Université de Nancy I
U.F.R. Sciences et Techniques Mathématiques, Informatique et Automatique
Centre de Recherche en Informatique de Nancy
Institut National de Recherche en Informatique et Automatique

LA RECONNAISSANCE AUTOMATIQUE DE LA PAROLE ET LES MODELES MARKOVIENS CACHES

modèles du second ordre et distance de Viterbi à optimalité locale

THÈSE

pour l'obtention du

Doctorat de l'Université de Nancy I

(Spécialité Informatique)

par

Abdelaziz KRIOUILE

Soutenue le 23 Octobre 1990

Composition du jury :

Président : J.-M. PIERREL

Rapporteurs : G. PERENNOU
R. SCHOTT

Examinateurs : J.-P. HATON
J.-F. MARI
M. NAJIM

Université de Nancy I
U.F.R. Sciences et Techniques Mathématiques, Informatique et Automatique
Centre de Recherche en Informatique de Nancy — **Institut National de Recherche en Informatique et Automatique**

LA RECONNAISSANCE AUTOMATIQUE DE LA PAROLE ET LES MODELES MARKOVIENS CACHES

modèles du second ordre et distance de Viterbi à optimalité locale

THÈSE

pour l'obtention du

Doctorat de l'Université de Nancy I

(Spécialité Informatique)

par

Abdelaziz KRIOUILE

Soutenue le 23 Octobre 1990

Composition du jury :

Président : J.-M. PIERREL

Rapporteurs : G. PERENNOU
R. SCHOTT

Examinateurs : J.-P. HATON
J.-F. MARI
M. NAJIM

A travers ces quelques lignes, je tiens à remercier vivement tous les membres du jury :

Jean-Paul HATON, Professeur à l'Université de Nancy I et Directeur de recherches INRIA, qui a bien voulu m'accuellir au sein de son équipe et diriger ce travail. Sa connaissance, sa rigueur scientifique, son esprit critique, et son enthousiasme, m'ont aidé à acquérir une intéressante formation de chercheur. Je lui exprime ma sincère gratitude.

Jean-François MARI, Maître de conférence à l'Université de Nancy II, pour la bienveillante attention avec laquelle il a suivi mon travail. Je lui suis reconnaissant pour les conseils qu'il m'a prodigués tout au long de ce travail, et de m'avoir initié aux techniques des modèles markoviens cachés avec tant de gentillesse.

Jean-Marie PIERREL, Professeur à l'Université de Nancy I, qui a volontiers accepté de juger cette thèse et de présider son jury.

Mohamed NAJIM, Professeur à l'Université de Bordeaux I, qui m'a fait l'honneur de se déplacer spécialement pour siéger dans le jury de cette thèse.

René SCHOTT, Professeur à l'Université de Nancy I, qui a consacré beaucoup de temps pour être rapporteur de cette thèse. Il a été un lecteur critique, qui a su mettre l'accent sur les points à éclaircir.

Guy PERENNOU, Professeur à l'Université Paul Sabatier de Toulouse, qui a bien voulu être rapporteur de ce travail, et de se déplacer spécialement pour faire partie du jury. Je le remercie sincèrement pour l'enthousiasme avec lequel il m'a fait part de ses remarques.

Cette thèse a été réalisée au sein de l'équipe RFIA du CRIN et je tiens à remercier tous ceux qui de près ou de loin ont facilité ce travail. Je remercie également tous les membres de l'équipe en particulier Dominique FOHR, Mahieddine DJOUDI et Belgacem YAHIAOUI pour leur amical soutien. Mes remerciements s'adressent à Noureddine LAZRAK, Mohamed ELKHOMSI, Abdellatif MEZRIOUI ainsi que tous les membres du laboratoire CRIN pour leur amical appui et la création d'une agréable ambiance de travail.

En fin que tous mes proches trouvent ici le témoignage de ma plus sincère gratitude et sentiment pour leur soutien et appui ainsi que leurs encougements de tous moments.

Résumé

Des travaux intensifs sur la reconnaissance automatique de la parole utilisant les modèles stochastiques ont été réalisés durant les cinq dernières années. L'application des modèles markoviens cachés (HMM) du premier ordre a conduit à des résultats impréssionnants dans le domaine de la reconnaissance de mots isolés et de la parole continue.

Notre objectif était de montrer que l'apport des modèles markoviens cachés à la reconnaissance automatique de la parole est d'autant plus important qu'on mène des réflexions fondamentales sur les modèles markoviens eux même et sur la façon de les appliquer.

Notre travail consiste en deux améliorations significatives des modèles markoviens :

- La modélisation des HMMs du second ordre.
- Une nouvelle distance de Viterbi à optimalité locale.

Nous avons développé une nouvelle formulation de l'algorithme de Baum-Welch et une extension de l'algorithme de Viterbi, qui rendent les modèles markoviens cachés du second ordre efficaces en calcul pour des applications en temps réel. Nous avons pu montrer, à travers les résultats d'un reconnaisseur de mots isolés, qu'il y avait une nette amélioration du taux de reconnaissance avec le second ordre par rapport au premier ordre. L'extension à des HMMs d'ordre plus élevé a été aussi discutée.

Enfin, nous avons proposé une nouvelle stratégie d'utilisation de l'algorithme de Viterbi dans la phase de reconnaissance de la parole continue. Elle est basée sur une procédure de comparaison d'optimums locaux calculés par l'algorithme de Viterbi dans une fenêtre de trames de largeur fixe et à distance fixe pour les différents modèles HMM. Cette stratégie, par bloc, a donné de meilleurs résultats que les versions classiques de l'algorithme de Viterbi. Elle laisse entrevoir des résultats encourageants et permet une forte interaction avec d'autres processeurs acoustiques pour confirmer ou infirmer un segment décodé.

Mots clés :

Reconnaissance automatique de la parole
Modèles markoviens cachés du premier et du second ordre
Algorithme de Viterbi
Algorithme Baum-Welch

Table des matières

Introduction

Cadre du présent travail

Un des buts de la recherche actuelle en informatique est de rendre plus facile la communication homme-machine. En particulier, elle s'intéresse au développement de nouveaux modes de communication : langage naturel, écriture, vision, parole,.... La parole est sans aucun doute le mode de communication le plus naturel, le plus spontané et le plus direct pour l'homme.

Dès lors, l'intérêt des recherches en reconnaissance automatique de la parole s'impose. C'est dans ce cadre que s'insère notre présent travail : les modèles markoviens cachés et leur application à la reconnaissance automatique de la parole..

Les modèles markoviens cachés sont des processus stochastiques simples en application, riches en propriétés et fondés sur des bases mathématiques solides. C'est pourquoi, ils permettent de modéliser plusieurs phénomènes physiques avec grand succès. Ce qui explique leur dominance dans les systèmes actuels de la reconnaissance automatique de la parole (R.A.P.). Ils sont appliqués dans ces systèmes de reconnaissance de la parole suivant deux axes principaux :

- L'utilisation directe des algorithmes existants, en améliorant le taux de reconnaissance par l'introduction de contraintes concernant le champ de l'application du système.
- L'amélioration des algorithmes existants, en introduisant des modifications ou des variantes par rapport à leur aspect d'origine, dans le but de mieux adapter leur application au domaine de la parole.

Notre contribution s'inscrit dans le deuxième axe. Elle concerne :

- La formulation et le développement du passage au second ordre des modèles markoviens cachés.
- La proposition d'une stratégie d'utilisation de l'algorithme de reconnaissance pour la parole continue.

Présentation de ce mémoire

Après cette introduction, le premier chapitre expose la nature des problèmes de la modélisation par les modèles markoviens cachés et leurs principales solutions.

Dans le deuxième chapitre, nous montrons le lien entre les modèles markoviens cachés et la reconnaissance automatique de la parole et nous présentons les différentes applications existantes.

Le troisième chapitre expose la première partie de notre travail concernant la formulation et le développement des modèles markoviens cachés du second ordre, leur application à un système de reconnaissance de mots isolés et la généralisation de cette formulation à un ordre quelconque. Les modèles markoviens cachés (HMM) que nous avons utilisés sont des modèles discrets utilisant les symboles d'une quantification vectorielle. Le nouveau reconnaisseur du second ordre a été comparé à celui du premier ordre sur un corpus de chiffres français prononcés isolément. Les résultats montrent l'utilité et l'efficacité de ce passage au second ordre.

Le quatrième chapitre présente la deuxième partie de notre travail visant le développement de variantes de l'algorithme de reconnaissance de Viterbi pour la reconnaissance de la parole continue et leur comparaison expérimentale. En particulier, nous proposons une stratégie d'utilisation de l'algorithme de Viterbi par bloc, qui s'avère intéressante. Dans nos applications, nous nous sommes bornés au niveau de décodage acoustico-phonétique qui occupe une place importante et reste à l'heure actuelle un problème majeur en reconnaissance de la parole continue. Les différentes versions de l'algorithme de Viterbi ont été testées sur une segmentation manuelle des phrases de Combescure phonétiquement équilibrées.

Enfin, après avoir rapporté, les principaux résultats obtenus, nous dégageons une conclusion sur ce travail.

1
Théorie des modèles markoviens cachés

1.1 Introduction

L'avantage des modèles markoviens par rapport à plusieurs autres types de modélisation, réside dans le fait qu'ils sont fondés sur des bases théoriques fortes et rigoureuses. Ce qui explique leur grande utilité dans plusieurs domaines d'applications.

Dans ce chapitre, nous allons définir, dans un premier temps, les modèles markoviens et les modèles markoviens cachés. Après quoi, nous allons voir comment ces derniers permettent, grâce à leur bases théoriques, de résoudre les problèmes fondamentaux correspondant aux types d'applications que nous voulons traiter : la reconnaissance automatique de la parole.

Les définitions et les notations utilisées sont basées sur celles employées dans [Levinson 83a, Rabiner 88].

1.2 Définitions

1.2.1 Les processus markoviens discrets

Définitions

- Une *variable aléatoire* réelle est une fonction mesurable à valeurs dans $\mathcal{R}$:

$$X : \Omega \longrightarrow \mathcal{R}$$
$$\omega \longmapsto X(\omega)$$

- Un *processus stochastique* est une famille $(X(t), t \in T)$ de variables aléatoires définies sur un espace Ω. Il tend à représenter l'évolution du système en fonction du temps. Un processus stochastique met en oeuvre des modèles probabilistes spécifiques dans le but de gérer l'incertitude et le manque d'information qui entâchent les formes à

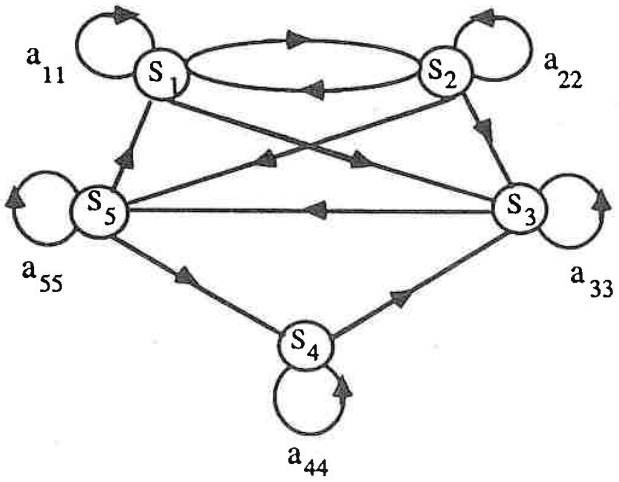


Figure 1.1. Exemple de chaîne stochastique

reconnaître. Dans le cas de R.A.P. ce phénomène est assez courant et n'importe quel codage aussi intelligent qu'il soit peut entraîner des ambiguités et nécessite de ce fait des corrections.

- Un processus stochastique est *discret* si les variables aléatoires sont en nombre fini ou dénombrables : X(1),X(2),...,X(T) dans le cas fini. Si l'ensemble des valeurs réelles prises par le processus (espace d'états) est fini ou dénombrable nous parlons de *chaîne* stochastique [figure 1.1].

- Un processus stochastique discret vérifie *la propriété de Markov* si pour tout t, nous avons la relation probabiliste suivante :

$$P(X(t) = s_i/X(t-1) = s_j, X(t-2) = s_k, ...) = P(X(t) = s_i/X(t-1) = s_j)$$

pour simplifier les notations, nous noterons, par la suite, X(t) par q_t et la propriété de Markov deviendra, donc, comme suit :

$$P(q_t = s_i/q_{t-1} = s_j, q_{t-2} = s_k, ...) = P(q_t = s_i/q_{t-1} = s_j)$$

- Une *chaîne de Markov discrète* est donc :
  - Un processus stochastique discret
  - Le nombre d'états est fini ou dénombrable
  - La propriété de Markov est vérifiée
- Une chaîne de Markov est *stationnaire* si pour tout t et k nous avons :

$$P(q_t = s_i/q_{t-1} = s_j) = P(q_{t+k} = s_i/q_{t+k-1} = s_j)$$

Nous définissons alors une matrice (de probabilité) de *transitions* :

$$A = [a_{ij}] \quad avec \quad a_{ij} = P(q_t = s_j/q_{t-1} = s_i) \quad 1 \leq i \leq N \; et \; 1 \leq j \leq N$$

avec N le nombre d'états.

L'étude d'une chaîne de Markov discrète stationnaire [figure 1.1] est l'étude de la matrice de transitions A associée (matrice stochastique). Les propriétés des coefficients de transitions d'états sont:

$$\forall i, j \;\; a_{ij} \geq 0 \quad , \quad \forall i \;\; \sum_{j=1}^{N} a_{ij} = 1$$

Exemple

Dans une station météo, en chaque jour à midi le temps est observé et classifié. Nous modélisons l'évolution du temps par une chaîne de Markov discrète d'espace d'états fini stationnaire [figure 1.2] :

- état 1 : "pluie"
- état 2 : "nuage"
- état 3 : "soleil"

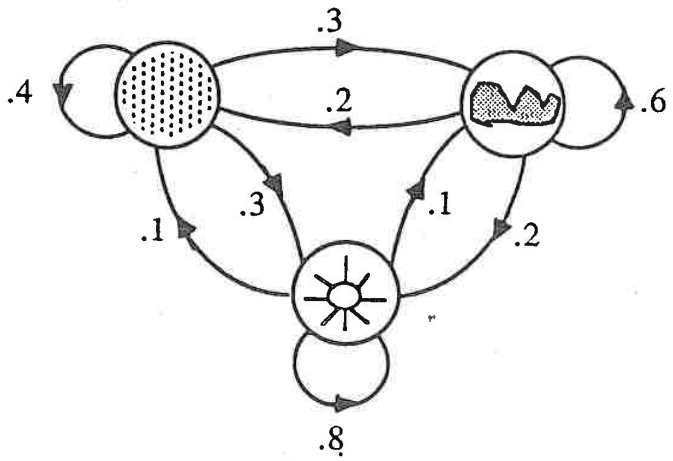


Figure 1.2. modélisation de l'évolution du temps par une chaîne

avec la matrice de transition : $A = [a_{ij}] =$

$$\begin{bmatrix} 0.4 & 0.3 & 0.3 \\ 0.2 & 0.6 & 0.2 \\ 0.1 & 0.1 & 0.8 \end{bmatrix}$$

Problème : Le temps d'un jour est "soleil", quelle est la probabilité pour que le temps des sept jours qui le suivent soit :

`"soleil , soleil , pluie , pluie , soleil , nuage , soleil" ?`

Nous définissons la suite d'observations comme :

$$O = (s_3, s_3, s_3, s_1, s_1, s_3, s_2, s_3)$$

et nous calculons $P(O/modèle, q_1 = s_3)$.

$$\begin{aligned}P(O/modèle, q_1 = s_3) &= P(s_3, s_3, s_3, s_1, s_1, s_3, s_2, s_3)\\ &= P(s_3)P(s_3/s_3)P(s_3/s_3)P(s_1/s_3)P(s_1/s_1)P(s_3/s_1)P(s_2/s_3)P(s_3/s_2)\\ &=\Pi_3(a_{33})^2 a_{31}a_{11}a_{13}a_{32}a_{23}\\ &=(1.0)(0.8)^2(0.1)(0.4)(0.3)(0.1)(0.2)\\ &=(1.536).10^{-4}\end{aligned}$$

avec $\Pi_i = P(q_1 = s_i)$ pour $1 \leq i \leq N$.

Problème : Le modèle est dans un état donné. Quelle est la probabilité pour qu'il reste dans le même état pour exactement d jours ?

Pour résoudre ce problème nous considérons la suite d'observations suivante :

$$O = (q_1 = s_i, q_2 = s_i, ..., q_d = s_i, q_{d+1} = s_j)$$

avec $s_j \neq s_i$ et nous calculons $P(O/modèle, q_1 = s_i)$.

$$P(O/modèle, q_1 = s_i) = (a_{ii})^{d-1}(1 - a_{ii}) = p_i(d)$$

avec p_i la probabilité de durée implicite d'état.
Le nombre de jours en moyen pendant lesquels un modèle peut rester dans un état s_i est donc :

$$\bar{d}_i = \sum_{d=1}^{\infty} d.p_i(d) = \frac{1}{1 - a_{ii}}$$

nous pouvons avoir, donc, en moyenne :

- 1.66 jours de suite de pluie,
- 2.5 jours de suite de nuage,
- 5 jours de suite de soleil.

1.2.2 Les modèles markoviens cachés

Définition

Bien que plusieurs travaux, au début, se référaient à ces modèles par : "fonction probabiliste d'une chaîne de Markov", l'appellation *"modèle markovien caché"* (**H**idden **M**arkov **M**odel), inventée par L.P.Neuwirth en 1970, est bien connue aujourd'hui.

Un *modèle markovien caché discret du premier ordre* est une chaîne de Markov stationnaire générant un processus stochastique à deux composantes : une cachée et l'autre observable.

Par la suite un modèle markovien caché sera noté **HMM** .

exemple 1

Le but de cet exemple est de montrer la différence entre un modèle markovien caché et un modèle markovien observable.

Une série de "Pile ou Face" est faite. Le nombre de fois est inconnu, seulement le résultat de chaque "Pile ou Face" est révélé. Exemple de suite d'observations :

$$O = O_1 O_2 ... O_T = PPFFPFFFPPPP...F$$

Nous voulons construire un HMM pour expliquer la suite d'observations. Les problèmes qui se posent sont alors :

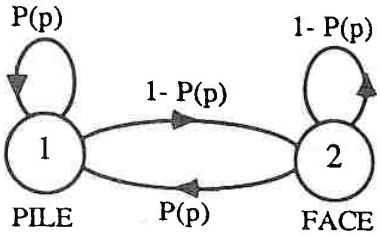
1. Quels sont les états dans ce modèle ?
2. Combien d'états seront utilisés ?
3. Quelles sont les probabilités des transitions d'états ?

Les trois cas de figure montrent la différence entre un modèle markovien observable, où la suite d'états S, correspondante à O, est bien définie (l'observation P correspond à l'état 1 et l'observation F correspond à l'état 2) [figure 1.3], et un modèle markovien caché à deux états [figure 1.4] ou à trois états [figure 1.5], où on ne peut pas définir la suite d'état S directement à partir de la suite d'observation O.

exemple 2

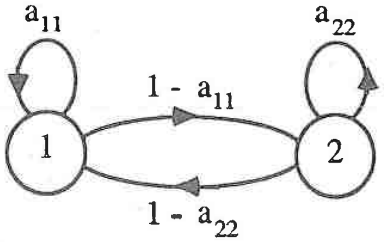
L'exemple des Urnes de Polya [Poritz 88] reflète parfaitement les deux composantes du processus stochastique d'un HMM construit comme suit :

Supposons que nous avons deux Urnes : Urne1 et Urne2. Chaque Urne a son propre mélange de balles rouges et noires. Soient $b_1(r)$, $b_2(r)$ les fractions des balles rouges



O = PPFFPFPPFFP.....
S = 11221211221.....

Figure 1.3. Un modèle markovien observable pour modéliser "Pile ou Face"

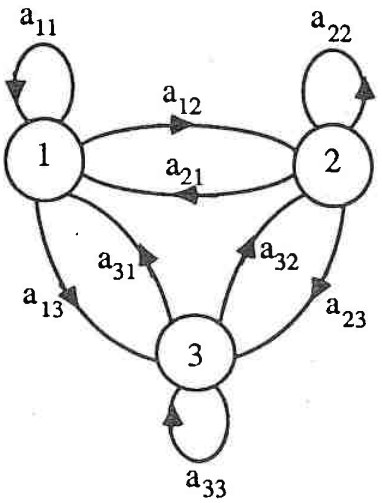


O = PPFFPFPPFFP.....
S = 21122212212.....

P(p) = p_1
P(F) = 1 - p_1

P(p) = p_2
P(F) = 1 - p_2

Figure 1.4. Un modèle markovien caché à deux états pour modéliser "Pile ou Face"



O = PPFFPFPPPFFP.....
S = 31233112313....

Etat	1	2	3
P(p)	p_1	p_2	p_3
P(F)	$1 - p_1$	$1 - p_2$	$1 - p_3$

Figure 1.5. Un modèle markovien caché à trois états pour modéliser "Pile ou Face"

respectivement dans l'Urne1 et l'Urne2. Soient $b_1(n)$, $b_2(n)$ les fractions des balles noires respectivement dans l'Urne1 et l'Urne2. Nous avons :

$$b_1(r) + b_1(n) = 1$$

$$b_2(r) + b_2(n) = 1$$

Et soient trois gobelets : Gobelet0, Gobelet1 et Gobelet2, chaque Gobelet a son propre mélange de pierres portant des marques. La marque sur une pierre est "état 1" ou "état 2". Soient a_{01}, a_{11}, a_{21} les fractions des pierres marquées "état 1" respectivement dans Gobelet0, Gobelet1 et Gobelet2. De même a_{02}, a_{12}, a_{22} les fractions des pierres marquées "état 2" respectivement dans Gobelet0, Gobelet1, et Gobelet2. Nous avons :

$$a_{01} + a_{02} = 1$$

$$a_{11} + a_{12} = 1$$

$$a_{21} + a_{22} = 1$$

Générons une suite d'observations des couleurs des balles $O = (O_1, O_2, ..., O_T)$ comme suit :

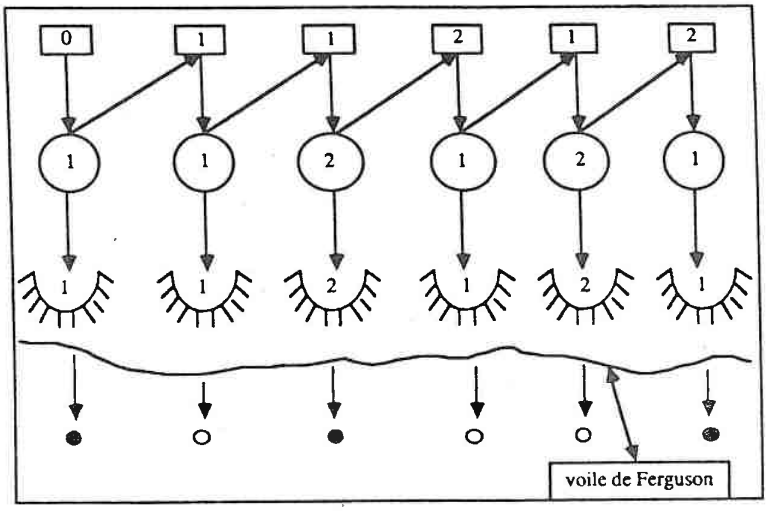


Figure 1.6. Un HMM du premier ordre à deux états

Tirons aléatoirement une pierre du Gobelet0; sa marque est appelée "état s_1". Tirons une balle aléatoirement de l'Urnes_1; sa couleur est O_1. Maintenant, tirons une pierre aléatoirement du Gobelets$_1$; sa marque est appelée "état s_2". Continuons dans cette voie, en utilisant l'état courant pour obtenir à la fois l'observation courante et l'état suivant jusqu'à un total de T observations [figure 1.6].

Ce mécanisme, génératif pour créer une suite d'obsevations, est un processus stochastique avec une composante cachée : En générant la suite d'observations des couleurs O, une suite de pierres $S = (s_1, s_2, ..., s_T)$ est aussi générée. Lorsque S n'est pas observée, elle est alors une suite cachée (ou chemin caché). Le voile de Ferguson cache cet unique échantillonnage des Gobelets. L'observateur obtient seulement une information probabiliste concernant les pierres. Le paramètre vecteur du modèle stochastique est :

$$\lambda = (a_{01}, a_{02}, a_{11}, a_{12}, a_{21}, a_{22}, b_1(r), b_1(n), b_2(r), b_2(n))$$

En résumé, nous avons défini à présent un modèle markovien caché du premier ordre à deux états. Il est un modèle du premier ordre puisque chaque état successeur est selectionné comme une fonction probabiliste du dernier état prédécesseur.

Bien que nous n'ayons que deux états dans cet exemple, un nombre fini, S, des états est possible. Dans ce cas, l'exemple va correspondre à S Urnes et S+1 Gobelets avec des

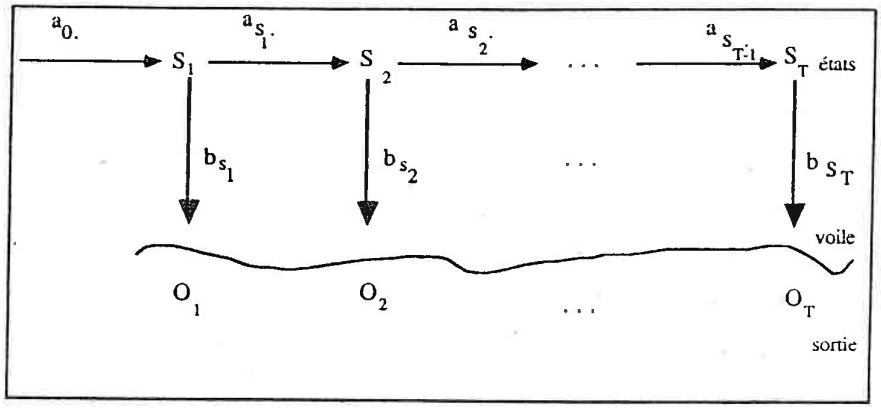


Figure 1.7. Génération d'une suite d'états et d'une suite d'observations à partir d'un HMM

mélanges de pierres portant S marques différentes. Il convient de noter aussi par S le nom de la suite des états. Le vecteur de probabilité $\Pi = (a_{01}, a_{02}, ..., a_{0s})$ associé dans l'exemple au Gobelet0 est la distribution initiale. La matrice stochastique $A = [a_{rs}]$, où la $r^{ème}$ rangée est associée au Gobelet$r$, est la matrice de transition. Π et A constituent une chaîne de Markov du premier ordre. La suite d'états cachées S est produite par cette chaîne [figure 1.7].

Un modèle markovien caché est dit ayant un alphabet de sortie fini si les articles observés s'étendent à un ensemble fini de k éléments. Dans l'exemple, les sorties sont $\{balle-rouge, balle-noire\}$ et k=2 (Autres exemples : l'alphabet du français plus l'espace entre mots, les symboles d'un dictionnaire de quantification vectorielle, l'ensemble des phonèmes dans un langage,...).

Pour chaque état s, le vecteur $b_s = (b_s(1), ..., b_s(k))$ (associé dans l'exemple à l'Urne s) est appelé vecteur de probabilité de sortie pour l'état s. Ces probabilités de sortie tracent la suite d'états S à partir de la suite des observations O.

Alternativement, les observations peuvent s'étendre dans un ensemble continu (ou dénombrable infini). Dans le choix de sortie continue, chaque état s est associé à son propre paramètre de densité de probabilité b_s.

Un modèle markovien caché est représenté par son vecteur paramètre $\lambda = (\Pi, A, b_1, b_2, ..., b_S)$. Dans le cas d'un alphabet fini (discret), la matrice $B = [b_1, b_2, ..., b_S]$ s'appelle la matrice de probabilités d'observations et le vecteur λ devient (Π, A, B).

t	1	2	3	4	5	6	7	8	...	T
Etat	q_1	q_2	q_3	q_4	q_5	q_6	q_7	q_8	...	q_T
Observation	o_1	o_2	o_3	o_4	o_5	o_6	o_7	o_8	...	o_T

Figure 1.8. Une suite d'observations générée par un HMM

Les éléments d'un HMM

Comme nous venons de le voir dans l'exemple précédent, les éléments d'un HMM sont :

1. N : Le nombre des états du modèle.
   - $S = \{s_1, s_2, ..., s_N\}$ est l'ensemble des états
   - Nous désignons un état au temps t par q_t ($q_t \in S$)
2. M : Le nombre des symboles d'observations distincts.
   - $V = \{v_1, v_2, ..., v_M\}$ est l'ensemble des symboles d'observations
   - Nous désignons un symbole au temps t par O_t ($O_t \in V$)
3. La distribution des probabilités des transitions des états.
   - $A = [a_{ij}]$
   - $a_{ij} = P(q_{t+1} = s_j / q_t = s_i)$ avec $1 \leq i, j \leq N$
4. La distribution des probabilités des symboles d'observations dans chaque état j.
   - $B = [b_j(k)]$
   - $b_j(k) = P(O_t = v_k / q_t = s_j)$ avec $1 \leq i \leq N$ et $1 \leq k \leq M$
5. La distribution de la probabilité d'être initialement dans un état.
   - $\underline{\Pi} = [\underline{\Pi}_i]$
   - $\Pi_i = P(q_1 = s_i)$ avec $1 \leq i \leq N$

En résumé, les éléments d'un HMM sont : N, M et λ avec $\lambda = (\Pi, A, B)$. Par souci de simplification, un modèle HMM sera désigné, dans la suite, par son vecteur paramètre λ.

La génération des observations dans un HMM [figure 1.8] se fait par la procédure suivante :

1. Pas $t = 1$. Choix de l'état initial, $q_1 = s_i$, avec la distribution Π_i.
2. Choix de l'observation, $O_t = v_k$, avec la probabilité $b_i(k)$.
3. Transition au nouvel état, $q_{t+1} = s_j$, avec la probabilité a_{ij}.
4. Pas $t = t + 1$. Si $t \leq T$ Alors retour à l'étape 2 Sinon fin de procédure.

avec T étant la longueur d'une suite d'observations.

1.3 Résolution des problèmes fondamentaux

Un HMM étant donné, il y a trois problèmes de base qui doivent être résolus pour qu'un tel modèle puisse être utilisé efficacement dans le monde des applications réelles. Ces problèmes sont les suivants :

1. Evaluation : Soient la suite d'observations O et un modèle λ, comment évaluer la probabilité $P(O/\lambda)$?
2. Chemin le plus probable (estimation de la partie cachée du modèle) : Soient la suite d'observation O et un modèle λ, comment trouver une suite d'états $Q = q_1 q_2 ... q_T$ qui soit optimale selon un critère convenable?
3. Adaptation optimale du modèle (problème d'apprentissage) : Comment ajuster les paramètres du modèle λ pour maximiser $P(O/\lambda)$?

1.3.1 Evaluation de la probabilité d'observation

Evaluation directe

La probabilité d'une suite d'observations O, sachant qu'un modèle λ est donné, est la somme sur tous les chemins d'états, Q, possibles des probabilités conjointes de O et de Q par rapport à ce modèle :

$$P(O/\lambda) = \sum_Q P(O, Q/\lambda) = \sum_Q P(O/Q, \lambda) P(Q/\lambda)$$

or :

$$P(Q/\lambda) = \Pi_{q_1} a_{q_1 q_2} a_{q_2 q_3} ... a_{q_{T-1} q_T}$$

et :

$$P(O/Q, \lambda) = b_{q_1}(O_1) b_{q_2}(O_2) ... b_{q_T}(O_T)$$

d'où :

$$P(O/\lambda) = \sum_{q_1,q_2,\ldots,q_T} \Pi_{q_1} b_{q_1}(O_1) a_{q_1 q_2} b_{q_2}(O_2) \ldots a_{q_{T-1} q_T} b_{q_T}(O_T)$$

Pour calculer $P(O/\lambda)$ par cette formule directe, il faut $N^T - 1$ additions et $N^T(2T-1)$ multiplications, soit environ $2TN^T$ opérations. Ce qui est énorme. Par exemple pour $N = 5$ et $T = 10$ il nous faut 20.5^{10} opérations environ.

Heureusement, il y a l'algorithme *Forward-Backward* qui permet une résolution agréable de ce premier problème.

Les fonctions Forward et Backward

Considérons la variable *Forward*, $\alpha_t(i)$, définie comme la probabilité de la suite d'observations partielle de 1 à t, terminant à l'état s_i du modèle λ :

$$\alpha_t(i) = P(O_1 O_2 \ldots O_t, q_t = s_i/\lambda)$$

Le calcul de $\alpha_t(i)$ se fait de façon récursive :

1. Initialisation

$$\alpha_1(i) = \Pi_i b_i(O_1) \quad ; \quad 1 \leq i \leq N$$

2. Induction

$$\alpha_{t+1}(j) = [\sum_{i=1}^{N} \alpha_t(i) a_{ij}] b_j(O_{t+1}) \quad ; \quad 1 \leq t \leq T-1 \quad 1 \leq j \leq N$$

3. Terminaison

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

Pour calculer $P(O/\lambda)$ par cette méthode inductive il nous faut $N + N(N+1)(T-1)$ multiplications et $(N-1)N(T-1)$ additions, soit une complexité en N^2T [figure 1.9]. Par exemple, pour $N = 5$ et $T = 10$ il faut 250 opérations environ.

