voir annexe). En second lieu, pour effectuer l'étape de test nous avons utilisé une base contenant 2245 phrases prononcées par 900 locuteurs et sélectionnées de la base SPEECHDAT 5000 (voir annexe) et qui constitue la même base de test utilisée pour l'évaluation de toutes les autres méthodes décrites dans ce manuscrit.

Une série d'expériences a été menée sur la base d'apprentissage pour fixer le nombre de neurones dans la couche cachée du réseau et les meilleures performances ont été atteintes pour un nombre de 2 neurones. Notre réseau sera donc composé de 6 neurones dans la couche d'entrée, 2 neurones dans la couche cachée et un neurone de sortie.

Les résultats obtenus par l'utilisation du PMC ont montré une amélioration par rapport à ceux réalisés par les différentes mesures de confiance à base des trois types de moyennes. Dans les figures 7.1 et 7.2, nous comparons les performances du PMC avec celles de la mesure de confiance à base de la moyenne harmonique normalisée qui présentait la meilleure mesure de confiance à base de moyenne en terme de performances. Les courbes représentées par ces figures illustrent respectivement les courbes ROC et les courbes rappel/précision des deux méthodes précédemment décrites.

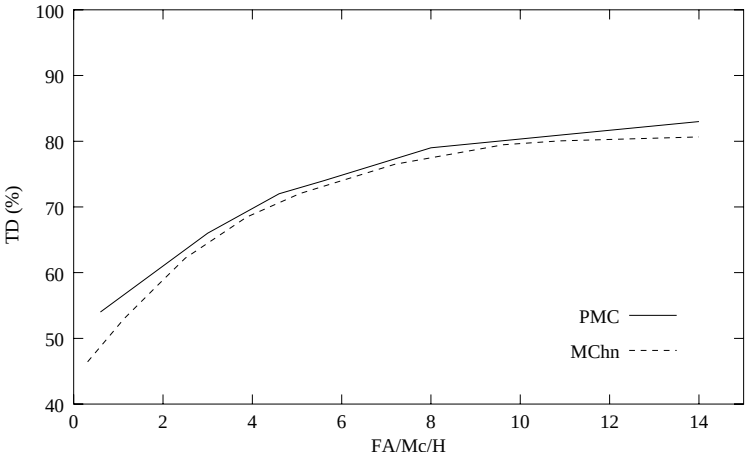

FIG. 7.1 – Courbes ROC de la mesure de confiance à base de moyenne harmonique normalisée et du PMC en utilisant un vecteur composé de six mesures de confiance.

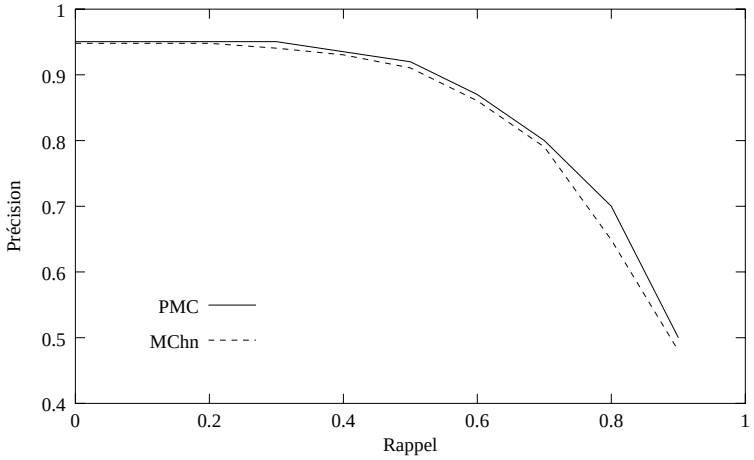

FIG. 7.2 – Courbes rappel/précision de la mesure de confiance à base de moyenne harmonique normalisée et du PMC en utilisant un vecteur composé de six mesures de confiance.

Ainsi, la combinaison des six mesures de confiance à base de moyenne arithmétique, géométrique, harmonique et leurs normalisées respectives, nous a permis d'améliorer légèrement les résultats. En effet, nous obtenons avec cette méthode, un FOM de 71.6% et une efficacité de 75.3%. Il s'agit d'une amélioration de l'ordre de 1.3% au niveau FOM et de l'ordre de 1.1% au niveau efficacité par rapport aux résultats obtenus par la mesure de confiance utilisant la moyenne harmonique normalisée des probabilités des observations acoustiques locales des phonèmes. Nous rappelons que cette dernière méthode a donné les meilleures performances des six méthodes considérées comme entrées du PMC.

Toujours, en quête d'amélioration, nous avons utilisé le classifieur SVM qui nous permet de combiner les différentes mesures de confiance à base des trois types de moyennes afin de classer l'ensemble des mots clés reconnus en mots clés corrects et mot clés insérés. Nous avons donc travaillé avec deux types de fonctions noyau (paragraphe 3.3.4), à savoir la fonction noyau linéaire et le noyau RBF. Pour déterminer les valeurs du coefficient C pour le noyau linéaire et du couple (C, γ) pour le noyau RBF, nous avons effectué une série d'expériences sur notre base d'apprentissage en faisant varier C sur l'ensemble des valeurs 0.001, 0.01, 0.1, 1, 10, 100 et 1000. Les meilleurs résultats ont été obtenus pour $C = 10$ pour la fonction noyau linéaire et (C, γ)=(10, 0.05) pour la fonction noyau RBF.

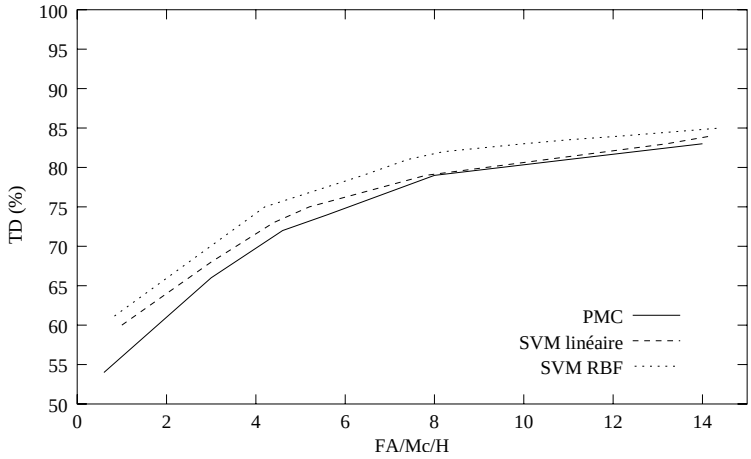

FIG. 7.3 – Courbes ROC obtenues par la combinaison de six mesures de confiance.

Pour comparer les résultats obtenus par les SVM et par le PMC, nous avons tracé les courbes ROC et les courbes rappel/précision relatives aux PMC, SVM avec noyau linéaire et SVM avec noyau RBF. Ces courbes sont données au niveau des figures 7.3 et 7.4. Comme valeurs de FOM, nous avons atteint les 75.4% avec les SVM à noyau RBF et les 73.2% avec les SVM à noyau linéaire comparant à 71.6% obtenu par le PMC. De même pour les valeurs d'efficacité qui ont atteint les 78.5% avec les SVM à noyau RBF et les 76.5% avec les SVM à noyau linéaire comparant à 75.3% obtenu par le PMC. L'intervalle de confiance considéré pour ces valeurs est de l'ordre de 1.4%.

Ainsi, nous remarquons que l'utilisation des différentes mesures de confiance nous a permis d'améliorer les résultats notamment en utilisant les SVM RBF comme moyen de combinaison. Ceci peut être expliqué par le fait que nous exploitons plus d'informations pour prendre la décision de rejeter ou d'accepter un mot clé donné.

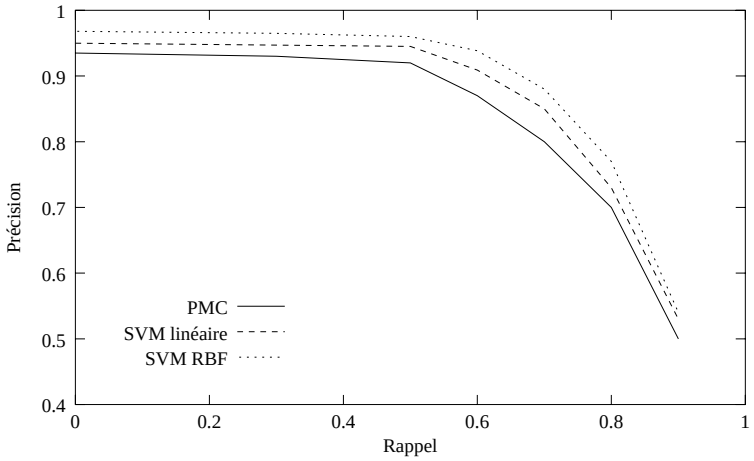

FIG. 7.4 – Courbes rappel/précision obtenues par la combinaison de six mesures de confiance.

7.5 Représentation vectorielle des mots

Nous avons remarqué qu'en utilisant la notion de mesure de confiance, nous prenons le risque de perdre une quantité importante d'informations puisque nous résumons plusieurs valeurs en une seule. D'où l'idée d'exploiter directement ces informations brutes sans passer par l'étape du calcul de la mesure de confiance. Ces informations seront utilisées par un processus de classification pour déterminer l'ensemble des mots clés prononcés.

Comme nous l'avons déjà décrit, dans un processus de classification, nous avons besoin de deux éléments importants. Le premier est le choix du classifieur à utiliser et le deuxième est le choix de la construction des vecteurs représentatifs de nos données. Ces vecteurs nous serviront comme entrée au classifieur en question. Dans notre travail, nous avons choisi d'utiliser les classifieurs PMC et SVM précédemment décrits et qui nous ont donné de bons résultats en les essayant avec les six mesures de confiance. Il nous reste alors à choisir le deuxième élément qui consiste à représenter chaque mot clé reconnu par un vecteur caractéristique.

Dans une représentation vectorielle, nous pouvons introduire plusieurs informations qui caractérisent au mieux un mot clé. Dans notre cas, nous disposons essentiellement de deux types d'informations qui sont les probabilités acoustiques locales de chaque phonème et la durée de ses états. Ces deux informations sont obtenues après une phase d'alignement de la sortie du système de reconnaissance sur les modèles HMM des phonèmes et elles représentent respectivement une caractéristique acoustique et une caractéristique temporelle. Nous allons tester ces deux types d'informations afin de bien représenter nos mots clés. Dans une étape ultérieure, nous combinons aussi ces deux informations dans l'optique d'une amélioration au niveau de la représentation des mots et au niveau des performances de notre système de détection.

7.5.1 Utilisation des probabilités d'observations acoustiques locales

Nous savons que chaque mot est composé d'une séquence de phonèmes et qu'à chaque phonème correspond une probabilité d'observation acoustique locale. Cette probabilité constitue une information acoustique importante permettant de bien caractériser les mots à classer. En adoptant une telle représentation, chaque mot clé sera caractérisé par un vecteur contenant les

probabilités acoustiques locales des phonèmes qui le constituent. L'utilisation d'un tel vecteur aidera sans doute le classifieur à bien discriminer ses classes puisqu'il ne s'agit pas d'une seule mesure, mais bien au contraire, il s'agit d'un vecteur de probabilités. Pour ces raisons nous avons adopté cette représentation dans laquelle un vecteur caractéristique d'un mot clé donné peut être représenté comme suit :

$$\vec{V}_{Mot} = [P(O_1|Ph_1), P(O_t|Ph_i), ..., P(O_T|Ph_N)]$$

Nous avons testé cette représentation vectorielle des mots avec les deux classifeurs PMC et SVM. Après une série d'expériences en utilisant notre base d'apprentissage, nous avons fixé pour le PMC, le nombre de neurones dans la couche cachée à 4. Le nombre de neurones dans la couche d'entrée est fixé à 10 représentant le nombre de phonèmes maximal dans un mot clé. La sortie du réseau contient un seul neurone. Concernant le classifieur SVM, nous avons utilisé les deux fonctions noyaux linéaire et RBF en fixant la valeur de C à 10 pour le noyau linéaire et le couple (C, γ) à (10, 1.3) pour le noyau RBF.

Les figures 7.5 et 7.6 présentent les courbes ROC et les courbes rappel/précision correspondantes à l'utilisation du PMC et des SVM avec une représentation vectorielle des mots à base des probabilités d'observations acoustiques locales.

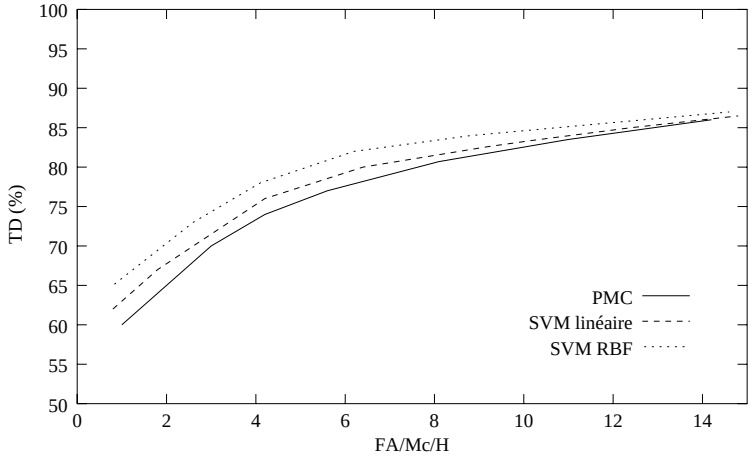

FIG. 7.5 – Courbes ROC obtenues par une représentation vectorielle à base de probabilités acoustiques locales des phonèmes.

En adoptant cette représentation des mots, le PMC nous a donné une valeur de FOM égale à 74% et une efficacité de 77.4% avec un intervalle de confiance de 1.4%. Les SVM linéaires ont amélioré les résultats en augmentant la valeur de FOM à 76% et l'efficacité à 79.2%. Les meilleures performances ont été enregistrées en utilisant les SVM à noyau RBF avec un FOM de 78.4% et une efficacité de 81.3% pour un intervalle de confiance égale à 1.3%.

Le gain réalisé par cette méthode nous confirme l'idée que l'utilisation d'un classifieur tel que le PMC ou les SVM et plus intéressante que la simple mesure de confiance et que l'utilisation des probabilités acoustiques locales des phonèmes à l'état brut comme entrées du classifieur améliore

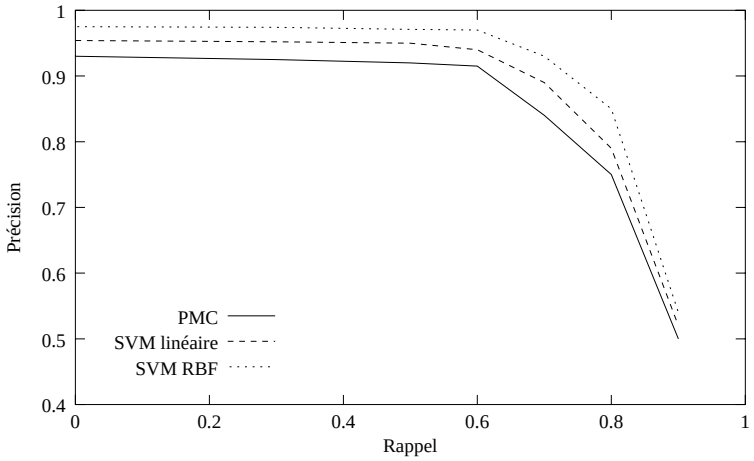

FIG. 7.6 – Courbes rappel/précision obtenues par une représentation vectorielle à base de probabilités acoustiques locales des phonèmes.

les résultats par rapport à ceux obtenus par la combinaison de six mesures de confiance qui ont été calculées en utilisant ces mêmes probabilités.

7.5.2 Utilisation du nombre de trames par état

Les classifieurs PMC ou SVM permettent de classer un ensemble de mots clés en deux classes, une classe des mots clés réellement prononcés correspondant à la classe "correct" et une deuxième classe, celle des mots clés insérés c'est-à-dire la classe "incorrect". Pour cela, il nous suffit de définir le vecteur d'entrée du classifieur choisi. Nous proposons ici une autre représentation vectorielle, différente de celle des probabilités où nous utilisons le nombre de trames par état dans chaque phonème [6]. Un mot sera donc représenté par un vecteur à 30 éléments correspondants aux nombres de trames de chaque état dans chaque phonème du mot en question.

De la même manière, nous utilisons pour tester cette nouvelle représentation, les deux classifieurs PMC et SVM. Il nous faut donc, avant d'entamer la phase de test, réaliser une série d'expériences afin de fixer les paramètres de ces deux classifieurs. À l'issue de ces expériences, nous avons pu fixer pour le PMC, le nombre de neurones dans la couche cachée à 10, le nombre de neurones dans la couche d'entrée à la valeur de 30 et un seul neurone pour la sortie. Pour le classifieur SVM, nous avons utilisé deux fonctions noyaux, la fonction linéaire et la fonction RBF. Les expériences que nous avons réalisées nous ont permis aussi de déterminer les valeurs des paramètres C à 1 pour la fonction noyau linéaire et (C, γ) à (1, 1.1) en ce qui concerne la fonction noyau RBF.

Les résultats obtenus par les deux méthodes de classification PMC et SVM en utilisant la représentation vectorielle à base de nombre de trames par état de chaque phonème constituant le mot clé, sont représentés par les figures 7.7 et 7.8. Les courbes illustrées par ces deux figures correspondent respectivement aux courbes ROC et celles de rappel/précision obtenues par nos deux types de classifieurs.

[6] Nous rappelons que dans notre travail, nous avons modélisé chaque phonème par un HMM à trois états.

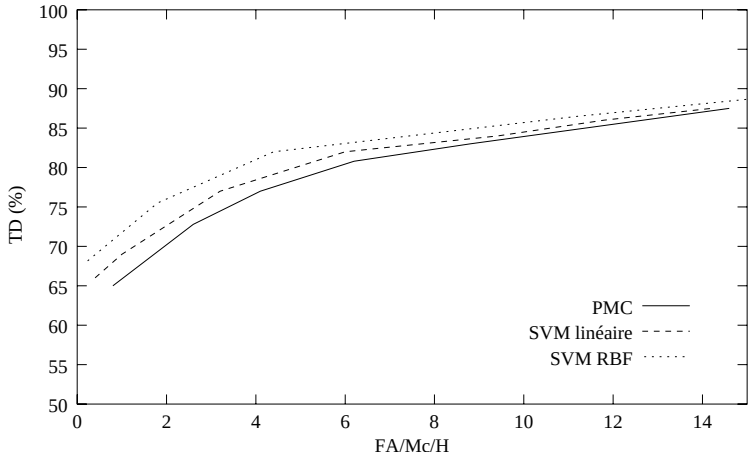

FIG. 7.7 – Courbes ROC obtenues par une représentation vectorielle à base du nombre de trames par état dans chaque phonème.

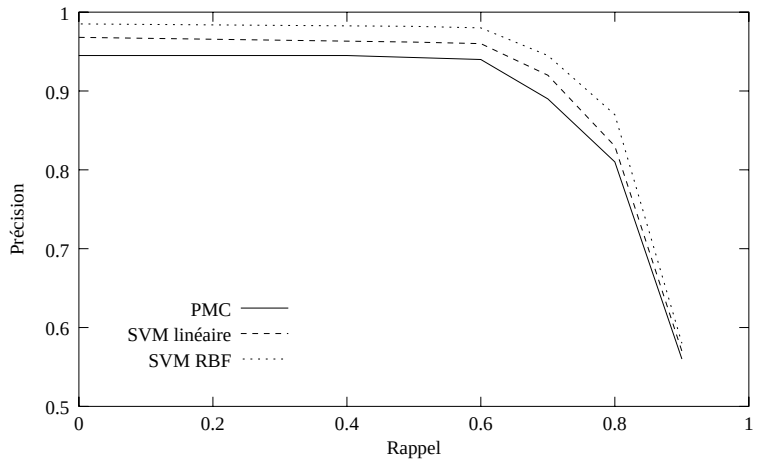

FIG. 7.8 – Courbes rappel/précision obtenues par une représentation vectorielle à base de nombre de trame par état dans chaque phonème.

Les valeurs de FOM réalisées en utilisant cette représentation vectorielle est de 77.4% pour le PMC, de 79.2% pour les SVM linéaires et de 81.2% pour les SVM RBF. Au niveau efficacité les valeurs obtenues sont les suivantes : 80.4% pour le PMC, 81.5% pour les SVM linéaires et 82.2% pour les SVM RBF. Pour toutes ces valeurs l'intervalle de confiance est de l'ordre de 1.2%.

7.5.3 Représentation vectorielle mixte

Afin de mieux représenter les mots donnés en sortie du système de reconnaissance, nous avons voulu exploiter le maximum d'information pouvant les caractériser, en l'occurrence les deux types de données déjà utilisés et qui représentent deux informations complémentaires au niveau de chaque mot : les probabilités d'observations acoustiques locales caractérisant le mot d'un point de vue acoustique et le nombre de trames dans chaque état, qui constitue une information d'ordre temporel. Ces deux types de représentations ont déjà donné de très bons résultats, leur combinaison ne peut qu'améliorer encore plus les performances. Un mot sera donc représenté par un vecteur de 40 éléments composé des probabilités acoustiques locales des phonèmes et du nombre de trames de chaque état dans chaque phonème.

Nous avons évalué les capacités de cette représentation mixte avec les deux types de classifieurs PMC et SVM. Le PMC utilisé est un réseau à 40 neurones d'entrée, 12 neurones dans la couche cachée et un neurone de sortie. Pour les SVM à noyau linéaire, nous avons fixé la valeur du paramètre C à 10 et pour les SVM à noyau RBF, nous avons pu déterminer les valeurs de (C, γ) à (10, 4).