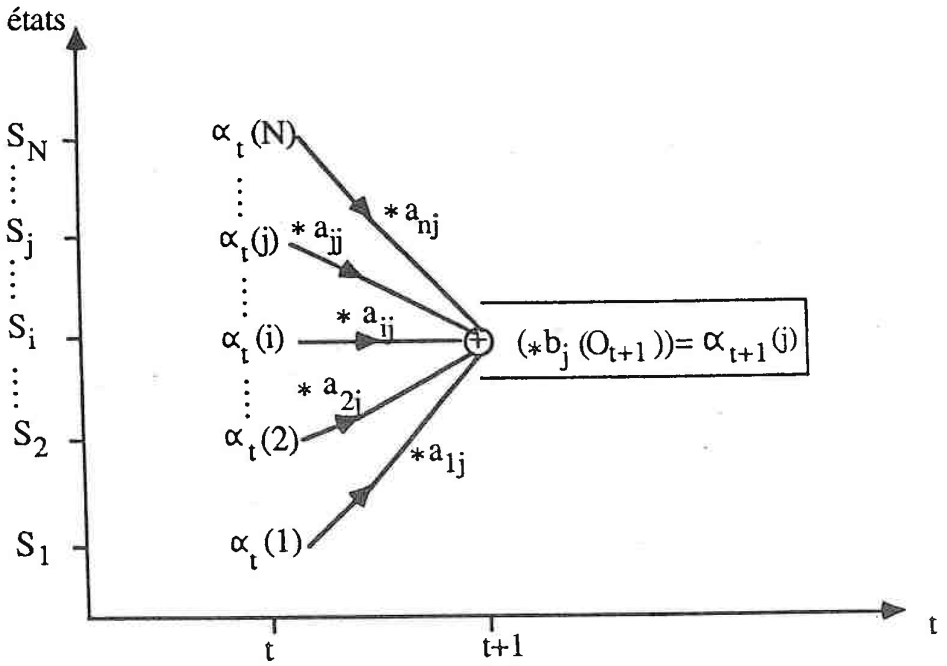


Figure 1.9. Calcul de α_t par induction

Considérons maintenant la variable *Backward*, $\beta_t(i)$, définie comme la probabilité de la suite d'observations partielle de $t+1$ à T, sachant qu'à l'instant t on était dans l'état s_i du modèle λ :

$$\beta_t(i) = P(O_{t+1}O_{t+2}...O_T/q_t = s_i, \lambda)$$

β_t se déduit de β_{t+1} de façon inductive :

1. Initialisation

$$\beta_T(i) = 1 \quad ; \quad 1 \leq i \leq N$$

2. Induction

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j) \quad T-1 \geq t \geq 1 \quad 1 \leq i \leq N$$

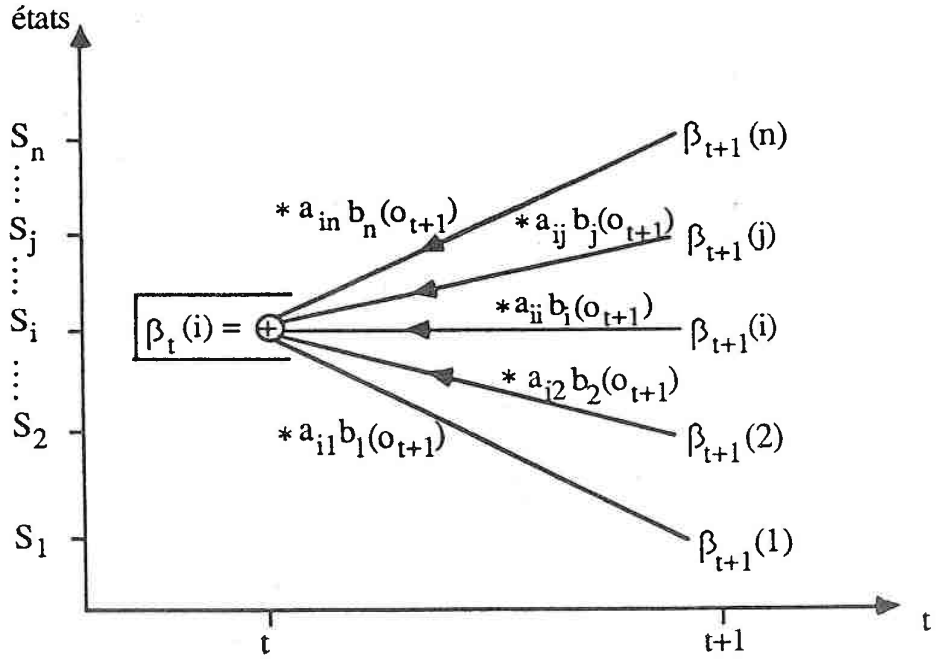


Figure 1.10. Calcul de β_t par induction

Le calcul de β_t est aussi en N^2T [figure 1.10].

Les variables Forward et Backward seront aussi utiles pour résoudre le troisième problème : l'estimation du modèle.

1.3.2 Chemin optimal

L'essentiel dans une application utilisant les HMMs est la détermination, de façon optimale, de la composante cachée du processus, étant donné la composante observable, O, et les informations probabilistes sur le modèle (N,M et λ). La difficulté de ce problème est la définition du critère d'optimisation. Il y a plusieurs choix pour ce critère. Par exemple : accepter l'état qui a le plus de chance individuellement. Ce qui revient à choisir au temps t l'état qui maximise :

$$P(q_t = s_i/O, \lambda)$$

D'autres critères sont utilisés, tels que le choix des états qui ont le plus de chance deux à deux ou trois à trois, dans certaines applications (Un inconvénient évident à signaler dans ces types de choix est qu'une partie des contraintes de transitions entre états ne sera pas prise en compte) [Rabiner 88]. Mais le critère le plus naturel, et le plus utilisé, est celui de trouver la meilleure suite d'états (chemin) qui maximise :

$$P(Q/O, \lambda)$$

Ce qui est équivalent à maximiser :

$$P(Q, O/\lambda)$$

La solution par ce dernier critère est souvent appelée la suite d'états de *Viterbi* parce qu'elle est trouvée en utilisant l'algorithme de *Viterbi* [Forney 73] qui est basé sur les techniques de la programmation dynamique.

L'algorithme de Viterbi

Pour trouver le meilleur chemin, $Q = (q_1, q_2, ..., q_T)$, pour une suite d'observations donnée, $O = (O_1, O_2, ..., O_T)$, nous définissons la quantité $\delta_t(i)$ comme la probabilité du meilleur chemin partiel amenant à l'état s_i à l'instant t guidé par les t premières observations :

$$\delta_t(i) = \max_{q_1, q_2, ..., q_{t-1}} P(q_1 q_2 ... q_t = s_i, O_1 O_2 ... O_t/\lambda)$$

Par induction, nous pouvons calculer $\delta_{t+1}(j)$ à partir des $\delta_t(i)$:

$$\delta_{t+1}(j) = [\max_i \delta_t(i) a_{ij}].b_j(O_{t+1})$$

Et nous devons garder trace de la suite d'états qui donne le meilleur chemin amenant à l'état s_i à l'instant t dans un tableau ψ. L'algorithme de *Viterbi* se résume par les quatre étapes suivantes :

1. Initialisation

$$\delta_1(i) = \Pi_i b_i(O_1) \quad pour \quad 1 \leq i \leq N$$

$$\psi_1(i) = 0$$

2. Induction

$$\delta_t(j) = \max_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}] . b_j(O_t) \quad pour \quad 1 \leq j \leq N$$

$$\psi_t(j) = \arg \max_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}] \quad pour \quad 1 \leq j \leq N$$

3. Terminaison

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

$$q_T^* = \arg \max_{1 \leq i \leq N} [\delta_T(i)]$$

4. Chemin d'états retenu (backtracking)

$$q_t^* = \psi_{t+1}(q_{t+1}^*) \quad pour \quad T-1 \geq t \geq 1$$

La fonction arg permet de mémoriser l'indice i, entre 1 et N, avec lequel on atteind le maximum des quantités $(\delta_{t-1}(i)a_{ij})$.

Le coût des opérations dans cet algorithme est en N^2T multiplications [figure 1.11]. Les difficultés d'ordre numérique peuvent être résolues en implantant une alternative de Viterbi utilisant les logarithmes. Le coût des opérations est ainsi devenu en N^2T additions.

Comparaison de Viterbi avec Forward-Backward

Ce deuxième problème, concernant le décodage, peut parfois se limiter à comparer les probabilités accumulées (P^*) des différents modèles HMMs pour choisir celui qui est le plus adapté à la forme inconnue à reconnaître, c'est-à-dire sans avoir recours à un cheminement arrière pour déterminer le chemin d'états optimal (le cas des mots isolés, par exemple).

L'algorithme Forward-Backward (celui utilisant les fonctions Forward et Backward) peut, lui aussi, servir à ce type de décodage. Par la suite nous présentons une comparaison entre cet algorithme et celui de Viterbi pour un tel décodage :

- L'algorithme Forward-Backward utilise la probabilité :

$$P_{F-B} = \sum_{Q} P(Q, O/\lambda)$$

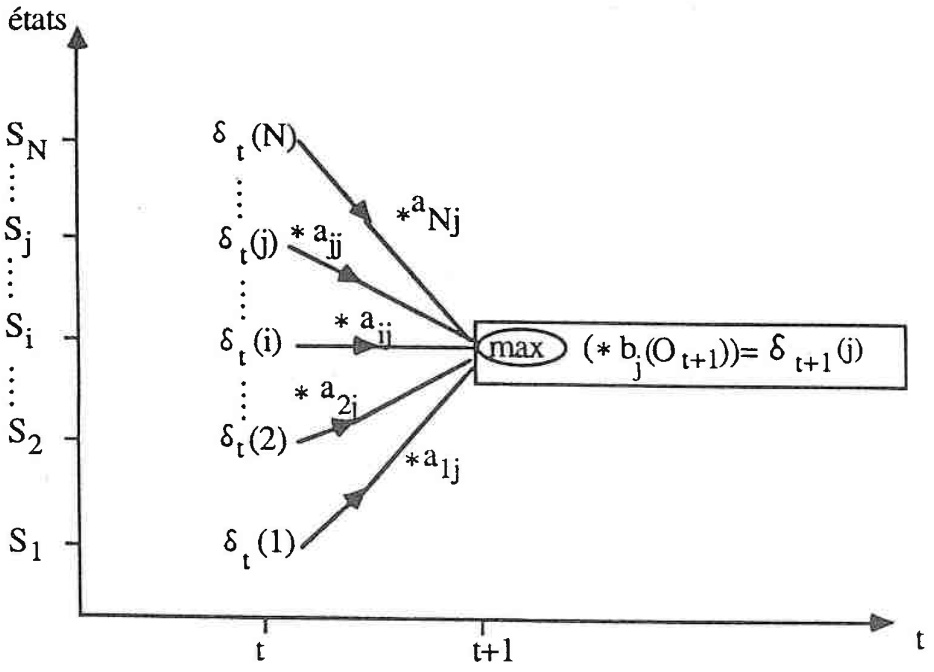


Figure 1.11. Calcul de δ_t par induction

- L'algorithme de Viterbi utilise la probabilité :

$$P_V = \max_Q P(Q, O/\lambda)$$

- Pour évaluer la probabilité qu'un mot soit produit par un modéle, c'est la quantité P_{F-B} qu'il faut utiliser parce qu'elle représente la probabilité d'observation du mot en ayant le modéle λ. Pourtant, l'algorithme de Viterbi permet de calculer plus simplement une autre quantité : la probabilité du meilleur alignement P_V qui dans les faits est peu différente de la probabilite P_{F-B}.
- Les difficultés d'ordre numérique peuvent être résolues dans le cas de l'algorithme de Viterbi en considérant les logarithmes des probabilités utilisées dans les deux premières étapes de cet algorithme (Dans l'autre algorithme aussi, mais moins efficacement).
- L'algorithme de Viterbi permet d'avoir le meilleur alignement temporel entre observations et états [Forney 73,Rabiner 88].

Dans ce type de problème de décodage, nous constatons que l'algorithme de Viterbi est généralement choisi. A plus forte raison, il est souvent utilisé dans le cas où on a besoin

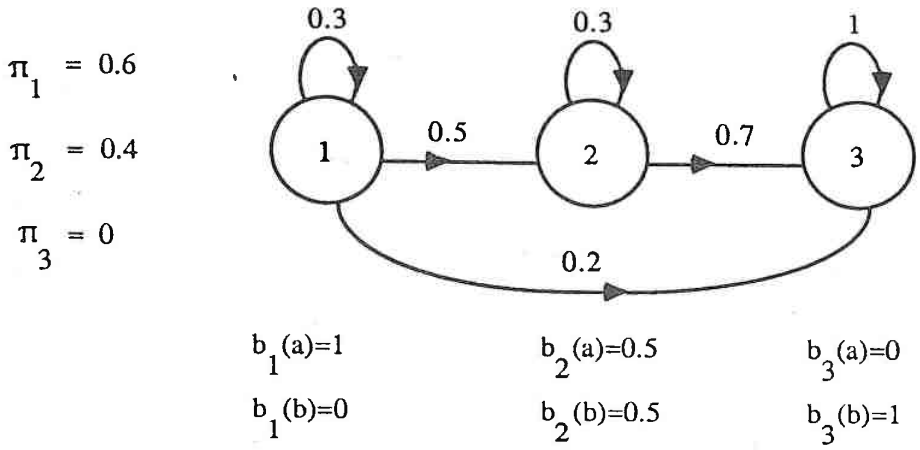


Figure 1.12. Un HMM à trois états

de décoder un chemin d'états optimal (le cas des mots enchaînés ou de la parole continue, par exemple).

Nous allons utiliser le modèle markovien caché à trois états 1,2 et 3 et à deux symboles a et b, illustré dans la figure 1.12, pour donner un exemple de calcul de la probabilité d'observation totale par la fonction Forward et un exemple de la détermination du chemin le plus probable par l'algorithme de Viterbi :

- La figure 1.13 illustre le calcul de $P(aabb/\lambda)$ par la fonction Forward.
- La figure 1.14 illustre le calcul effectué par l'algorithme de Viterbi et montre comment est déterminé le chemin le plus probable pour produire la suite d'observations aabb :

$$s_1 s_2 s_3 s_3$$

Pour les phases de reconnaissance dans nos diverses applications, c'est, bien sûr, l'algorithme de Viterbi qui est choisi.

1.3.3 Estimation du modèle HMM

La grande difficulté dans une utilisation des HMMs est celle de déterminer, les paramètres (Π, A, B) qui maximisent la probabilité de la suite d'observations, $P(O/\lambda)$. Les difficultés de calcul nous empêchent d'avoir une solution d'optimisation globale : Il n'y a pas

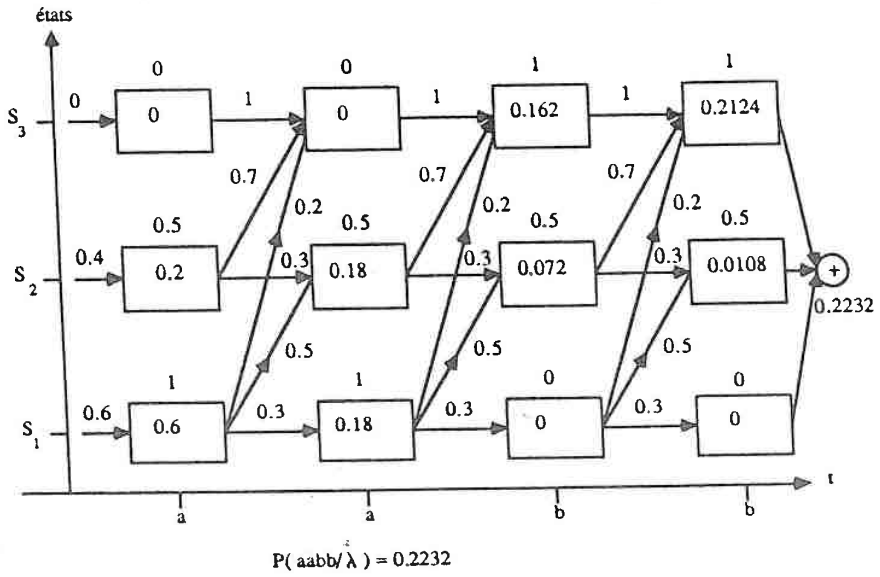


Figure 1.13. Exemple de calcul de $P(O/\lambda)$

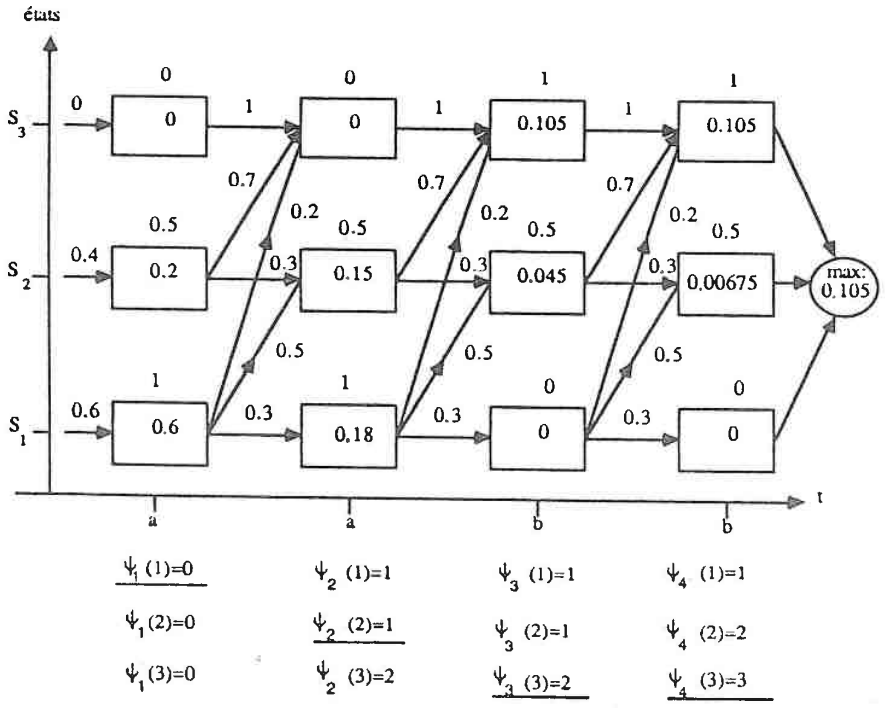


Figure 1.14. Exemple de calcul par l'algorithme de Viterbi

une méthode analytique directe pour résoudre ce problème. Donc, les solutions utilisées ne présentent que des optimisations locales telles que les techniques du gradient et les procédures de ré-estimations.

L'idée de base pour les procédures de ré-estimations est de construire une suite de modèles affinés au fur et à mesure.

Dans le but d'utiliser une procédure de ré-estimation, (l'algorithme de Baum-Welch), nous définissons les quantités suivantes :

Les formules de ré-estimations

Nous définissons $\xi_t(i,j)$, la probabilité d'être dans l'état s_i à l'instant t, et dans l'état s_j à l'instant t+1 sachant que nous avons le modèle λ et la suite d'observations O :

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

De cette définition, nous déduisons facilement que :

$$\xi_t(i,j) = \frac{P(q_t = s_i, q_{t+1} = s_j, O/\lambda)}{P(O/\lambda)}$$

En développant le numérateur du terme de droite de cette égalité et en pensant à introduire les fonctions Forward et Backward, nous aboutissons à une formule très intéressante au niveau algorithmique (illustrée dans la figure 1.15) :

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

Nous définissons aussi la quantité $\gamma_t(i)$ comme la probabilité d'être dans l'état s_i à l'instant t sachant que nous avons le modèle λ et la suite d'observations O :

$$\gamma_t(i) = P(q_t = s_i / O, \lambda)$$

soit :

$$\gamma_t(i) = \sum_{j=1}^{N} P(q_t = s_i, q_{t+1} = s_j / O, \lambda)$$

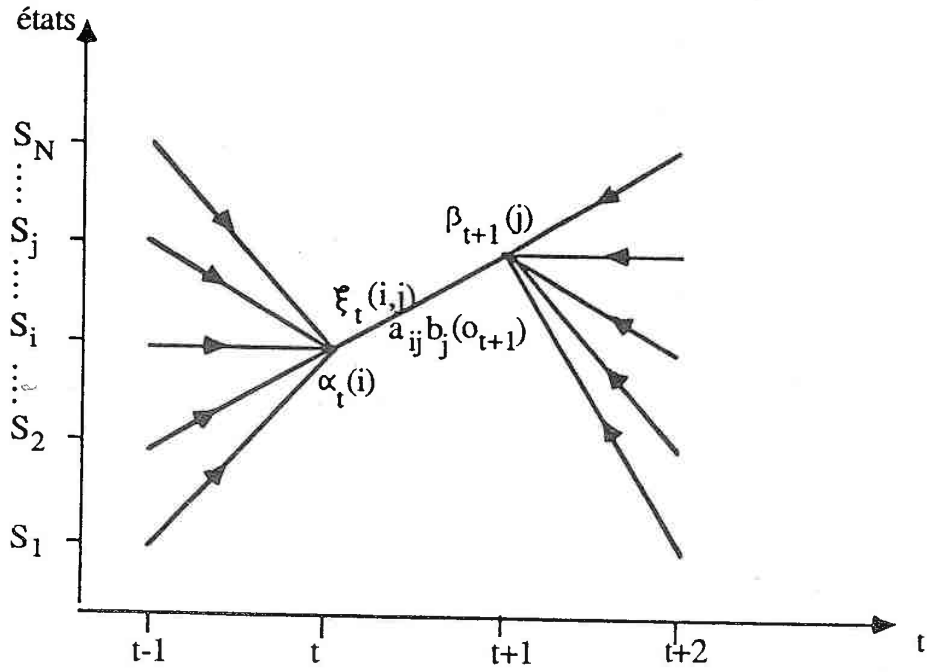


Figure 1.15. Calcul de ξ_t à partir des fonctions α_t *et* β_t

Nous déduisons, donc, la relation entre $\xi_t(i,j)$ et $\gamma_t(i)$:

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

D'après la définition de $\gamma_t(i)$, une sommation sur le temps t de cette quantité peut être interprétée comme une information sur le nombre de fois possibles d'être dans l'état s_i ou comme une information sur le nombre de transitions possibles à partir de l'état s_i. De manière similaire, une sommation sur le temps t de la quantité $\xi_t(i,j)$ peut être interprétée comme une information sur le nombre de transitions possibles à partir de l'état s_i à l'état s_j. Ainsi, des formules de ré-estimations convenables pour Π, A et B peuvent être :

$\bar{\Pi}_i =$ *Le nombre de fois possibles d'être dans l'état* s_i *à l'instant* 1

$$\bar{a}_{ij} = \frac{\textit{Le nombre de transitions possibles à partir de l'état } s_i \textit{ à l'état } s_j}{\textit{Le nombre de transitions possibles à partir de l'état } s_i}$$

$$\bar{b}_j(k) = \frac{\textit{Le nombre de fois possibles d'être dans l'état } s_j \textit{ en observant le symbole } v_k}{\textit{Le nombre de fois possibles d'être dans l'état } s_j}$$

soient :

$$\bar{\Pi}_i = \gamma_1(i)$$

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

$$\bar{b}_j(k) = \frac{\sum_{t=1, O_t=v_k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}$$

Notons que les contraintes stochastiques sur ces formules sont automatiques :

$$\sum_{i=1}^{N} \bar{\Pi} = 1 \quad , \quad \sum_{j=1}^{N} \bar{a}_{ij} = 1 \quad , \quad \sum_{k=1}^{M} \bar{b}_j(k) = 1$$

Théorème de Baum

Les formules de ré-estimations, que nous venons d'exposer de façon intuitive, sont en fait déduites du théorème de Baum [Baum 67] appliqué à la fonction suivante :

$$P(O/\lambda) = \sum_{Q} \Pi_{q_1} b_{q_1}(O_1) a_{q_1 q_2} b_{q_2}(O_2) ... a_{q_{T-1} q_T} b_{q_T}(O_T)$$

$P(O/\lambda)$ s'exprime comme un polynôme homogène à coefficients positifs.

L'énoncé du **théorème** est :

Soit le polynôme homogène suivant :

$$P(Z_1, Z_2, ..., Z_n) = \sum_{\mu_1,...,\mu_n} c_{\mu_1...\mu_n} Z_1^{\mu_1} Z_2^{\mu_2} ... Z_n^{\mu_n}$$

avec

$$c_{\mu_1...\mu_n} \geq 0 \quad et \quad \sum_{i=1}^{n} \mu_i = d$$

alors la transformation :

$$\mathcal{T}(Z_i) = \frac{Z_i \frac{\partial P}{\partial Z_i}}{\sum_j Z_j \frac{\partial P}{\partial Z_j}}$$

transforme l'ensemble :

$$D = \{Z_i, Z_i \geq 0, \sum_i Z_i = 1\}$$

dans lui même et vérifie :

$$P((\mathcal{T}(Z_i))_{i=1,..,n}) \geq P((Z_i)_{i=1,...,n})$$

L'inégalité est stricte sauf si les Z_i de D représentent un point critique de P.

Pour le polynôme $P(O/\lambda)$, les Z_i sont les Π_i, a_{ij} et les $b_j(k)$. Les $\mathcal{T}(Z_i)$ sont alors les $\bar{\Pi}_i$, $\bar{a}_{ij}$ et les $\bar{b}_j(k)$.

Donc, si $\lambda = (\Pi, A, B)$ est le modèle initial et $\bar{\lambda} = (\bar{\Pi}, \bar{A}, \bar{B})$ le modèle ré-estimé, nous avons, d'après ce théorème, deux cas possibles :

1. Le modèle initial, λ, définit un point critique (ce qui est rare) de la fonction de vraisemblance, $P(O/\lambda)$. Dans ce cas, nous avons :

$$\bar{\lambda} = \lambda$$

2. Le modèle $\bar{\lambda}$ est plus probable que le modèle λ dans le sens de l'inégalité du théorème :

$$P(O/\bar{\lambda}) > P(O/\lambda).$$

Nous avons donc trouvé un modèle $\bar{\lambda}$ à partir duquel la suite d'observations a plus de chance d'être produite.

En conclusion, nous utilisons $\bar{\lambda}$ à la place de λ, et nous répétons la ré-estimation jusqu'à un certain point limite (en général, on choisit un nombre d'itérations convenable). Le modèle résultant est, ainsi, un HMM obtenu par le **M**aximum de **V**raisemblance.

L'algorithme de Baum-Welch

En résumant ce que nous venons de voir, l'algorithme de *Baum-Welch* [Baum 70, Baum 72] est l'algorithme itératif suivant :

1. Fixer des valeurs initiales

   a_{ij}^0 , $b_j^0(k)$, Π_i^0 *pour* $1 \leq i \leq N$, $1 \leq j \leq N$, $1 \leq k \leq M$

2. Calculer

   $\xi_t(i,j)$, $\gamma_t(i)$ *pour* $1 \leq i \leq N$, $1 \leq j \leq N$, $1 \leq t \leq T-1$

   en utilisant les fonctions Forward et Backward.

3. Nouvelles estimations

   $\bar{\Pi}_i$, $\bar{a}_{ij}$, $\bar{b}_j(k)$, *pour* $1 \leq i \leq N$, $1 \leq j \leq N$, $1 \leq k \leq M$

   d'où $\bar{\lambda}$.

4. Recommencer en 2 jusqu'à un certain point limite.

Remarques :

- Le choix du modèle initial influe sur les résultats.
- L'algorithme converge vers des valeurs de paramètres qui forment un point critique de $P(O/\lambda)$. Donc, nous obtenons un maximum local ou un point d'inflexion. D'où la nécessité d'un bon choix du modèle initial pour éviter les points d'inflexion.
- Le test d'arrêt est en général le nombre des itérations qui est fixé empiriquement.

- Pour avoir une estimation convenable du modèle, les ré-estimations se font sur un ensemble de plusieurs suites d'observations appelées corpus d'apprentissage. Donc la taille du corpus d'apprentissage influe, elle aussi, sur les résultats.

- Les formules de ré-estimations peuvent être dérivées de la maximisation de la fonction auxiliaire de $\bar{\lambda}$:

$$R(\lambda, \bar{\lambda}) = \sum_{Q} P(O, Q/\lambda) \log(P(O, Q/\bar{\lambda}))$$

  En fait, il est prouvé que :

$$\max_{\bar{\lambda}}[R(\lambda, \bar{\lambda})] \quad \Longrightarrow \quad P(O/\bar{\lambda}) \geq P(O/\lambda)$$

  et que cette fonction de vraisemblance converge éventuellement vers un point critique [Baum 70,Baum 72].
  Cette méthode appelée **EM** (*Expectation Modification*) est, ainsi, une technique équivalente à la détermination de $\bar{\lambda}$ par le théorème de Baum.

- Puisque l'algorithme de Baum-Welch se base sur le calcul des fonctions Forward et Backward, il est possible de trouver dans la littérature cet algorithme désigné par l'algorithme Forward-Backward.

- D'après les formules récursives de calcul de α et de β, il est clair que lorsque T croît, α et β tendent vers 0. Pour un grand nombre d'observations, la fonction de probabilité aura ainsi une valeur trop petite pour être représentée dans un calculateur. Ce problème désigné dans la littérature par le problème de l'"underflow" sera traité dans la partie consacrée aux techniques d'implantation.

En conclusion, cette méthode du *Maximum de Vraisemblance* est la plus utilisée dans les applications. C'est celle que nous avons choisie dans les phases d'apprentissage de nos systèmes.

1.3.4 Autres méthodes d'apprentissage

Diverses variantes et alternatives des algorithmes de Baum-Welch et de Viterbi ont été développées afin d'apporter des améliorations ou des simplifications au niveau de l'apprentissage (et/ou de reconnaissance). Nous allons en citer quelques unes concernant l'apprentissage. Cette étape est très délicate. On se pose toujours la question : Quelle est la procédure qui, à partir d'un corpus d'apprentissage donné, calculera les valeurs des paramètres qui donneront le meilleur taux de reconnaissance? Généralement, seule l'expérimentation permet de comparer réellement deux méthodes.

Apprentissage par la méthode de l'algorithme de Viterbi

Au lieu de calculer la probabilité $P(O/\lambda)$, l'objectif pour cette méthode est de maximiser la probabilité $P(O, Q/\lambda)$. Les formules de ré-estimations sont identiques à celles

développées par l'algorithme de Baum-Welch. Sauf, pour les fonctions α et β, les sommes ($\sum$) sont remplacées par des maximums (max).

L'algorithme de Baum-Welch est moins sensible que celui de Viterbi à la qualité d'initialisation des paramètres des modèles, car tous les chemins de longueur T sont pris en compte durant le processus itératif d'estimation. Le problème de l'"underflow" est facilement résolu pour l'algorithme de Viterbi par le passage aux logarithmes.

Méthode mixte Baum-Welch / Viterbi

Afin d'éviter le problème de l'"underflow", Jouvet [Jouvet 87] utilise une version intermédiaire entre l'algorithme de Baum-Welch et celui de Viterbi : Pour chaque instant et chaque transition, on calcule la probabilité d'émission d'observation le long du meilleur chemin passant par cette transition (et non la somme comme dans l'algorithme de Baum-Welch), ce qui permet d'utiliser les logarithmes des probabilités comme dans l'algorithme de Viterbi. Les informations le long des divers chemins sont pondérées en fonction de cette probabilité d'émission de façon à obtenir un poids égal à 1 le long du chemin optimal et un poids de plus en plus faible au fur et à mesure que l'on s'en "éloigne".

Apprentissage évolutif

Un algorithme, basé sur la procédure Forward-Backward, permet aux modèles d'être améliorés tant qu'il existe de nouvelles occurrences du mot [Sugawara 86]. Ceci est fait par le calcul d'une moyenne pondérée prenant en compte des statistiques effectuées sur l'apprentissage initial et des statistiques dérivées de la nouvelle occurrence entrée. Cette technique, intéressante pour changer d'application ou pour adapter les modèles à de nouveaux locuteurs, nécessite cependant de nombreux calculs.

Méthode du maximum d'information mutuelle

Cette méthode (MMI), développée au centre IBM (New York) par Bahl et Brown [Bahl 86,Brown 87,Bahl 87], part du fait que les hypothèses faites sur les modèles utilisés ne sont pas totalement correctes quand on utilise la procédure d'estimation du maximum du vraisemblance habituelle. En effet, la vraie distribution de la parole n'est pas forcément un membre de la famille de distributions utilisées dans un reconnaisseur. Aussi, afin de mettre en relief les différences entre mots, les paramètres des divers modèles sont estimés ensemble. Cette méthode est reconnue comme fournissant de bonnes performances de reconnaissance dans [Brown 87,Merialdo 88c,Merialdo 88a].

Apprentissage correctif

Cette alternative, présentée dans [Bahl 87,Bahl 88], part de la même remarque faite dans l'approche précédente (MMI) tout en jugeant les valeurs des paramètres sur les erreurs des tests plutôt que sur la vraisemblance des données d'apprentissage. Elle vise, ainsi, à minimiser le nombre d'erreurs de la reconnaissance en ajustant les valeurs des paramètres de façon que les mots corrects soient plus probables et les mots incorrects soient moins probables. La convergence de l'algorithme n'est pas encore prouvée , mais

les expériences [Bahl 87,Bahl 88] suggèrent qu'il l'est, et il mène à des diminutions des erreurs de la reconnaissance plus significatives qu'avec l'estimation par le maximum de vraisemblance.

Méthode du minimum d'information de discrimination

Cette approche (MDI) itérative pour modéliser des HMMs des sources d'information a été proposée par Ephraim, Dembo et Rabiner [Ephraim 87] et reprise dans [Tishby 88] pour la reconnaissance des locuteurs. Elle vise à minimiser l'information de discrimination (entropie relative) entre la source et le modèle. Cette approche n'exige pas la supposition faite généralement : la source à modéliser est un processus de Markov caché. L'algorithme est initialisé par le modèle estimé par l'approche du maximum de vraisemblance de Baum-Welch et décrémente alternativement l'information de discrimination sur toute les distributions de probabilité de la source qui concorde avec les mesures données et sur tous les HMMs. Cet algorithme est une généralisation de celui de Baum-Welch, il est descendant pour la mesure de l'information de discrimination et sa convergence locale est prouvée.

Corrélation du temps explicite

Pour palier l'inconvénient de supposer l'indépendance entre les observations successives, les HMMs sont généralisés dans [Wellekens 87] en définissant une nouvelle probabilité d'émission d'observations qui prend en compte la corrélation entre les vecteurs de traits successifs, pour le cas continu. Les formules de ré-estimations sont, ainsi, redéfinies, à la fois, pour l'algorithme de Baum-Welch et pour l'apprentissage par la méthode de Viterbi [Wellekens 87]. La reconnaissance utilisant ces nouveaux modèles n'exigera que le remplacement des probabilités d'emission classiques par celles corrélées dans l'algorithme de Viterbi.

1.4 Alternatives au modèle HMM discret

1.4.1 Modèle HMM continu

Jusqu'à présent, nous n'avons considéré que le cas où les observations sont caractérisées par des symboles discrets obtenus à partir d'un alphabet fini. Par conséquent, nous pouvons utiliser des densités de probabilités discrètes à chaque état du modèle. Le problème avec cette approche, au moins pour certaines applications, est que les observations proviennent de phénomènes continus. Bien qu'il soit possible de quantifier de tels signaux continus à l'aide d'un dictionnaire, il y a une sérieuse dégradation d'information, associée à cette quantification. Il sera, alors, avantageux d'être capable d'utiliser des HMMs avec des densités de probabilités d'observations continues.