Les résultats obtenus par cette représentation mixte avec les deux types de classifieurs sont représentés par les figures 7.9 et 7.10. Nous avons tracé respectivement les courbes ROC et celles de rappel/précision. Les meilleurs performances ont été réalisées par les SVM à noyau RBF où on atteint les 82.9% comme valeur de FOM et les 83.1% pour l'efficacité avec un intervalle de confiance de 1.2%. Les SVM à noyau linéaire ont donné aussi de bons résultats en atteignant les 81.5% comme valeur de FOM et les 82.4% pour l'efficacité, l'intervalle de confiance est égale à 1.2%. Le PMC aussi a réalisé les meilleurs de ses performances en utilisant cette représentation mixte. En effet, nous obtenons à ce niveau un FOM de l'ordre de 79.4% et une efficacité de 81.5% avec un intervalle de confiance de l'ordre de 1.2%.

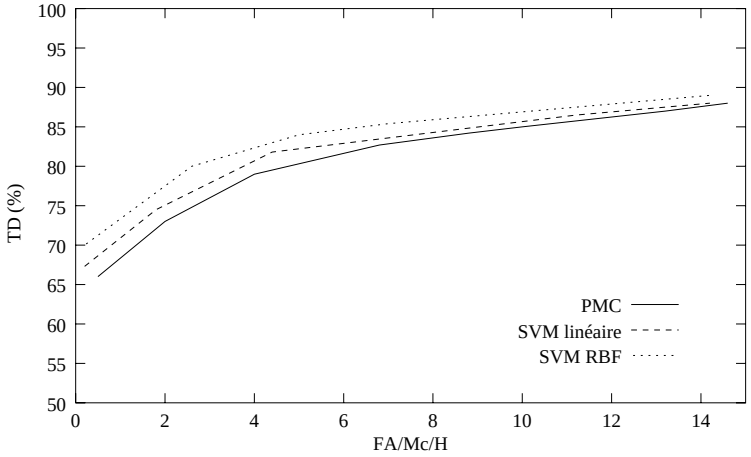

Fig. 7.9 – Courbes ROC obtenues par une représentation vectorielle mixte.

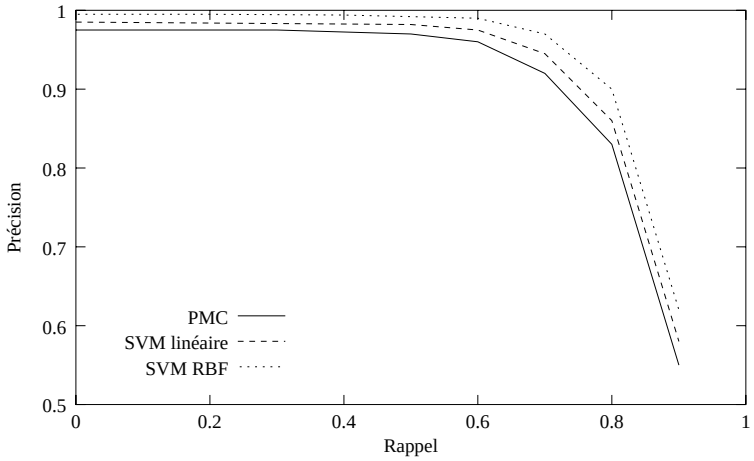

FIG. 7.10 – Courbes rappel/précision obtenues par une représentation vectorielle mixte.

7.6 Classification multi-classes pour la détection de mots clés

Dans ce chapitre, nous avons utilisé jusqu'à maintenant deux classifieurs, le PMC et les SVM et nous avons considéré seulement deux classes, la classe "correct" qui correspond à tous les mots clés réellement prononcés et la classe "incorrect" attribuée aux mots clés insérés. Dans cette approche, tous les mots clés corrects sont représentés par une seule classe or les mots clés sont différents puisque nous avons en total 20 mots clés, d'où l'idée d'attribuer à chaque mot clé une classe distincte. Ceci va permettre au classifieur de mieux discriminer les données d'entrée et donc de mieux séparer les classes des différents mots. Pour ce faire, nous avons proposé d'utiliser, en premier lieu, un PMC à plusieurs sorties et, en second lieu, un SVM multi-classes.

Pour évaluer cette alternative de détection de mots clés, nous avons adopté une représentation vectorielle mixte, dans laquelle chaque mot clé est représenté par un vecteur composé des probabilités d'observations acoustiques locales des phonèmes et du nombre de trames de chaque état dans chaque phonème constituant le mot en question. Nous avons choisi cette représentation car elle nous a donné les meilleurs résultats avec le PMC à une seule sortie et avec les SVM biclasses.

Pour déterminer les différents paramètres des deux types de classifieurs, nous avons mené une série d'expériences sur la base d'apprentissage. Nous avons donc fixé pour le classifieur PMC le nombre de neurones de la couche cachée à 28, le nombre de neurones de la couche d'entrée à 40 et enfin le nombre de neurones de la couche de sortie à 21. Pour le classifieur SVM, nous avons fixé la valeur de C à 10 pour le noyau linéaire et le couple (C, γ) à (10, 4) pour le noyau RBF.

Les résultats présentés sur les figures 7.11 et 7.12 concernent respectivement les courbes ROC et les courbes rappel/précision en utilisant le PMC à 21 sorties et les SVM multi-classes avec 21 sorties aussi.

Nous obtenons pour les SVM à noyau RBF une valeur de FOM de l'ordre de 83.2% et une efficacité de 83.4%. Ensuite, pour les SVM à noyau linéaire, nous avons atteint la valeur de 81.7% comme FOM et la valeur de 82.5% pour l'efficacité. Enfin, pour le PMC, nous avons réalisé un FOM de l'ordre de 79.8% et une efficacité de 81.7%. L'intervalle de confiance considéré est de

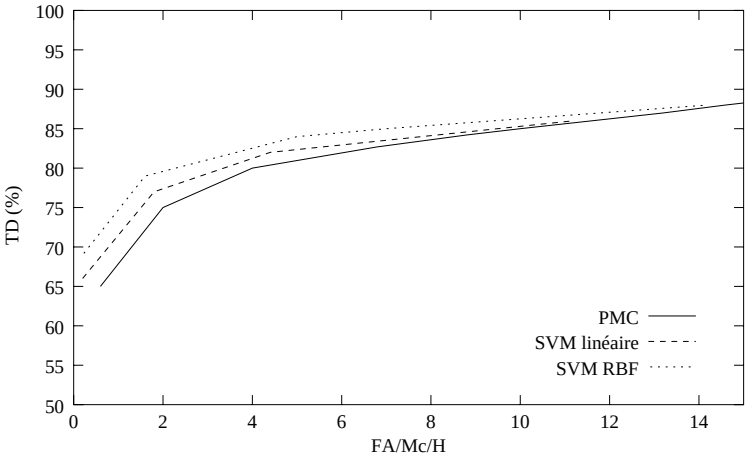

FIG. 7.11 – Courbes ROC obtenues par une classification multi-classes.

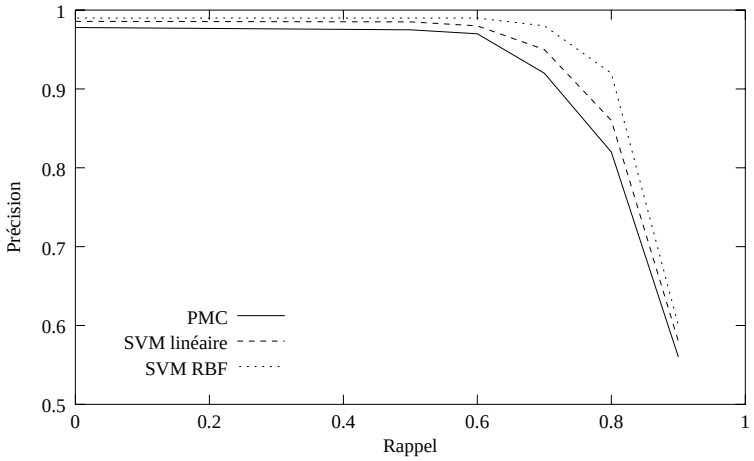

FIG. 7.12 – Courbes rappel/précision obtenues par une classification multi-classes.

l'ordre de 1.3%

Les performances obtenues par l'utilisation des classifieurs multi-classes sont comparables à celles obtenues par les classifieurs biclasses (une classe "correct" pour tous les mots clés corrects et une classe "incorrect" pour les mots insérés). En effet, le fait d'avoir plusieurs classes à la sortie du classifieur augmente le risque de chevauchement entre les classes, c'est-à-dire qu'un mot clé de la classe i peut être reconnu comme étant un mot appartenant à la classe j. Ce type d'erreurs augmente le nombre de fausses acceptations et diminue donc le nombre de mots clés corrects réellement prononcés.

7.7 Conclusion

Nous venons de présenter dans ce chapitre les performances obtenues par les deux classifieurs PMC et SVM pour la classification des mots clés reconnus afin de détecter les mots clés réellement prononcés. Après avoir étudié différentes mesures de confiance à base de trois types de moyennes dans le chapitre 6, nous nous sommes intéressés à la combinaison de ces mesures afin d'améliorer les performances de notre système de détection. Nous avons donc, tout d'abord commencé par l'utilisation d'un PMC avec un nombre de neurones d'entrée correspondant au nombre des mesures de confiance à combiner et un seul neurone pour la sortie. Ce réseau nous aide à décider d'accepter ou de rejeter le mot présenté en entrée. Puis, nous avons proposé l'utilisation du classifieur SVM avec deux fonctions noyaux, la fonction linéaire et la fonction RBF. Ensuite, afin d'améliorer les résultats, nous avons proposé trois représentations vectorielles de l'entrée du classifieur à base de probabilités d'observations acoustiques locales et du nombre de trames par état dans chaque phonème dans un mot. Enfin, nous nous sommes intéressés à l'utilisation d'un classifieur multi-classes, avec un PMC à plusieurs sorties, chacune représentant un mot clé donné, et un SVM multi-classes.

Les résultats obtenus par les différentes représentations vectorielles montrent que chacune a contribué à sa façon à l'amélioration des performances. En effet, passant d'une simple mesure de confiance utilisée pour prendre la décision d'accepter ou de rejeter un mot clé vers une combinaison de plusieurs mesures, nous avons remarqué que cette dernière a amélioré les performances de l'ordre de 1.2% au niveau FOM et efficacité par rapport à la meilleure mesure de confiance qui est la moyenne harmonique normalisée. Dans le même contexte et afin de bien représenter le mot en entrée du classifieur, nous nous sommes orientés vers les données sources utilisées pour calculer ces mesures de confiance, qui sont les probabilités d'observations acoustiques locales des phonèmes. Afin d'introduire l'aspect temporel, nous avons introduit ensuite une représentation vectorielle à base du nombre de trames par état dans chaque phonème d'un mot clé. Ensuite, nous avons proposé une représentation vectorielle plus complète qui utilise les probabilités acoustiques et le nombre de trames par état. Les résultats obtenus prouvent en premier lieu que les représentations à base de vecteur mixte sont les meilleures et en second lieu les SVM à base de fonction noyau RBF donnent toujours les meilleurs résultats et que les SVM linéaires sont plus performants que le PMC. Ceci peut être expliqué d'une part par le fait que les SVM réalisent une projection des entrées dans un espace de caractéristiques de plus grande dimension ce qui facilite la discrimination des données. D'autre part, le PMC utilise le principe de minimisation du risque empirique alors que les SVM se basent sur le principe de minimisation du risque structurel qui est plus efficace que le premier.