Cependant pour utiliser une densité continue de probabilité d'observations, certaines restrictions doivent être précisées sur la forme du modèle de la fonction de densité de probabilité pour pouvoir assurer que les paramètres de cette fonction peuvent être estimés de façon consistante. La représentation la plus générale pour laquelle une procédure de ré-

estimation a été bien formulée est une fonction de mélange finie de la forme [Rabiner 88] :

$$b_j(x) = \sum_{m=1}^{M} c_{jm} . \eta[x, \mu_{jm}, U_{jm}] \qquad 1 \leq j \leq N$$

où :
j : est l'état courant du HMM;
x : est le vecteur observation à modéliser (d est le nombre de composantes);
M : est le nombre de mélanges;
c_{jm} : est le gain du $m^{ème}$ mélange à l'état j;
η : est une densité symétrique (exemple : Gaussienne);
μ_{jm} : est le vecteur moyen pour le mélange m, à l'état j;
U_{jm} : est la matrice (d,d) de covariance pour le mélange m, à l'état j (si X_u et X_v, $1 \leq u, v \leq d$, sont deux variables aléatoires réelles qui prennent pour valeurs respectivement les valeurs des composantes u et v des vecteurs observations à modéliser dans l'état j pour le mélange m, les éléments de la matrice U_{jm} seront de la forme $\text{Var}(X_u)=E(X_u^2) - E(X_u)^2$ si $u = v$ et $\text{Covar}(X_u, X_v)=E(X_u X_v) - E(X_u)E(X_v)$ si $u \neq v$, avec E : la moyenne).

Les gains des mélanges doivent satisfaire aux contraintes stochastiques :

$$c_{jm} \geq 0 \qquad 1 \leq j \leq N, \quad 1 \leq m \leq M \quad et \quad \sum_{m=1}^{M} c_{jm} = 1 \qquad 1 \leq j \leq N$$

ainsi que pour la fonction de densité de probabilité :

$$\int_{-\infty}^{+\infty} b_j(x)dx = 1 \qquad 1 \leq j \leq N$$

La figure 1.16 présente un état d'un HMM, ayant une fonction de densité de probabilité de M mélanges et son ensemble d'états équivalent, ayant un unique mélange par état.

Il a été démontré que les formules de ré-estimations pour les coefficients de la densité de mélange sont les suivantes [Baum 70,Baum 72] :

$$\bar{c}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j) [\frac{c_{jk}\eta(O_t, \mu_{jk}, U_{jk})}{\sum_{m=1}^{M} c_{jm}\eta(O_t, \mu_{jm}, U_{jm})}]}{\sum_{t=1}^{T} \gamma_t(j)}$$

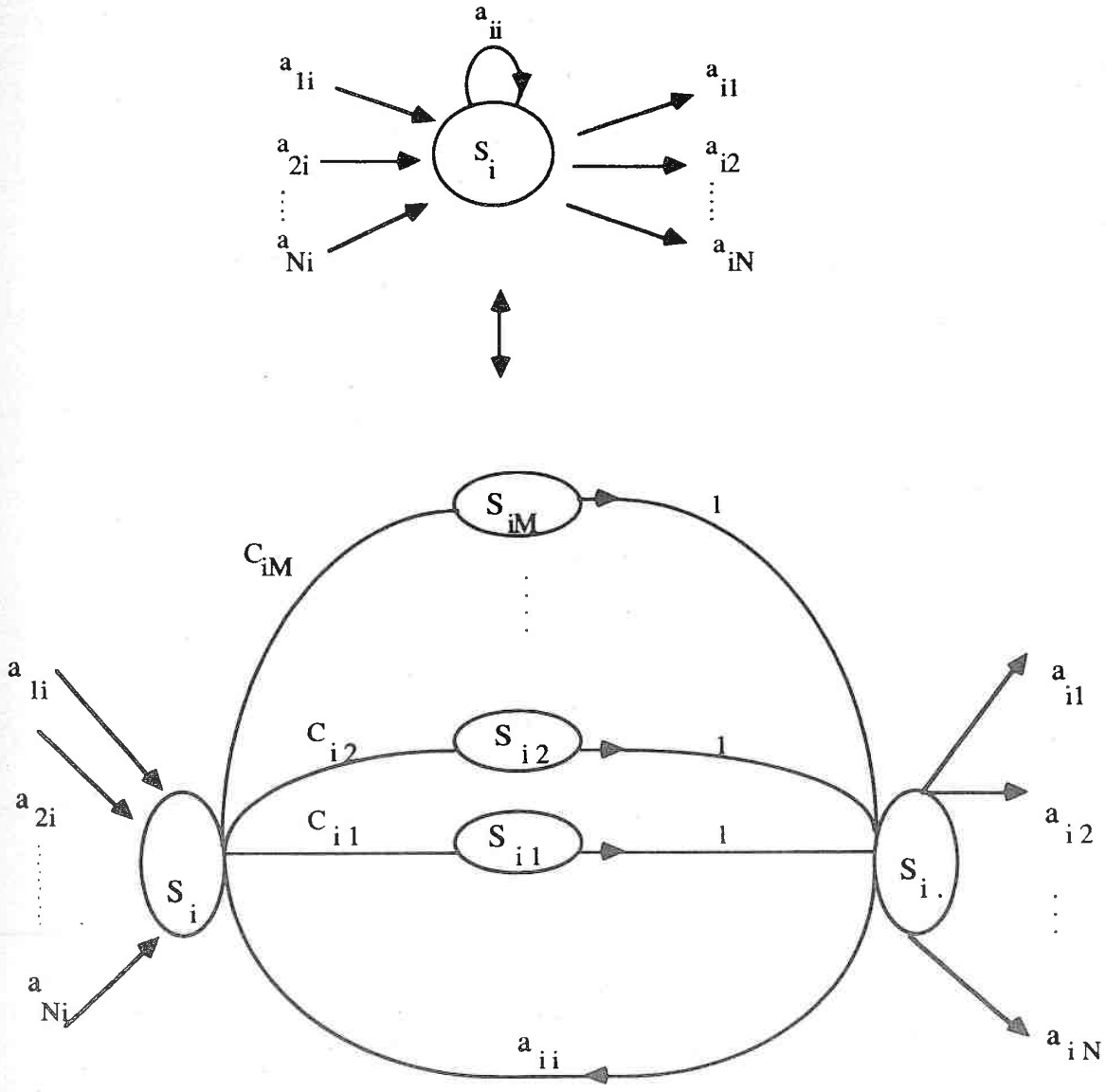


Figure 1.16. équivalent d'un état de M mélanges

$$\bar{\mu}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j)[\frac{c_{jk}\eta(O_t,\mu_{jk},U_{jk})}{\sum_{m=1}^{M} c_{jm}\eta(O_t,\mu_{jm},U_{jm})}]O_t}{\sum_{t=1}^{T} \gamma_t(j)[\frac{c_{jk}\eta(O_t,\mu_{jk},U_{jk})}{\sum_{m=1}^{M} c_{jm}\eta(O_t,\mu_{jm},U_{jm})}]}$$

$$\bar{U}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j)[\frac{c_{jk}\eta(O_t,\mu_{jk},U_{jk})}{\sum_{m=1}^{M} c_{jm}\eta(O_t,\mu_{jm},U_{jm})}](O_t-\mu_{jk})(O_t-\mu_{jk})'}{\sum_{t=1}^{T} \gamma_t(j)[\frac{c_{jk}\eta(O_t,\mu_{jk},U_{jk})}{\sum_{m=1}^{M} c_{jm}\eta(O_t,\mu_{jm},U_{jm})}]}$$

où : γ_t est la quantité définie dans le premier chapitre pour déterminer les formules de ré-estimations dans le cas des HMMs discrets et O_t le vecteur observation à la trame t correspondant au symbole observation O_t dans le cas du discret.

En général, on choisit pour η une densité gaussienne et on aura alors affaire à une fonction $b_j(x)$ de densité de probabilité appelée multi-gaussienne. Dans le cas d'un seul mélange gaussien, la fonction se réduit alors à la formule gaussienne bien connue :

$$b_j(O_t) = \frac{1}{(2\pi)^{\frac{1}{2}}\sigma_j} \exp \frac{-(O_t-m_j)^2}{2\sigma_j^2}$$

avec :

$$\bar{m}_j = \frac{\sum_{t=1}^{T} \gamma_t(j) O_t}{\sum_{t=1}^{T} \gamma_t(j)} \quad , \quad \bar{\sigma}_j^2 = \frac{\sum_{t=1}^{T} \gamma_t(j)(O_t-m_j)^2}{\sum_{t=1}^{T} \gamma_t(j)}$$

En résumé, plusieurs types de densité de probabilité d'émission peuvent être utilisés, mais les modèles les plus utilisés sont les fonctions de densité de probabilités discrètes et les distributions multivariables gaussiennes.

Dans le cas discret, la sortie est une séquence de symboles appartenant à un alphabet fini. Pour celà, une quantification vectorielle est souvent appliquée sur les vecteurs d'entrée [Levinson 83b,Cravero 84,Cravero 86,Mariani 87,Merialdo 87]; ensuite les HMMs sont mis en correspondance avec la séquence des symboles appartenant à l'alphabet et dérivés de l'entrée.

Dans le cas continu, la distribution associée à chaque état est la plupart du temps de la forme d'une densité de probabilité multivariable gaussienne [Juang 85a,Rabiner 85, Juang 86,Jouvet 87]. Il semble en effet selon divers auteurs, dont notamment [Rabiner 83] et [Juang 85b], que les distributions continues donnent de meilleures performances de reconnaissance que les distributions discrètes. Une comparaison de ces méthodes a été faite dans [Rabiner 83] au niveau de la complexité des calculs des paramètres pour les modèles de mots isolés multi-locuteurs.

Notons, cependant, que le nombre des paramètres à estimer augmente considérablement dans le cas continu. Pour diminuer ce nombre et pouvoir mener pratiquement les calculs, on se ramène presque toujours à deux restrictions majeures :

1. Le choix a priori de la loi de probabilité des vecteurs acoustiques.
2. La supposition d'indépendance entre ces vecteurs acoustiques. Chaque vecteur observation étant indépendant de toute autre vecteur.

On choisit, en génèral, des densités gaussiennes et on suppose, en plus, que la matrice des covariance est diagonale.

Pour éviter d'avoir à faire une quelconque hypothèse sur la loi de probabilité des vecteurs acoustiques d'une part, et afin de réduire le nombre des paramètres des modèles HMMs pour ne pas exiger un très grand corpus d'apprentissage d'autre part, nous avons choisi pour nos applications d'implanter des modèles discrets. Nous procédons, ainsi, à une quantification vectorielle préalable dans chaque application afin de réduire les entrées acoustiques à un dictionnaire fini.

1.4.2 Modèle HMM semi-continu

Dans le cas du HMM discret, la classification du vecteur symbole, O_t, à partir d'un vecteur acoustique, x, peut ne pas être précise. Par exemple, le vecteur x, peut être à une distance à peu près équivalente des deux vecteurs symboles ou plus. Par ailleurs, le HMM discret fait moins de suppositions concernant la nature des observations; il est plus facile à implanter et offre un calcul plus rapide par rapport au HMM continu. Cependant, les erreurs de la QV et l'information perdue par ses opérations font que les performances du HMM discret sont significativement inférieures à celles du HMM continu quand les données d'apprentissage sont suffisantes pour ce dernier. La raison majeure des utilisations du HMM discret est que le HMM continu a beaucoup plus de paramètres à estimer, et que ceci est souvent peu compatibles avec des données d'apprentissage limitées.

Dans le cas du HMM discret, l'information perdue par la QV peut être diminuée si, durant le décodage, le vecteur acoustique observation est utilisé par le HMM discret au lieu du symbole observation discret correspondant. En partant de cette idée, Huang et Jack [Huang 88] avaient le souci d'intégrer des qualités du HMMs discret et du HMM continu pour former un HMM plus utile. Ils ont, ainsi, proposé un HMM semi-continu (SCHMM) visant à placer la distortion de la quantité vectorielle (QV) dans la méthodologie générale des HMMs sous un cadre probabiliste. En effet, ils ont proposé un SCHMM qui étend le HMM discret en remplaçant la probabilité du symbole observation discret par une combinaison des probabilités de symboles d'observations discrets et des fonctions de densités de probabilités continues dérivées du dictionnaire de la QV. Les fonctions de densités de probabilités continues de ce dictionnaire sont utilisées dans l'algorithme du décodage, Viterbi modifié, pour pondérer les vecteurs acoustiques observés et les paramètres du HMM discret. Ainsi, avec les paramètres du HMM discret, la QV est supprimée dans la procédure du décodage et le vecteur acoustique d'observation est utilisé. La fonction de densité de probabilité qu'un état s_j produit un vecteur observation x à l'instant t sera alors, dans un SCHMM, comme suit :

$$f(x/q_t = s_j) = \sum_{k=1}^{M} f(x/O_t = v_k) P(O_t = v_k/q_t = s_j)$$

Après quelques simplifications de notations, cette formule peut s'écrire :

$$f_j(x) = \sum_{k=1}^{M} f(x/v_k) b_j(k)$$

où :

M : est le nombre de symboles dans le dictionnaire de la QV.
v_k : est un élément du dictionnaire (vecteur symbole).
Les fonctions de densités de probabilités du dictionnaire, $f(x/v_k)$ $(1 \leq k \leq M)$, peuvent être estimées durant la QV, en les supposant Gaussiennes. Elle peuvent donc être représentées par la moyenne et la covariance de chaque vecteur symbole. L'équation précédente peut être simplifiée en ne considérant que le terme maximal, par exemple, ou les termes les plus significatifs.

L'algorithme d'apprentissage reste inchangé pour obtenir les paramètres du HMM discret $(b_j(k),...)$. Avec l'équation précédente, il est possible de combiner, dans le processus du décodage, les caractéristiques de distortion du dictionnaire de la QV et les paramètres du HMM discret sous un cadre probabiliste unifié. Le décodage direct du vecteur acoustique observation est, donc, devenu possible avec les paramètres du HMM discret.

Les résultats expérimentaux dans [Huang 88] montrent que la précision de la reconnaissance avec les modèles SCHMMs est améliorée par rapport à celle des HMMs discrets et à celle des techniques de la programmation dynamique. En effet, en locuteur-indépendent (50 locuteurs pour l'apprentissage, 49 locuteurs pour le test, chiffres isolés), l'amélioration du taux de reconnaissance, par rapport à 88.9% celui du HMM discret, est de :

- 0.5% pour un SCHMM avec une matrice de covariance supposée diagonale.
- 1.3% pour un SCHMM avec une matrice de covariance pleine. L'inconvénient est que ce cas demande beaucoup plus de données d'apprentissage.

1.5 Conclusion

Les fonctions Forward, Backward, les algorithmes de Viterbi et de Baum-Welch, présentés dans ce chapitre, forment un tout pour mettre en oeuvre ces modèles stochastiques particuliers que sont les HMMs dans un système de reconnaissance de la parole ou de toute autre application du même type. Mais, d'autres problèmes et considérations techniques, autour de ces bases théoriques, doivent être pris en compte pour pouvoir implanter de tels systèmes. Les chapitres suivants permettront l'éclaircissement de ces problèmes et la présentation des diverses solutions proposées.

La théorie des HMMs est, aussi, à la base de plusieurs applications autres que la reconnaissance automatique de la parole. Citons, par exemple :

- La modélisation des langages [Cave 80,Katz 87]

- La théorie de codage [Bahl 74]
- Le traitement du signal [Esin 77]
- La modélisation des finances [Cover 84]
- Le contrôle biologique [Vardi 85]
- La biostatistique [Ott 77]
- La reconnaissance de l'écriture [Faray 79,Kordi 87,Kundu 88,Anigbogu 89].
- Le codage d'images [Mao 90]

2
Modèles markoviens cachés et reconnaissance de la parole

2.1 Généralités

2.1.1 Reconnaissance automatique de la parole

Un des buts de la recherche actuelle en informatique est de rendre plus facile la communication homme-machine, en particulier grâce au développement de nouveaux modes de communication (langage naturel, écriture, vision, parole,...). Dans ce contexte, la communication parlée occupe une place privilégiée, par l'importance et la spécificité de la parole pour l'homme (chacun est, ou croit être, un expert en parole), et aussi du fait des immenses possibilités qu'offrirait une communication orale sans trop de contraintes avec une machine [Haton 85].

Il est évident qu'il est plus agréable, lors d'une communication avec une machine, d'établir le dialogue directement comme on le ferait avec un être humain. Or le dialogue s'est toujours fait à partir d'un clavier ou avec des moyens encore plus rudimentaires. Un clavier ne permet pas de réagir spontanément lors d'une apparition d'un phénomène particulier dans un processus industriel par exemple. Le pilote d'avion pourra être libéré des tâches manuelles qu'il a à faire devant son tableau de bord. La parole libère également l'opérateur de son poste de travail, au lieu de garder le regard rivé sur son écran, celui-ci peut se déplacer, effectuer une autre tâche, en attendant d'être averti par le terminal du déroulement de la première tâche. La parole peut être introduite dans un système de surveillance des malades graves, ainsi le médecin de garde peut être averti de tout changement d'état de ses patients. Nous pouvons imaginer de nombreuses applications une fois que la reconnaissance automatique de la parole (R.A.P.) est bien maîtrisée [Boyer 87a]. De nombreuses applications du dialogue oral homme-machine font intervenir simultanément plusieurs volets de traitement automatique de la parole, plus ou moins maitrisés : reconnaissance, synthèse, transmission, vérification du locuteur, ... Nous ne nous intéresserons qu'à la reconnaissance automatique de la parole dont les recherches, jusqu'à nos jours, n'ont pas pu parvenir, malgré leur intensité, à des résultats convenables, surtout dans le domaine de la reconnaissance de la parole continue.

La R.A.P. débutait à l'époque des années 40, de façon presque inaperçue, en URSS (premiers travaux par Mjasnikov en 1943) et en Suisse [Dreyfus 49]. D'ailleurs les rétrospectives historiques commencent souvent avec les travaux menés aux USA [Davis 52]..etc.

Avec l'informatique, la numérisation du signal de parole va ouvrir la voie au traitement du signal (FFT,LPC,Cepstre), à l'analyse des données (analyse factorielle, des correspondances, multidimensionnelle). Lorsqu'au début des années 50 les chercheurs vont s'attaquer au problème de la reconnaissance, ils croyaient que le problème allait être rapidement résolu. A la fin des années 60, à partir des résultats obtenus par une cinquantaine de systèmes de reconnaissance, un premier bilan a pu être dressé : il est possible d'obtenir un taux de reconnaissance de 95% pour des mots isolés, d'un vocabulaire limité, enregistré dans de bonnes conditions, avec un petit nombre de locuteurs. Ainsi, en 1969, un chercheur des laboratoires BELLs, J.PIERCE tire une sonnette d'alarme en remettant en cause fondamentalement la R.A.P. [Pierce 69]. Son argumentation repose sur le fait que la communication fonctionne entre locuteur et auditeur parce que tous deux ont en commun, non seulement la connaissance du langage, mais aussi l'intelligence de la situation. En conséquence, W.A.LEA ([Lea 70]) propose de construire des systèmes de reconnaissance avec de la parole "limitée". En 1971, il y a eu le lancement du programme ambitieux ARPA-SUR par le département de la défense des USA. Son but est d'accepter la parole continue, avec un vocabulaire de 1000 mots, en utilisant une syntaxe artificielle, pour une tâche déterminée. Cinq laboratoires sont contractants, trois systèmes vont être présentés. Un seul d'entre eux (HARPY de Carnegie-Mellon University) va atteindre des spécifications révisées : 184 phrases, 5 locuteurs, 1011 mots, 5% d'erreurs sémantiques (42% de reconnaissance phonétique).

A la même époque, et ensuite, la recherche française était également très active dans ce secteur, et produisit des systèmes intéressants : Myrtille I et II [Haton 76,Pierrel 82] au C.R.I.N. de Nancy; Keal [Mercier 77] au C.N.E.T. de Lannion; Esope [Mariani 78] au L.I.M.S.I. à Orsay; le système de l'I.C.P. à Grenoble [Groc 80] et Arial II [Perennou 82] au C.E.R.F.I.A. de Toulouse.

A partir de 1976, à la suite du rapport ARPA-SUR un certain nombre de certitudes ont commencé à s'installer dans la communauté scientifique internationale : le signal de parole n'est pas un objet physique comme les autres et les techniques classiques de traitement du signal, d'analyses de données, de reconnaissance des formes ne permettront pas de résoudre, seules, les problèmes. Il faut obligatoirement prendre en compte les connaissances phonétiques, phonologiques, lexicales, syntaxiques, pragmatiques, voire sémantiques. Grâce à celles-ci, il est possible d'améliorer le taux de reconnaissance à la sortie du décodage acoustico-phonétique, mais la reconnaissance automatique bute à ce premier niveau de traitement [Boe 88].

Depuis, en se fixant des contraintes : vocabulaire déterminé, nombre limité de locuteurs, apprentissage, structures syntaxiques formalisées,...,la R.A.P. a vu des systèmes fournissant des performances convenables. Les premiers systèmes de reconnaissance de mots isolés ont été commercialisés aux Etats-Unis. Actuellement, des centaines de systèmes sont proposés sur le marché par de nombreuses firmes américaines, japonaises, françaises, anglaises, etc. Cependant, de nombreux problèmes fondamentaux demeurent ouverts, surtout, concernant la parole continue de nombreux locuteurs, prononcée naturellement.

De façon générale, la R.A.P. consiste à transcrire l'onde acoustique en symboles. La grande complexité de la R.A.P. provient essentiellement de certaines caractéristiques spécifiques du signal vocal. La parole est caractérisée par une grande variabilité à la fois interlocuteur et intralocuteur. En effet, le conduit vocal présente des différences phy-

siologiques d'un locuteur à un autre. Par ailleurs, les limitations d'ordre mécanique de l'appareil phonatoire conduisent à une forte dépendance contextuelle des sons émis. Les phénomènes articulatoires, le contrôle des cordes vocales, le volume sonore ne sont pas précis. Ils varient en fonction du contexte, l'émotion et la fatigue. Lorsque la vitesse d'élocution est trop élevée, les positions articulatoires normales ne sont pas atteintes. Il en résulte de grandes variations difficiles à analyser.
Pour une même personne, l'élocution d'une phrase ou d'un mot n'est jamais constante au cours d'une répétition. En effet, la diction fluctue avec l'humeur, l'état de santé, l'environnement dans lequel évolue le locuteur. Ainsi, la vitesse d'élocution, la fréquence fondamentale du signal de parole, l'énergie du son émis et le rythme sont susceptibles d'évoluer de façon significative.
Cette variabilité augmente lorsqu'on passe d'un locuteur à un autre. Parmi les facteurs introduisant de grandes différences de prononciation entre différents locuteurs on peut citer : l'âge, le sexe et l'accent régional.

Une autre caractéristique du signal de parole est l'absence de marques de segmentation entre les mots d'un même énoncé. De ce fait, la localisation des mots dans le signal vocal est une opération difficile nécessitant le plus souvent le recours à des connaissances linguistiques.

En résumé, les facteurs de complexité dans le traitement automatique de la reconnaissance de la parole sont :

- Reconnaissance mono-locuteur, multi-locuteur ou locuteurs indépendants
- Reconnaissance de mots isolés, mots enchaînés ou parole continue
- Taille du vocabulaire
- Langage très contraint ou non
- Environnement bruité ou non

2.1.2 Les approches du problème

Il est clair que les méthodes à appliquer pour la R.A.P. sont étroitement liées aux différents types de problèmes à résoudre. Pour la reconnaissance "mono-locuteur, mots enchaînés" ou "multi-locuteur, petit vocabulaire, mots isolés", par exemple, une méthode globale semble parfaitement convenir, tandis que pour des "grands vocabulaires, multi-locuteurs, parole continue", la méthode analytique s'impose :

- Les méthodes globales consistent à comparer globalement le mot inconnu avec tous les mots du vocabulaire. Le mot du vocabulaire qui satisfait au mieux le critère de comparaison avec le mot à identifier est retenu si le taux de ressemblance dépasse un certain seuil. Un inconvénient de ces méthodes est qu'elles nécessitent le stockage de plusieurs références pour chaque mot du vocabulaire et le temps de réponse du système augmente avec la taille du vocabulaire .
- Les méthodes analytiques, fondées sur une analyse très fine du signal de parole, consistent à représenter chaque mot par une séquence d'unités phonétiques (phonèmes,

diphonèmes, syllabes ...). Les unités sont décrites une fois pour toutes ce qui est un avantage pour la taille des applications : Le stockage de ces descriptions d'unités est indépendant de la taille du vocabulaire.

Les systèmes de la R.A.P. utilisent principalement trois approches suivant le type de la tâche à résoudre :

- *"DTW"* : Lorsque les mots sont prononcés isolément, leur identification se ramène à un problème classique de reconnaissance de formes qui peut être résolu par une méthode globale. La programmation dynamique fournit à ce type de problème une solution optimale largement utilisée [Sakoe 71,Sakoe 78,Haton 82]. Le même principe peut être étendu, de plusieurs façons, à la reconnaissance de mots enchaînés, c'est-à-dire prononcés continûment mais sans lien logique (syntaxique) entre eux [Levinson 80,Sauter 84,Charpillet 84,Boyer 87a].

- *"IA"* : Les approches analytiques, relevant naturellement des traitements de type intelligence artificielle, sont surtout utilisées pour la reconnaissance de la parole continue. Les approches globales sont difficilement généralisables au cas de la parole continue, tant par le volume des informations à stocker pour représenter les formes acoustiques d'un grand lexique que par la relative inefficacité des algorithmes de comparaison des formes devant la variabilité de la parole naturelle. La méthode IA consiste à considérer que la reconnaissance de la parole continue ne peut être résolue qu'en utilisant à la fois des connaissances spécifiques (acoustique, phonétique, linguistique,...), et des techniques générales de recherche de solution dans un espace [Haton 85]. Parmi les systèmes de reconnaissance existants, un grand nombre relève de l'approche IA. Nous citons Hearsay I et II, Harpy, Hwin aux USA, Myrtille I et II [Pierrel 82] et Aphodex [Fohr 86] du C.R.I.N., Sonex du L.I.M.S.I. [Stern 86] et Serac du C.N.E.T. Lannion [Gilloux 84].

- *"HMM"* : L'approche stochastique, issue de la théorie de l'information, consiste à modéliser l'ensemble des opérations de reconnaissance de parole en termes de processus stochastiques. Les données probabilistes sont alors apprises au préalable à partir d'un corpus important de prononciations d'apprentissage. Le processus stochastique est en général un processus de Markov du premier ordre (le plus souvent un HMM). Cette approche est adaptée aux systèmes multi-locuteur, mots isolés ou mots enchaînés et nous verrons aussi qu'elle est utilisable pour la parole continue. C'est celle qui nous intéresse dans le présent travail.

La quantification vectorielle (QV) est une approche statistique qui peut être utilisée dans un système de R.A.P. ou dans une de ses étapes. Elle consiste, dans l'espace des paramètres du signal, à définir, à partir du nuage des points représentant les prononciations d'apprentissage, par une technique d'analyse de données (clustering) quelques points (prototypes) représentatifs pour chaque mot à reconnaître [Flocon 84,Gray 84]. La QV est bien adaptée aux systèmes multi-locuteur, elle donne une repésentation statistique de chaque mot du vocabulaire à partir des prononciations d'un grand nombre de locuteurs. Elle est plutôt utilisée pour la reconnaissance de mots isolés. Cependant, par exemple, une extension à la reconnaissance de la parole continue était présentée dans [Gourinda 88].

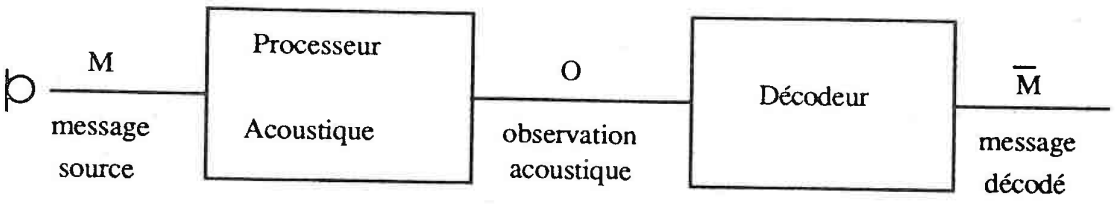


Figure 2.1. R.A.P. en théorie de l'information

2.1.3 Approche de la théorie de l'information

La R.A.P. peut être perçue comme un problème de décodage classique en théorie de l'information [Jelinek 75,Baker 75b,Baker 75a,Baker 75c,Bahl 83]. La formulation du problème s'énonce comme suit [figure 2.1] :

- Le locuteur est considéré comme une source qui émet des messages.
- Chaque message, M, traverse un canal appelé canal acoustique composé de l'appareil phonatoire du locuteur, de l'espace de propagation entre le locuteur et le microphone et du processeur acoustique qui transforme le signal reçu en une suite acoustique (suite d'observations O)
- Le décodeur linguistique transforme cette suite en un message décodé, $\bar{M}$.

Pour trouver le message $\bar{M}$, on a le choix entre deux types de décodage :

- Le décodage à maximum de vraisemblance, qui consiste à maximiser $P(O/M)$ sur l'ensemble des messages possibles. Ce premier type de décodage est préférable lorsque l'on n'a aucune idée sur les probabilités a priori des messages, i.e. qu'on les considère comme toutes égales, ce qui peut être le cas lorsqu'on fait simplement de la reconnaissance de mots [Calliope 89].
- Le décodage à minimum de probabilité d'erreur. La théorie de l'information affirme que la probabilité d'erreur sur le message décodé est minimale [Falaschi 88] si le décodeur a pour sortie le message qui a la probabilité d'être transmis, la plus élevée quand l'observation O est connue. C'est à dire :

$$P(\bar{M}/O) = \max_{M} P(M/O)$$

et en appliquant la formule de Bayes, on obtient :

$$P(\bar{M}/O) = \max_{M} \frac{P(O/M)P(M)}{P(O)}$$

C'est ce type de décodage qui est adapté à la reconnaissance du discours parce qu'on ne désire pas forcément que les probabilités de suites de mots quelconques soient identiques. Le dénominateur de l'équation ne dépend pas de M et il suffit de maximiser le numérateur, la suite de mots décodée est donc :

$$\bar{M} = \arg\max_{M} P(O/M)P(M)$$

Comme précisé dans [Cerf 86], cette approche, basée sur la théorie de l'information, fait ressortir quatre problèmes à résoudre pour construire un système de la R.A.P. :

1. La paramétrisation : choisir les paramètres acoustiques qui caractérisent l'observation, et réaliser le processeur acoustique qui les extrait du signal.
2. La modélisation acoustique : élaborer un modèle acoustique qui indique la façon dont les messages sont transmis sous forme d'observations acoustiques, en permettant de calculer $P(O/M)$.
3. La modélisation linguistique : élaborer un modèle linguistique qui permette d'estimer la probabilité de chaque message, $P(M)$.
4. La maximisation : comment trouver pratiquement le meilleur message.

La puissance de cette formulation tient dans sa généralité. En effet, on n'a pas encore précisé quel était le langage (ensemble des messages) considéré, ni quelle était l'observation acoustique, ni comment on réalise les différents modèles (en particulier, on n'a pas encore parlé de source de Markov). Mais cette formule indique quels sont les éléments du système, et surtout, comment se réalise la coopération des niveaux acoustique et linguistique. Cette coopération se fait à partir des probabilités des modèles acoustique et linguistique, en maximisant leur produit.

En théorie de l'information, il est courant de modéliser une source quelconque par une source de Markov. Ces sources ont l'avantage d'être simples, leurs propriétés sont bien connues d'un point de vue théorique, il est facile de voir quelle approximation elles font d'une source réelle, et enfin il existe des algorithmes puissants pour les utiliser efficacement [Cerf 86].

Les sources markoviennes cachées sont particulièrement appropriées pour décrire le signal de parole : D'une part, elles permettent une grande réduction du nombre de paramètres à estimer par rapport à une simple source de Markov tout en gardant un aspect Markovien. D'autre part, il existe des algorithmes qui permettent de calculer les paramètres optimaux d'une source Markovienne cachée en fonction d'une suite d'observations de

symboles produits, sans même connaître la suite exacte des états qui a produit cette suite d'observations. Par exemple l'algorithme itératif de Baum-Welch provenant de l'égalité introduite par Baum dans [Baum 72]. Comme nous l'avons déjà vu, cet algorithme cherche, à partir d'une structure initiale de la source de Markov M, à maximiser la probabilité $P(O/M)$ que la suite d'observations O, soit produite par la source.

2.1.4 Historique

Nous allons présenter par la suite quelques dates qui ont marqué l'histoire de l'évolution des recherches sur les chaînes de Markov et leur application à la R.A.P. :

- 1913 : La théorie des chaînes de Markov est bien connue. Markov a utilisé ses processus pour analyser le langage [Markov 13].
- 1948..1951 : Les travaux de Shannon sur la théorie de l'information utilisant les chaînes de Markov [Shannon 48,Shannon 51].
- 1957 : Les travaux de Bellman sur la programmation dynamique [Bellman 57].
- 1958..1961 : Les premières applications intéressantes. Par exemple :
  - Les modèles probabilistes d'urnes [Feller 58].
  - Calcul direct du maximum de vraisemblance [Hartley 58].
  - La suite d'états est observée dans une chaîne de Markov [Billingsley 61].
- 1966..1974 : C'est une période marquante dans l'histoire des HMMs. Pendant cette période sont découvertes des algorithmes puissants pour l'estimation des paramètres du modèle HMM d'une part et pour le décodage de la suite cachée d'autre part :
  - Baum, Eagon, Soules, Weiss et Petrie [Baum 66,Baum 67,Baum 68,Petrie 69, Baum 70,Baum 72] travaillaient sur la récupération des états cachés, l'estimation itérative du maximum de vraisemblance pour déterminer les paramètres du modèle et les preuves de consistance de ces estimations. Ils essayaient, aussi, d'inclure des généralisations permettant, ainsi, d'avoir lieu à d'autre études dans le futur telles que les HMMs à durée d'état variable [Ferguson 80b] et à densités de probabilités continues multivariables [Liporace 82].
  - La construction des deux algorithmes puissants de décodage utilisés en théorie de l'information et qui vont servir par la suite au décodage de la parole. D'une part, l'algorithme de Viterbi [Viterbi 67,Forney 73] qui a une complexité de calcul (dans le pire cas) linéaire avec la longueur de la suite d'observations à décoder et qui permet de connaître la suite des états du modèle correspondante au meilleur chemin. D'autre part, l'algorithme de Jelinek [Jelinek 69] qui fait une exploration arborescente des chemins possibles en utilisant une pile.
  - Notons aussi qu'en 1970, l'utilisation des HMMs devient plus claire grâce à leur description par l'exemple des urnes de la part de Neuwirth qui a employé pour la première fois l'expression "modèle Markovien caché" (Hidden Markov Model) au lieu de "fonction probabilistique d'une chaîne de Markov" (probabilistic function of a Markov Chain).