Type du représentation		**FOM**	**Efficacité**
Mesure de confiance	PMC	71.6%	75.3%
	SVM linéaire	73.2%	76.5%
	SVM RBF	75.4%	78.5%
Vecteur de probabilité	PMC	74%	77.4%
	SVM linéaire	76%	79.2%
	SVM RBF	78.4%	81.3%
Vecteur de nombre de trame	PMC	77.4%	80.4%
	SVM linéaire	79.2%	81.5%
	SVM RBF	81.2%	82.2%
Vecteur mixte	PMC	79.4%	81.5%
	SVM linéaire	81.5%	82.4%
	SVM RBF	82.9%	83.1%
Multi-classes	PMC	79.8%	81.7 %
	SVM linéaire	81.7%	82.5 %
	SVM RBF	83.2%	83.4 %

TAB. 7.1 – Tableau récapitulatif des résultats obtenus en utilisant différentes représentations vectorielles.

L'utilisation d'un classifieur a prouvé son efficacité en améliorant les performances de notre système de détection. En effet, nous avons atteint la valeur de 82.9% au niveau FOM et la valeur de 83.1% au niveau efficacité en utilisant les SVM à noyau RBF. Les meilleurs résultats obtenus pour les SVM linéaires biclasses sont : un FOM de l'ordre de 81.5% et une efficacité de 82.4%. Concernant le PMC biclasses, les meilleures performances réalisées sont : 79.4% comme FOM et 81.5% comme valeur d'efficacité.

La dernière étape de ce chapitre a été consacrée aux classifieurs multi-classes, où nous avons utilisé en premier lieu, un PMC à plusieurs sorties et en second lieu, nous avons employé un SVM multi-classes. Les résultats obtenus sont comparables à ceux obtenus par les classifieurs biclasses avec la même représentation vectorielle mixte. En effet, introduire un classifieur multi-classes permet de faire une bonne discrimination entre les mots clés puisque chaque mot sera représenté par une classe. En revanche, il augmente le nombre de substitutions puisqu'il a le choix d'attribuer un mot en entrée à une classe parmi plusieurs. Les meilleurs résultats sont obtenus par les SVM RBF pour lesquels, nous avons obtenu un FOM de 83.2% et une efficacité de 83.4%.

Conclusions

La détection de mots clés, sujet de cette thèse est sans doute une étape clé pour plusieurs applications interactives utilisant la parole comme moyen de communication. Elle consiste à améliorer les performances de telles applications en les aidant à détecter le sens des énoncés émis et ce en dévoilant seulement les mots porteur de sens pour l'application en question. La liste des mots clés les plus significatifs pour l'application est déterminée dans une étape antérieure : dans notre travail, par exemple, nous avons défini un ensemble de 20 mots clés.

Dans la littérature, la détection de mots clés a donné lieu à une grande variété de travaux depuis les années 80. En effet, dès l'apparition des applications interactives, les chercheurs du domaine ont adopté cette technique comme solution pour pouvoir comprendre et satisfaire les requêtes des utilisateurs sans les restreindre à suivre un vocabulaire rigide et limité. Nous distinguons alors deux approches principales utilisées pour détecter les mots clés d'une application donnée. La première, dans l'ordre chronologique, est basée sur l'utilisation des modèles poubelles. La deuxième, qui constitue l'alternative la plus simple, utilise une ou plusieurs mesures de confiance pour décider si un mot reconnu représente bien un mot clé réellement prononcé ou si au contraire il s'agit d'un mot inséré. Inspirés de cette dernière approche, nous avons aussi proposé une méthode, à notre sens plus intéressante que le simple seuil de décision, qui utilise d'autres méthodes de classification plus élaborées pour classer les mots clés reconnus dans deux classes distinctes. Dans notre travail, nous proposons l'utilisation du PMC et des SVM, sachant que ce sont les derniers qui nous ont donné les meilleurs performances.

La détection de mots clés peut être simplement réalisée à l'aide d'un système de reconnaissance à grand vocabulaire. Il s'agit de modéliser tous les mots clés et les mots hors-vocabulaire afin de les reconnaître pour ne garder enfin que l'ensemble des mots clés prédéfinis. En réalisant cette expérimentation, nous avons obtenu un taux de détection faible de l'ordre de 50%. Ceci nous a incité à adopter des méthodes de détection plus performantes. Tout d'abord, nous avons commencé notre étude par une méthode de la littérature qui consiste à créer un état poubelle au fur et à mesure de la phase de reconnaissance et qui a pour rôle l'absorption des mots hors-vocabulaire. Le score de cet état est calculé comme étant la moyenne des N meilleurs scores locaux des états des modèles phonétiques utilisés pour décrire les mots clés. Ensuite, nous avons proposé un modèle poubelle représenté par un GMM. Un GMM est un modèle HMM à seul état dont la fonction de densité est composée d'un mélange de gaussiennes et dont les paramètres sont fixés lors de la phase d'apprentissage. Enfin, nous avons proposé un modèle combiné dans lequel, nous utilisons un GMM appris durant la phase d'apprentissage et un état poubelle déterminé au fur et à mesure de la phase de reconnaissance.

Comme alternative aux modèles poubelles, nous avons proposé une nouvelle approche à base de boucle de phonèmes. Nous avons suggéré une favorisation de la reconnaissance des phonèmes

constituant un mot clé donné. Pour cela, nous avons proposé trois fonctions de récompense à savoir la fonction constante, la fonction affine et la fonction sigmoïde. Ces deux dernières fonctions dépendent du nombre de phonèmes constituant le mot clé en question. Les meilleures performances sont obtenues par la récompense sigmoïdale qui nous donne un FOM de l'ordre de 67.9%. Bien que le modèle combiné ait donné une meilleure valeur de FOM que la récompense sigmoïdale, nous pensons que cette dernière présente plus d'avantages qu'un modèle poubelle quelle que soit sa définition. En effet, avec une reconnaissance à base de boucle de phonèmes, on n'est plus restreint à un modèle poubelle bien déterminé, les mots hors-vocabulaire sont remplacés par des séquences de phonèmes qui seront ignorés dans la phase de détection.

Après avoir étudié l'utilisation des modèles poubelles pour la détection de mots clés, nous nous sommes penché sur la notion de mesure de confiance qui permet de rejeter les mots les moins fiables donnés en sortie du système de reconnaissance. Nous avons proposé trois mesures de confiance. Ces mesures sont calculées comme étant les moyennes arithmétique, géométrique et harmonique des probabilités d'observations acoustiques locales des phonèmes composant un mot clé reconnu. Ensuite, nous avons introduit la notion de durée dans nos calculs de mesures de confiance et nous avons établi trois autres mesures qui représentent une normalisation des trois précédentes. Il s'agit donc des moyennes arithmétique, géométrique et harmonique normalisées. La mesure la plus performante est celle calculée comme une moyenne harmonique normalisée des probabilités d'observations. Elle nous a fournit une valeur de FOM de l'ordre de 70.3%.

Visant toujours à améliorer nos résultats nous avons défini deux nouvelles mesures de confiance, un rapport et une distance de vraisemblance. L'idée sous-jacente à ces mesures est de considérer une double reconnaissance. La première est basée sur une grammaire composée de l'ensemble des mots clés et du modèle de silence, la deuxième est une reconnaissance à base de boucle de phonèmes. Pour calculer la mesure de confiance d'un mot, nous avons utilisé les probabilités d'observations au niveau le plus élémentaire, celui de la trame. Comme nous l'avons remarqué, l'introduction de la notion de normalisation a amélioré les résultats, nous avons donc opté à une normalisation de ces deux dernières mesures de confiances Les performances obtenues par cette normalisation ont prouvé l'efficacité de l'approche. Le meilleur résultat est fourni par la distance de vraisemblance normalisée qui est de l'ordre de 71.5% pour la mesure de FOM.

La complémentarité des approches à base de modèles poubelles et de mesures de confiance, nous a incité à étudier leur combinaison. Pour ce faire, le modèle poubelle est utilisé en premier lieu pour absorber un maximum de mots hors-vocabulaire. En second lieu, nous établissons une étape de vérification supplémentaire des mots clés reconnus afin de rejeter les mots les moins fiables et ce à l'aide d'une mesure de confiance. Dans notre cas, nous avons employé le modèle poubelle combiné et la mesure de confiance "moyenne harmonique normalisée". Ce modèle hybride a enregistré une amélioration par rapport aux deux approches qu'il utilise et il a présenté les meilleurs performances trouvées jusque là. Ce modèle atteint une valeur de FOM de l'ordre de 72.5% ce qui confirme l'idée "combiner pour améliorer".

Une autre forme de combinaison se présente en remarquant que nos trois types de mesures de confiance (moyennes arithmétique, géométrique et harmonique) exploitent différemment les mêmes informations à savoir, les probabilités d'observations acoustiques locales des phonèmes. Dans ce cas, les méthodes de combinaison les plus connues sont le vote majoritaire et la combinaison linéaire. Pour perfectionner cette combinaison, nous avons utilisé une méthode de classification PMC. Avec un PMC à 6 entrées, chacune correspondent à une mesure de confiance

(moyennes arithmétique, géométrique, harmonique et leurs normalisées respectives) et une seule sortie, nous avons obtenu une valeur de FOM de l'ordre de 71.6%. Les SVM ont montré leur efficacité dans plusieurs applications et ils constituent une méthode innovante dans le domaine de classification à apprentissage statistique, ils se basent sur le principe de minimisation du risque structurel qui a fait preuve d'une grande capacité de généralisation. Les SVM se présentent donc comme une autre alternative pour réaliser la combinaison des six mesures de confiance. Dans notre travail, nous avons considéré deux types de SVM, les SVM à fonction noyau linéaire et les SVM à fonction noyau RBF. En adoptant comme entrée de ces SVM les six mesures de confiance, nous avons pu atteindre les 75.4% comme valeur de FOM avec les SVM à noyau RBF.

Ces études nous ont mené à deux conclusions. Tout d'abord,la considération de plusieurs informations nous a permis d'améliorer les performances. Ensuite, l'utilisation des méthodes de classification PMC et SVM nous a été bénéfique. C'est pourquoi nous avons décidé d'utiliser ces deux classifieurs avec différentes représentations des mots comme approche originale pour la détection de mots clés.

Concernant les représentations des mots, nous avons exploité deux types d'informations, les probabilités d'observations acoustiques locales des phonèmes et le nombre de trames de chaque état dans chaque phonème composant le mot à classer. Nous avons proposé trois représentations : la première utilise les probabilités d'observations phonétiques, la deuxième est basée sur le nombre de trames par état et la troisième combine les deux précédentes en un seul vecteur mixte. Ces trois représentations ont été testées en utilisant le PMC, les SVM à noyau linéaire et les SVM à noyau RBF afin de classer les mots reconnus en deux classes, "correct" et "incorrect". La classes "correct" correspond à l'ensemble des mots clés réellement prononcés et la classe "incorrect" correspond aux mots clés insérés. Ce sont les SVM à noyau RBF qui ont donné les meilleurs résultats avec la représentation mixte. Nous obtenons alors la meilleure valeur de FOM qui est de l'ordre de 82.9%.

Dans le cas des deux classifieurs PMC et SVM, tous les mots clés ont été représentés par une seule classe, la classe "correct". En réalité, il s'agit de 20 mots clés différents donc il vaut mieux considérer 20 classes et non pas une seule. Pour cette raison, nous avons augmenté le nombre de neurones de sortie du PMC à 21 et au lieu des SVM biclasses, nous avons employé les SVM multi-classes avec 21 sorties. Les résultats obtenus par ces deux méthodes sont comparables à ceux obtenus par le PMC et les SVM biclasses, l'amélioration enregistrée étant relativement faible. Ceci est dû principalement au fait que les méthodes de classification utilisées gèrent un ensemble de 20 classes ce qui augmente le nombre substitutions et diminue par conséquent le nombre de mots clés corrects.

Dans cette thèse, nous avons présenté différentes contributions dans le domaine de la détection des mots clés. Nos contributions se répartissent en trois branches. Au niveau des modèles poubelles, nous avons proposé l'utilisation des GMM et des modèles hybrides ainsi que trois méthodes de récompense dans le cas d'une reconnaissance à base de boucle de phonèmes. Au niveau des mesures de confiance, notre apport a consisté en différentes mesures originales qui utilisent deux types d'informations, les probabilités des observations acoustiques et la durée des états des phonèmes. Enfin, nous avons présenté une nouvelle approche pour résoudre le problème de détection de mots clés. Il s'agit d'assimiler ce dernier à un problème de classification en utilisant les SVM. Comparé aux résultats obtenus par les modèles poubelles et les mesures de confiance, notre approche a donné les meilleures performances.

Perspectives

Nous envisageons un certain nombre de perspectives pour la poursuite de nos travaux.

Tout d'abord, au niveau des modèles poubelles, nous pensons à l'introduction de la notion de syllabe dans la construction de nos modèles. Nous envisageons aussi l'utilisation d'autres modèles poubelles qui dépendent du nombre de phonèmes dans les mots hors-vocabulaire. Nous aurons, dans ce cas, un modèle poubelle pour chaque ensemble de mots ayant le même nombre de phonèmes. Une autre idée, qui nous parait originale, consiste à créer des modèles poubelles par type de phonèmes au fur et à mesure de l'étape de reconnaissance.

Pour la reconnaissance à base de boucle de phonèmes, nous envisageons d'étudier d'autres types de fonctions de récompense permettant d'améliorer les performances. Enfin, nous estimons qu'il sera important d'essayer de combiner les modèles poubelles avec la méthode à base de fonctions de récompense.

Nous avons pu remarquer lors de l'utilisation des mesures de confiance une grande variation des performances obtenues en passant d'une mesure à une autre. C'est par exemple le cas des mesures à base de moyenne géométrique et de moyenne harmonique normalisée. Il serait donc profitable de rechercher d'autres mesures de confiance plus efficaces pour la détection de mots clés.

Tout au long de l'utilisation des classifieurs, nous avons constaté l'importance de deux facteurs dans l'amélioration des résultats : la représentation vectorielle des mots et le type de classifieur. Pour ces raisons, nous proposons de perfectionner les représentations des mots en ajoutant un maximum d'informations d'ordre acoustique et éventuellement linguistique. Il serait aussi rentable de rechercher d'autres fonctions noyaux pour les SVM, autres que la fonction linéaire et la fonction RBF et qui soient adaptées à la nature de nos données.

Une des perspectives les plus attrayantes consiste à intégrer notre système de détection dans une plate-forme réelle de dialogue oral homme machine. En effet, notre travail peut être exploité par plusieurs types d'applications réelles comme par exemple, pour la consultation orale de bases de données, ou pour l'indexation des documents audio et encore pour mener des recherches sur le Web en utilisant la parole. Pour cela, il nous faut envisager un certain nombre de points. La première étape à réaliser consiste à utiliser des bases d'évaluation contenant de la parole spontanée. La deuxième concerne l'augmentation du nombre des mots clés considérés.

Nous envisageons enfin d'aborder le domaine de la compréhension automatique de la parole, suite logique de nos travaux. Les systèmes de compréhension actuels élaborés dans notre équipe utilisent un ensemble de concepts sémantiques. Chaque concept regroupe une liste de mots clés partageant les mêmes propriétés sémantiques et exprimant un sens bien déterminé. Il serait donc

avantageux d'utiliser nos modèles pour détecter les mots clés des concepts sémantiques dans ce qui a été prononcé. Ceci nous permettra d'associer à chaque énoncé la liste de ses concepts sémantiques qui nous permettront de le comprendre dans une étape ultérieure.

Annexe

Description des bases de données SPEECHDAT

Base de données Speechdat(II) FDB-1000 du français (référence ELDA : S0061)

Cette base de données téléphonique du français a été conçue pour le développement et la validation de systèmes de reconnaissance de la parole. Elle comprend 48 occurrences (40 obligatoires et 8 optionnelles) pour 1 017 locuteurs différents, collectés à travers le réseau téléphonique fixe. Cette base a été produite par Matra Nortel Communications et sponsorisée par la Commission européenne (CEC DGXIII), sous le projet LE2-4001. 17 locuteurs ont été ajoutés au 1 000 locuteurs d'origine afin de correspondre aux besoins de la base. La base de données respecte les spécifications du projet SpeechDat(II). Elle contient principalement des fichiers de parole et leur transcription orthographique.

Les fichiers de parole sont stockés en séquences d'échantillons de 8 bit, 8 kHz, loi-A avec un en-tête de fichier de 16 bytes et ne sont pas compressés, selon les spécifications de SpeechDat.

Chaque énoncé est stocké dans un fichier séparé (extension de fichier FRA) et est accompagné d'un fichier d'étiquetage ASCII SAM (extension de fichier FRO).

Contenu du corpus :

- 5 mots de commande
- 1 séquence de 10 chiffres isolés
- 4 chiffres connectés : 1 numéro permettant d'identifier la feuille de prompt (5+ chiffres), 1 numéro de téléphone (9-11 chiffres), 1 numéro de carte de crédit (14-16 chiffres), 1 code confidentiel (6 chiffres)
- 3 dates : 1 date spontanée (ex. anniversaire), 1 date lue, 1 date relative et générale
- 2 expressions utilisant un mot de commande
- 1 chiffre isolé
- 3 mots épelés (séquences de lettres) : 1 spontané, ex. prénom du locuteur, 1 nom de ville, 1 mot réel/artificiel pour couverture
- 1 montant en devises
- 1 nombre entier naturel
- 5 noms provenant d'un annuaire de renseignements téléphoniques + 1 nom épelé : 1 spontané, ex. prénom du locuteur, 1 ville de naissance/d'origine du locuteur (spontané), 1 nom de ville parmi les plus fréquentes (ensemble de 500), 1 nom de compagnie parmi les plus fréquentes (ensemble de 500), 1 répétition de "prénom nom", 1 ville de naissance épelée
- 2 questions, incluant des oui/non "vagues" : 1 question à prédominance "oui", 1 question à prédominance "non"
- 9 phrases phonétiquement riches
- 2 phrases comportant une notion de temps : 1 jour (spontané), 1 phrase comportant une notion de temps
- 8 mots phonétiquement riches

Base de données SpeechDat(II) FDB-5000 du français (référence ELDA : S0076)

La base de données SpeechDat(II) FDB-5000 du français contient 5040 locuteurs français (2347 hommes, 2693 femmes) enregistrés à travers le réseau téléphonique fixe français. La base de données SpeechDat a été collectée et annotée par MATRA NORTEL COMMUNICATIONS. Cette base est répartie sur 18 CD-ROM, chacun comprenant 300 sessions (sauf pour le CD-ROM 4, qui comprend 100 sessions). Les bases de données SpeechDat(II) ont été réalisées selon les spécifications du projet SpeechDat(II) et validées par SPEX, Pays-Bas.

Les fichiers de parole sont stockés en séquences d'échantillons de 8 bit, 8 kHz, loi-A avec un en-tête de fichier de 16 bytes et ne sont pas compressés, selon les spécifications de SpeechDat.

Chaque énoncé est stocké dans un fichier séparé (extension de fichier FRA) et est accompagné d'un fichier d'étiquetage ASCII SAM (extension de fichier FRO).

Contenu du corpus :

- 5 mots de commande ;
- 1 séquence de 10 chiffres isolés ;
- 4 chiffres connectés : 1 numéro permettant d'identifier la feuille de prompt (5+ chiffres), 1 numéro de téléphone (9-11 chiffres), 1 numéro de carte de crédit (14-16 chiffres), 1 code confidentiel (6 chiffres) ;
- 3 dates : 1 date spontanée (ex. anniversaire), 1 date lue, 1 date générale ;
- 2 expressions utilisant un mot de commande ;
- 1 chiffre isolé ;·
- 3 mots épelés (séquences de lettres) : 1 spontané, ex. prénom du locuteur, 1 nom de ville provenant d'un annuaire de renseignements téléphoniques, 1 mot réel/artificiel pour couverture ;
- 1 montant en devises ;
- 1 nombre entier naturel ;
- 5 noms provenant d'un annuaire de renseignements téléphoniques + 1 nom épelé : 1 spontané, ex. prénom du locuteur, 1 ville de naissance/d'origine du locuteur (spontané), 1 nom de ville parmi les plus fréquentes (sur 500), 1 nom de compagnie parmi les plus fréquentes (sur 500), 1 répétition de "prénom nom", 1 ville de naissance épelée ;
- 2 questions incluant des oui/non "vagues" : 1 question à prédominance "oui", 1 question à prédominance "non" ;
- 9 phrases phonétiquement riches ;
- 2 phrases comportant une notion de temps : 1 jour (spontané), 1 phrase comportant une notion de temps ;
- 8 mots phonétiquement riches.

Les classes d'âge sont réparties comme suit : 215 locuteurs de moins de 16 ans, 2531 locuteurs entre 16 et 30 ans, 1208 locuteurs entre 31 et 45 ans, 910 locuteurs entre 46 et 60 ans et 176 locuteurs de plus de 60 ans.

Un lexique de prononciation avec sa transcription phonétique en SAMPA est également fourni.

Publications personnelles

Y. Ben Ayed, D. Fohr, J. P. Haton, G. Chollet, "Improving the Performance of a Keyword Spotting System by Using Support Vector Machines" IEEE Automatic Speech Recognition and Understanding Workshop. (ASRU 2003), 1-4 décembre 2003.