  - En 1974, Ferguson a exposé le problème de façon plus explicite et modulaire tout en introduisant ce que nous appelons, aujourd'hui, le voile de Ferguson entre la suite d'états cachée et l'observateur [chapitre 1]. Ce style d'exposition, que nous retrouvons dans [Ferguson 80a,Rabiner 86a], est celui que nous avons adopté au premier chapitre en considérant les trois problèmes de base.

- 1975 : C'est à cette époque qu'une telle approche stochastique (l'utilisation des HMMs) a été introduite parallèlement par un groupe d'IBM [Jelinek 75,Bahl 75] et par Baker à CMU [Baker 75b,Baker 75a,Baker 75c] pour résoudre des problèmes de la R.A.P..

- 1976..1982 : Nous pouvons qualifier cette époque de période de test et d'évaluation de cette nouvelle technique pour la R.A.P.. Nous citons par exemple :

  - Les travaux du groupe IBM-USA sur la R.A.P. continue [Jelinek 76,Bakis 76, Bahl 76,Bahl 79,Bahl 80,Jelinek 80]
  - Les travaux sur l'apprentissage à partir des données insuffisantes [Dempster 77, Jelinek 80].

- 1983..1990 : Ces dernières années, la tendance à utiliser les chaînes de Markov dans les systèmes de la R.A.P. domine. Nous trouvons ainsi un très grand nombre d'applications pour :

  - les mots isolés, en tant qu'alternative à la programmation dynamique [Rabiner 83, Rabiner 84b,Rabiner 86b,Juang 86,Poritz 86,Averbuch 86,Euler 88],
  - les mots enchaînés [Rabiner 85,Rabiner 86c,Rabiner 87,Mari 85b],
  - la parole continue [Bahl 83,Bahl 89,Levinson 87,Lee 88b,Schwartz 84,Schwartz 85, Chow 87,Kubala 88],
  - la localisation de mot (word spotting) [Rosemberg 87,Dours 88,Dours 89,Rohlicek 89];

  des applications pour :

  - les petits vocabulaires [Levinson 83b,Mari 85b,Rabiner 86b,Wellekens 86],
  - les grands vocabulaires [Rabiner 84b,Averbuch 86,Cerf 86,Derouault 87,Lee 88a, Bahl 89]
  - les très grands vocabulaires [Merialdo 87,Merialdo 88b];

  des applications :

  - mono-locuteur [Merialdo 88c,Derouault 89],
  - multi-locuteurs [Boyer 88],
  - indépendantes du locuteur [Rabiner 85,Euler 88,Jouvet 87,Lee 88a]

Cette énorme activité ne se borne pas, bien sûr, aux modèles Markoviens cachés de base, mais plusieurs notions, provenant des extensions théoriques de ces modèles et de leur variantes, ont été introduites dans le but d'améliorer le taux de reconnaissance des applications (MMI, HMMC, mélange, durée, MDI,..etc).

Nous assistons, donc, à l'apparition d'un très grand nombre de systèmes utilisant cette approche basée sur les HMMs. Bien que nous ayons étudié plusieurs systèmes de ce

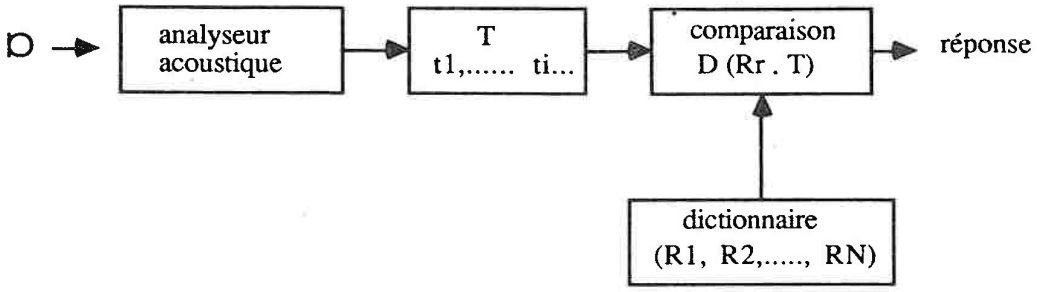


Figure 2.2. reconnaissance de mots isolés [Calliope 89]

type, nous nous bornerons par la suite, principalement, aux travaux du groupe IBM-USA, du groupe IBM-Paris, des laboratoires Bells et BBN et quelques travaux de l'université Carnegie-Mellon (CMU).

2.2 Application à la R.A.P.

2.2.1 Reconnaissance de mots isolés

La reconnaissance de mots isolés, où tous les mots prononcés sont supposés être séparés par des silences de durées supérieures à quelques dixièmes de secondes, se fait, essentiellement, par l'approche globale : Chaque mot du vocabulaire est représenté par une ou plusieurs références acoustiques.

La structure d'un tel système est représentée sur la figure 2.2.

La première phase d'un système de reconnaissance de la parole analyse et paramétrise le signal vocal. Il s'agit de minimiser la quantité d'information nécessaire à la séparation des éléments du vocabulaire. Les paramètres sont, le plus souvent, évalués sur des segments de signal de 20 à 50 ms avec un pas de 5 à 20 ms. Ainsi, un mot R sera représenté par une suite de vecteurs r_i dont les coordonnées $r_{i,n}$ sont les paramètres retenus :

$$R = r_1, r_2, ..., r_i, ..., r_T$$
$$r_i = r_{i,1}, r_{i,2}, ..., r_{i,p}$$

où T est la longueur du mot en trame et, p, le nombre de paramètres évalués sur chaque segment.

Dans l'approche stochastique markovienne qui est similaire à l'approche DTW, on représente chaque référence acoustique R, sous la forme d'un HMM λ. Disposant de n

occurrences d'un même mot $O = O^1, O^2, ..., O^n$, l'évaluation des paramètres du modèle λ est obtenue en maximisant la probabilité de l'ensemble d'apprentissage :

$$P(O/\lambda) = \prod_{k=1}^{n} P(O^k/\lambda)$$

où $P(O^k/\lambda)$ est la probabilité que la $k^{ème}$ séquence d'observations O^k soit générée par le modèle λ. Les formules de ré-estimations sont, par conséquent, légèrement modifiées. La formule de ré-estimation de a_{ij}, par exemple, devient alors :

$$\bar{a}_{ij} = \frac{\sum_{k=1}^{n} \sum_{t=1}^{T_k - 1} \xi_t(i,j)}{\sum_{k=1}^{n} \sum_{t=1}^{T_k - 1} \gamma_t(i)}$$

Les autres formules de ré-estimations sont traitées de la même façon.

L'algorithme itératif de Baum-Welch permet, ainsi, de résoudre ce problème. Il suffit d'initialiser les paramètres par une estimation grossière et après chaque itération, les probabilités du modèle, λ, sont réestimées en fournissant un meilleur modèle, $\bar{\lambda}$, tel que :

$$P(O/\bar{\lambda}) \geq P(O/\lambda)$$

Le processus prend fin lorsqu'il n'y a plus de changement significatif.

Les distributions de probabilités d'émission peuvent être approximées directement à partir de la suite de paramétrisation ($O^i = r_1^i, r_2^i, ..., r_T^i$), si on utilise un HMM continu, ou après une étape de quantification vectorielle ($O^i = O_1^i, O_2^i, ..., O_T^i$ avec O_j^i un élément du dictionnaire de classification représentant le vecteur r_j^i), si on utilise un HMM discret.

Le mot à reconnaître passe par les mêmes étapes que les occurrences d'apprentissage. A partir de sa suite d'observations obtenue O, l'algorithme de Viterbi nous permet d'identifier le modèle, λ^u $(1 \leq u \leq U,$ U : *le nombre de mots dans le vocabulaire*), qui a donné la plus grande valeur de la quantité :

$$\max_{Q} P(Q, O/\lambda)$$

avec Q : un chemin d'états.
Le mot de référence correspondant à ce modèle, λ^u, sera le mot reconnu.

Nous allons présenter au chapitre trois un nouveau reconnaisseur de mots isolés basé sur les HMMs du second ordre. Ce reconnaisseur est comparé à celui utilisant les HMMs du premier ordre sur un corpus de chiffres isolés.

2.2.2 Reconnaissance de mots enchaînés

Les techniques employées pour la reconnaissance de mots isolés, peuvent être étendues à la reconnaissance de séquences de mots prononcés sans pauses (mots enchaînés). Cependant, aux problèmes soulevés par la reconnaissance de mots isolés, viennent s'ajouter, entre autres, la difficulté de la segmentation de la séquence en mots, le problème de la coarticulation entre mots adjacents, et l'accroissement de la quantité de calcul requise par les algorithmes de comparaison.

Les mots composant la séquence à reconnaître ne sont pas séparés par des silences. Leurs frontières ne peuvent donc pas être déterminées par simple analyse de l'enveloppe du signal. Il est alors possible d'envisager deux méthodes pour effectuer la segmentation en mots [Boyer 87a,Calliope 89] :

- Une première méthode consiste à définir des règles de segmentation pour un vocabulaire donné en se basant sur des critères acoustiques. Sambur et Rabiner [Sambur 76] ont établi de telles règles pour les chiffres anglais. Cependant, cette méthode ne peut s'appliquer facilement à des vocabulaires plus importants comprenant des mots de plus d'une ou deux syllabes. De plus, les erreurs inévitables commises à ce niveau viennent limiter les performances du système avant même de considérer l'étage de reconnaissance.

- Une autre solution consiste à ne pas faire de segmentation préalable, les frontières des mots sont alors déterminées au cours de la phase de reconnaissance. Cette méthode repose sur l'idée que le principe de la reconnaissance de mots isolés peut s'appliquer directement à la reconnaissance de mots enchaînés si l'on considère que la forme à identifier est l'image acoustique de toute la phrase [Sakoe 79].

C'est à cette deuxième solution que nous allons nous intéresser dans la suite. Nous verrons comment l'algorithme de Viterbi, qui est utilisé pour la reconnaissance de mots isolés, peut se généraliser à la reconnaissance de mots enchaînés.

Jusqu'à présent la généralisation de l'algorithme de Viterbi est faite, essentiellement, de deux manières inspirées de l'approche programmation dynamique pour la reconnaissance de mots enchaînés [Sakoe 79,Myers 81,Bridle 82,Gagnoulet 82,Gauvain 82]. Ces deux manières sont :

- Les algorithmes "construction par niveaux" (Level Building), utilisés principalement dans les laboratoires Bells [Rabiner 85,Rabiner 88]. La suite optimale de HMMs, correspondant à la suite d'observations à reconnaître, est obtenue par un processus itératif utilisant l'algorithme de Viterbi. Ce processus est répété à travers un nombre de niveaux équivalent au nombre maximum de mots acceptés dans une chaîne à reconnaître.
  L'inconvénient principal de cet algorithme est la récursivité qui est introduite sur le nombre de mots pouvant être contenus dans la forme inconnue. Par conséquent, il est impératif de fixer arbitrairement le nombre maximal de mots pouvant être contenus dans la phrase à identifier afin de donner une condition d'arrêt à l'algorithme de comparaison.

- Les algorithmes de Viterbi généralisés à des réseaux décrivant les grammaires des suites de mots possibles [Mari 85a,Bahl 83,Rabiner 89,Jouvet 86,Jouvet 87]. La suite de mots prononcés est obtenue en gardant trace des transitions entre états dans le réseau linguistique, definissant le meilleur chemin décodé par l'algorithme.

L'apprentissage

Lorsque deux mots sont prononcés successivement sans être séparés par un silence, il y a un phénomène de coarticulation qui, en première approximation, se traduit par une modification de la structure spectrale de chacun des mots au niveau de leur frontière commune. La séquence de deux mots ainsi prononcée n'est donc pas identique à la séquence obtenue par juxtaposition de ces deux mots prononcés isolément.

Une façon de prendre en compte la coarticulation dans la reconnaissance de mots enchaînés, est d'estimer les modèles de mots en utilisant des références extraites de séquences de mots enchaînés segmentées soit manuellement soit par des algorithmes de segmentation automatique. Dans ce cas, en général, plusieurs références sont conservées pour chaque mot suivant les différents contextes et l'apprentissage se décompose, le plus souvent, en deux phases :

- Une première phase au cours de laquelle les références sont estimées à partir d'au moins une prononciation de chaque mot isolé ou d'une segmentation manuelle.
- Une seconde phase où les références d'apprentissage sont extraites de séquences de mots enchaînés qui sont segmentées en les comparant aux références isolées, par exemple, par une procédure d'alignement de Viterbi ou Forward-Backward.

2.2.3 Reconnaissance de la parole continue

Bien que les méthodes les plus adaptées à la reconnaissance de la parole continue soient les méthodes analytiques, plusieurs tentatives ont été faites pour la généralisation des méthodes de reconnaissance globales [Gauvain 83]. Plus particulièrement, les HMMs semblent intéressants dans ce type de généralisation, étant donné leur succès dans les problèmes de la reconnaissance de mots isolés et enchaînés. La reconnaissance de la parole continue est basée sur l'utilisation des unités de parole plus petites que le mot. Les systèmes de ce type de reconnaissance sont marqués par quatre choix importants :

1. Choix des unités de parole : phonème, demi-syllabe, syllabe, etc.. Bien que perceptivement la notion de phonème (unité élémentaire de son) paraisse naturelle et la moins complexe, c'est une unité extrêmement variable. Il n'existe pas obligatoirement de segment (portion de signal) qui puisse lui correspondre.
2. Choix du modèle de mot : possibilité d'utiliser des réseaux déterministes avec les représentations explicites de chaque mot en fonction des unités de parole choisies. Par exemple les réseaux statistiques ou les approches lexicales.
3. Choix de la syntaxe : possibilité d'utiliser des diagrammes d'états de toutes les combinaisons de mots possibles. Par exemple des grammaires stochastiques ou formelles.

4. Choix de la sémantique : description de la tâche du reconnaisseur.

Dans plusieurs systèmes actuels, la théorie des HMMs a été appliquée à chacun des trois premiers choix précédents : représentation des unités de parole, des mots et/ou de la syntaxe par des modèles HMMs.

Dans notre travail, nous ne nous sommes intéressés qu'au premier niveau de ce type de reconnaissance : Le décodage acoustico-phonétique qui occupe une place importante, et reste encore à l'heure actuelle un problème majeur. Parce qu'une mauvaise transformation de l'onde vocale en unités phonétiques étiquetées (un mauvais décodage acoustico-phonétique) entraînera forcément des mauvaises interprétations dans les niveaux supérieurs. Concernant les méthodes globales et particulièrement celle des HMMs, la reconnaissance phonétique se fait, généralement, en reprenant les approches de la reconnaissance des mots enchaînés. Ceci limite les performances avec de nombreux locuteurs par la quantité des informations à stocker et par le temps de calcul.

2.3 Des systèmes prototypes

2.3.1 Le système des laboratoires Bells (1985)

Ce système exposé en 1985 par Rabiner et Levinson [Rabiner 85] intégre la quantification vectorielle, les HMMs et l'algorithme "Construction par niveaux" (ou LB) pour donner un système, HMM/LB, de reconnaissance de mots enchaînés, indépendant du locuteur et à syntaxe directe. Ce système a un temps de calcul modeste, cependant ses performances sont comparables aux précédents reconnaisseurs d'ordre de calcul plus grand, par exemple le système DTW/LB [Myers 81,Myers 82].

La structure générale du reconnaisseur HMM/LB est illustrée dans la figure 2.3. Il utilise une procédure de normalisation de l'énergie globale, appliquée à la chaîne de mots enchaînés entière [Rabiner 84a]. Chaque mot du vocabulaire est modélisé par un HMM. Un modèle de durée de mot gaussien a été incorporé dans l'algorithme LB. Une syntaxe formelle du langage est décrite par un diagramme de transition d'états. Pour évaluer les performances, un système d'information et de réservation des lignes aériennes a été mis en oeuvre. Il utilise un vocabulaire de 129 mots et une syntaxe déterministe de 144 états, 450 transitions d'états et 21 états finaux, générant plus de 6.10^9 phrases [Levinson 78, Levinson 80,Myers 82].
Les modèles en mode locuteur indépendant ont été estimés à partir d'un apprentissage de 100 locuteurs (50 hommes et 50 femmes) prononçant chacun les mots isolément du vocabulaire. Un ensemble de 6 locuteurs expérimentés (5 hommes et 1 femmes) a été utilisé pour le test. Chacun a prononcé 51 phrases suivant la grammaire aérienne. Le taux d'erreurs de mots est de 6.7%.

En conclusion, un système de reconnaissance de mots enchaînés, de dimensions moyennes, peut être efficacement implanté en utilisant des HMMs et l'algorithme LB.

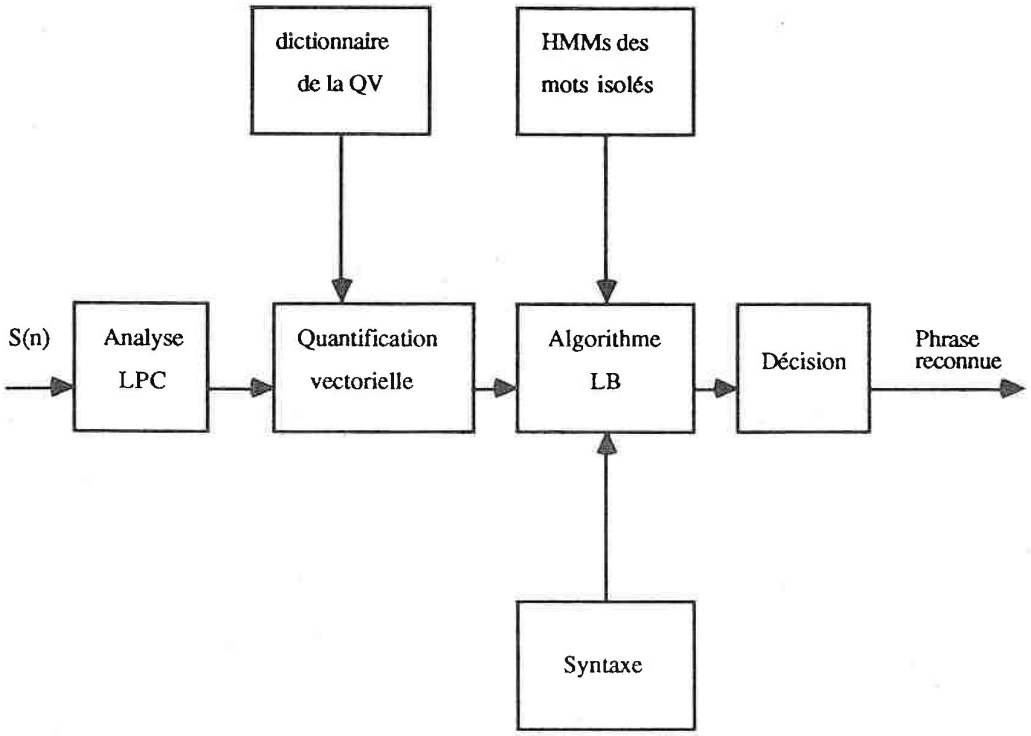


Figure 2.3. Le système HMM/LB

2.3.2 Le système BYBLOS de BBN (1987)

Le système BYBLOS, décrit de façon entière en 1987 par les gens des laboratoires BBN [Chow 87], est conçu pour la reconnaissance de la parole continue en intégrant des sources de connaissance acoustique, phonétique, lexicale, linguistique pour accomplir des performances de reconnaissance élevées. L'approche de base, comme décrit dans [Schwartz 85,Chow 86], utilise des modèles HMMs dépendants des contextes des coarticulations phonétiques.

La structure générale de BYBLOS est illustrée dans la figure 2.4 qui montre les différents modules, sources de connaissance et leurs intéractions dans le système complet. Le module "Apprentisseur" a pour but de produire une base de données des HMMs dépendant des contextes des phonèmes [Chow 87]. Le module "Générateur" compile cette base de données des modèles phonétiques pour produire une base de données des HMMs des mots du vocabulaire. Le décodeur est une stratégie de recherche par faisceau utilisant un algorithme de Baum-Welch modifié [Chow 86].

Le système a été testé pour plusieurs tâches différentes. Par exemple, pour le mode "locuteur dépendant" avec 15 mn de parole d'apprentissage par locuteur, le taux de reconnaissance de mots est de 98.8% pour une tâche de 334 mots de vocabulaire.

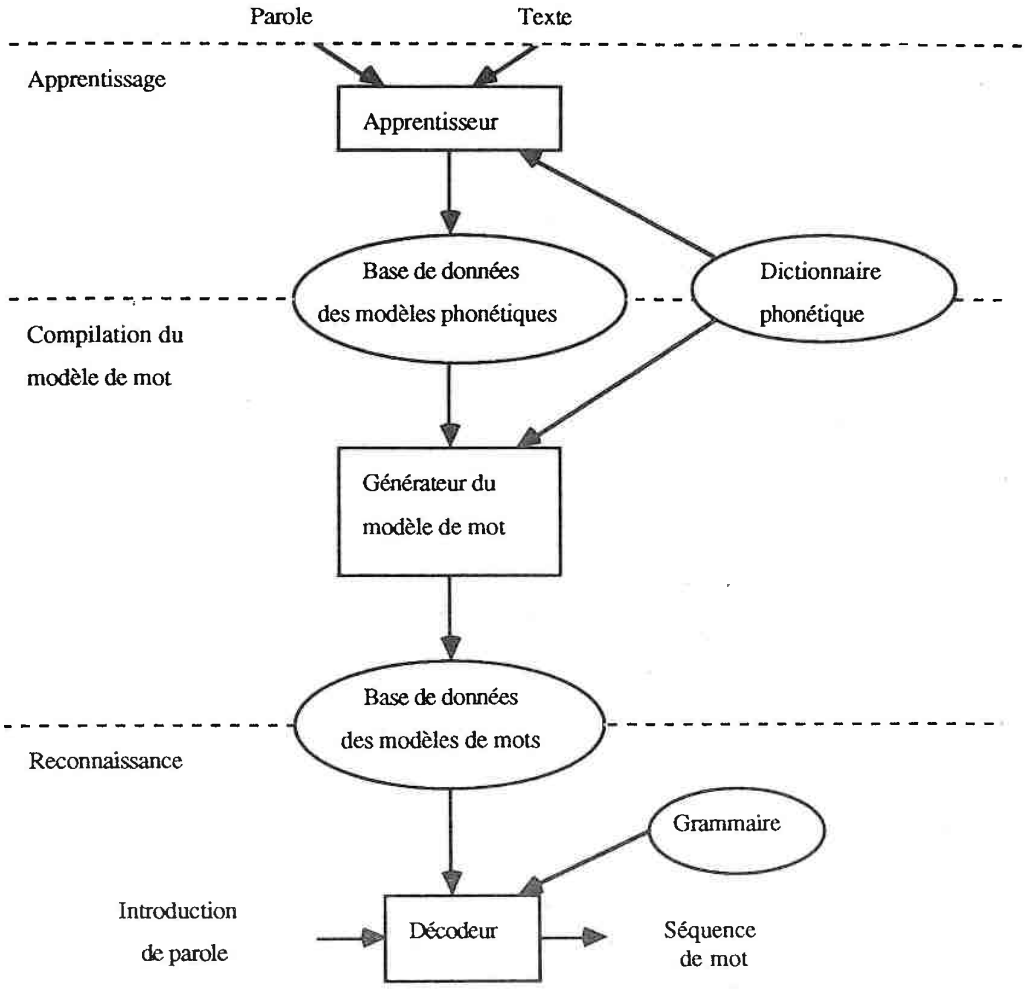


Figure 2.4. Diagramme du système BYBLOS

En conclusion, le système BYBLOS, basé sur l'intégration de plusieurs sources de connaissance, est un système efficace pour la reconnaissance de la parole continue, grand vocabulaire et dépendant du locuteur.

2.3.3 Le système d'IBM-Paris (1987-88)

Le système du centre IBM-Paris fait de la reconnaissance de parole sur un très grand vocabulaire dans le but de faire de la dictée automatique [Merialdo 87,Merialdo 88b]. Les phrases sont prononcées avec des courtes pauses entre les syllabes, de plus le système est monolocuteur. Une syllabe phonétique correspond à une suite d'unités phonétiques et à chaque unité est associée un HMM de 7 états.

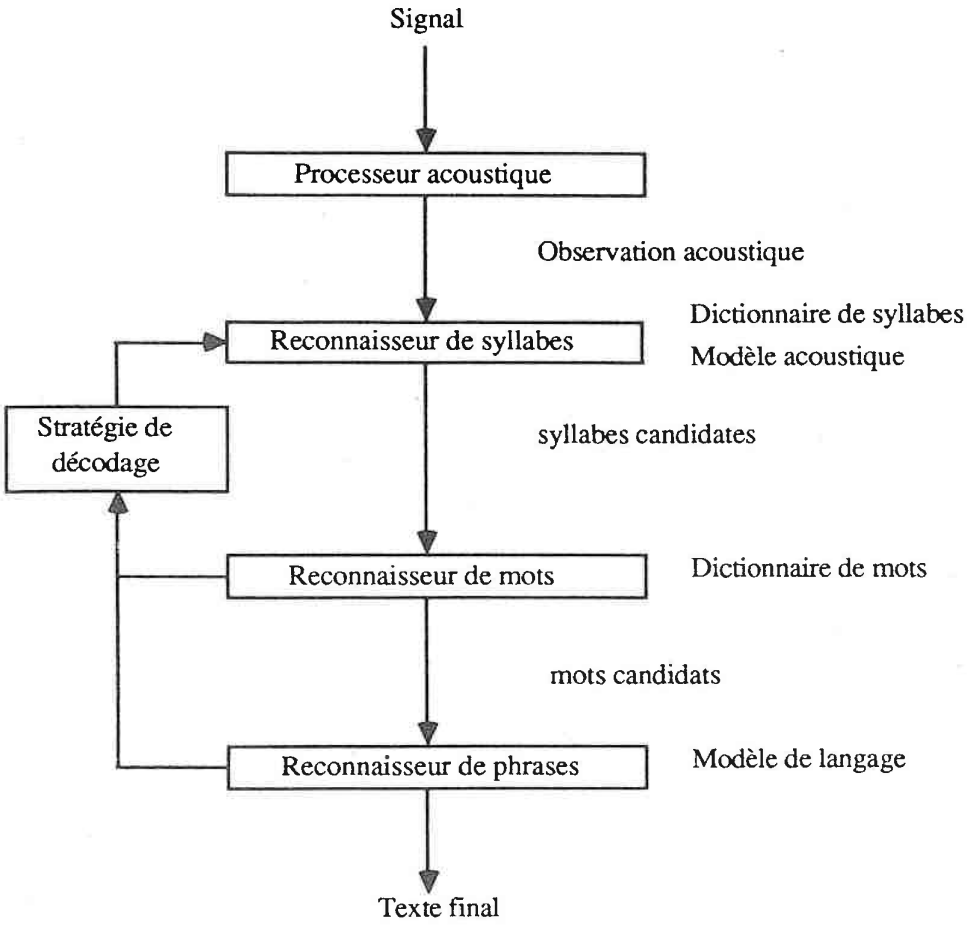


Figure 2.5. Diagramme du système d'IBM-Paris

Ce système est basé sur une stratégie de décodage multi-niveaux. Sa structure générale est illustrée dans la figure 2.5. Après le décodage d'un sous-ensemble de syllabes candidates pour une syllabe à reconnaître, on procède à une composition d'unités successives pour former des mots en utilisant un dictionnaire, puis à une composition de mots successifs pour former des phrases partielles, en utilisant les contraintes du modèle linguistique. Lorsque de nouveaux mots candidats sont produits, ces phrases partielles sont prolongées par ces nouveaux mots. Le modèle linguistique utilisé est le modèle de langage tri-classes décrit en [Derouault 86].

Pour évaluer les performances du système un corpus de 250 phrases, contenant tous les contextes correspondant aux 130 HMM phonétiques, a été utilisé pour l'apprentissage. Le test utilise un texte à reconnaître de 722 mots (79 phrases). Le taux d'erreurs sur les mots est de 12.7% pour un dictionnaire de 200000 mots couvrant toutes les phrases du texte.

En conclusion, ce système d'IBM-Paris testé avec plusieurs dictionnaires, confirme l'utilité des grands vocabulaires pour les systèmes de dictée automatique.

2.3.4 Le système SPHINX (1988)

Le système SPHINX a été conçu à l'université de Carnegie-Mellon (U.S.A.), pour la reconnaissance de la parole continue, indépendante du locuteur et avec grand vocabulaire [Lee 88a]. C'est un reconnaisseur basé sur les HMMs, utilisant trois dictionnaires de Q.V. à partir de trois traits variables dérivés de l'analyse LPC. SPHINX utilise deux types de HMM : des modèles de phonémes indépendants des contextes et des modèles de phonèmes dépendants des contextes dans les mots courts fréquement utilisés. Chaque HMM doit produire trois symboles de QV à chaque trame. 153 modèles de phonèmes sont alors estimés par l'algorithme de Baum-Welch à partir d'une base de données d'apprentissage de 4160 phrases (104 locuteurs avec 40 phrases chacun).
SPHINX a été appliqué à la tâche de direction de ressource de 997 mots utilisée par le projet DARPA. Le taux de reconnaissance de mots avec une grammaire "Bigram" est de 93%.

En conclusion, les résultats obtenus par SPHINX montrent la faisabilité de la reconnaissance de la parole continue, indépendante du locuteur et avec un grand vocabulaire à l'aide des modèles HMMs.

2.3.5 Le système d'IBM-NY (1989)

Le système présenté en 1989 par le centre d'IBM-NY est conçu pour la reconnaissance de la parole continue, "locuteur-dépendant" et avec grand vocabulaire [Bahl 89]. Dans les travaux précédents, le groupe de parole IBM (à New York) a construit un système de reconnaissance des phrases lues continuement à partir d'une grammaire d'états finis de 250 mots du vocabulaire et aussi, à partir d'un corpus naturel de 1000 mots [Bahl 83]. Puis, il a construit le système Tangora qui est en fait une classe de reconnaisseurs de mots isolés, grand vocabulaire, implanté sur ordinateur personel IBM-AT [Averbuch 86,Averbuch 87]. Nous trouvons, ainsi, Tangora-5 et Tangora-20 utilisant respectivement un vocabulaire de 5000 et de 20000 mots. Le système présenté en 1989 combine et étend les travaux précédents dans le but de reconnaître des phrases lues continuement à partir d'un corpus naturel couvert par un vocabulaire de 5000 mots.

Ce système de reconnaissance est constitué d'un processeur acoustique, d'un modèle du canal acoustique, d'un décodeur grossier, d'un modèle du langage et d'un décodeur détaillé. Le modèle du canal acoustique est un modèle acoustique allophonique contextuel. A chaque allophone correspond un modèle HMM. Les 200 vecteurs prototypes de la QV utilisés par le processeur acoustique sont sélectionnés par un mode itératif dans l'intention d'optimiser l'efficacité du modèle acoustique allophonique. Le décodeur grossier produit une courte liste de mots candidats. Le reconnaisseur utilise le modèle du langage trigramme standard d'IBM décrit en [Bahl 83]. Le décodeur détaillé est un module de recherche par hypothèse qui est basé sur l'algorithme à pile du décodage séquentiel décrit en [Jelinek 69].

Ce système a été testé sur un vocabulaire de 5000 mots fréquemment utilisés. Un ensemble de 10 locuteurs masculins lisant des documents de 2000 phrases a servi pour

l'apprentissage. Le test était fait sur 50 phrases construites à l'aide des mots du vocabulaire. Le taux d'erreurs de mots est de 11%. Presque le tiers des erreurs a été causé par le décodeur grossier qui n'a pas retenu le bon candidat.

Les perfomances de ce système de parole continue, grand vocabulaire et dépendant du locuteur sont surtout attribuées au modèle acoustique allophonique et à la méthode de QV utilisée.

2.4 Conclusion

Ce chapitre a montré la grande utilité des HMMs dans le domaine de la reconnaissance de la parole. Malgré les multiples applications des HMMs à ce domaine, nous constatons, à travers les systèmes présentés succintement, que leur application se fait presque toujours de la même façon (utilisation des HMMs du premier ordre, reconnaissance par une machine concaténant les modèles HMMs ou par une méthode de construction par niveau analogue à celle utilisée dans la programmation dynamique). Notre contribution porte sur l'amélioration des HMMs eux-mêmes et sur la façon de les appliquer à la reconnaissance de la parole. C'est l'objet des deux chapitres suivants.

3
Modèles markoviens cachés du second ordre

3.1 Introduction

Ce chapitre concerne la première partie de notre travail, traitant la formulation et l'implantation des HMMs du second ordre appliqués à la R.A.P.. Il ouvre ainsi une nouvelle voie de recherche et d'applications par l'extension des algorithmes Baum-Welch et de Viterbi à ce type de modèles. Les résultats de ces modèles du second ordre comparés à ceux du premier ordre, montrent l'utilité et l'efficacité de cette approche.

3.2 HMM du second ordre

3.2.1 Définition

Un modèle markovien caché discret du second ordre est une chaîne de Markov du second ordre stationnaire générant un processus stochastique à deux composantes : une cachée et l'autre observable. C'est-à-dire, un modèle HMM où la propriété de Markov :

$$P(q_t = s_i/q_{t-1} = s_j, q_{t-2} = s_k, q_{t-3} = s_l, ...) = P(q_t = s_i/q_{t-1} = s_j)$$

devient comme suit :

$$P(q_t = s_i/q_{t-1} = s_j, q_{t-2} = s_k, q_{t-3} = s_l, ...) = P(q_t = s_i/q_{t-1} = s_j, q_{t-2} = s_k)$$

3.2.2 Pourquoi les HMMs du second ordre?

La réponse à cette question peut être formulée de plusieurs façons. La plus simple étant : puisque nous sommes parvenus à formuler les algorithmes d'apprentissage et de reconnaissance correspondant aux HMMs du second ordre et que nous avons surmonté les

difficultés d'implantation (d'ordre numérique), il serait intéressant de les appliquer à la reconnaissance de la parole . Ceci étant, il y a d'autres raisons préalables de choisir une telle voie : théoriquement, le second ordre permet l'incorporation de plus d'informations dans la procédure de reconnaissance, et plus de précisions dans les estimations des probabilités de la phase d'apprentissage. En effet, la contrainte qui consiste à dire : "le fait d'être à l'instant t à l'état s_i ne dépend que de l'état où on était à l'instant $t-1$", est plus légère que celle de considérer "le fait d'être à l'instant t à l'état s_i dépend de l'état où on était à l'instant $t-1$ et de celui où on était à l'instant $t-2$".