Y. Ben Ayed, D. Fohr, J. P. Haton, G. Chollet, "A New Keyword Spotting Approach Based on Reward Function" Seventh International Symposium on Signal Processing and Its Applications (ISSPA'2003), 1-4 juillet 2003.

Y. Benayed, D. Fohr, J. P. Haton, G. Chollet, "Confidence Measures for Keyword Spotting Using Support Vector Machines" IEEE International Conference on Acoustics, Speech and Signal Processing. (ICASSP'2003), 06-10 Avril 2003.

Y. Benayed, D. Fohr, J. P. Haton, G. Chollet, "Recognition and Rejection Performance in Wordspotting Systems Using Support Vector Machines" 2nd WSEAS International Conference on Signal, Speech and Processing, (WSEAS ICOSSIP 2002) , 25-28 septembre 2002.

Y. Benayed, D. Fohr, J. P. Haton, G. Chollet, "Keyword Spotting Using Support Vector Machines" Fifth International Conference on Text, Speech and Dialogue (TSD'2002), 9-12 septembre 2002.

Y. Ben Ayed, D. Fohr, J. P. Haton, G. Chollet, "Recognition Performance in Wordspotting Systems Using Hidden Markov modeling techniques" International Workshop speech and computer (SPECOM'2002) , 2-5 septembre 2002.

Y. Benayed, D. Fohr, J. P. Haton, G. Chollet, "Support Vector Machines for Keyword Spotting" International Workshop speech and computer (SPECOM'2002) , 2-5 septembre 2002.

Y. Ben Ayed, D. Fohr, J. P. Haton, G. Chollet, "Détection de Mots Clés dans un Flux de Parole par les Modèles de Markov Cachés" Rapport Interne, N. LORIA : A02-R-033, 2002.

Y. Benayed, D. Fohr, J. P. Haton, G. Chollet, C. Antoine "Détection de Mots Clés dans un Flux de Parole" Quatrièmes Rencontres des Jeunes Chercheurs en Parole (RJC'2001), Mons-Belgique Septembre 2001.

Bibliographie

[Aubert *et al.*, 1993] X. Aubert, R. HaebUmbach, et H. Ney. Improvement in connected digit recognition using linear discriminant analysis and mixture densities. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 648–651, 1993.

[Bahl *et al.*, 1983] L. R. Bahl, F. Jelinek, et R. L. Mercer. A maximum likelihood approach to continuous speech recognition. *IEEE Transactions Pattern Analysis and Machine Intelligence*, 5 :179–190, 1983.

[Bahl *et al.*, 1986] L. R. Bahl, P. F. Brown, P. V. Suza, et L. R. Mercer. Maximum mutual information estimation of hidden Markov model parameters for speech recognition. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 49–52, 1986.

[Bakis, 1976] R. Bakis. Continuous speech word recognition via centisecond acoustic states. In *91st Meeting of the Acoustical Society of America*, 1976.

[Bayya, 2000] A. Bayya. Rejection in speech recognition system with limited training. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, pages 305–308, 2000.

[Benayed, 1999] Y. Benayed. Réalisation et comparaison de différents types de réseaux de neurones artificiels pour la reconnaissance automatique de la parole. Master's thesis, Ecole Nationale d'Ingénieurs de Sfax, 1999.

[Bernardis et Bourlard, 1998] G. Bernardis et H. Bourlard. Confidence measures in hybrid HMM/ANN speech recognition. In *Proceedings of the First Workshop on Text, Speech, Dialogue*, pages 159–164, 1998.

[Bezie et Lockwood, 1993] O. Bezie et P. Lockwood. Beamsearch and traceback in the frame synchronous two level algorithm. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, pages 511–514, 1993.

[Blanz *et al.*, 1996] V. Blanz, B. Schölkopf, H. Bulthoff, C. Burges, V. Vapnik, et T. Vetter. Comparison of view-based object recognition algorithms using realistic 3D models. In *Proceedings of the International Conference on Artificial Neural Networks*, pages 251–256, 1996.

[Boite *et al.*, 1993] J. M. Boite, H Bourlard, B. D'hoore, et M. Haesen. A new approach towards keyword spotting. In *Proceedings of the European Conference On Speech Communication and Technology*, pages 1273–1276, 1993.

[Boite et Kunt, 1987] R. Boite et M. Kunt. *Traitement de la Parole*. Press Polytechniques Romandes, 1987.

[Boser *et al.*, 1992] B. E. Boser, I. Guyon, et V. Vapnik. A training algorithm for optimal margin classifiers. In *Computational Learing Theory*, pages 144–152, 1992.

[Bottou *et al.*, 1994] L. Bottou, C. Cortes, J. Drucker, I. Guyon, Y. LeCunn, U. Muller, E. Sackinger, P. Simard, et V. Vapnik. Comparaison of classifier methods : a case study in handwri-

ting digit recognition. In *Proceedings of the International Conference on Pattern Recognition*, pages 77–87, 1994.

[Bourlard *et al.*, 1994] H. Bourlard, B. D'hoore, et J. M. Boite. Optimizing recognition and rejection performance in wordspotting system. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 373–376, 1994.

[Bouwman et Hulstijn, 1998] G. Bouwman et J. Hulstijn. Dialogue strategy redesign with reliability measures. In *First International Conference on Language Resources and Evaluation*, pages 191–198, 1998.

[Bradley, 1997] A. P. Bradley. The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(6) :1145–1159, 1997.

[Burges, 1998] C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2) :121–167, 1998.

[Calliope, 1989] Calliope. *Traitement automatique de la parole*. Masson, 1989.

[Caminero *et al.*, 1997] J. Caminero, L. Hernandez-Gomez, C. de la Torre, et C. Martin. Improving utterance verification using hierarchical confidence measures in continuous natural numbers recognition. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 891–894, 1997.

[Cardin *et al.*, 1991] R. Cardin, Y. Normandin, et R. De Mori. High performance connected digit recognition using maximal mutual information estimation. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 533–536, 1991.

[Cercadillo et Gomez, 1993] J. A. Cercadillo et A. H. Gomez. Grammar learning and word spotting using recurrent neural networks. In *Proceedings of the European Conference On Speech Communication and Technology*, pages 21–23, 1993.

[Chapelle *et al.*, 2002] O. Chapelle, V. Vapnik, O. Bousquet, et S. Mukherjee. Choosing multiple parameters for support vector machines. *Machine Learning*, 46(1-3) :131–159, 2002.

[Charlet *et al.*, 2001] D. Charlet, G. Mercier, et D. Jouvet. On combining confidence measures for improved rejection of incorrect data. In *Proceedings of the European Conference On Speech Communication and Technology*, 2001.

[Chigier, 1992] B. Chigier. Rejection and keyword spotting algorithms for a directory assistance city name recognition application. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 93–96, 1992.

[Chou *et al.*, 1992] W. Chou, B. H. Juang, et C. H. Lee. Segmented GPD training of HMM based speech recognizer. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, pages 473–476, 1992.

[Ciarlet, 1994] P. Ciarlet. *Introduction à l'analyse numérique matricielle et l'optimisation*. Masson, 1994.

[Clary et Hansen, 1992] G.J. Clary et J.H.L. Hansen. A novel speech recognizer for keyword spotting. In *Proceedings of the International Conference on Spoken Language Processing*, pages 13–16, 1992.

[Cole *et al.*, 1995] R. Cole, L. Hirschman, L. Atlas, M. Beckman, A. Bierman, M. Bush, M. Clements, J. Cohen, O. Garcia, B. Hanson, H. Hermansky, S. Levinson, K. Mckeown, N. Morgan, D. G. Novick, M. Ostendorf, S. Oviatt, C. Weinstein, S. Zahorian, et V. Zue. The challenge of spoken language system : research directions for the nineties. *IEEE Transactions on Speech and Audio Processing*, 1 :1–21, 1995.

[Cortes et Vapnik, 1995] C. Cortes et V. Vapnik. Support-vector networks. *Machine Learning*, 20(3) :273–297, 1995.

[Cox et Rose, 1996] S. Cox et R. C. Rose. Confidence measures for the switchboard database. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 511–514, 1996.

[Davis et Mermelstein, 1980] S. B. Davis et P. Mermelstein. Comparaison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 28(4) :357–366, 1980.

[Deroo, 1998] O. Deroo. *Modèles dépendants du contexte et méthodes de fusion de données appliqués à la reconnaissance de la parole par modèles hybrides HMM/MLP*. PhD thesis, Faculté Polytechnique de Mons, 1998.

[Egan, 1975] J. P. Egan. *Signal detection theory and ROC analysis*. Academic Press, 1975.

[Eide *et al.*, 1995] E. Eide, H. Gish, P. Jeanrenaud, et A. Mielke. Understanding and improving speech recognition performance through the use of diagnostic tools. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, pages 221–224, 1995.

[Fohr *et al.*, 2000] D. Fohr, O. Mella, et C. Antoine. The automatic speech recognition engine espere experiments on telephone speech. In *Proceedings of the International Conference on Spoken Language Processing*, pages 246–250, 2000.

[Fontaine *et al.*, 1997] V. Fontaine, C. Ris, et J. Boite. Non linear discriminant analysis for improved speech recognition. In *Proceedings of the European Conference On Speech Communication and Technology*, pages 2071–2075, 1997.

[Foote *et al.*, 1995] J. T. Foote, G. J. F. Jones, K. Spärck Jones, et S. J. Young. Talker-independent keyword spotting for information retrieval. In *Proceedings of the European Conference On Speeech Communication and Technology*, pages 2145–2148, 1995.

[Friedman, 1996] J. Friedman. Another approch to polychotomous classification. Rapport technique, Departement of Statistics Stanford University, 1996.

[Furey *et al.*, 2000] T. S. Furey, N. Christianini, N. Duffy, D. W. Bednarski, M. Schummer, et D. Hauessler. Support vector machine classification and validation of cancer tissue samples using microarray expression data. *Bioinformatics*, 16(10) :906–914, 2000.

[Furui, 1981] S. Furui. Cepstral analysis technique for automatic speaker verification. *IEEE Transactions on Acoustics Speech and Audio Processing*, 29 :254–272, 1981.

[Ganapathiraju, 2002] A. Ganapathiraju. *support vector machines for speech recognition*. PhD thesis, Faculty of Mississipi State University, 2002.

[Garcia *et al.*, 1998] P. Garcia, J. C. Segura, J. Diaz, A. J. Rubio, M. C. Benitez, V. Sanchez, A. M. Peinado, J. M. Lopez, J. L. Perez, A . De la Torre, et R. Lopez-cozar. Speech recognition and new technologies in communications : a continuous speech recognition-based switchboard with an answering module for e-mailing voice messages. In *Proceedings of the first International Conference on Language Resources and Evaluation*, pages 1263–1266, 1998.

[Gauvain et Lee, 1994] J. L. Gauvain et C. Lee. Maximum a-posteriori estimation for multivariate gaussian mixture observations of markov chains. *IEEE Transactions on Speech and Audio Processing*, 2(2) :291–298, 1994.