Dans les systèmes actuels, l'application des HMMs est encore réduite à ceux du premier ordre. Une seule application à la reconnaissance de l'écriture était faite par Kundu et coll. [Kundu 88]; ils ont montré l'avantage des HMMs du second ordre sur ceux du premier ordre. Ils ont signalé la difficulté d'implantation des HMMs du second ordre due à l'indisponibilité de leurs formules de ré-estimations dans la littérature. Ils étaient alors obligés de calculer les probabilités de transitions manuellement. Nous proposons, ainsi, notre premier travail dans cette voie de recherche : le développement des formules de ré-estimations des HMMs du second ordre, d'une part, et l'implantation d'un reconnaisseur de mots isolés utilisant ces modèles, d'autre part.

3.2.3 Modèles équivalents

Nous venons de voir qu'il y aura incorporation de plus d'informations dans un modèle HMM du second ordre que dans son homologue du premier ordre possédant le même nombre d'états et de transitions. Ce qui laisse espérer de meilleures performances de reconnaissance.
Mais, il reste une autre question à soulever : Pourquoi utilisons-nous des HMMs du second ordre et cherchons des difficultés alors que nous pouvons transformer un modèle du second ordre en un modèle équivalent du premier ordre ?
C'est peut être la raison pour laquelle la plupart des travaux de recherche du domaine n'utilisent que les HMMs du premier ordre.

Pour répondre à cette question, il suffit de comprendre le sens donné au mot équivalent, utilisé ici, dont l'origine est la théorie des chaînes de Markov. La différence essentielle entre un HMM et une chaîne de Markov est la matrice de probabilités d'observations, ainsi, l'équivalence dont on parle est celle entre matrices de transitions. La figure 3.1 présente un exemple de construction d'un modèle équivalent en transition (b) à partir d'un modèle du second ordre (a). En effet une probabilité de transition d'un modèle du second ordre :

$$a_{ijk} = P(q_t = k / q_{t-1} = j, q_{t-2} = i)$$

sera équivalente à une probabilité de transition :

$$a_{ij,jk} = P(q_t = jk / q_{t-1} = ij)$$

d'un modèle du premier ordre où les transitions, ij, de celui du second ordre sont considérées comme des états du modèle et la transition $ij \longrightarrow jk$ dans ce modèle est possible lorsque la double transition $i \longrightarrow j \longrightarrow k$ est possible dans celui du second ordre. Pour avoir une vraie équivalence entre un HMM du premier ordre et un autre du second ordre, il nous faut, en plus de l'équivalence entre matrices de transitions, celle entre matrices de

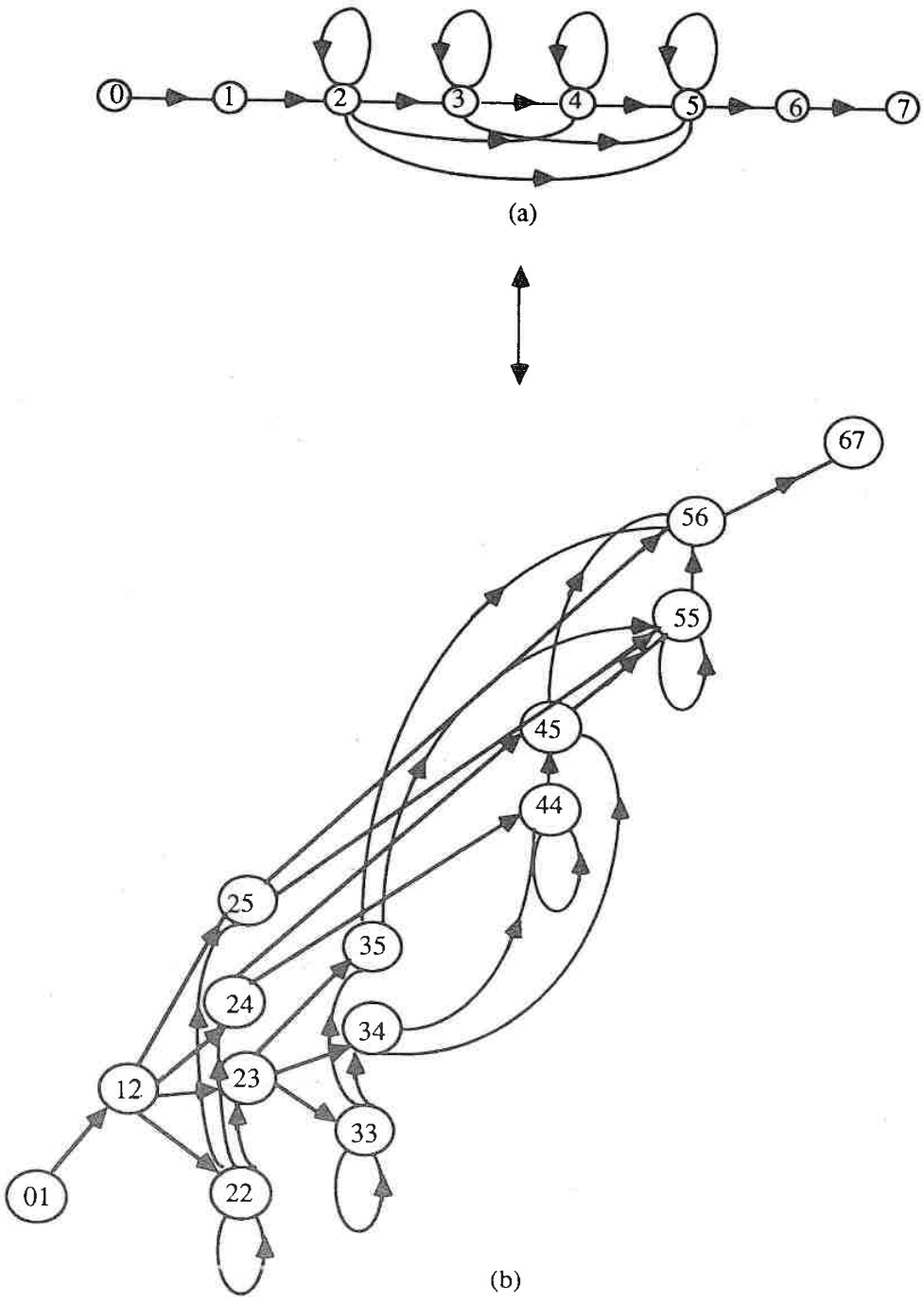


Figure 3.1. Un HMM du second ordre (a) et son équivalent du premier ordre (b)

probabilités d'observations. Ce qui suppose un corpus d'apprentissage infini compte tenu du nombre d'états qui n'est plus le même dans les deux modèles. En effet, la probabilité d'observation :

$$b_{ij}(k) = P(O_t = v_k / q_t = ij)$$

du modèle équivalent serait équivalente dans un HMM du second ordre à la probabilité :

$$b_{i \longrightarrow j}(k) = P(O_t = v_k / q_{t-1} = i, q_t = j)$$

que si on l'utilise à la place de :

$$b_j(k) = P(O_t = v_k / q_t = j),$$

nous aurons une équivalence parfaite entre les deux modèles et il ne serait pas intéressant de traiter un HMM du second ordre de ce type. L'intéret de considérer un HMM du second ordre avec une probabilité d'observation :

$$b_j(k) = P(O_t = v_k / q_t = j)$$

réside dans le fait qu'on aura à estimer beaucoup moins de paramètres que dans un HMM équivalent où on ne peut plus utiliser ce type de probabilité puisque la probabilité d'observation ne peut plus dépendre que de deux états ($q_t = ij$). Pour confirmer ce que nous venons de dire une comparaison expérimentale est effectuée entre les deux modèles équivalents de la figure 3.1. Les résultats seront exposés au paragraphe 3.6..

En conclusion, au plan théorique, on sait construire les modèles du premier ordre que l'on peut rendre équivalent aux modèles du deuxième ordre. Mais comme cela suppose un nombre d'états beaucoup plus grand, il est plus intéressant de traiter directement le modèle du deuxième ordre et notamment d'y adapter les algorithmes de Viterbi et de Baum-Welch.

3.3 Extension de l'algorithme de Viterbi

3.3.1 Technique d'extension

Une extension théorique de l'algorithme de Viterbi, pour le second ordre, a été développée dans [He 88] par l'utilisation d'une technique permettant de réduire l'ordre du HMM et ainsi se ramener au cas bien connu des HMMs du premier ordre [White 78,Forney 73] :
Soit $X = (X(1), X(2), ..., X(T))$ un processus de Markov du second ordre de N états. Nous noterons, toujours, la variable aléatoire X(t) par q_t.
Soit $O = (O_1, O_2, ..., O_T)$ une suite d'observations du processus markovien caché λ (M : le nombre de symboles d'observations). Pour chaque instant t, l'observation O_t ne dépend que de l'état courant (correspondant à cet instant).
Posons les définitions suivantes :

$$\Pi_i = P(q_1 = s_i / \lambda)$$

$$a_{ij} = P(q_2 = s_j / q_1 = s_i, \lambda)$$

$$a_{ijk} = P(q_{t+2} = s_k / q_{t+1} = s_j, q_t = s_i, \lambda)$$

$$b_k(l) = P(O_t = v_l / q_t = s_k, \lambda)$$

Le critère de l'algorithme de Viterbi, déjà vu, était de trouver la meilleure suite d'états qui maximise :

$$P(Q/O, \lambda)$$

Ce qui est équivalent à maximiser :

$$P(Q, O/\lambda)$$

avec $Q = (q_1, q_2, .., q_T)$.

or :

$$P(Q, O/\lambda) = P(O/Q, \lambda)P(Q/\lambda)$$

avec :

$$P(O/Q, \lambda) = b_{q_1}(O_1)b_{q_2}(O_2)...b_{q_T}(O_T)$$

et :

$$P(Q/\lambda) = \Pi_{q_1} a_{q_1 q_2} a_{q_1 q_2 q_3} ... a_{q_{T-2} q_{T-1} q_T}$$

Introduisons une suite d'états combinée :

$Y = (y_1, y_2, ..., y_T)$ où $y_1 = q_1$ et $y_i = q_{i-1} q_i$ *pour* $2 \leq i \leq T$

En considérant les définitions du premier ordre pour la suite Y, nous aurons :

$$\Pi_{y_1} = \Pi_{q_1} \quad , \quad a_{y_1 y_2} = a_{q_1 q_2} \quad et \quad a_{y_{i-1} y_i} = a_{q_{i-2} q_{i-1} q_i} \quad pour \quad 3 \leq i \leq T$$

et :

$$P(Q, O/\lambda) = \Pi_{y_1} b_{q_1}(O_1) a_{y_1 y_2} b_{q_2}(O_2) a_{y_2 y_3} b_{q_3}(O_3) ... a_{y_{T-1} y_T} b_{q_T}(O_T)$$

Cette égalité est la même que celle obtenue à partir du modèle du premier ordre. Par suite l'équation pour les suites partielles sera, pour $3 \leq t \leq T$:

$$P(q_1 q_2 ... q_t, O_1 O_2 ... O_t/\lambda) = P(q_1 q_2 ... q_{t-1}, O_1 O_2 ... O_{t-1}/\lambda) a_{y_{t-1} y_t} b_{q_t}(O_t) \qquad (1)$$

Pour trouver le maximum de $P(q_1 q_2 ... q_t, O_1 O_2 ... O_t/\lambda)$ pour chaque y_t nous aurons seulement besoin de :

- mémoriser le maximum de $P(q_1 q_2 ... q_{t-1}, O_1 O_2 ... O_{t-1}/\lambda)$ pour chaque y_{t-1}
- étendre chacune de ces probabilités à différents y_t par le calcul de l'équation (1)
- sélectionner le maximum de $P(q_1 q_2 ... q_t, O_1 O_2 ... O_t/\lambda)$ pour chaque y_t

En incrémentant le temps t de 1 à T et en répétant ces trois étapes, le maximum de $P(Q, O/\lambda)$ pour chaque y_T peut être finalement trouvé. La plus grande de ces probabilités maximales nous donnera le y_T cherché et nous aurons, de même, la suite d'états Y^* menant à cette probabilité maximale. Cette suite optimale combinée Y^*, donnera automatiquement la suite optimale Q^* correspondante.

Nous venons d'exposer, en fait, l'algorithme de Viterbi, qui mérite une formulation plus claire en revenant aux états normaux, les q_t, au lieu des états combinés, les y_t, qui ne sont qu'un moyen pour y arriver.

Remarquons que cette technique d'introduire une suite d'états combinés peut être utilisée pour une extension facile aux modèles du troisième ordre.

3.3.2 Algorithme de Viterbi pour le second ordre

Pour trouver le meilleur chemin, $Q = (q_1, q_2, ..., q_T)$, pour une suite d'observations donnée, $O = (O_1, O_2, ..., O_T)$, nous définissons la quantité $\delta_t(i, j)$ comme la probabilité du meilleur chemin partiel amenant à l'état s_j à l'instant t, passant par l'état s_i à l'instant t-1 et guidé par les t premières observations :

$$\delta_t(i, j) = \max_{q_1 q_2 ... q_{t-2}} P(q_1 q_2 ... q_{t-1} = s_i, q_t = s_j, O_1 O_2 ... O_t/\lambda)$$

Par induction, nous pouvons calculer $\delta_{t+1}(j,k)$ à partir des $\delta_t(i,j)$:

$$\delta_{t+1}(j,k) = [\max_i \delta_t(i,j)a_{ijk}].b_k(O_{t+1})$$

et on garde la trace de la suite d'états qui donne le meilleur chemin amenant à l'état s_j à l'instant t par l'état s_i à l'instant t-1 dans un tableau ψ.
Les quatre étapes suivantes résument l'extension de l'algorithme de Viterbi aux HMMs du second ordre :

1. Initialisation

$$\delta_1(i) = \Pi_i.b_i(O_1) \quad pour \quad 1 \leq i \leq N$$

$$\delta_2(i,j) = \delta_1(i).a_{ij}.b_j(O_2) \quad pour \quad 1 \leq i \leq N \quad et \quad 1 \leq j \leq N$$

2. Récurrence pour $3 \leq t \leq T$, $1 \leq i \leq N$ et $1 \leq j \leq N$

$$\delta_t(j,k) = max_{1 \leq i \leq N}[\delta_{t-1}(i,j)a_{ijk}].b_k(O_t)$$

$$\psi_t(j,k) = \arg_i \max_{1 \leq i \leq N}[\delta_{t-1}(i,j)a_{ijk}]$$

3. Terminaison

$$P^* = max_{1 \leq i,j \leq N}[\delta_T(i,j)]$$

$$q^*_{T-1} = \arg_i \max_{1 \leq i,j \leq N}[\delta_T(i,j)]$$

$$q^*_T = \arg_j \max_{1 \leq i,j \leq N}[\delta_T(i,j)]$$

4. Chemin d'états retenu (backtracking)

$$q^*_t = \psi_{t+2}(q^*_{t+1}, q^*_{t+2}) \quad pour \quad T-2 \geq t \geq 1$$

Le coût des opérations est approximativement N fois le nombre des multiplications dans l'algorithme de Viterbi pour le premier ordre. Soit une complexité en N^3T dans le pire cas, l'algorithme est donc toujours linéaire avec la longueur T de la suite d'observations à décoder. En plus le passage aux logarithmes nous permet d'utiliser une alternative en N^3T additions (Dans nos expériences N ne dépasse pas la valeur 10).

3.4 Extension de l'algorithme Baum-Welch

3.4.1 Nouvelles fonctions Forward et Backward

Pour la phase de reconnaissance, le problème est résolu par l'extension précédente de l'algorithme de Viterbi. Mais, la question qui reste à résoudre est : comment estimer un HMM du second ordre ? Nous avons proposé une extension de l'algorithme d'apprentissage Baum-Welch, basée sur de nouvelles définitions des fonctions Forward et Backward, adaptées à l'évolution demandée [Kriouile 90b] :

Considérons la fonction *Forward*, $\alpha_t(i,j)$, définie comme la probabilité conjointe d'avoir la suite d'observations partielle, de 1 à t, et d'être à l'instant $t-1$ à l'état s_i et à l'état s_j à l'instant t :

$$\alpha_t(i,j) = P(O_1O_2...O_t, q_{t-1} = s_i, q_t = s_j/\lambda)$$

Lemme :

$$\alpha_{t+1}(j,k) = [\sum_{i=1}^{N} \alpha_t(i,j)a_{ijk}].b_k(O_{t+1}) \quad pour \quad 2 \leq t \leq T-1$$

Démonstration :

$\alpha_{t+1}(j,k)$

$= P(O_1O_2...O_{t+1}, q_t = s_j, q_{t+1} = s_k/\lambda)$

$= \sum_{i=1}^{N} P(O_1O_2...O_{t+1}, q_{t-1} = s_i, q_t = s_j, q_{t+1} = s_k/\lambda)$

$= \sum_{i=1}^{N} P(O_1O_2...O_t, q_{t-1} = s_i, q_t = s_j, q_{t+1} = s_k, O_{t+1}/\lambda)$

$= \sum_{i=1}^{N} P(O_1O_2...O_t, q_{t-1} = s_i, q_t = s_j/\lambda)P(q_{t+1} = s_k/O_1O_2...O_t, q_{t-1} = s_i, q_t = s_j, \lambda)$
$P(O_{t+1}/O_1O_2...O_t, q_{t-1} = s_i, q_t = s_j, q_{t+1} = s_k, \lambda)$

$= \sum_{i=1}^{N} P(O_1O_2...O_t, q_{t-1} = s_i, q_t = s_j/\lambda)P(q_{t+1} = s_k/q_{t-1} = s_i, q_t = s_j, \lambda)P(O_{t+1}/q_{t+1} = s_k, \lambda)$

$= \sum_{i=1}^{N} \alpha_t(i,j)a_{ijk}b_k(O_{t+1})$

d'où la formule cherchée.

Puisque :

$$\alpha_T(i,j) = P(O_1O_2...O_T, q_{T-1} = s_i, q_T = s_j/\lambda)$$

nous déduisons la valeur de $P(O/\lambda)$:

$$P(O/\lambda) = \sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_T(i,j)$$

Le lemme précédent permet de calculer $\alpha_t(i,j)$ de façon récursive :

1. Initialisation

   $\alpha_2(i,j) = \Pi_i b_i(O_1)a_{ij}b_j(O_2)$ *pour* $1 \leq i \leq N$ *et* $1 \leq j \leq N$

2. Récurrence pour $2 \leq t \leq T-1$

   $\alpha_{t+1}(j,k) = [\sum_{i=1}^{N} \alpha_t(i,j)a_{ijk}].b_k(O_{t+1})$ *pour* $1 \leq j \leq N$ *et* $1 \leq k \leq N$

3. Terminaison

$$P(O/\lambda) = \sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_T(i,j)$$

Le coût de calcul est approximativement N fois le nombre des opérations qui calculent la fonction Forward du premier ordre. Soit une complexité, en N^3T dans le pire cas, toujours linéaire avec T.

Considérons, maintenant, la variable *Backward*, $\beta_t(i,j)$, définie comme la probabilité de la suite d'observations partielle de $t+1$ à T, sachant qu'on était à l'état s_i à l'instant $t-1$ et à l'état s_j à l'instant t (probabilité que le modèle émette O_{t+1}^T en partant de la transition s_is_j) :

$$\beta_t(i,j) = P(O_{t+1}O_{t+2}...O_T/q_{t-1} = s_i, q_t = s_j, \lambda)$$

Lemme :

$$\beta_t(i,j) = \sum_{k=1}^{N} a_{ijk}b_k(O_{t+1})\beta_{t+1}(j,k) \quad pour \quad T-1 \geq t \geq 2$$

Démonstration :

$\beta_t(i,j)$

$= P(O_{t+1}O_{t+2}...O_T/q_{t-1} = s_i, q_t = s_j, \lambda)$

$= \sum_{k=1}^{N} P(O_{t+1}O_{t+2}...O_T, q_{t+1} = s_k/q_{t-1} = s_i, q_t = s_j, \lambda)$

$= \sum_{k=1}^{N} P(q_{t+1} = s_k, O_{t+1}, O_{t+2}...O_T/q_{t-1} = s_i, q_t = s_j, \lambda)$

$= \sum_{k=1}^{N} P(q_{t+1} = s_k/q_{t-1} = s_i, q_t = s_j, \lambda) P(O_{t+1}/q_{t-1} = s_i, q_t = s_j, q_{t+1} = s_k, \lambda)$
$P(O_{t+2}...O_T/q_{t-1} = s_i, q_t = s_j, q_{t+1} = s_k, O_{t+1}, \lambda)$

$= \sum_{k=1}^{N} P(q_{t+1} = s_k/q_{t-1} = s_i, q_t = s_j, \lambda) P(O_{t+1}/q_{t+1} = s_k, \lambda) P(O_{t+2}...O_T/q_t = s_j, q_{t+1} = s_k)$

$= \sum_{k=1}^{N} a_{ijk} b_k(O_{t+1}) \beta_{t+1}(j,k)$

d'où la formule à démontrer.

Ce lemme permet de calculer $\beta_t(i,j)$ de façon récursive :

1. Initialisation

$$\beta_T(i,j) = 1 \quad pour \quad 1 \leq i \leq N \quad et \quad 1 \leq j \leq N$$

2. Récurrence pour $T - 1 \geq t \geq 2$

$$\beta_t(i,j) = \sum_{k=1}^{N} a_{ijk} b_k(O_{t+1}) \beta_{t+1}(j,k) \ pour \ 1 \leq i \leq N \ et \ 1 \leq j \leq N$$

3. Terminaison

Le calcul de $\beta_t(i,j)$ jusqu'à $t = 2$ suffit pour les formules de réestimations.

Le coût de calcul des $\beta_t(i,j)$ est lui aussi en N^3T.

3.4.2 Les formules de ré-estimations

Bien entendu, les formules de ré-estimations vont être modifiées par rapport à celles du premier ordre, puisque les calculs vont être faits à partir des nouvelles fonctions Forward et Backward. Mais, les définitions de $\xi_t(i,j)$ et de $\gamma_t(i)$ seront les mêmes. Soient :

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

et

$$\gamma_t(i) = P(q_t = s_i / O, \lambda).$$

Mais il nous faut définir une autre quantité $\eta_t(i,j,k)$ qui est la probabilité de passer par les états s_i , sj *et* s_k respectivement à l'instant $t-1$, t *et* $t+1$ pendant l'émission de la suite d'observations O :

$$\eta_t(i,j,k) = P(q_{t-1} = s_i, q_t = s_j, q_{t+1} = s_k / O, \lambda)$$

Lemme :

$$\eta_t(i,j,k) = \frac{\alpha_t(i,j) a_{ijk} b_k(O_{t+1}) \beta_{t+1}(j,k)}{P(O/\lambda)} \quad pour \quad 1 \leq t \leq T-1$$

Démonstration :

à partir de la définition de $\eta_t(i,j,k)$ nous déduisons facilement l'égalité suivante :

$$\eta_t(i,j,k) = \frac{P(q_{t-1} = s_i, q_t = s_j, q_{t+1} = s_k, O/\lambda)}{P(O/\lambda)}$$

En développant le numérateur du terme de droite, nous obtenons les égalités suivantes :

$$P(q_{t-1} = s_i, q_t = s_j, q_{t+1} = s_k, O/\lambda)$$

$$= P(O_1 O_2 ... O_t, q_{t-1} = s_i, q_t = s_j, q_{t+1} = s_k, O_{t+1}, O_{t+2} ... O_T/\lambda)$$

$$= P(O_1 O_2 ... O_t, q_{t-1} = s_i, q_t = s_j/\lambda) P(q_{t+1} = s_k/O_1 O_2 ... O_t, q_{t-1} = s_i, q_t = s_j, \lambda)$$
$$P(O_{t+1}/O_1 O_2 ... O_t, q_{t-1} = s_i, q_t = s_j, q_{t+1} = s_k, \lambda)$$
$$P(O_{t+2} ... O_T/O_1 O_2 ... O_t, q_{t-1} = s_i, q_t = s_j, q_{t+1} = s_k, O_{t+1}, \lambda)$$

$$= P(O_1 O_2 ... O_t, q_{t-1} = s_i, q_t = s_j/\lambda) P(q_{t+1} = s_k/q_{t-1} = s_i, q_t = s_j, \lambda) P(O_{t+1}/q_{t+1} = s_k)$$
$$P(O_{t+2} ... O_T/q_t = s_j, q_{t+1} = s_k, \lambda)$$

$$= \alpha_t(i,j) a_{ijk} b_k(O_{t+1}) \beta_{t+1}(j,k)$$

d'où la relation très importante du lemme précédent.

D'autre part, puisque les $\xi_t(i,j)$ et les $\gamma_t(i)$ peuvent s'écrire à partir de leur définitions de la façon suivante :

$$\xi_t(i,j) = \sum_{k=1}^{N} P(q_t = s_i, q_{t+1} = s_j, q_{t+2} = s_k/O, \lambda)$$

et

$$\gamma_t(i) = \sum_{j=1}^{N} P(q_t = s_i, q_{t+1} = s_j/O, \lambda)$$

nous déduisons les relations entre ces trois quantités, qui vont permettre de calculer $\xi_t(i,j)$ et $\gamma_t(i)$ à partir de $\eta_t(i,j,k)$, elle aussi calculée par la formule du lemme précédent :

$$\xi_t(i,j) = \sum_{k=1}^{N} \eta_{t+1}(i,j,k)$$

et

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

Nous avons déjà vu que, d'après la définition, une sommation sur le temps t de la quantité $\gamma_t(i)$ peut être interprétée comme le compte moyen d'être dans l'état s_i pendant

l'émission de la suite O, ou comme le compte moyen des transitions dont le départ est l'état s_i pendant l'émission de la suite O. De même, une sommation sur le temps t de la quantité $\xi_t(i,j)$ peut être interprétée comme le compte moyen de la transition $s_i s_j$ pendant l'émission de la suite O. De manière similaire, la définition de $\eta_t(i,j,k)$ nous permet d'interpréter la sommation sur le temps t de cette quantité, comme le compte moyen de la double transition $s_i s_j s_k$ pendant l'émission de la suite O. Ainsi, des formules de ré-estimations convenables pour Π_i , a_{ij} , a_{ijk} *et* $b_k(l)$ peuvent être :

$$\bar{\Pi}_i = \textit{Le compte moyen d'être dans l'état } s_i \textit{ à l'instant } 1$$

$$\bar{a}_{ij} = \frac{\textit{Le compte moyen des transitions de l'état } s_i \textit{ à l'instant } 1 \textit{ à l'état } s_j \textit{ à l'instant } 2}{\textit{Le compte moyen des transitions partant de l'état } s_i \textit{ à l'instant } 1}$$

$$\bar{a}_{ijk} = \frac{\textit{Le compte moyen de la double transition } s_i s_j s_k}{\textit{Le compte moyen de la transition } s_i s_j}$$

$$\bar{b}_k(l) = \frac{\textit{Le compte moyen d'être dans l'état } s_k \textit{ avec l'observation du symbole } v_l}{\textit{Le compte moyen d'être dans l'état } s_k}$$

soient :

$$\bar{\Pi}_i = \gamma_1(i)$$

$$\bar{a}_{ij} = \frac{\xi_1(i,j)}{\gamma_1(i)}$$

$$\bar{a}_{ijk} = \frac{\sum_{t=1}^{T-2} \eta_{t+1}(i,j,k)}{\sum_{t=1}^{T-2} \xi_t(i,j)}$$

$$\bar{b}_k(l) = \frac{\sum_{t=1, O_t=v_l}^{T} \gamma_t(k)}{\sum_{t=1}^{T} \gamma_t(k)}$$

Notons que les contraintes stochastiques sur ces formules sont évidentes :

$$\sum_{i=1}^{N} \bar{\Pi}_i = 1$$

$$\sum_{j=1}^{N} \bar{a}_{ij} = \frac{\sum_{j=1}^{N} \xi_1(i,j)}{\gamma_1(i)} = 1$$

$$\sum_{k=1}^{N} \bar{a}_{ijk} = \frac{\sum_{t=1}^{T-2} \sum_{k=1}^{N} \eta_{t+1}(i,j,k)}{\sum_{t=1}^{T-2} \xi_t(i,j)} = 1$$

$$\sum_{l=1}^{M} \bar{b}_k(l) = 1$$

3.4.3 Algorithme de Baum-Welch pour le second ordre

Les formules de ré-estimations que nous venons d'exposer nous permettent de proposer une extension de l'algorithme d'apprentissage utilisé pour les HMMs du premier ordre. Cet algorithme que nous appelons *Baum-Welch pour le second ordre*, est résumé dans les quatre étapes suivantes :

1. Fixer des valeurs initiales

$$\Pi_i^0 \quad , \quad a_{ij}^0 \quad , \quad a_{ijk}^0 \quad , \quad b_k^0(l)$$

pour $1 \leq i \leq N \quad , \quad 1 \leq j \leq N \quad , \quad 1 \leq k \leq N \quad , \quad 1 \leq l \leq M$

2. Calculer :

$$\eta_t(i,j,k) \quad , \quad \xi_t(i,j) \quad , \quad \gamma_t(i)$$

pour $1 \leq i \leq N \quad , \quad 1 \leq j \leq N \quad , \quad 1 \leq k \leq N \quad , \quad 1 \leq t \leq T$

en utilisant les nouvelles fonctions Forward et Backward.

3. Nouvelles estimations

calculer :

$$\bar{\Pi}_i \quad , \quad \bar{a}_{ij} \quad , \quad \bar{a}_{ijk} \quad , \quad \bar{b}_k(l)$$

pour $1 \leq i \leq N \quad , \quad 1 \leq j \leq N \quad , \quad 1 \leq k \leq N \quad , \quad 1 \leq l \leq M$

en utilisant les formules de ré-estimations.

4. Recommencer en 2, jusqu'à un certain point limite, avec les nouvelles valeurs estimées :

$$\Pi_i = \bar{\Pi}_i \quad , \quad a_{ij} = \bar{a}_{ij} \quad , \quad a_{ijk} = \bar{a}_{ijk} \quad , \quad b_k(l) = \bar{b}_k(l)$$

pour $1 \leq i \leq N \quad , \quad 1 \leq j \leq N \quad , \quad 1 \leq k \leq N \quad , \quad 1 \leq l \leq M$

Les remarques suivantes que nous avons faites pour le cas du premier ordre sont conservées pour celui du second ordre :

- Le choix du modèle initial influe sur les résultats.
- Le test d'arrêt est en général le nombre des itérations, fixé empiriquement.
- Pour avoir une estimation convenable du modèle, les ré-estimations se font sur un ensemble de plusieurs suites d'observations. Les dimensions de ce corpus d'apprentissage

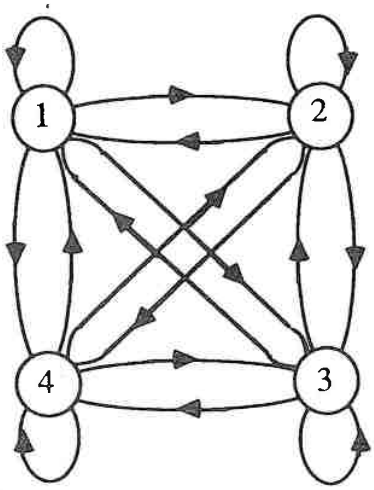


Figure 3.2. exemple de modèle ergodique

influent sur les résultats.

Pour conclure sur cette extension, dont la méthode peut être reprise pour un passage au troisième ordre, remarquons qu'il n'est pas intéressant d'étudier la complexité de cet algorithme Baum-Welch pour le second ordre, car, une fois l'apprentissage terminé, nous n'allons travailler qu'avec les modèles appris. Ceci étant, nous avons vu que les paramètres de base, α_t *et* β_t, de cet algorithme se calculent avec une complexité linéaire en T.

3.5 Techniques d'implantation

3.5.1 Types des HMMs

Jusqu'à présent, nous n'avons pas encore spécifié la structure des HMMs utilisée dans les algorithmes d'apprentissage et de reconnaissance précédents. Les structures sont nombreuses et diversifiées, nous allons citer, par la suite, trois types de structures intéressantes du point de vue des applications :

1. Le modèle général où aucune transition n'est absente. Ce modèle, dans lequel nous pouvons aller de n'importe quel état à n'importe quel autre en une seule transition, est appelé modèle ergodique. Un tel modèle est illustré, pour 4 états, dans la figure 3.2.

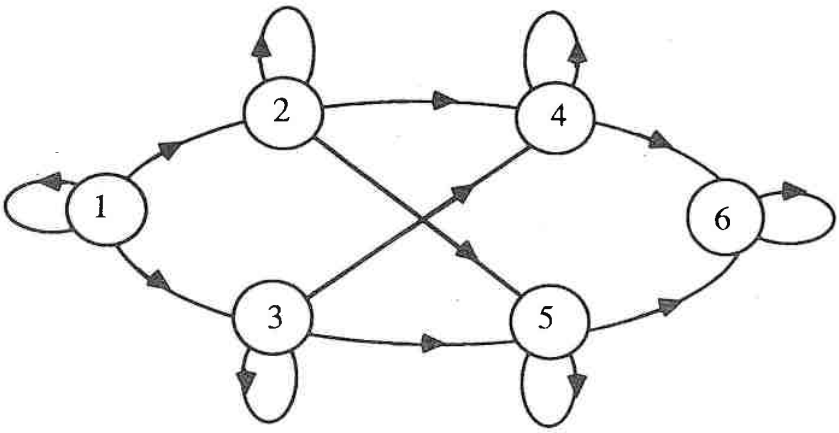


Figure 3.3. exemple de modèle à branches parallèles

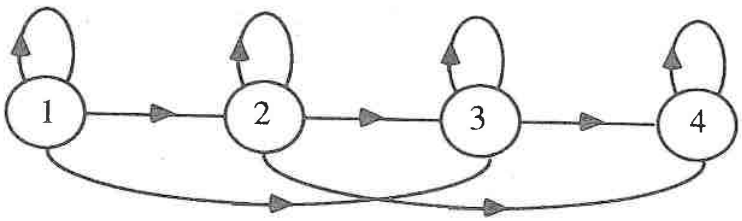


Figure 3.4. exemple de modèle Gauche-Droite

2. Le modèle à branches parallèles. Ce type de modèle peut être utile pour certaines applications par la symétrie qu'il présente dans sa structure. Un exemple de ce modèle, à 6 états, est illustré dans la figure 3.3.

3. Le modèle Gauche-Droite défini par les trois propriétés suivantes :

   - La première observation est produite alors que la chaîne de Markov se trouve dans le premier état;
   - La dernière observation est générée alors que la chaîne arrive sur l'état final;
   - Une fois que la chaîne quitte un état, elle ne peut y revenir, d'où le nom de modèle Gauche-Droite :

   $$\Pi_i = 1 \quad si \quad i = 1, \quad \Pi_i = 0 \quad si \quad i \neq 1 \quad et \quad a_{ij} = 0 \quad si \quad j < i$$

   On choisit, en général, en plus $a_{ij} = 0 \ \ si \ \ j > i + \delta \ \ avec \ \ \delta = 1, 2, ...$ C'est le cas du modèle de Bakis qui prend pour δ la valeur 2. Un tel modèle, à 4 états, est illustré dans la figure figure 3.4.

Pour les applications où le signal évolue avec le temps, telles que la reconnaissance de la parole, le modèle Gauche-Droite est plus adapté pour modéliser une telle propriété. Ainsi, l'indice de la séquence d'états associée au signal augmente avec le temps. L'avantage d'utiliser de tels modèles réside dans le fait qu'ils introduisent des contraintes temporelles fortes sur la chaîne de Markov, ce qui est bien adapté à la reconnaissance de mots isolés, enchaînés et le décodage phonétique. Les modèles ergodiques peuvent être plus utiles pour les niveaux supérieurs dans un système de reconnaissance de la parole continue [Rabiner 88], par exemple, où chaque état représente une unité phonétique.

Notons, aussi, que les contraintes du modèle Gauche-Droite n'affectent pas les formules de ré-estimations, parce qu'un paramètre initialement nul reste toujours nul durant les ré-estimations.

3.5.2 Problème de l'"underflow"

Si, pour l'algorithme de Viterbi, les difficultés d'ordre numérique peuvent être résolues en passant aux logarithmes, ce n'est pas le cas pour l'algorithme de Baum-Welch, qui contient à la fois des produits et des sommes dans les quantités qu'il calcule. En effet, cet algorithme d'apprentissage nécessite le calcul des probabilités partielles Forward, $\alpha_t(i)$, et Backward, $\beta_t(i)$ pour $(1 \leq t \leq T)$ et $(1 \leq i \leq N)$. D'après les formules récursives de calcul de α et de β, il est clair que lorsque T croît, α et β tendent vers 0. Pour un grand nombre d'observations, la fonction de probabilité aura ainsi une valeur trop petite pour être représentée dans un calculateur.

Diverses méthodes ont été proposées pour remédier à cet inconvénient, dont celle consistant à effectuer les calculs à partir d'approximations des logarithmes de somme. Dans [Rabiner 83,Dours 89], nous trouvons des méthodes reposant sur l'introduction des coefficients de normalisation des fonctions α et β. Cette normalisation se répercute donc au niveau des autres formules de ré-estimations.

Une autre approche propose d'introduire un facteur d'échelle qu'on multiplie par la probabilité $\alpha_t(i)$ pour permettre à cette dernière de rester dans une échelle convenable.

Une opération similaire est alors faisable pour le calcul de $\beta_t(i)$. Il suffit d'effacer toute trace de ce coefficient à la fin du calcul pour retrouver les valeurs réelles. Le facteur d'échelle, indépendant de i, utilisé dans [Levinson 83a,Levinson 83b] est :

$$c_t = \frac{1}{\sum_{i=1}^{N} \alpha_t(i)}$$

Dans le cas du second ordre, nous pouvons utiliser le facteur d'échelle suivant :

$$c_t = \frac{1}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_t(i,j)}$$

Notons par $\hat{\alpha}_t(i,j)$ et $\hat{\beta}_t(i,j)$ les valeurs échelonnées correspondant à $\alpha_t(i,j)$ et $\beta_t(i,j)$. Le calcul de $\alpha_{t+1}(j,k)$, par exemple, est basé sur les valeurs $\hat{\alpha}_t(i,j)$, ainsi le facteur d'échelle c_{t+1} dans $\alpha_{t+1}(j,k)$ représente seulement l'échelle de l'itération courante. Les relations entre les deux types de valeurs seront alors :

$$\hat{\alpha}_t(i,j) = [\prod_{r=1}^{t} c_r]\alpha_t(i,j) = C_t \alpha_t(i,j)$$

$$\hat{\beta}_t(i,j) = [\prod_{r=t}^{T} c_r]\beta_t(i,j) = D_t \beta_t(i,j)$$

avec :

$$C_t = \prod_{r=1}^{t} c_r$$

et

$$D_t = \prod_{r=t}^{T} c_r$$

Ces deux relations peuvent être démontrées facilement par récurrence. En effet :

$$\hat{\alpha}_1(j,k) = c_1 \alpha_1(j,k)$$

et

$$\hat{\alpha}_{t+1}(j,k) = c_{t+1}[\sum_{i=1}^{N} \hat{\alpha}_t(i,j) a_{ijk}] b_k(O_{t+1})$$

$$\hat{\alpha}_{t+1}(j,k) = c_{t+1}[\sum_{i=1}^{N}(\prod_{r=1}^{t} c_r)\alpha_t(i,j)a_{ijk}]b_k(O_{t+1})$$

$$\hat{\alpha}_{t+1}(j,k) = c_{t+1}(\prod_{r=1}^{t} c_r)[\sum_{i=1}^{N}\alpha_t(i,j)a_{ijk}]b_k(O_{t+1})$$

d'où :

$$\hat{\alpha}_{t+1}(j,k) = (\prod_{r=1}^{t+1} c_r)\alpha_{t+1}(j,k)$$

La deuxième relation peut être, elle aussi, démontrée de manière similaire.

La formule de ré-estimation pour a_{ij}, par exemple, en termes de probabilités partielles sera :

$$\bar{a}_{ijk} = \frac{\sum_{t=1}^{T-2}\hat{\alpha}_{t+1}(i,j)a_{ijk}b_k(O_{t+2})\hat{\beta}_{t+2}(j,k)}{\sum_{t=1}^{T-2}\sum_{l=1}^{N}\hat{\alpha}_{t+1}(i,j)a_{ijl}b_l(O_{t+2})\hat{\beta}_{t+2}(j,l)}$$

soit :

$$\bar{a}_{ijk} = \frac{\sum_{t=1}^{T-2}C_{t+1}\alpha_{t+1}(i,j)a_{ijk}b_k(O_{t+2})D_{t+2}\beta_{t+2}(j,k)}{\sum_{t=1}^{T-2}\sum_{l=1}^{N}C_{t+1}\alpha_{t+1}(i,j)a_{ijl}b_l(O_{t+2})D_{t+2}\beta_{t+2}(j,l)}$$

Or :

$$C_{t+1}D_{t+2} = (\prod_{r=1}^{t+1} c_r)(\prod_{r=t+2}^{T} c_r) = \prod_{r=1}^{T} c_r = C$$

où C est une quantité indépendante de t. Il en résulte que la formule de ré-estimation est la même qu'avec α et β d'origine. Cette procédure d'échelle traite de la même façon les autres formules de ré-estimations.

Dans nos implantations, nous avons utilisé cette dernière approche avec un facteur d'échelle utilisant le maximum au lieu de la somme (donc toujours indépendant de i et de j) :

$$c_t = \frac{1}{\max_{1\leq i,j\leq N}\alpha_t(i,j)}$$

3.5.3 Problème de données insuffisantes

Un autre problème associé à l'apprentissage des paramètres de HMM via la méthode de ré-estimation est que la suite d'observations pour cet apprentissage est nécessairement finie. Or, Baum et Petrie [Baum 66] ont montré que l'estimation du maximum de probabilité des paramètres d'un HMM converge vers les vraies valeurs quand le nombre d'observations tend vers l'infini. L'implication de ce théorème est que, en apprentissage, il y a souvent un manque d'échantillons représentatifs des différents événements : on devra utiliser autant d'observations que possible. Cette première solution qui consiste à augmenter la taille de données d'observations pour l'apprentissage n'est souvent pas possible. Une seconde solution est de réduire la taille des modèles (i.e. le nombre d'états, le nombre de symboles dans chaque état etc.). Même si, cette solution est toujours possible, il y a souvent des raisons physiques d'avoir choisi tel ou tel modèle et donc la taille des modèles ne peut pas être changée.

Le problème de l'insuffisance du nombre des observations dans l'apprentissage, implique que certains événements de faible probabilité peuvent ne pas apparaître dans cet ensemble fini d'observations. Aussi, ces événements se voient-ils affectés d'une probabilité nulle dont la valeur peut avoir par la suite des répercussions très défavorables. Les deux méthodes les plus connues pour pallier ce problème sont celle de Jelinek et Mercer [Jelinek 80] et celle de Levinson et Rabiner [Levinson 83a].

La méthode de Jelinek et Mercer utilise des informations supplémentaires par rapport à celles de la séquence d'observations pour estimer les petites probabilités.

Levinson et Rabiner utilisent une méthode plus simple, consistant à ajouter des contraintes supplémentaires aux paramètres des modèles pour empêcher les valeurs des paramètres de descendre en dessous d'un certain seuil. Par exemple, on peut spécifier une contrainte, qui, pour des modèles avec des symboles discrets comme les nôtres, vise à empêcher les valeurs de la matrice B d'être inférieures au seuil ε, c'est-à-dire :

$$b_j(k) \geq \varepsilon;$$

ou pour des modèles de distributions continues vise à empêcher les valeurs de la diagonale de la matrice de covariance U_{jm} d'être inférieures au seuil δ, c'est-à-dire :

$$U_{jm}(r,r) \geq \delta$$

C'est cette dernière approche que nous avons choisie pour l'implantation des HMMs discrets, en testant plusieurs valeurs de ε pour choisir la meilleure valeur permettant de diminuer les effets d'insuffisance de données d'apprentissage. L'algorithme d'estimation est inchangé puisque les contraintes sont appliquées comme un post-processeur aux formules de ré-estimations, si une contrainte est violée, le paramètre relevant est corrigé et tous les autres paramètres correspondants sont reéchelonnés pour que les contraintes stochastiques soient respectées. Une telle technique, post-processeur, a été appliquée à plusieurs problèmes de traitement de la parole avec grand succès [Rabiner 88].

3.5.4 Initialisation des HMMs

De nombreuses techniques d'apprentissage des HMMs et notamment celle de Baum-Welch, démontrée dans [Baum 72], se heurtent, dans leur processus d'optimisation, à des extrema locaux qui ne sont pas forcément globaux dans l'espace des paramètres des modèles, ce qui empêche les nouvelles estimations d'évoluer.

Une méthode pour converger vers l'optimum global, consiste à perturber aléatoirement les paramètres afin de s'éloigner du domaine de l'optimum local [Paul 85,Jouvet 87].

Une deuxième méthode pour réduire le nombre d'itérations au niveau de l'apprentissage et pour augmenter la chance d'atteindre le maximum global, est que les paramètres doivent être correctement initialisés. La question clef est : comment choisir les estimations initiales des paramètres HMM, pour que le maximum local atteint, soit le maximum global? A la base, il n'y a aucune méthode simple ni directe permettant d'estimer les valeurs initiales des paramètres HMM. Des expériences ont montré que des estimations initiales aléatoires, ou bien des estimations équiprobables de Π et de la matrice A sont adéquates pour donner des ré-estimations convenables de ces paramètres. Cependant, pour les paramètres de la matrice B, l'expérience a montré que des bonnes estimations initiales sont utiles dans la cas de symboles discrets et sont obligatoires dans le cas des distributions continues [Rabiner 88]. De telles estimations peuvent être extraites de l'ensemble de données d'apprentissage (ou d'une partie de cet ensemble) à l'aide des segmentations manuelles ou d'une procédure de segmentation automatique [Russel 86].

Nous avons choisi d'utiliser, essentiellement, l'initialisation équiprobable, puisque nos modèles sont des HMMs discrets et des tests effectués sur des initialisations à partir des données d'apprentissage ont donné pour notre part presque les mêmes résultats.

3.6 Comparaison expérimentale

3.6.1 Conditions expérimentales

Pour évaluer cette extension au second ordre, nous nous sommes placés dans les mêmes conditions expérimentales d'une étude comparative, faite au sein de notre laboratoire, entre quatre méthodes statistiques de reconnaissance de la parole [Boyer 88]. Ces méthodes sont succintement :

- Classification des formes acoustiques [Divoux 88] : Cette méthode consiste à construire un ensemble de références classifiées durant la phase d'apprentissage dans le but de réduire la redondance des données. Elle est basée sur une notion de distance entre deux mots utilisant les scores donnés par une programmation dynamique [Jambu 78,Rabiner 79,Briant 84,Divoux 85].
- Comparaison d'automates d'états finis [Martino 87] : Après une quantification vectorielle, cette méthode repose sur une technique permettant la construction d'un automate d'états finis pour chaque mot et la reconnaissance se fait par une programmation dynamique adaptée aux automates d'états finis.
- Programmation dynamique et quantification vectorielle (DTW+VQ) : Les détails de cette méthode bien connue dans les applications de reconnaissance des formes, en

particulier dans la reconnaissance des mots isolés et enchaînés, peuvent être consultés dans [Burton 84,Martino 85,Boyer 87b] et dans [Linde 80] pour l'algorithme Linde-Buzo-Gray de construction du dictionnaire des prototypes.

- La modélisation stochastique utilisant les HMMs du premier ordre. Elle repose, comme déjà vu, sur l'algorithme d'apprentissage Baum-Welch (ou Forward-Backward) pour le premier ordre [Baum 72,Baker 75c,Liporace 82,Rabiner 83] et sur celui de Viterbi pour la reconnaissance [Viterbi 67,Forney 73].

Ces quatres méthodes que nous allons appeler respectivement : **Classification, Automate, DTW+VQ, HMM1**, ont été expérimentées dans les mêmes conditions, pour pouvoir les comparer convenablement. Le corpus utilisé est un ensemble de prononciations des chiffres en français (0 à 9), enregistrées par 60 locuteurs (30 hommes et 30 femmes) . Chaque locuteur a prononcé 4 fois chacun des chiffres. La moitié du corpus (15 hommes et 15 femmes) a servi pour l'apprentissage, et le reste pour le test. Dans ces expériences, une distinction a été faite entre un système multi-locuteurs capable de reconnaître les mots prononcés par les locuteurs qui ont été utilisés pendant la phase d'apprentissage et un système indépendant du locuteur capable de reconnaître les mots prononcés par des locuteurs totalement inconnus au système.

Le corpus est enregistré sur une machine Masscomp avec une fréquence d'échantillonnage de 12 khz sur 10 bits. L'analyse spectrale est une FFT utilisant une fenêtre de Hamming de 256 points. Le recouvrement est de 100 points.

Dans nos expériences le dictionnaire de la QV est obtenu par classification de la parole des 30 locuteurs de l'ensemble d'apprentissage en 128 classes. Chaque symbole de sortie est un élément du dictionnaire, déterminé par la quantification du spectre de parole calculé chaque 8 ms.

3.6.2 Nombre d'états d'un HMM

L'obligation de définir explicitement au départ la structure des modèles HMMs utilisés, nous a poussé à mener une étude expérimentale sur le choix du nombre d'états de ces modèles.

Différents nombres d'états ont été testés dans diverses applications : modèles d'unités phonétiques à deux états de Jelinek [Jelinek 76], modèles à N états de Bakis qui est largement utilisé [Billi 82,Bourlard 86], etc. Une étude portant sur la reconnaissance de nombres avec des HMMs de différents nombres d'états a été faite par Russel [Russel 86] : Les meilleurs résultats sont obtenus pour un modèle possédant un grand nombre d'états (i.e. 20). De nombreux autres résultats optent pour des modèles plus restreints (très souvent 5 à 7 états) [Levinson 83b,Rabiner 83,Juang 85a,Juang 85b,Rabiner 85,Sugawara 85, Juang 86]. Dans une étude préliminaire, K.M.Ponting [Ponting 88] utilise un critère statistique afin de déterminer automatiquement le nombre d'états optimal d'un modèle à partir d'un corpus d'apprentissage.

Le nombre d'états d'un HMM peut donc être déterminé par deux approches :

1. Une approche purement expérimentale, en menant des tests sur plusieurs HMMs de nombres d'états différents, pour déterminer le nombre d'états optimal à partir des meilleurs résultats [Russel 86].

2. Une approche qui essaie de donner à un HMM une explication physique explicite, telle que le fait de considérer qu'un HMM est composé d'un nombre fini d'états, représentant les différents segments stables de l'unité, et d'arcs de transitions entre ces états modélisant les variations spectrales [Dours 89].

Cette dernière approche peut être traitée à partir de deux classes de réflexions :

- La première est de considérer un nombre d'états correspondant approximativement au nombre de sons (par exemple, nombre de phonèmes à l'intérieur d'un mot). D'où, des modèles de 2 à 10 états qui seront appropriés.

- La deuxième est de considérer un nombre d'états correspondant approximativement au nombre moyen des observations d'un ensemble d'occurrences de l'unité de reconnaissance. De cette manière, chaque état correspondra grossièrement à un intervalle d'observations.

Notons que la plupart des applications utilisent une restriction qui consiste à considérer un nombre d'états unique pour tous les modèles HMMs. En général, cette restriction n'a pas une grande influence sur les performances, et elle simplifie la programmation.

En ce qui nous concerne, nous avons considéré un nombre d'états unique pour tous les modèles HMMs. Un état initial et un état final sont souvent ajoutés à un modèle du premier ordre pour des raisons techniques (simplification de la programmation). Ce qui n'a pas d'effet sur les performances du modèle. Pour les mêmes raisons, et comme il est indiqué clairement sur la figure 3.1, nous avons ajouté deux états de chaque côté du modèle du second ordre. Ce qui permet une simplification technique évidente, sans influence sur le modèle qui peut être considéré comme un modèle de cinq états dans la figure 3.1(a) au lieu de huit (pas d'observations aux états 0 et 1,...).

Notre étude a pour but de comparer les taux de reconnaissance fournis par ce modèle de 5 états [figure 3.1(a)] et par ceux de la même topologie ayant respectivement 6, 7, 8 et 9 états. Ainsi, notre système utilisant les HMMs du second ordre (HMM2), représenté sur la figure 3.5, est testé en "locuteur-indépendent" avec le corpus précédent des chiffres, en partant de ces modèles initiaux respectifs. Les résultats de cette expérience représentés par la courbe de la figure 3.6, indiquent que les meilleurs taux de reconnaissance obtenus sont ceux donnés par les modèles de 6 et 7 états.

3.6.3 Test en monolocuteur

Dans un premier temps, nous avons étudié le comportement du système HMM2 en monolocuteur. La figure 3.7 illustre le nombre de prononciations mal reconnues par trois locuteurs choisis aléatoirement parmi 60 locuteurs du corpus des chiffres précédent. Les colonnes de cette figure correspondent aux résultats obtenus en utilisant pour l'apprentissage respectivement une, deux et trois prononciations par chiffre. Nous remarquons que pour trois prononciations par chiffre, l'apprentissage permet une reconnaissance à 100% pour les trois locuteurs.

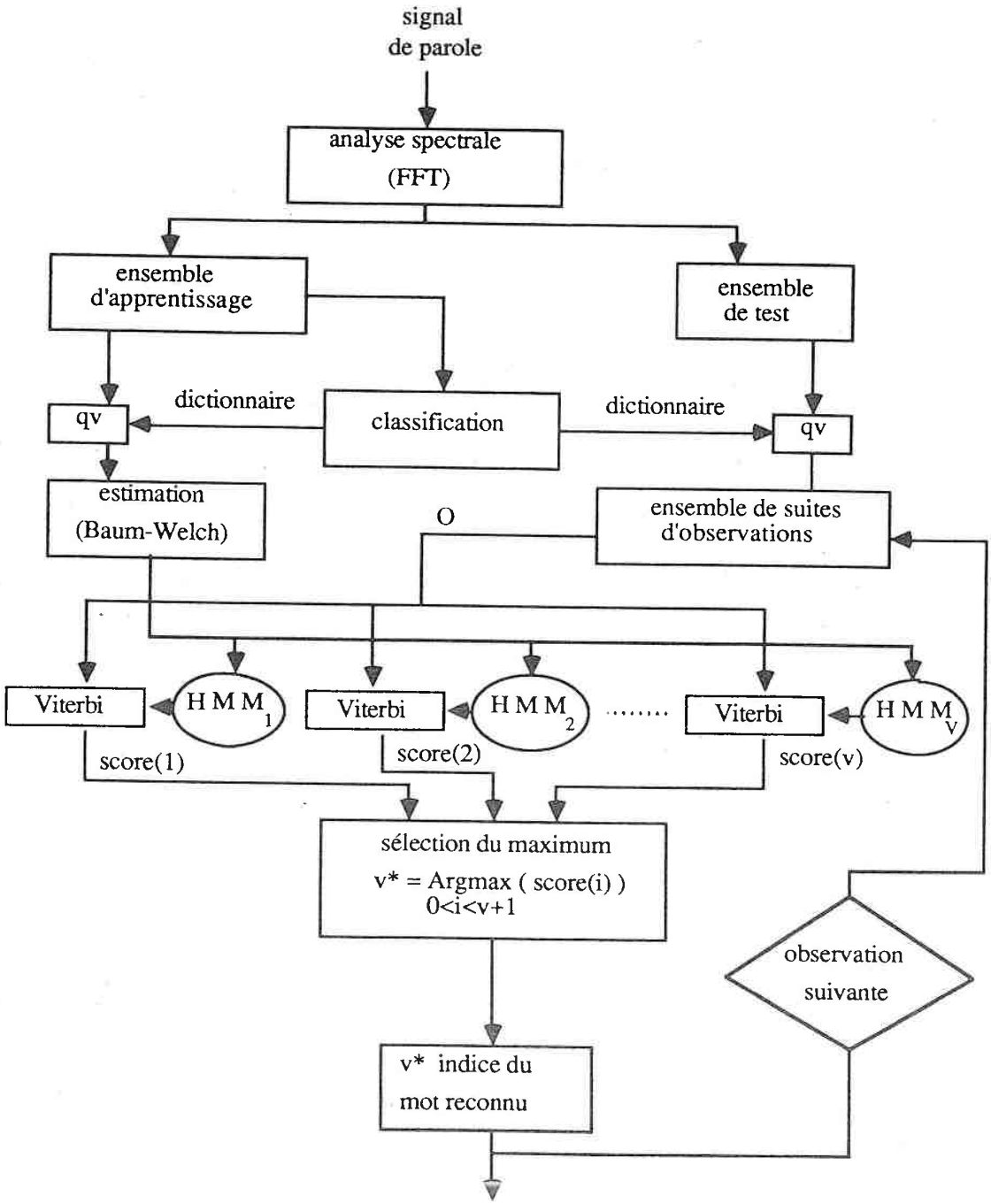


Figure 3.5. Diagramme du système HMM2

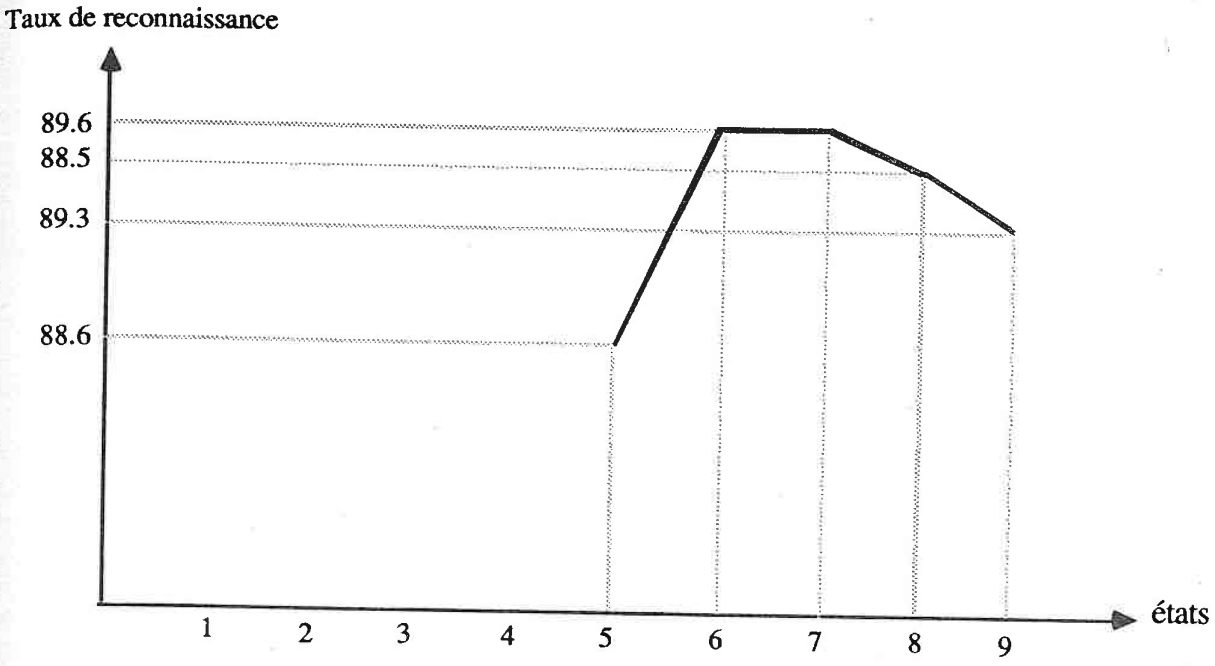


Figure 3.6. Comparaison des nombres d'états

Nombre d'occurrences pour l'apprentissage		10	20	30
Nombre d'occurrences pour le test		30	20	10
Nombre d'occurrences mal reconues par le :	1er locuteur	2	1	0
	2e locuteur	2	0	0
	3e locuteur	0	1	0

Figure 3.7. Test en monolocuteur

	HMM1 (homologue)	HMM1 (équivalent)	HMM2
Taux de reconnaissance (%)	87.2	86.9	89.4

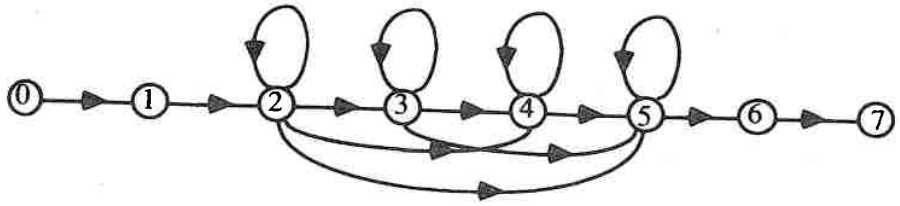
Figure 3.8. Comparaison des systèmes HMM (5 états, 3 itérations)

3.6.4 Première comparaison

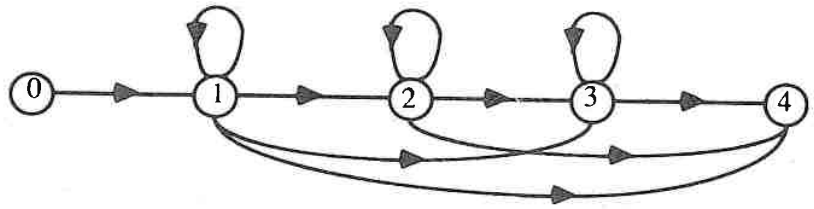
Dans cette première comparaison entre les deux systèmes HMMs, du premier et du second ordre, nous nous sommes intéressés à la question du passage au modèle équivalent évoquée au début de ce chapitre. Ainsi, le système HMM2 est testé avec le corpus des chiffres, en partant du modèle initial illustré sur la figure 3.1(a). Son modèle équivalent du premier ordre, illustré sur la figure 3.1(b), est utilisé avec ce même corpus pour tester le système HMM1 qui a en fait, à un ordre près pour Baum-Welch et l'algorithme de Viterbi, la même configuration que celle de la figure 3.5.

Les résultats de la figure 3.8 sont les taux de reconnaissance globaux obtenus après un test indépendant du locuteur dans les conditions expérimentales décrites précédemment (30 locuteurs pour l'apprentissage, 30 locuteurs pour le test), où le nombre des itérations sur le corpus d'apprentissage est 3. La troisième colonne indique le taux obtenu par le système du second ordre initialisé par le modèle de la figure 3.1(a), la première et la deuxième montrent les taux obtenus par le système du premier ordre initialisé respectivement par l'homolgue du même modèle à la figure 3.9(b) et par son équivalent de la figure 3.1(b). Nous remarquons que les résultats donnés par le système utilisant les HMMs du second ordre sont meilleurs que les deux autres. La diminution du taux d'erreurs de reconnaissance représente un pourcentage supérieur à 17%.

3.6.5 Deuxième comparaison



(a)



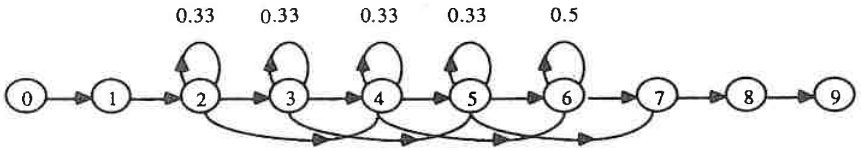
(b)

Figure 3.9. Un HMM du second ordre (a) et son homologue en premier ordre (b)

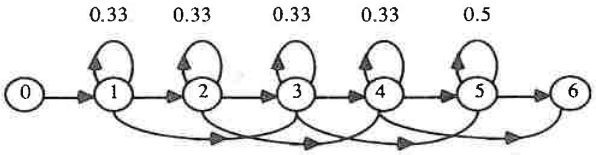
Pour apprécier les performances du reconnaisseur utilisant les HMMs du second ordre d'une part, et les comparer avec celles données par les quatre méthodes illustrées dans le paragraphe "Conditions expérimentales" d'autre part, nous avons recommencé le test dans les conditions expérimentales communes aux cinq méthodes mais avec le modèle HMM à 7 états de la figure 3.10(b) utilisé dans [Boyer 88], avec son homologue du second ordre repésenté à la figure 3.10(a) et avec l'équivalent de ce dernier en premier ordre illustré à la figure 3.10(c).

La figure 3.11 représente les taux de reconnaissance obtenus par les cinq systèmes en modes multi-locuteurs et locuteur-indépendant, toujours avec trois itérations d'apprentissage. Elle informe aussi sur le temps demandé par chaque système HMM pour reconnaître les 1200 mots du corpus test. Nous obtenons, ainsi, en moyenne :

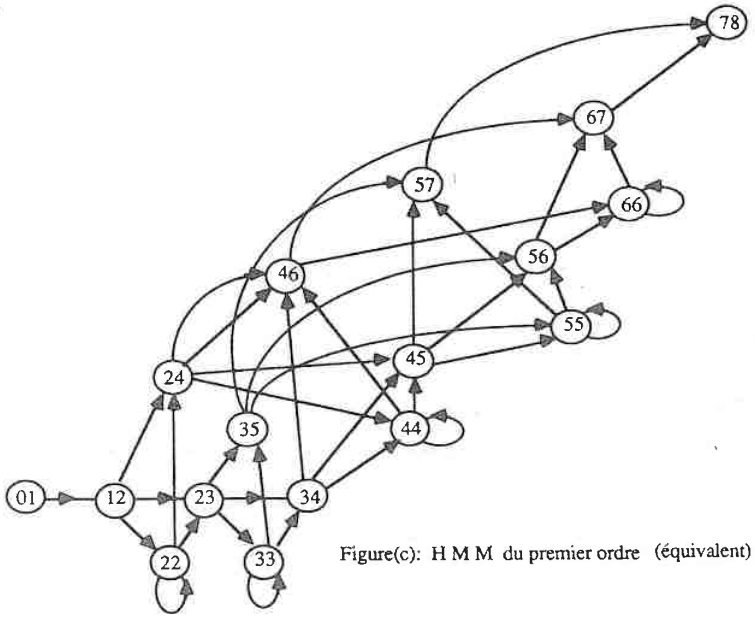
- 0.53 secondes/mot, pour le système HMM1 utilisant le modèle homologue,
- 1.35 secondes/mot, pour le système HMM1 utilisant le modèle équivalent,
- 1.62 secondes/mot, pour le système HMM2.



Figure(a): H M M du second ordre



Figure(b): H M M du premier ordre (homologue)



Figure(c): H M M du premier ordre (équivalent)

Figure 3.10. Un HMM du second ordre (a), son homologue (b) et son équivalent (c) en premier ordre

	Classification	Automate	DTW+VQ	HMM 1		HMM 2
				homol.	équi.	
Multi-locuteurs	86	86	91	96	97	99
locuteur-indépendant (valeurs relatives du temps/mot)	81 (1)	80 (1)	87 (1)	91 (0.1)	91 (0.27)	93 (0.32)

Figure 3.11. Taux de reconnaissance des cinq systèmes (7 états, 3 itérations)

Pour une comparaison plus profonde entre HMM1 et HMM2, la figure 3.12 donne leur matrices de confusion. Théoriquement, le système HMM2 fournit une modélisation plus précise, les taux de reconnaissance des chiffres doivent s'améliorer par rapport à ceux du système HMM1. Nous remarquons qu'au contraire des autres chiffres, le taux de reconnaissance du "sept" se dégrade pour le système HMM2. Il est souvent confondu avec le chiffre cinq. Cette confusion a, évidemment, une explication acoustico-phonétique. Cette dégradation du taux de reconnaissance du chiffre "sept", peut être expliquée, a priori, par les données d'apprentissage qui sont devenues insuffisantes pour bien estimer le modèle du second ordre correspondant. Ce modèle qui a naturellement plus de paramètres que son homologue du premier ordre. D'ailleurs, le modèle équivalent a presque la même matrice de confusion que celle du modèle homologue du premier ordre, sauf que le taux de reconnaissance du chiffre "sept" est de 76%. Ce résultat est conforme avec le fait que le modèle du second ordre a moins de paramètres à estimer que son modèle équivalent et avec l'explication précédente de l'insuffisance des données d'apprentissage.

Dans le but d'avoir plus de précision sur les deux systèmes, la figure 3.13 illustre une autre comparaison entre HMM1 et HMM2 par rapport au nombre des itérations d'apprentissage. Il montre, aussi, l'influence du nombre des itérations sur les résultats de la reconnaissance. Nous remarquons que pour cette application sur les chiffres, il est inutile d'aller au delà de 3 itérations et qu'il a fallu ces 3 itérations pour avoir le taux de reconnaissance maximal dans les deux systèmes.