[Gelin et Wellekens, 1996] P. Gelin et C. Wellekens. Keyword spotting for video soundtrack indexing. In *Proceedings of the International Conference on Spoken Language Processing*, pages 299–302, 1996.

[Gelin, 1997] P. Gelin. *Détection de mots clés dans un flus de parole : Application à l'indexation de documents multimédia*. PhD thesis, Ecole Polytechnique Fédérale de Lausanne, 1997.

[George *et al.*, 1994] M. George, P. Jospa, et A. Soquet. Articulatory trajectories generated by the control of the vocal tract by a neural network. In *Proceedings of the International Conference on Spoken Language Processing*, pages 583–586, 1994.

[Glorennec, 1993] P. Y. Glorennec. A general class of fuzzy inference systems application to identification and control. In *Proceedings of the European Conference On Speech Communication and Technology*, 1993.

[Godin et Lockwood, 1989] C. Godin et P. Lockwood. DTW schemes for continuous speech recognition : a unified view. *Computer Speech and Language*, 3 :169–198, 1989.

[Gorin *et al.*, 1997] A. L. Gorin, G. Riccardi, et J. H. Wright. How May I Help You ? *Speech Communication*, 23 :113–127, 1997.

[Guermeur *et al.*, 1999] Y. Guermeur, A. Elisseeff, et H. Paugam-Moisy. Estimating the sample complexity of a multi-class discriminant model. In *Proceedings of the International Conference on Artificiel Neural Networks*, pages 310–315, 1999.

[Gunn *et al.*, 1997] S. R. Gunn, M. Brown, et K. M. Bossely. Network performance assessment for neurofuzzy data modelling. *Intelligent Data Analysis*, 1208 :313–323, 1997.

[Gupta et Soong, 1998] S. Gupta et F. K. Soong. Improved utterance rejection using length dependent thresholds. In *Proceedings of the International Conference on Spoken Language Processing*, pages 795–798, 1998.

[Halberstadt et Glass, 1998] A. Halberstadt et J. Glass. Heterogeneous measurements and multiple classifiers for speech recognition. In *Proceedings of the International Conference on Spoken Language Processing*, pages 995–998, 1998.

[Haton *et al.*, 1991] J. P. Haton, J. M. Pierrel, G. Perennou, J. Caelen, et J. L. Gauvain. *Reconnaissance automatique de la Parole*. dunod, 1991.

[Hazen et Bazzi, 2001] T. J. Hazen et I. Bazzi. A comparaison and combination of methods for word detection and word confidence scoring. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, pages 1096–1099, 2001.

[Hernandez-Abrego et Marino, 2000] G. Hernandez-Abrego et B. Marino. Contextual confidence measures for continuous speech recognition. In *Proceedings of International Conference on Acoustics, Speech, and Signal Processing*, pages 1803–1806, 2000.

[Higgins et Wohlford, 1985] A. L. Higgins et R. E. Wohlford. Keyword recognition using template concatenation. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 1233–1236, 1985.

[Hornik *et al.*, 1989] K. Hornik, M. Stinchcombe, et H. White. Multilayer feedforward nets are universal approximators. *Neural Networks*, 2 :359–366, 1989.

[Hsu et Lin, 2001] C. Hsu et C. Lin. A comparison of methods for multi-class support vector machines. Rapport technique, Department of Computer Science and Information Engineering, National Taiwan University, 2001.

[James et Young, 1994] D. A. James et S. J. Young. A fast lattice-based approach to vocabulary independent wordspotting. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 377–380, 1994.

[Jiang et Lee, 2003] H. Jiang et C. H. Lee. A new approach to utterance verification based on neighborhood information in model space. *IEEE Transactions on Speech and Audio Processing*, 11 :425–434, 2003.

[Joachims, 1998] T. Joachims. Text categorization with support vector machines : learning with many relevant features. In *10th European Conference on Machine Learning*, pages 137–142, 1998.

[Jones *et al.*, 1995] G. J. F. Jones, J. T. Foote, K. Sparck Jones, et S. J. Young. Video mail retrieval : the effect of word spotting accuracy on precision. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, pages 309–312, 1995.

[Jouvet et Monné, 1999] D. Jouvet et J. Monné. Recognition of spelled names over the telephone and rejection of data out of the spelling lexicon. In *Proceedings of the European Conference On Speech Communication and Technology*, pages 283–286, 1999.

[Juang, 1985] B. H. Juang. Maximum likelihood estimation for mixture multivariate stochastic observations of markov chains. *ATT Technical Journal*, 64 :1235–1246, 1985.

[Kamppari et Hazen, 2000] S. O. Kamppari et T. J. Hazen. Word and phone level acoustic confidence scoring. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, pages 1799–1802, 2000.

[Kapadia *et al.*, 1993] S. Kapadia, V. Valtchev, et S. J. Young. MMI training for continuous phoneme recognition on the TIMIT database. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 491–494, 1993.

[Kemp et Waibel, 1998] T. Kemp et A. Waibel. Unsupervised training of a speech recognizer using TV broadcasts. In *Proceedings of the International Conference on Spoken Language Processing*, pages 2207–2210, 1998.

[Kharroubi *et al.*, 2001] J. Kharroubi, D. Petrovska, et G. Chollet. Combining gmm's with support vector machines for text independent speaker verification. In *Proceedings of the European Conference on Speech Communication and Technology*, pages 1761–1764, 2001.

[Kharroubi, 2002] J. Kharroubi. *Etude de techniques de classement "machines à vecteurs supports" pour la vérification automatique du locuteur*. PhD thesis, Ecole Nationale Supérieure des Télécommunications de Paris, 2002.

[Klatt, 1977] D. H. Klatt. Review of the arpa speech understanding project. *J. Acoust. Soc. Am*, 62 :1345–1366, 1977.

[Klemm *et al.*, 1995] H. Klemm, U. Kilian, et F. Class. Word- and phrase spotting with syllable-based garbage modelling. In *Proceedings of the European Conference On Speech Communication and Technology*, pages 2157–2160, 1995.

[Knerr *et al.*, 1990] S. Knerr, L. Personnaz, et G. Dreyfus. Single-layer learning revisited : a stepwise procedure for building and training a neural network. *Neurocomputing : Algoritms, Architectures and Applications*, 68, 1990.

[Koreman *et al.*, 1999] J. Koreman, B. Andreeva, et H. Strik. Acoustic parameters versus phonetic features in ASR. In *Proceedings of the International Congress of Phonetic Sciences*, pages 549–553, 1999.

[Krebel, 1998] U. Krebel. *Advances in kernel methods support vector learning*. MIT Press, 1998.

[Lai et Shi, 2001] C. S. Lai et B. E. Shi. A one-pass strategy for keyword spotting and verification. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, pages 2172–2175, 2001.

[Li *et al.*, 1995] H. Li, Y. Gong, et J.P.Haton. EM algorithms for probabilistic mapping networks. Rapport technique, INRIA/Lorraine, 1995.

[Lippmann *et al.*, 1994] R. P. Lippmann, E. I. Chang, , et C. R. Jankowski. Wordspotter trainning using figure-of-merit back propagation. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, pages 389–392, 1994.

[Lleida *et al.*, 1993] E. Lleida, J. B. Marino, J. Salavedra, A. Bonafonte, E. Monte, et A. Martinez. Out-of-vocabulary word modelling and rejection for keyword spotting. In *Proceedings of the European Conference On Speech Communication and Technology*, pages 1265–1268, 1993.

[Maison et Gopinath, 2001] B. Maison et R. Gopinath. Robust confidence annotation and rejection for continuous speech rejection. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, pages 2300–2303, 2001.

[Mangasarian et Musicant, 2001] O. L. Mangasarian et D. R. Musicant. Lagrangian support vector machines. *Journal of Machine Learning Research*, 1 :161–177, 2001.

[Marcus, 1992] J. A. Marcus. A novel algorithm for HMM words spotting performance evaluation and error analysis. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 89–92, 1992.

[Mathan et Miclet, 1991] L. Mathan et L. Miclet. Rejection of extraneous input speech recognition applications using multi-layer perceptrons and the trace of HMM. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, pages 93–96, 1991.

[Mathan, 1991] L. Mathan. *Contributions à la reconnaissance de la parole pour des serveurs vocaux interactifs*. PhD thesis, Ecole Nationale Supérieure des Télécommunications, 1991.

[Mauuary, 1994] L. Mauuary. *Amélioration des performances des serveurs vocaux interactifs*. PhD thesis, Université de Rennes I, 1994.

[Mazor et Feng, 1993] B. Mazor et M. W. Feng. Improved a posteriori processing for keyword spotting. In *Proceedings of the European Conference On Speech Communication and Technology*, pages 2231–2234, 1993.

[Meliani et O'Shaughnessy, 1998] R. El Meliani et D. O'Shaughnessy. Specific language modelling for new-word detection in continuous-speech recognition. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 321–325, 1998.

[Moreau *et al.*, 2000] N. Moreau, D. Charlet, et D. Jouvet. Confidence measure and incremental adaptation for the rejection of incorrect data. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, 2000.

[Moreau et Jouvet, 1999] N. Moreau et D. Jouvet. Use of a confidence mesure based on frame level likelihood ratios for the rejection of incorrect data. In *Proceedings of the European Conference On Speech Communication and Technology*, pages 291–294, 1999.

[Moreau et Jouvet, 2000] N. Moreau et D. Jouvet. Détermination d'une mesure de confiance pour le rejet des entrées incorrectes. In *XXIIIèmes Journées d'Etudes sur la Parole*, pages 173–176, 2000.

[Moreno *et al.*, 2001] P. J. Moreno, B. Logan, et B. Raj. A boosting approach for confidence scoring. In *Proceedings of the European Conference On Speech Communication and Technology*, pages 2109–2112, 2001.

[Morgan *et al.*, 1990] D. P. Morgan, C. L. Scofield, T. M. Lorenzo, E. C. Real, et D. P Loconto. A keyword spotter which incorporates neural networks for secondary processing. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 113–116, 1990.

[Morgan *et al.*, 1991] D. Morgan, C. Scofield, et J. Adcock. Multiple neural network topologies applied to keyword spotting. In *Proceedings of International Conference on Acoustics, Speech, and Signal Processing*, pages 313–316, 1991.

[Mukherjee *et al.*, 1997] S. Mukherjee, E. Osuna, et F. Girosi. Nonlinear prediction of chaotic time series using support vector machines. In J. Principe, L. Giles, N. Morgan, et E. Wilson, editors, *International Workshop on Neural Networks for Signal Processing*, pages 511–515, 1997.

[Muller *et al.*, 2001] K. R. Muller, S. Mika, G. Ratsch, K. Tsuda, et B. Schölkopf. An introduction to kernel based learning algorithms. *Transactions on Neural Networks*, 12, 2001.

[Myers *et al.*, 1980] C. S. Myers, L. R. Rabiner, et A. E. Rosenberg. An investigation of the use of dynamic time warping for word spotting and connected speech recognition. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 173–177, 1980.