	0	1	2	3	4	5	6	7	8	9
0	99									
1		86			10					
2	4	1	90						1	
3				100						
4					99					
5		3			2	83		8		
6	2						84		10	
7	1				4	11		81		
8									98	
9			1							97

	0	1	2	3	4	5	6	7	8	9
0	99									
1	1	93			2					2
2			95							2
3				100						
4					99					
5		1				89		7		
6	2						85		9	1
7		2			2	14		78		
8									100	
9			1							98

Figure 3.12. Matrices de confusion des systèmes HMM1 et HMM2

taux de reconnaissance \ itérations	1	2	3	4	5	6
HMM 1 (homologue)	89	90	91	91	91	91
HMM 2	90	92	93	93	93	93

Figure 3.13. Influence du nombre d'itérations sur les taux de reconnaissance

Dans le cas réel, qui est la reconnaissance en mode indépendant du locuteur, le système HMM2 apporte une amélioration de 2% par rapport aux performances du système HMM1. Cette amélioration représente un pourcentage de diminution du taux d'erreurs de reconnaissance de 23%. D'où l'intérêt de passer à un ordre supérieur de HMM.

Le paragraphe suivant donne une généralisation des systèmes markoviens, permettant ainsi de passer à un ordre supérieur quelconque.

3.7 Généralisation théorique à un ordre quelconque

3.7.1 Généralisation

Pour terminer ce chapitre, nous proposons une généralisation des algorithmes précédents à un ordre supérieur quelconque, r. Cette généralisation ne sera qu'une extension théorique des définitions et des formules déjà présentées pour les HMMs du second ordre. Ainsi, nous commençons par définir les fonctions Forward et Backward correspondant à l'ordre r (r est un entier strictement positif) :

$$\alpha_t(i_1, i_2, .., i_r) = P(O_1 O_2 .. O_t, q_{t-r+1} = s_{i_1}, q_{t-r+2} = s_{i_2}, .., q_t = s_{i_r}/\lambda)$$

et :

$$\beta_t(i_1, i_2, .., i_r) = P(O_{t+1}O_{t+2}..O_T/q_{t-r+1} = s_{i_1}, q_{t-r+2} = s_{i_2}, .., q_t = s_{i_r}, \lambda)$$

Pour alléger les notations, nous allons noter un état s_i par son indice i.
Le calcul de la fonction Forward se fera par la récurrence suivante :

- Initialisation

$$\alpha_r(i_1, i_2, .., i_r) = \Pi_{i_1} b_{i_1}(O_1) a_{i_1 i_2} b_{i_2}(O_2) a_{i_1 i_2 i_3} b_{i_3}(O_3)..a_{i_1 i_2..i_r} b_{i_r}(O_r)$$

- Récurrence

$$\alpha_{t+1}(i_2, i_3, .., i_{r+1}) = [\sum_{i_1=1}^{N} \alpha_t(i_1, i_2, .., i_r) a_{i_1 i_2..i_{r+1}}].b_{i_{r+1}}(O_{t+1})$$

et nous aurons donc :

$$P(O/\lambda) = \sum_{i_1, i_2, .., i_r=1}^{N} \alpha_T(i_1, i_2, .., i_r)$$

De même, le calcul de la fonction Backward sera induit de la façon suivante :

- Initialisation

$$\beta_T(i_1, i_2, .., i_r) = 1$$

- Récurrence

$$\beta_t(i_1, i_2, .., i_r) = \sum_{i_{r+1}=1}^{N} a_{i_1 i_2 .. i_{r+1}} b_{i_{r+1}}(O_{t+1}) \beta_{t+1}(i_2, i_3, .., i_{r+1})$$

Pour une généralisation convenable des formules de ré-estimations nous avons défini les quantités suivantes :

$$\xi_t^{r+1}(i_1, i_2, .., i_{r+1}) = P(q_{t-r+1} = i_1, q_{t-r+2} = i_2, .., q_{t+1} = i_{r+1}/O, \lambda)$$

$$\xi_t^{r}(i_1, i_2, .., i_{r}) = P(q_{t-r+2} = i_1, q_{t-r+3} = i_2, .., q_{t+1} = i_{r}/O, \lambda)$$

...

$$\xi_t^{2}(i_1, i_2) = P(q_t = i_1, q_{t+1} = i_2/O, \lambda)$$

$$\gamma_t(i_1) = P(q_t = i_1/O, \lambda)$$

Puis, nous avons démontré les relations suivantes :

$$\xi_t^{r+1}(i_1, i_2, .., i_{r+1}) = \frac{\alpha_t(i_1, i_2, .., i_r) a_{i_1 i_2 .. i_{r+1}} b_{i_{r+1}}(O_{t+1}) \beta_{t+1}(i_2, i_3, .., i_{r+1})}{P(O/\lambda)}$$

$$\xi_t^{r}(i_1, i_2, .., i_r) = \sum_{i_{r+1}=1}^{N} \xi_{t+1}^{r+1}(i_1, i_2, .., i_{r+1})$$

...

$$\xi_t^2(i_1, i_2) = \sum_{i_3=1}^{N} \xi_{t+1}^3(i_1, i_2, i_3)$$

$$\gamma_t(i_1) = \sum_{i_2=1}^{N} \xi_t^2(i_1, i_2)$$

Les formules de ré-estimations seront donc celles utilisées dans l'algorithme d'apprentissage Baum-Welch pour les HMMs d'ordre r :

$$\bar{\Pi}_{i_1} = \gamma_1(i_1)$$

$$\bar{a}_{i_1 i_2} = \frac{\xi_1^2(i_1, i_2)}{\gamma_1(i_1)}$$

$$\bar{a}_{i_1 i_2 i_3} = P(q_3 = i_3 / q_1 = i_1, q_2 = i_2, \lambda) = \frac{\xi_2^3(i_1, i_2, i_3)}{\xi_1^2(i_1, i_2)}$$

...

$$\bar{a}_{i_1 i_2 .. i_r} = \frac{\xi_{r-1}^r(i_1, i_2, .., i_r)}{\xi_{r-2}^{r-1}(i_1, i_2, .., i_{r-1})}$$

$$\bar{a}_{i_1 i_2 ... i_{r+1}} = \frac{\sum_{t=r-1}^{T-2} \xi_{t+1}^{r+1}(i_1, i_2, .., i_{r+1})}{\sum_{t=r-1}^{T-2} \xi_t^r(i_1, i_2, .., i_r)}$$

$$\bar{b}_j(k) = \frac{\sum_{t=1, O_t = v_k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}$$

D'autre part, l'extension de l'algorithme de Viterbi se fait de la même manière que celle utilisée pour les HMMs du second ordre. Nous aurons ainsi un algorithme de Viterbi pour un ordre quelconque r basé sur la quantité :

$$\delta_t(i_1, i_2, .., i_r) = \max_{q_1 q_2 .. q_{t-r}} P(q_1 q_2 .. q_{t-r} q_{t-r+1} = i_1, .., q_t = i_r, O_1 O_2 .. O_t / \lambda)$$

et résumé par les quatres étapes suivantes :

- Initialisation

$$\delta_1(i_1) = \Pi_{i_1} b_{i_1}(O_1)$$

$$\delta_2(i_1, i_2) = \delta_1(i_1) a_{i_1 i_2} b_{i_2}(O_2)$$

...

$$\delta_r(i_1, i_2, .., i_r) = \delta_{r-1}(i_1, i_2, .., i_{r-1}) a_{i_1 i_2 .. i_r} b_{i_r}(O_r)$$

- Récurrence

$$\delta_t(i_2, i_3, .., i_{r+1}) = \max_{i_1}[\delta_{t-1}(i_1, i_2, .., i_r) a_{i_1 i_2 .. i_{r+1}}].b_{i_{r+1}}(O_t)$$

$$\psi_t(i_2, i_3, .., i_{r+1}) = \arg\max_{i_1}[\delta_{t-1}(i_1, i_2, .., i_r) a_{i_1 i_2 .. i_{r+1}}]$$

- Terminaison

$$P^* = \max_{i_1, i_2, .., i_r} [\delta_T(i_1, i_2, .., i_r)]$$

$$q^*_{T-r+1} = \arg_{i_1} \max_{i_1, i_2, .., i_r} [\delta_T(i_1, i_2, .., i_r)]$$

...

$$q^*_T = \arg_{i_r} \max_{i_1 i_2 .. i_r} [\delta_T(i_1, i_2, .., i_r)]$$

- Chemin d'états retenu

$$q^*_t = \psi_{t+r}(q^*_{t+1}, q^*_{t+2}, .., q^*_{t+r})$$

Remarques :

- Les démonstrations des formules de cette généralisation se font de manière similaire à celles développées pour les HMMs du second ordre.
- L'utilisation des HMMs d'ordre r suppose que le nombre N des états du modèle markovien initialement choisi soit strictement supérieur à r.
- Pour reconnaître une suite d'observations de longueur T, l'algorithme de Viterbi d'un HMM d'ordre r, utilise environ $N^{r+1}T$ opérations et de l'ordre de $N^{r+1}T$ comparaisons.

3.7.2 Perspectives

Après cette généralisation des algorithmes du système de reconnaissance HMM à un ordre quelconque deux grandes questions se posent alors :

- A quel ordre pourrons nous aller tout en ayant, d'un côté, de plus en plus de performances et tout en gardant, d'un autre côté, l'efficacité du calcul ? Aura-t-on une sorte de convergence pour les taux de reconnaissance ? et quels sont ses critères ?
- Est-il possible de créer un système général utilisant les HMMs de n'importe quel ordre suivant la demande de l'utilisateur, par exemple ?

Etant donné, l'évolution rapide du matériel informatique qui est de plus en plus performant, des éléments de réponse peuvent être donnés à ces deux questions loin du problème de l'efficacité de calcul ou non. En fait, pour une application donnée, le passage à un ordre supérieur implique l'augmentation du nombre de paramètres à estimer. Les données d'apprentissage qui sont fixées, seront alors insuffisantes à partir d'un certain ordre. L'idéal serait de passer à l'ordre supérieur tant que les données d'apprentissage permettent une bonne estimation des paramètres. Le choix d'un ordre HMM dépend donc, à priori, de la taille de l'application. Un système général utilisant plusieurs ordres HMM sera sans doute intéressant à envisager.

3.8 Conclusion

Nous avons montré dans ce chapitre l'utilisation pratique des HMMs du second ordre grâce aux formules de ré-estimations que nous avons mises au point pour pouvoir estimer automatiquement les paramètres de ces modèles.

La nette amélioration du taux de reconnaissance montre l'importance du passage au second ordre dans les modèles markoviens cachés. Le reconnaisseur de mots isolés utilisant ces modèles, ne peut, bien sûr, qu'être plus performant lorsqu'on lui apporte les améliorations habituelles, telles que le passage du discontinu au continu ou au semi-continu et la bonne initialisation des modèles par des prétraitements du corpus d'apprentissage. D'autre part, le passage à un système de reconnaissance de la parole continue utilisant ce type de modèles sera sans doute intéressant.

Motivés par ce surcroit d'efficacité, nous avons ensuite proposé dans le dernier paragraphe les éléments d'un système utilisant les HMMs d'un ordre quelconque. Une étude dans cet axe de recherche serait très utile.

4
L'algorithme VITERBI-BLOC

4.1 Introduction

Nous avons vu que les HMMs peuvent être utilisés pour la reconnaissance de la parole continue en généralisant l'algorithme de décodage. Ce chapitre fait l'objet de plusieurs versions de l'algorithme de Viterbi utilisées pour ce type de reconnaissance. Dans notre travail, nous ne nous sommes intéressés qu'au niveau de décodage acoustico-phonétique qui est une étape importante dans les systèmes de la reconnaissance de la parole continue. Plus précisément, nos différentes versions de l'algorithme de Viterbi sont testées avec une étape de segmentation en grandes classes phonétiques. Une de ces versions, appelée VITERBI-BLOC, propose une nouvelle solution au problème de la modélisation de la durée dans un système de reconnaissance de la parole continue utilisant les HMMs.

4.2 Etape de segmentation

4.2.1 Segmentation de la parole continue

Dans un processus de reconnaissance automatique de la parole continue, on fait souvent appel au décodage acoustico-phonétique. La transcription d'un signal continu en une suite discrète d'éléments appartenant à un vocabulaire fini (par exemple, l'ensemble des phonèmes), est un problème difficile et crucial, puisque les performances des niveaux supérieurs de reconnaissance et de dialogue homme-machine, dépendent fortement de la qualité de ce codage.

Dans le but de réduire l'indéterminisme, plusieurs systèmes de décodage acoustico-phonétique utilisent une étape de segmentation. La segmentation de la parole continue est un problème très délicat; il est difficile à résoudre puisqu'il n'existe pas d'unité élémentaire caractérisée par des invariants que l'on rechercherait; elle est cependant nécessaire pour réduire le flot d'information qui est issu du signal et pour faciliter la reconnaissance et réduire le nombre d'éléments à reconnaître.

Une bonne segmentation de la parole est un bon départ pour le décodage acoustico-phonétique. C'est dans ce cadre que nous avons réalisé plusieurs expériences servant de test aux algorithmes que nous allons proposer dans ce chapitre.

Dans les différents systèmes de reconnaissance analytique, nous distinguons trois démarches de segmentation différentes [Obrecht 85] :

- La segmentation avec reconnaissance :
  Dans ces méthodes la paramétrisation du signal est liée au fait que c'est un signal de nature parole avec ses caractéristiques propres (formants,traits,...). L'utilisation de ces connaissances permet une première reconnaissance. Ce type de segmentation fait très souvent appel à des notions utilisées en reconnaissance des formes (comparaison à des modèles prototypes,...) ou en intelligence artificielle (règles heuristiques,...). Il est le plus employé par les systèmes de reconnaissance de la parole continue. Sa caractéristique essentielle est de ne pas dissocier l'étape segmentation de l'étape identification. Cette identification peut être en éléments acoustiques fins (phonèmes,syllabes,...) [Klatt 80,Breant 82] ou de façon plus grossière (voyelles,fricatives,...) [Caelen 78,Atal 76]. Cet étiquetage grossier a pour but de guider l'étape suivante d'identification. Chaque segment obtenu ne correspond pas à une unité élémentaire à reconnaître, mais appartient à une catégorie d'unités élémentaires. Cette segmentation cherche en priorité à restreindre la recherche de l'unité élémentaire à une certaine fraction du dictionnaire.

- La segmentation séquentielle indicielle :
  La première étape de ces méthodes consiste comme précédemment à extraire du signal de parole, des paramètres pertinents, liés à la nature du signal de parole, d'où le nom indice, puis une fonction de segmentation où une distance est calculée à partir des seuls paramètres de signal, sans accès à quelque référence que ce soit. La segmentation est faite de manière séquentielle par franchissement de seuils, et donne les frontières d'unités élémentaires. Aucun étiquetage n'est fait.
  Au lieu d'utiliser une fonction de segmentation, comme dans les systèmes ARIAL II et dans [Ruske 81], le système HARPY [Lowerre 80] procède à une comparaison séquentielle des échantillons entre eux pour détecter une variation. Mais très souvent, un décodeur acoustique combine plusieurs méthodes de segmentation pour arriver à de bons résultats, tel que le système MOSES [Lochschmidt 82] et le système MYRTILLE [Haton 84].

- La segmentation par recherche de stationnarités :
  La recherche des segments est basée sur la notion de stationnarité utilisée en traitement de signal, la paramétrisation et les tests utilisés tiennent peu compte de la spécificité du signal de parole. De manière générale, le signal de parole est considéré comme une juxtaposition de zones stationnaires et de zones transitoires [Ariki 84].

4.2.2 Segmentation de la parole par des HMMs

Au CRIN, la tendance à choisir le premier type de segmentation est marquée par les prétraitements du système expert APHODEX pour le décodage acoustico-phonétique de la parole continue [Fohr 86]. Ces prétraitements sont des algorithmes procéduraux qui reposent sur différents critères simples (intensité d'énergie, barre d'explosion, voisement,...) pour segmenter le signal en grandes classes phonétiques : noyaux vocaliques, fricatives, plosives.

Dans le but de contribuer à l'amélioration de cette segmentation qui est la base de ce système expert de décodage acoustico-phonétique d'une part, et de tester les différents algorithmes de reconnaissance qui vont être proposés d'autre part, nous allons illustrer une approche pour segmenter la parole continue s'inscrivant toujours dans le premier type

de méthodes de segmentation, mais cette fois-ci reposant sur des modèles markoviens cachés [Kriouile 86]. Cette approche stochastique consiste à construire, par l'algorithme de Baum-Welch, un modèle markovien caché pour chacune des classes phonétiques plus ou moins fines, et un dernier modèle pour le silence. La reconnaissance se fera par une des versions de l'algorithme de Viterbi décrites aux paragraphes 5.3 et 5.5. Pour le moment, nous définissons les six grandes classes phonétiques suivantes :

- PV : plosives voisées (/b/ , /d/ , /g/).
- PS : plosives sourdes (/p/ , /t/ , /k/).
- FV : fricatives voisées (/v/ , /z/ , /z/).
- FS : fricatives sourdes (/f/ , /s/ , /ʃ/).
- VY : voyelles.
- RE : le reste des phonèmes.

Ainsi, une comparaison des deux segmentations serait intéressante :

- La segmentation par les procédures de prétraitement d'APHODEX.
- La segmentation par cette méthode stochastique appliquée à ces six classes (sans oublier le silence).

Mais le plus intéressant serait d'exploiter les résultats des deux segmentations [Haton 87]. Par exemple, modifier APHODEX pour raisonner en parallèle sur plusieurs segmentations ou réaliser un système plus général (ayant une architecture de Blackboard par exemple) réalisant une interaction entre les deux méthodes; cette voie constituera la suite de ce travail.

4.3 Premières versions de l'algorithme de reconnaissance

4.3.1 VITERBI-réseau

En reconnaissance de mots isolés, nous avons vu que le mot reconnu est celui dont le modèle donne le score maximum par l'algorithme de Viterbi. Par contre, le problème du décodage, en reconnaissance de mots enchaînés et de la parole continue, se ramène souvent à la recherche d'un chemin optimal dans un graphe. Deux types d'algorithmes sont en effet utilisés pour le décodage des mots enchaînés par les HMMs :

- Les algorithmes "construction par niveaux".
- Les algorithmes de Viterbi généralisés à des réseaux phonétiques décrivant les grammaires des suites de mots possibles.

Le premier type d'algorithme présente une faiblesse majeure en reconnaissance de la parole continue, à cause de la récursivité introduite sur le nombre d'unités de référence contenues dans la phrase inconnue. Par contre jusqu'à présent, le deuxième type d'algorithme est celui le plus souvent utilisé dans un système de décodage de la parole continue se basant sur des HMMs.

C'est avec ce dernier type que nous avons conçu notre premier système pour segmenter la parole continue. Les différents modèles des six classes phonétiques plus celui du silence, définies au paragraphe précédent, sont immergés dans un réseau plus important représentant la grammaire des suites de classes possibles. L'algorithme de Viterbi se généralise sur ce modèle phonétique [figure 4.1], et la suite des classes décodées est obtenue en gardant trace des transitions entre les modèles faisant partie du chemin optimal déterminé sur ce réseau.

4.3.2 VITERBI-réseau-3meilleurs

Pour avoir plus d'information sur les classes décodées, nous proposons un algorithme légèrement différent de VITERBI-réseau et donnant les trois meilleurs alignements. En combinant ces trois chemins, nous obtenons un treillis de traits phonétiques qui ne fera qu'augmenter le taux de reconnaissance. Rappelons qu'un point du plan de parcours de l'algorithme de Viterbi est déterminé par un instant t de la suite d'observations à reconnaître et un état j d'un modèle de référence. La modification apportée à l'algorithme pourra donc être résumée par le fait qu'au lieu de garder en un point du plan de parcours le score maximum et le chemin partiel correspondant, on gardera les trois scores maximum et les chemins partiels correspondants. Pour faciliter la compréhension de l'algorithme, nous exposons une version VITERBI-3meilleurs donnant les trois meilleurs chemins dans le cas isolé. Cette version se généralise facilement sur un réseau phonétique de la même façon que pour VITERBI-réseau, nous obtenons, ainsi, un algorithme VITERBI-réseau-3meilleurs pour le cas continu. L'algorithme VITERBI-3meilleurs sera résumé par les quatre étapes suivantes :

1. Initialisation pour $1 \leq i \leq N$

$$\delta_1^1(i) = \Pi_i b_i(O_1)$$
$$\delta_1^2(i) = \Pi_i b_i(O_1)$$
$$\delta_1^3(i) = \Pi_i b_i(O_1)$$

2. Récurrence pour $1 \leq t \leq T \quad et \quad 1 \leq j \leq N$

$$MAX1 = \max_{1 \leq i \leq N} [\delta_{t-1}^1(i) a_{ij}]$$

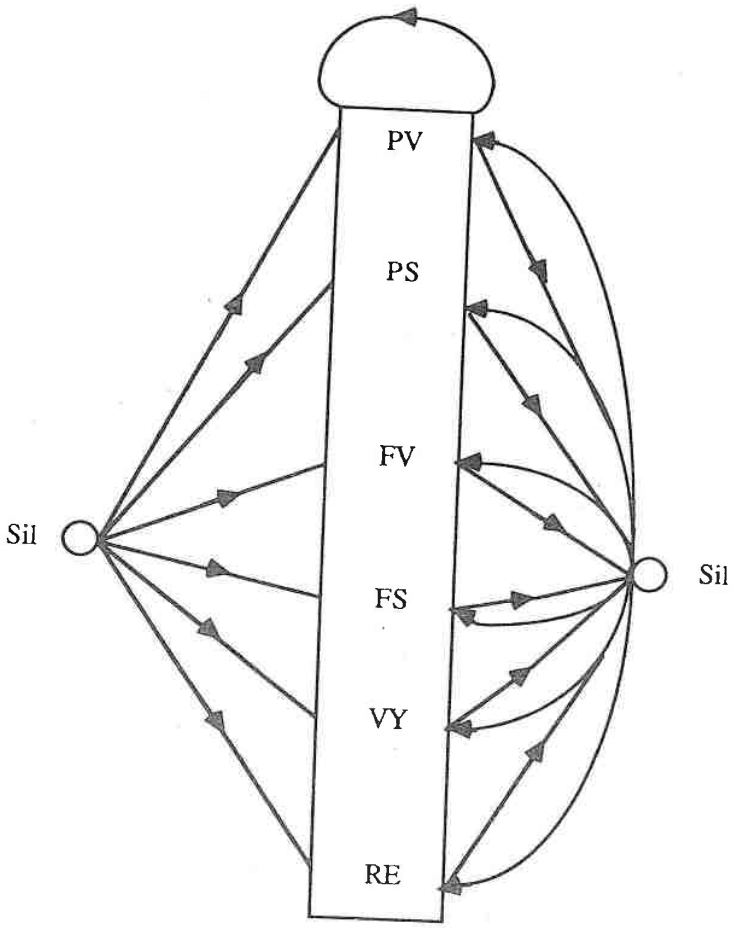


Figure 4.1. Grammaire des suites de classes phonétiques de longueur quelconque

$$MAX2 = \max_{1\leq i,k\leq N\ ,\ MAX1<MAX2} [\delta^1_{t-1}(i)a_{ij}\ \ ,\ \ \delta^2_{t-1}(k)a_{kj}]$$

$$MAX3 = \max_{1\leq i,k,l\leq N\ ,\ MAX2<MAX3} [\delta^1_{t-1}(i)a_{ij}\ \ ,\ \ \delta^2_{t-1}(k)a_{kj}\ \ ,\ \ \delta^3_{t-1}(l)a_{lj}]$$

$$\delta^1_t(j) = MAX1.b_j(O_t)$$
$$\delta^2_t(j) = MAX2.b_j(O_t)$$
$$\delta^3_t(j) = MAX3.b_j(O_t)$$

$$\psi^1_t(j) = \arg[MAX1]$$
$$\psi^2_t(j) = \arg[MAX2]$$
$$\psi^3_t(j) = \arg[MAX3]$$

3. Terminaison (on garde la trace des trois premiers scores)

$$P^*_1 = \max_{1\leq i\leq N} [\delta^1_T(i)]$$
$$P^*_2 = \max_{1\leq i\leq N} [\delta^2_T(i)]$$
$$P^*_3 = \max_{1\leq i\leq N} [\delta^3_T(i)]$$

$$q^*_{T,1} = \arg[P^*_1] \quad q^*_{T,2} = \arg[P^*_2] \quad q^*_{T,3} = \arg[P^*_3]$$

4. "backtracking" pour $T-1 \geq t \geq 1$

$$q^*_{t,1} = \psi^1_{t+1}(q^*_{t+1,1}) \quad q^*_{t,2} = \psi^2_{t+1}(q^*_{t+1,2}) \quad q^*_{t,3} = \psi^3_{t+1}(q^*_{t+1,3})$$

Le coût des opérations est multiplié à peu près par 3, de même que l'espace mémoire pour pouvoir mémoriser les trois premiers scores en chaque point, et leur alignement correspondant. Le nombre des comparaisons à effectuer fait augmenter encore plus le coût global.

En fait, nous avons développé un algorithme plus général que VITERBI-réseau-3meilleurs donnant les n meilleurs chemins. Nous avons choisi la valeur 3 comme compromis entre la complexité qui augmente, et l'amélioration des résultats. De plus les expériences ont permis de constater que le choix au delà de la valeur 3 ne changera presque pas les résultats.

Une comparaison des trois meilleurs chemins obtenus par cette version de l'algorithme de Viterbi, nous a permis de faire les deux remarques suivantes :

- Le meilleur chemin d'états de la grammaire obtenu, n'entraîne ni la meilleure segmentation, ni la meilleure concaténation des classes phonétiques.
- Une erreur du décodage ou de segmentation influe, généralement, sur le reste du chemin à décoder.

Ces deux remarques ont joué un rôle important sur nos orientations illustrées par la proposition de l'algorithme VITERBI-BLOC que nous allons présenter par la suite.

4.3.3 VITERBI-frontières

Nous supposons, dans ce cas, que les frontières des segments de la phrase à décoder sont connues. Il ne reste plus qu'à étiqueter ces segments. VITERBI-frontières est une version de l'algorithme de Viterbi qui, à partir de la suite d'observations de la phrase à décoder et des frontières connues, étiquetera la classe phonétique correspondante à chaque segment. C'est, en quelques sorte, à une initialisation près, la reconnaissance des classes isolées mises bout à bout.

Le développement de cette version a deux buts principaux :

1. Evaluer la qualité de l'apprentissage des modèles en contexte et juger la suffisance ou non du corpus d'apprentissage loin de tout problème de durée. Le taux de reconnaissance de cet algorithme est à rapprocher de celui de mots isolés.
2. Comparer et améliorer l'étiquetage du système APHODEX en partant de ses propres frontières de segments.

4.4 Modélisation de la durée

Bien qu'il présente différents avantages, tels que clarté, rigueur, efficacité et généralité, le modèle de base HMM est un modèle pauvre pour capturer et exprimer l'information concernant la variabilité de la durée des sons de la parole [Crystal 86]. Il favorise en effet les durées courtes, augmentant ainsi les probabilités d'insertion d'erreurs dans la reconnaissance de la parole. Ainsi, diverses méthodes ont été proposées pour améliorer la prise en compte de la durée [Rabiner 85,Rabiner 88, Cook 88,Russel 85,Russel 87,Russel 88,Bourlard 86,Levinson 86] ..etc.

4.4.1 Durée d'état explicite

Le point faible majeur d'un HMM classique est peut-être la modélisation de la durée d'état. Il est facile de démontrer que la densité de durée naturelle, $P_i(d)$, associée à l'état s_i est de la forme [Rabiner 88] :

$$P_i(d) = (a_{ii})^{d-1}(1 - a_{ii})$$

où a_{ii} est la probabilité de la transition boucle et $P_i(d)$ est la probabilité d'avoir d observations produites successivement à l'états s_i. Un simple calcul nous permet de remarquer la relation suivante :

$$P_i(d+1) = a_{ii}.P_i(d)$$

La durée la plus probable est la plus courte.

Pour la plupart des phénomènes physiques, cette densité géométrique de la durée d'état qui décroît exponentiellement est inadaptée. Par conséquent, on préfère souvent modéliser la densité de la durée d'état avec une densité de durée explicite. Cette idée de durée variable a été développée originellement par Ferguson [Ferguson 80b] pour modéliser le pitch dans la parole. La séquence des événements d'un HMM de densité de durée d'état variable, $P_i(d)$, sera alors :

1. Un état initial, $q_1 = s_i$, est choisi selon la distribution d'état initial Π_i;
2. Une durée d_1 est choisie selon la densité de durée d'état, $P_1(d_1)$;
3. Des observations $O_1O_2...O_{d_1}$ sont choisies selon la densité d'observations jointe $b_1(O_1O_2...O_{d_1})$. En général, on suppose l'indépendance des observations, ainsi :

$$b_1(O_1O_2...O_{d_1}) = \prod_{t=1}^{d_1} b_1(O_t)$$

4. L'état suivant, $q_2 = s_j$, est choisi selon les probabilités de transitions, $a_{q_1q_2}$, avec la contrainte de ne pas avoir de transitions boucles dans le modèle, ($a_{ii} = 0$, $1 \leq i \leq N$).

Cette dernière contrainte sur les transitions boucles est bien illustrée dans la figure 4.2 qui montre une représentation de deux états dans un modèle standard et sa transformation dans un modèle à durée d'état explicite. Plus clairement, la correspondance entre différents éléments intervenant dans cette modification est illustrée dans la table de la figure 4.3.
On suppose que l'état initial, q_1, commence à $t = 1$ et l'état final fini à $t = T$. Les intervalles d'observations de durée d'état sont tous inclus dans la suite d'observations $O_1O_2...O_T$. Les fonctions Forward et Backward sont, ainsi, modifiées comme suit :

$$\alpha_t(i) = P(O_1O_2...O_t, s_i \ \ fini \ \ à \ \ t/\lambda)$$

et

$$\beta_t(i) = P(O_{t+1}O_{t+2}...O_T/s_i \ \ fini \ \ à \ \ t, \lambda)$$

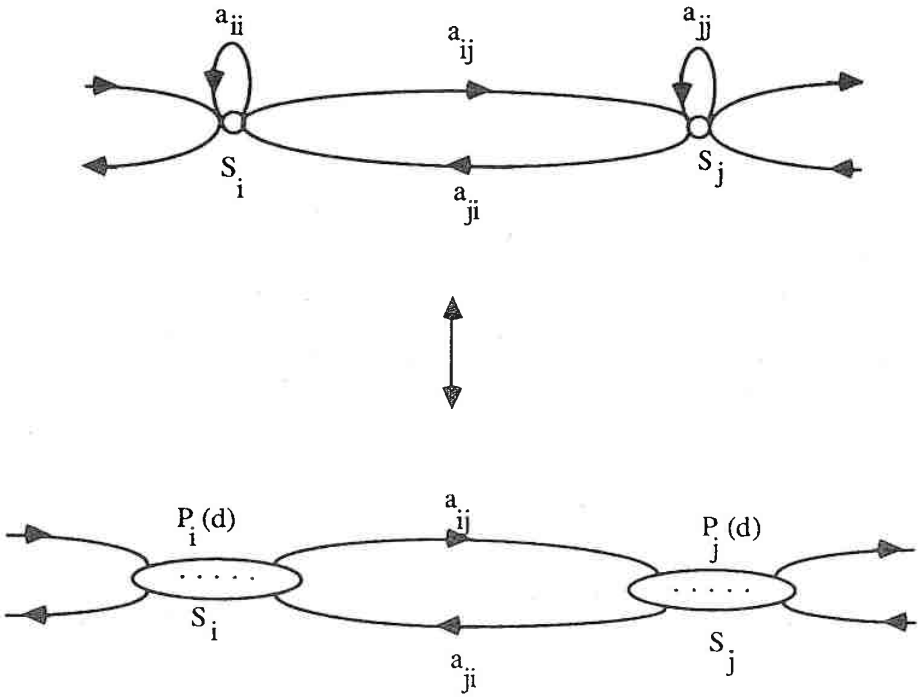


Figure 4.2. représentation d'états avec durée explicite

t	1	$d_1 + 1$	$d_1 + d_2 + 1$
état	q_1	q_2	q_3
durée	d_1	d_2	d_3
observations	$O_1 O_2 \ldots\ldots O_{d_1}$	$O_{d_1+1} \ldots O_{d_1+d_2}$	$O_{d_1+d_2+1} \ldots O_{d_1+d_2+d_3}$

Figure 4.3. correspondance entre différents éléments

Nous avons, plus précisément [Russel 87] :

$$\alpha_t(i) = P(O_1 O_2 ... O_t, q_t = s_i / q_{t+1} \neq s_i, \lambda)$$

et

$$\beta_t(i) = P(O_{t+1} O_{t+2} ... O_T, q_{t+1} \neq s_i / q_t = s_i, \lambda)$$

Ces fonctions Forward et Backward se calculent, aussi, de façon récursive grâce aux formules récurrentes suivantes [Russel 87,Codogno 87] :

$$\alpha_t(j) = \sum_{i=1}^{N} \sum_{d \leq t} \alpha_{t-d}(i) a_{ij} P_j(d) \prod_{\theta=1}^{d} b_j(O_{t-d+\theta})$$

et

$$\beta_t(i) = \sum_{j=1}^{N} \sum_{d \leq T-t} a_{ij} P_j(d) (\prod_{\theta=1}^{d} b_j(O_{t+\theta})) \beta_{t+d}(j)$$

Pour pouvoir définir les formules de ré-estimations avec la durée d'état explicite, nous allons définir d'abord, de façon similaire au précédent chapitre, les deux quantités probabilistes suivantes :

- La probabilité de séjourner en s_j le temps d en provenant de l'état s_i :

$$\xi_t(i,j,d) = P(q_t = s_i, q_{t+1} = s_j, q_{t+2} = s_j, ..., q_{t+d} = s_j, q_{t+d+1} \neq s_j / O, \lambda)$$

- La probabilité de séjourner en s_j le temps d :

$$\gamma_t(j,d) = P(q_t \neq s_j, q_{t+1} = s_j, q_{t+2} = s_j, ..., q_{t+d} = s_j, q_{t+d+1} \neq s_j / O, \lambda)$$

Nous déduisons la relation évidente :

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

Ces deux quantités probabilistes sont, ainsi, calculées à partir des deux fonctions récurrentes Forward et Backward, grâce à la relation importante suivante :

$$\xi_t(i,j,d) = \frac{\alpha_t(i) a_{ij} P_j(d) (\prod_{\theta=1}^{d} b_j(O_{t+\theta})) \beta_{t+d}(j)}{P(O/\lambda)}$$

Les formules de ré-estimations s'écrivent alors [Nakagawa 88,Poritz 88] :

$$\bar{\Pi}_i = \gamma_1(i,1)$$

$$\bar{a}_{ij} = \frac{\sum_t \sum_{d \leq t} \ \xi_t(i,j,d)}{\sum_t \sum_{d \leq t} \ \gamma_t(i,d)}$$

$$\bar{b}_j(k) = \frac{\sum_t \sum_{d \leq t} \sum_{\theta=1, O_{t-d+\theta}=v_k}^{d} \ \gamma_t(j,d)}{\sum_t \sum_{d \leq t} \sum_{\theta=1}^{d} \ \gamma_t(j,d)}$$

$$\bar{P}_j(d) = \frac{\sum_t \ \gamma_t(j,d)}{\sum_t \sum_{\tau \leq t} \ \gamma_t(j,\tau)}$$

Pour simplifier les calculs, on suppose souvent une limitation D au temps de séjour dans un état, c'est à dire $P_j(d)$ est calculée seulement pour $1 \leq d \leq D$. Avec cette restriction, nous avons illustré sur la figure 4.4 une transition du modèle HMM à durée d'état explicite et sa représentation équivalente dans le cas des HMMs standards [Nakagawa 88].