[Myers et Rabiner, 1981] C. S. Myers et L. R. Rabiner. A dynamic time-warping algorithm for connected word recognition. *IEEE Transations on Acoustics, Speech, and Signal Processing*, 29 :351–363, 1981.

[Neti *et al.*, 1997] C. Neti, S. Roukos, et E. Eide. Word-based confidence measures as a guide for stack search in speech recognition. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, pages 883–886, 1997.

[Osuna *et al.*, 1997a] E. Osuna, R. Freund, et F. Girosi. Support vector machines : Training and applications. Rapport technique AIM-1602, Center for Biological and Computational Learning, Massachusetts Institue on Technology, Cambridge, 1997.

[Osuna *et al.*, 1997b] E. Osuna, R. Freund, et F. Girosi. Training support vector machines : an application to face detection. In *Proceedings of the International Conference on computer Vision and Pattern Recognition*, pages 130–136, 1997.

[Palmer et Ostendorf, 2001] D. D. Palmer et M. Ostendorf. Improved word confidence estimation using long range features. In *Proceedings of the European Conference On Speech Communication and Technology*, 2001.

[Platt *et al.*, 2000] J. C. Platt, N. Cristianini, et J. Shawe-Taylor. Large margin dags for multiclass classification. *In Advances in Neural Information Processing Systems, MIT Press*, 12 :547–553, 2000.

[Pols, 1997] L. C. W. Pols. Flexible human speech recognition. In *Proceedings of the Workshop on Automatic Speech Recognition and Understanding*, pages 273–283, 1997.

[Pontil et Verri, 1997] M. Pontil et A. Verri. Properties of support vector machines. Rapport technique, INFM, Département de physique, Université de Genève, 1997.

[Rabiner et Juang, 1989] L. Rabiner et B. H. Juang. A tutorial on hidden markov models and selected application in speech recognition. *Proceeding of IEEE*, 77 :257–285, 1989.

[Rabiner et Juang, 1993] L. Rabiner et B. H. Juang. *Fundamentals of Speech Recognition*. Prentice-Hall, 1993.

[Rabiner et Schmidt, 1980] L. R. Rabiner et C. E. Schmidt. Application of dynamic time-warping to connected-digit recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 28 :337–388, 1980.

[Ramalingam *et al.*, 1999] C. S. Ramalingam, Y. Gong, L. P. Netsch, W. Anderson, J. Godfrey, et Y. Kao. Speaker-dependent name dialing in a car environment with out-of-vocabulary

rejection. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, pages 165–168, 1999.

[Reynolds, 1994] D.A. Reynolds. Speaker identification and verification using gaussian mixture models. In *In Workshop on Automatic Speaker Recognition and Verification*, pages 27–30, 1994.

[Riccardi *et al.*, 1996] G. Riccardi, R. Pieraccini, et E. Bocchieri. Stochasic automata for language modeling. *Computer Speech and Language*, 10 :265–293, 1996.

[Riccardi *et al.*, 1997] G. Riccardi, A. L. Gorin, A. Ljolje, et M. Riley. A spoken language system for automated routing. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 1143–1146, 1997.

[Rijsbergen, 1979] C. Van Rijsbergen. *Information Retrieval.* Dept. of Computer Science, University of Glasgow, 1979.

[Rivlin *et al.*, 1996] Z. Rivlin, M. Cohen, V. Abrash, et T. Chung. A phone-dependent confidence measure for utterance rejection. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, pages 515–518, 1996.

[Rohlicek *et al.*, 1989] J. R. Rohlicek, W. Russell, S. Roukos, et H. Gish. Continuous hidden markov modeling for speaker independent word spotting. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 627–630, 1989.

[Rohlicek *et al.*, 1993] J. R. Rohlicek, P. Jeanrenaud, K. Ng, H. Gish, B. Musicus, et M. Shiu. Phonetic training and language modeling for word spotting. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 459–462, 1993.

[Rose *et al.*, 1998] R. C. Rose, H. Yao, G. Ricardi, et J. Wright. Integration of utterance verification with statistical language modeling and spoken language understanding. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 237–240, 1998.

[Rose et Hofstetter, 1993] R. C. Rose et E. M. Hofstetter. Task independent wordspotting using decision tree based allophone clustering. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 467–470, 1993.

[Rose et Paul, 1990] R. C. Rose et D. B. Paul. A hidden markov model based keyword recognition system. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 129–132, 1990.

[Rose, 1992] R. C. Rose. Discriminant wordspotting techniques for rejection of non-vocabulary utterances in unconstrained speech. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 105–108, 1992.

[Rose, 1995] R. C. Rose. Keyword detection in conversational speech utterances using hidden markov model based continuous speech recognition. *Computer, Speech and Language*, 9 :309–333, 1995.

[Roxane, 1995] L. Roxane. *Au sujet des algorithmes de recherche des systèmes de reconnaissance de la parole à grands vocabulaires.* PhD thesis, Computer Science Université McGill, Montréal, 1995.

[Ruske et Schotola, 1991] G. Ruske et T. Schotola. The effiency of demi-syllable segmentation in the recognition of spoken words. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 701–704, 1991.

[Schölkopf *et al.*, 1996] B. Schölkopf, O. Burges, et V. Vapnik. Incorporating invariances in support vector learning machines. In *Proceedings of the International Conference on Articial Neural Networks*, volume 1112, pages 47–52. Springer Lecture Notes in Computer Science, 1996.

[Schölkopf, 2000] B. Schölkopf. Statistical learning and kernel methods. Rapport technique, Microsoft Research, 2000.

[Schmidt *et al.*, 1997] M. Schmidt, J. Golden, et H. Gish. Gmm sample statistic log-likelihood for text independent speaker recognition. In *Proceedings of the European Conference on Speech Communication and Technology*, 1997.

[Sebastiani, 2002] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1), 2002.

[Silaghi et Bourlard, 1999] M.C. Silaghi et H. Bourlard. Posterior-based keyword spotting approches without filler models. Rapport technique, Laboratoire d'intelligence artificielle, département informatique, EPFL Lausanne, 1999.

[Silaghi et Bourlard, 2000] M.C. Silaghi et H. Bourlard. A new keyword spotting approach based on iterative dynamic programming. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, pages 1966–1969, 2000.

[Sukkar et Lee, 1996] R. A. Sukkar et C.-H. Lee. Vocabulary independent discriminative utterance verification for nonkeyword rejection in subword based speech recognition. *IEEE Transactions on Speech and Audio Processing*, 4 :420–429, 1996.

[Sukkar et Wilpon, 1993] R. A. Sukkar et J. G. Wilpon. A two pass classifier for utterance rejection in keyword spotting. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 451–454, 1993.

[Sukkar, 1994] R. A. Sukkar. Rejection for connected digit recognition based on GPD segmental discrimination. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 393–396, 1994.

[Tadj, 1995] C. Tadj. *méthodes connexionnistes de quantification vectorielle à apprentissage compétitif. Application à la détection de mots clé.* PhD thesis, Ecole Nationale Supérieure des Télécommunications, 1995.

[Torre et Acero, 1994] C. Torre et A. Acero. Discriminative training of garbage model for non-vocabulary utterance rejection. In *Proceedings of the International Conference on Spoken Language Processing*, pages 475–478, 1994.

[Vapnik, 1995] V. Vapnik. *The nature of statistical learning theory.* Springer Verlag, New York, 1995.

[Vapnik, 1998] V. Vapnik. *Statistical learning theory.* John Wiley and Sons, 1998.

[Vergyri, 2000] D. Vergyri. Use of word level side information to improve speech recognition. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, pages 1823–1826, 2000.

[Verlinde *et al.*, 2000] P. Verlinde, G. Chollet, et M. Acheroy. Multi-modal identity verification using expert fusion. *Information Fusion*, 1(1) :17–33, 2000.

[Weintraub *et al.*, 1997] M. Weintraub, F. Beaufays, Z. Rivlin, Y. Konig, et A. Stolcke. Neural-network based measures of confidence for word recognition. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 887–890, 1997.

[Wessel *et al.*, 1999] F. Wessel, K. Macherey, et H. Ney. A comparison of word graph and n-best list based confidence measures. In *Proceedings of the European Conference On Speech Communication and Technology*, pages 315–318, 1999.

[Wessel *et al.*, 2001] F. Wessel, R. Schliter, K. Macherey, et H. Ney. Confidence measure for large vocabulary continuous speech recognition. In *IEEE Transactions on speech and audio processing*, pages 288–298, 2001.

[Weston et Watkins, 1998] J. Weston et C. Watkins. Multi-class support vector machines. Rapport technique, CSD-TR-98-04, Department of Computer Science, Royal Holloway, University of London, Egham, TW20 0EX, UK, 1998.

[Weston et Watkins, 1999] J. Weston et C. Watkins. Support vector machines for multiclass pattern recognition. In *Proceedings of the Seventh European Symposium On Artificial Neural Networks*, 1999.

[Wilcox et Bush, 1991] L. D. Wilcox et M. A. Bush. HMM based wordspotting for voice editing and indexing. In *Proceedings of the European Conference On Speeech Communication and Technology*, pages 25–28, 1991.

[Williams et Renals, 1997] G. Williams et S. Renals. Confidence measures for hybrid HMM/ANN speech recognition. In *Proceedings of the European Conference On Speech Communication and Technology*, pages 1955–1958, 1997.

[Williams, 1999] D. A. G. Williams. *knowing what you don't know : roles for confidence measures in automatic speech recognition.* PhD thesis, Departement of computer science University of Sheffield, 1999.

[Wilpon *et al.*, 1990] J. G. Wilpon, L. R. Rabiner, C. Lee, et E. R. Glodman. Automatic recognition of keywords in unconstrained speech using hidden Markov models. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 38 :1870–1878, 1990.

[Yapanel, 1997] U. Yapanel. *Garbage modeling techniques for a turkish keyword spotting system.* PhD thesis, Istanbul Teknik university, 1997.

[Zeppenfeld et Waibel, 1992] T. Zeppenfeld et A. Waibel. A hybrid neural network, dynamic programming word spotter. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, pages 77–80, 1992.

[Zhang *et al.*, 2001] Y. Zhang, R. Lee, et A. Madievski. Confidence measure estimation for large vocabulary speaker independent continuous speech recognition system. In *Proceedings of the European Conference On Speech Communication and Technology*, 2001.

[Zhang et Rudnicky, 2001] R. Zhang et A. I. Rudnicky. Word level confidence annotation using combinations of features. In *Proceedings of the European Conference On Speech Communication and Technology*, 2001.

[Zue *et al.*, 1997] V. Zue, S. Seneff, J. Glass, L. Hetherington, E. Hurley, H. Meng, C. Pao, J. Polifroni, R. Schloming, et P. Schmid. From interface to content : translingual access and delivery of one-line information. In *Proceedings of the European Conference On Speech Communication and Technology*, pages 2227–2230, 1997.

[Zwicker et Feldtkeller, 1981] E. Zwicker et R. Feldtkeller. *Psychoacoustique : L'oreille, récepteur d'information.* Masson, 1981.