Dans le cas d'un HMM continu à durée d'état explicite, à la place de ré-estimer $b_j(k)$ par la formule précédente, les quantités c_{jk}, μ_{jk} *et* U_{jk} sont ré-estimées par des formules identiques à celles utilisées dans le cas du HMM continu standard, il suffit de remplacer $\gamma_t(j)$ par $[\sum_{d \leq t} \ \gamma_t(j,d)]$.

De la même façon, l'algorithme de décodage de Viterbi, est modifié. Sa quantité probabiliste, $\delta_t(i)$, est définie comme suit :

$$\delta_t(i) = \max_{q_1, q_2, \ldots, q_{t-1}} P(q_1 q_2 \ldots q_t = s_i, q_{t+1} \neq s_i, O_1 O_2 \ldots O_t / \lambda)$$

Cette quantité peut être calculée récursivement par [Levinson 87,Levinson 88] :

$$\delta_t(j) = \max_{1 \leq i \leq N} [\max_{d \leq t} [\delta_{t-d}(i) a_{ij} P_j(d) \prod_{\theta=1}^{d} b_j(O_{t-d+\theta})]]$$

pour $1 \leq j \leq N$ *et* $1 \leq t \leq T$.

Si nous mémorisons simultanément :

$$\psi_t(j) = (i^*, d^*) = \arg_{i,d} [\delta_t(j)],$$

nous pourrons tracer le cheminement arrière à partir de l'état final :

$$q_T^* = \arg \max_j [\delta_T(j)]$$

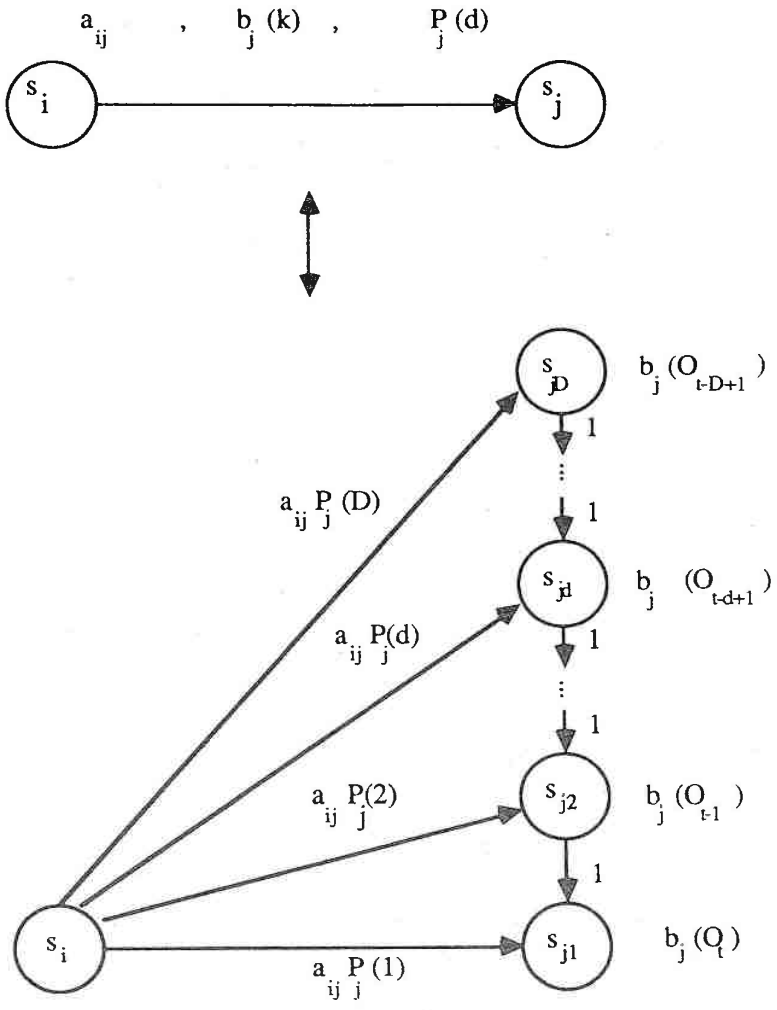


Figure 4.4. équivalent d'une transition avec durée explicite

pour reconstruire la séquence d'état optimal.

4.4.2 Inconvénients

Comme nous venons de le voir, Il est assez facile d'incorporer une densité de durée d'état explicite dans les algorithmes de décodage et d'apprentissage. Pour plusieurs problèmes d'application, la qualité de la modélisation est nettement améliorée quand ces densités de durée variable sont utilisées. Cependant, il y a des inconvénients non négligeables dans leur utilisation. Par exemple, dans le cas de séjour limité dans un état, la relation récurrente de l'algorithme de Viterbi devient :

$$\delta_t(j) = \max_{1\leq i\leq N} [\max_{1\leq d\leq D} [\delta_{t-d}(i)a_{ij}P_j(d)\prod_{\theta=1}^{d} b_j(O_{t-d+\theta})]]$$

ce qui entraîne à la fois que :

- le stockage est multiplié par, environ, D fois celui dans le cas des HMMs standards.
- les opérations de calcul sont multipliées par, environ, $\frac{D^2}{2}$ fois celles dans le cas des HMMs avec durée implicite (standard).

Il est raisonnable, dans plusieurs problèmes de traitement de la parole, de choisir par exemple D de l'ordre de 20, le calcul augmente de 200 fois environ.

Pour l'algorithme d'apprentissage, en plus de ces deux inconvénients, il y aura un grand nombre de paramètres, associés à chaque état, à estimer en plus des paramètres HMM usuels, et en général les données seront insuffisantes pour bien estimer la durée $P_j(d)$.

4.4.3 Alternatives

En général, pour alléger certains de ces problèmes, les applications utilisent des alternatives à la densité d'état explicite précédente :

- Une première méthode consiste à utiliser une densité de durée d'état paramétrique au lieu de non-paramétrique. Par exemple, l'utilisation d'une densité Gaussienne :

$$P_j(d) = N(d, \mu_j, \sigma_j^2).$$

La plupart des systèmes utilisent la densité de probabilité Gamma [Rabiner 88] :

$$P_j(d) = \frac{\eta_j^{\nu_j} d^{\nu_j - 1} exp(-\eta_j d)}{\Gamma(\nu_j)}.$$

Les formules de ré-estimations pour les paramètres η_j et ν_j ont été déduites et

utilisées avec de bons résultats [Codogno 87,Levinson 86,Levinson 87,Levinson 88]. D'autres travaux ont utilisé une durée d'état explicitée par une densité de probabilité de Poisson [Russel 85,Russel 87,Bourlard 86].

- Une deuxième solution pour obtenir une distribution de probabilité de la durée est de supposer une loi d'évolution temporelle convenable pour la probabilité de transition et de déduire alors la probabilité de la durée résultante. Falaschi a proposé une loi exponentielle, décroissante avec le temps d, de séjour dans un état j, pour la probabilité de la transition-boucle [Falaschi 88] :

$$a_{jj}(d) = exp(-\mu d)$$

La loi de probabilité de la durée résultante sera alors :

$$P_j(d) = (1 - exp(-\mu d)) \prod_{\theta=0}^{d-1} exp(-\mu\theta)$$

L'avantage de cette formulation est qu'une approximation de la relation récurrente de l'algorithme de Viterbi peut être utilisée sans augmentation de la complexité. La méthode consiste à ajouter à chaque état une variable (d_{0i} pour l'état s_i) qui contient le premier instant d'entrée dans l'état. Quand on calcule la probabilité du meilleur chemin terminant en s_j à l'instant t, l'état s_i^* qui maximise :

$$\delta_t(j) = \max_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}(t - d_{0i})].b_j(O_t)$$

est utilisé pour mettre à jour la variable d_{0j}, suivant qu'on est déjà dans un état ou on vient de l'être. C'est à dire :

$$\begin{array}{rcll} d_{0j} & = & d_{0j} & si \quad s_j = s_i^* \\ & & t & si \quad s_j \neq s_i^* \end{array}$$

avec la probabilité de transition, $a_{ij}(t)$, définie comme suit :

$$\begin{array}{rcll} a_{ij}(t) & = & exp(-\mu_i t) & si \quad s_i = s_j \\ & & a_{ij}(1 - exp(-\mu_i t)) & si \quad s_i \neq s_j \end{array}$$

Les HMMs utilisant cette approche sont des HMMs non stationnaires, puisque la stationnarité des chaînes de Markov n'est plus respectée (a_{ij} dépend du temps t). Ils peuvent aussi être appelés, comme les HMMs utilisant une densité de durée paramètrique ou non, modèles semi-markoviens (HSMM), parce que

les probabilités de transitions ne dépendent pas uniquement du présent état, mais aussi du temps de séjour écoulé dans cet état. Nous trouvons aussi, dans la littérature, les notations CVDHMM et VDHMM pour parler des modèles markoviens cachés de durée variable continus et discrets, respectivement.

- Une troisième solution pour incorporer de l'information sur la durée d'état est réalisée par un post-processeur [Rabiner 88,Rabiner 89]. En effet, la probabilité de la durée d'état, $P_j(d)$, a été mesurée directement à partir des séquences de parole d'apprentissage segmentées. Ainsi, l'estimation de $P_j(d)$ est strictement heuristique (on utilise souvent une procédure d'alignement temporel de Viterbi). La mise en oeuvre de ces densités de durée heuristiques se fait en trois étapes :
  1. L'algorithme de Viterbi normal est utilisé pour donner, avec son "back-tracking", la meilleure segmentation, en suite d'états, de la séquence d'observations inconnue;
  2. La durée d_j, de chaque état j, est alors mesurée à partir de cet alignement temporel;
  3. Un post-processeur augmente le score logarithmique de Viterbi de la façon suivante :

$$\log P'(Q, O/\lambda) = \log P(Q, O/\lambda) + \alpha_d \sum_{j=1}^{N} \log[P_j(d_j)]$$

  où α_d est une échelle empirique multiplicative sur les durées d'états. La reconnaissance se fera alors à partir de ces nouveaux scores.

  Le coût de calcul du post-processeur est négligeable devant le coût général de la reconnaissance. Cependant, il apporte des performances significatives [Rabiner 88].

- Une quatrième solution pour représenter la durée, plus efficacement que dans un HMM standard, consiste en l'utilisation des modèles à états étendus, en identifiant chaque état dans un HMM standard, avec un sous-HMM ayant une fonction de densité de probabilité d'observations unique [Russel 87], ou en remplaçant chaque état-boucle (ayant une transition boucle non nulle) par un nombre de reproductions de lui-même [Codogno 87].

  L'avantage de cette approche est que l'algorithme de Viterbi n'est modifié en aucun point. L'apprentissage se fait en deux étapes : Un premier apprentissage, en général limité à un sous corpus de parole, avec les HMMs standards, suivi d'un alignement temporel avec une procédure de Viterbi, par exemple. Il est ainsi possible de compter la moyenne et la variance de séjour à chaque état-boucle [Russel 87,Codogno 87], et donc de déterminer un facteur de reproduction pour chacun des états [Codogno 87]. Un deuxième apprentissage complet se fera, ensuite, avec les HMMs à états étendus. Dans le cas des sous-HMMs ayant une fonction de densité de probabilité d'observations unique dans chaque identification, Les formules de ré-estimations dans l'algorithme d'apprentissage sont légèrement modifiées en prenant en compte :

$$\sum_{i=\phi_j}^{\phi_{j+1}-1} \gamma_t(i)$$

au lieu de :

$$\gamma_t(j)$$

et où ϕ_j est l'indice du premier état du sous-HMM remplaçant le $j^{ème}$ état standard.

Malgré la simplicité de cette approche, les résultats sont, parfois, plus intéressants qu'avec une durée explicite paramétrique [Russel 87,Codogno 87].

4.4.4 Durée d'unité de reconnaissance

Il y a une deuxième forme d'information sur la durée, qui peut être utilisée en alternative ou simultanément à la durée d'état explicite. Elle consiste à incorporer de l'information sur la durée de l'unité de reconnaissance globale, c'est à dire, à modéliser le temps de séjour dans un modèle HMM global sans s'occuper des différents états du modèle. Rabiner a utilisé une telle notion pour la reconnaissance de mots enchaînés, en considérant un modèle de durée gaussien pour chaque mot du vocabulaire. La moyenne et la variance, de chaque mot du vocabulaire, ont été estimées à partir des données d'apprentissage. Cette densité de durée a été introduite dans les scores de reconnaissance après chaque niveau de l'algorithme de décodage construction par niveaux ("level building") utilisé dans [Rabiner 85,Rabiner 88].
En ce qui concerne notre travail sur la reconnaissance phonétique, nous avons repris cette notion de durée d'unité avec une nouvelle stratégie d'utilisation de l'algorithme de Viterbi, basée sur des critères de comparaison des différents scores de Viterbi obtenus par blocs. Le paragraphe suivant présente cet algorithme à optimalité locale, appelé VITERBI-BLOC. Les résultats expérimentaux montrent l'efficacité d'une telle approche.

4.5 L'algorithme VITERBI-BLOC

4.5.1 Description de l'algorithme

Ce paragraphe décrit un nouvel algorithme adapté à la reconnaissance de la parole continue puisqu'il prend en compte la durée d'un segment [Kriouile 90b, Kriouile 90a]. Cette version appelée VITERBI-BLOC permet une segmentation et un étiquetage de la phrase inconnue au fur et à mesure de son exécution alors que l'algorithme de Viterbi attend la fin de la phrase pour revenir en arrière sur le meilleur chemin.
D'un point de vue probabiliste, il est plus facile de décoder, sans erreurs, un segment court qu'un segment long. Cet algorithme apporte une solution à ce problème de durée de segment en donnant des chances de décodage équiprobables à des segments de longueurs différentes.
L'identification d'un segment pendant le décodage peut être confirmée ou infirmée par d'autres processeurs ou sources de connaissance grâce à la stratégie locale de cet algorithme.

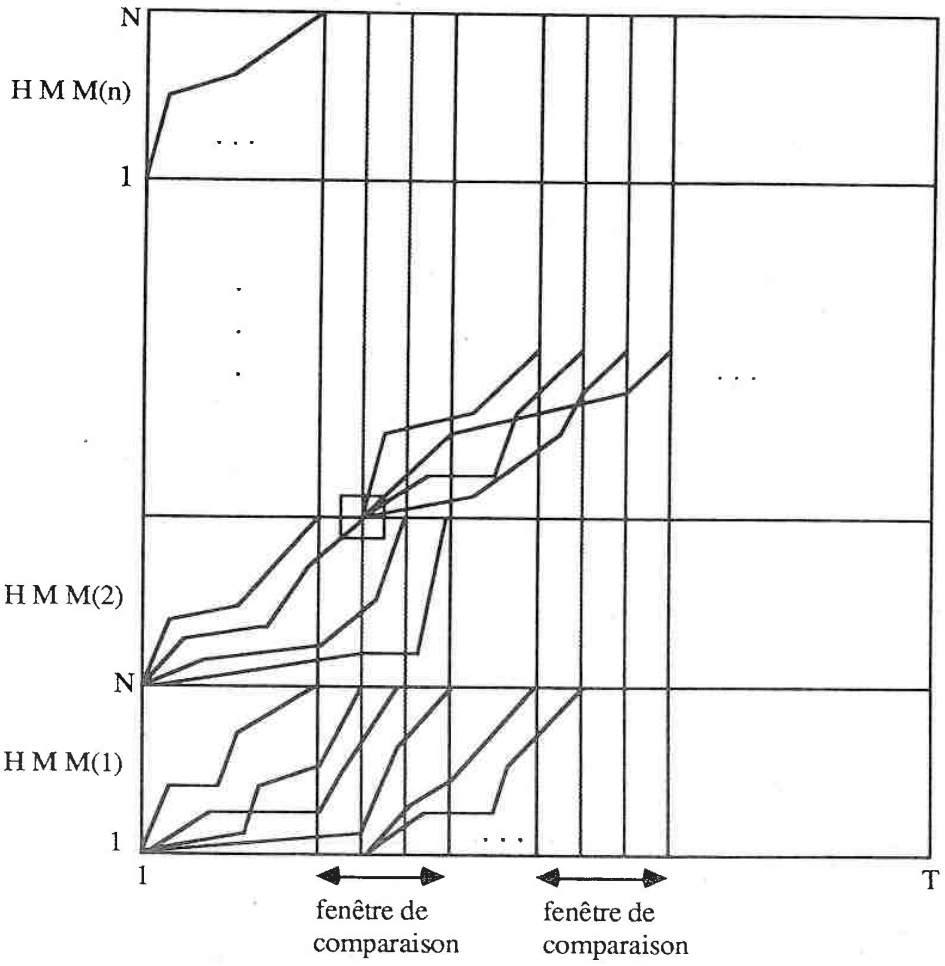


Figure 4.5. L'algorithme VITERBI-BLOC

La reconnaissance par l'algorithme VITERBI-BLOC est réalisée par un processus itératif. Le principe est basé sur une procédure de comparaison des probabilités calculées par l'algorithme de Viterbi. Ces probabilités sont celles correspondant aux états finaux des différents modèles HMMs dans une fenêtre de trames de largeur fixe et à distance fixe de la trame de départ de l'algorithme [figure 4.5]. Cette distance ainsi que la largeur de la fenêtre, sont définies par apprentissage. En effet, les occurrences d'apprentissage permettent d'estimer les longueurs des segments possibles de chaque classe.

Ce processus de reconnaissance se décompose de la façon suivante :

1. trame courante ⟵ trame de départ
2. Exécution de l'algorithme de Viterbi pour chaque modèle à partir de la trame courante
3. Mémorisation des probabilités obtenues dans la fenêtre de comparaison pour chaque modèle
4. Comparaison de ces probabilités et choix du segment correspondant au modèle favorable
5. trame courante ⟵ trame juste après le segment choisi
6. Recommencer les étapes 2, 3, 4 et 5 jusqu'à la fin de la suite d'observations à décoder.

4.5.2 Critères de comparaison

La procédure de comparaison (étape 4 du processus) des probabilités calculées par l'algorithme de Viterbi des différents modèles est le noyau de l'algorithme VITERBI-BLOC. C'est, en effet, le choix du critère de comparaison qui conditionnera l'efficacité de l'algorithme. Nous allons présenter deux critères qui ont servi de test pour l'algorithme.

Critère MAXDIFF

Le premier critère de comparaison, appelé MAXDIFF, est basé sur trois étapes :

1. Recherche du meilleur et du second modèle au niveau de chaque trame de la fenêtre de comparaison.
2. Calcul de la différence entre leur scores.
3. Comparaison de ces différences obtenues aux différentes trames de la fenêtre de comparaison. La plus grande différence permettra de choisir le modèle favorable et la trame finale du segment correspondant.

Ce critère permet de trouver le modèle qui se dègage le plus du lot des modèles se terminant dans la fenêtre de comparaison, il permet d'améliorer les résultats de l'algorithme de reconnaissance (paragraphe 5.6). Il n'utilise que des traits numériques tels que la comparaison et la différence des valeurs numériques : le modèle choisi est celui ayant le score de Viterbi le plus grand dans une trame et le plus éloigné des autres scores de différents modéles à cette même trame.

Critère MAXDUREE

Ce deuxième critère, basé sur l'introduction de la durée globale des segments à reconnaître, est appelé MAXDUREE. Ce critère, un peu plus compliqué que MAXDIFF, nous amène à modifier légèrement l'algorithme d'apprentissage de telle façon qu'au lieu d'obtenir un modèle $\lambda = (\pi, A, B)$ nous obtiendrons $\lambda = (\pi, A, B, C_\lambda)$, où C_λ est un vecteur durée déterminant une distribution de probabilité sur les différentes durée de la classe phonétique correspondante au modèle λ. L'ensemble des C_λ forme ainsi une matrice C où $C[i,t]$ est la probabilité d'avoir la longueur t pour le modèle i.

L'idée de base de ce critère est la suivante : pour comparer un score donné par l'algorithme de Viterbi à une trame t donnée avec celui de la trame suivante, ce score doit être multiplié par un certain coefficient probabiliste pour que la comparaison ait un sens. Parce que le passage d'une trame à la suivante dans le calcul de l'algorithme de Viterbi multiplie le score par deux valeurs de probabilité celles de transition et d'observation. Ce coefficient est une puissance d'ordre α de la probabilité $C[i,t]$ où i est le modèle correspondant à ce score. L'exposant α est une constante déterminée empiriquement pour une normalisation en durée des probabilités. D'ailleurs, l'expérience montre qu'avec $\alpha = 2$ nous obtenons les meilleurs résultats, ceci peut être expliqué intuitivement par le fait qu'en passant à la trame suivante le score de Viterbi se multiplie par deux nouvelles valeurs de probabilités. Mais, on ne peut pas affirmer ce que nous venons de dire parce que nous n'avons aucune idée sur l'ordre de ces valeurs.

Ce critère MAXDUREE est décrit par les étapes suivantes :

1. trame courante ⟵ trame de début de la fenêtre de comparaison
   MAXDUREE ⟵ 0
2. Recherche de la probabilité maximale entre celles obtenues pour les différents modèles et MAXDUREE, au niveau de la trame courante
3. MAXDUREE deviendra cette valeur maximale multipliée par un coefficient. Ce coefficient est une puissance d'ordre α de la probabilité $C[i,t]$ où i est le modèle correspondant à cette valeur maximale. L'exposant α est une constante déterminée empiriquement.
4. trame courante ⟵ trame suivante
5. Recommencer les étapes 2, 3, et 4 jusqu'à la trame de la fin de la fenêtre de comparaison
6. Le modèle et la trame correspondants à l'origine de la valeur gardée dans MAXDUREE, seront, respectivement, le modèle favorable choisi et la trame finale du segment qui lui correspond.

Avec les notations suivantes :

Tmin : trame de début de la fenêtre de comparaison.
Tmax : trame de la fin de la fenêtre de comparaison.
score(i,t) : score de l'algorithme de Viterbi à la trame t pour le modèle i.
N : nombre de modèles HMMs.

ce critère MAXDUREE peut être explicité de la façon suivante :

1. Initialisation :

$$MAXDUREE(Tmin) = \max_{1 \leq i \leq N} score(i, Tmin)$$

$$\arg MAXDUREE(Tmin) = \arg_i \max$$

$$trame = Tmin$$

2. Induction :

$$MAXDUREE(t+1) = \max_{1 \leq i \leq N,\ \arg MAXDUREE(t)} \Big\{ MAXDUREE(t) * C[\arg MAXDUREE(t), trame]^{\alpha}; score_{1 \leq i \leq N}(i, t+1) \Big\}$$

$$\arg MAXDUREE(t+1) = \arg_{i, \arg MAXDUREE(t)} \max$$

$trame =$ *si* $\arg MAXDUREE(t+1) \neq \arg MAXDUREE(t)$

alors t+1

3. Terminaison

$$modèle^* = \arg MAXDUREE(Tmax)$$

$$trame^* = trame$$

4.6 Expérimentation

4.6.1 Conditions expérimentales

Pour mesurer et comparer les performances de nos systèmes, nous avons utilisé le corpus, acquis au CRIN, de 57 phrases phonétiquement équilibrées de Combescure [Combescure 81,Fohr 86] prononcées à un rythme naturel d'élocution par cinq locuteurs masculins non professionnels. Les phrases étaient prononcées par un locuteur expérimenté et répétées par les locuteurs (mémoire immédiate). Ce mode opératoire était adopté pour éviter un rythme d'élocution trop lent et une intonation artificielle de type lecture. De ce fait, la prosodie et le rythme des locuteurs étaient homogénéisés (environ 14 phonèmes par seconde). Ces phrases ont été numérisées à une fréquence de 12 kHz sur 10 bits.

Segmentation manuelle

Le savoir-faire d'un expert phonéticien a permis de placer manuellement, à partir des spectrogrammes numériques et de la courbe de l'énergie globale, les marques de segmentation sur le corpus numérisé. Les taux de reconnaissance que nous donnerons par la suite sont tous fondés sur cette segmentation manuelle.

Paramétrisation du corpus

Dans une fenêtre de Hamming de 20 ms de durée, on calcule 12 coefficients cepstraux en tenant compte d'une échelle MEL. Cette fenêtre est déplacée de 10 ms afin de permettre un recouvrement des intervalles d'analyse. Chaque phrase est représentée par une matrice réelle (12 fois le nombre de trames d'analyse). Le nombre de trames dépend de la longueur du signal.

Classification

Chaque trame d'analyse est un point dans un espace vectoriel de dimension 11 normé par la distance euclidienne. En effet, la première dimension, fortement corrélée avec l'énergie du signal, est ignorée. L'algorithme de classification utilisé est celui initialement proposé par Buzo et Gray [Buzo 80,Gray 84]. Il effectue une partition géographique d'un nuage de points en deux en minimisant la distorsion moyenne de l'ensemble. On définit la distorsion d'un point par la plus petite distance de ce point aux centres de gravité des nuages. Chaque nuage de points est représenté par son centre de gravité. Cette famille de représentants (qu'on appelle aussi : classes, symboles ou prototypes) forme un dictionnaire (codebook) de prototypes. Nous avons utilisé dans nos expériences deux dictionnaires de 64 prototypes chacun. Les deux dictionnaires DIC5 et DIC12 sont respectivement obtenus après classification de 5 et 12 phrases.

Quantification vectorielle

Ce processus classe une trame d'analyse parmi l'ensemble des prototypes. Ce classement se fait en calculant la distance euclidienne de la trame aux différents prototypes et en choisissant celui le plus proche. Nous quantifions ainsi les 57 phrases paramétrées. Chaque phrase est représentée par une suite de prototypes éléments de DIC5 ou de DIC12 suivant que l'on utilise l'un ou l'autre.

Application

Pour estimer un HMM par classe phonétique (7 modèles :silence, PV, PS, FV, FS, VY et RE) nous avons utilisé 40 phrases du corpus quantifiées et leur segmentation manuelle pour l'apprentissage (8 phrases par locuteur). Les 17 autres ont servi pour tester les différentes versions de l'algorithme de reconnaissance proposées. Chaque phrase reconnue est comparée à sa segmentation manuelle. Les différentes étapes de ces expériences sont résumées dans le schéma de la figure 4.6.

4.6.2 Résultats expérimentaux

Plusieurs expériences ont été menées dans le but de déterminer le plus convenablement possible :

* Le type du modèle initial (test avec 5, 7 et 9 états).
* Le nombre des itérations de l'algorithme d'apprentissage sur les occurences de chaque classe, issues des 40 phrases (test avec 2, 3, 4 et 5 itérations).

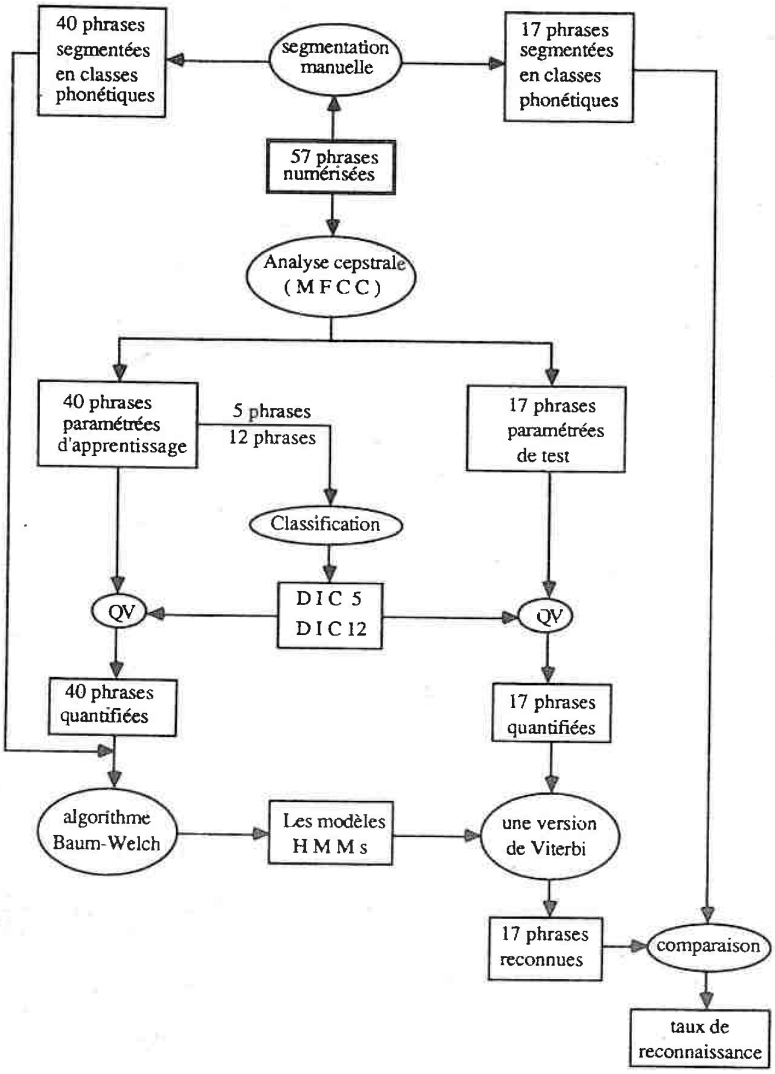


Figure 4.6. Schéma expérimental

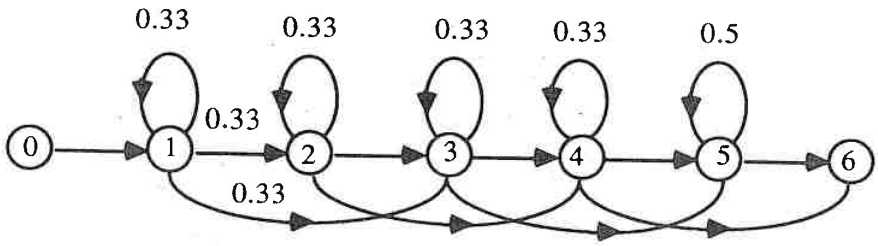


Figure 4.7. Modèle initial choisi

* La constante α, l'exposant utilisé dans MAXDUREE (test avec 1, 1.5, 2, 2.5 et 3) .

En moyenne, les meilleurs résultats sont ceux fournis par :

* le modèle initial illustré dans la figure [figure 4.7],
* 3 itérations,
* et α égal à 2.

Ce sont ces choix que nous avons utilisés pour les résultats exposés par la suite. Mais insistons, d'abord, sur le fait que nous avons constaté que plus le nombre d'états est grand et le nombre de transitions-boucles réduit, plus le deuxième et le troisième chemin fournis par l'algorithme VITERBI-réseau-3meilleurs, contiennent les frontières de segmentation correctes.

Matrice durée

Pour le critère de comparaison MAXDUREE utilisé par l'algorithme VITERBI-BLOC, nous avons déterminé pour nos expériences une matrice durée C formée de distributions de probabilités discrètes. Ces distributions sont calculées à partir des échantillons de longueurs des classes phonétiques. Ces échantillons sont obtenus à partir des occurrences de chaque classe se trouvant dans les 40 phrases d'apprentissage. Pour donner une possibilité d'avoir des longueurs de classe non existantes dans le corpus d'apprentissage, nous leur avons attribué une probabilité de petite valeur (10^{-4} dans nos expériences).

Exemple de phrase reconnue

Le tableau de la figure 4.8 illustre un exemple de phrase test reconnue par les différentes versions de l'algorithme de reconnaissance. Il présente aussi les résultats de VITERBI-frontières appliqué aux frontières correctes (colonne 4) et aux frontières des segments donnés par la segmentation du système expert APHODEX (colonne 7). Le spectrogramme de cette phrase test ("confie-moi à quoi tu penses") est illustré à la figure 4.9. A partir du tableau de la figure 4.8, nous remarquons, par exemple, qu'avec APHODEX deux classes sont mal reconnues ($(50,57)$ $VY \longrightarrow FS$, $(119,125)$ $RE \longrightarrow VY$), alors qu'elles sont bien reconnues avec VITERBI-BLOC-MAXDIFF et VITERBI-BLOC-MAXDUREE. Dans l'autre sens de complémentarité, les quatres versions de Viterbi ont mal reconnues deux classes ($(107,118)$ $PS \longrightarrow PV$, $(147,157)$ $VY \longrightarrow PS$) qu'APHODEX a bien reconnues. D'autres types de complementarité dans la reconnaissance de cette phrase peuvent se déduire facilement de la même table. Ces résultats montrent l'importance d'intégrer les deux approches dans une structure unique afin d'améliorer les systèmes de décodage acoustico-phonétique [Haton 87].

Résultats du corpus test

Les taux de reconnaissance de la figure 4.10 et les matrices de confusion des figures 4.11, 4.12, 4.13 illustrent les résultats de la reconnaissance des phrases test par les différentes versions, de l'algorithme de Viterbi, proposées. Les mêmes expériences ont été faites par les deux dictionnaires DIC5 et DIC12. Les résultats avec DIC12 sont légèrement améliorés qu'avec DIC5. Par conséquent, nous n'avons présenté que les matrices de confusion correspondant à l'utilisation de DIC12. Ces matrices de confusion montrent que les taux de reconnaissance sont fortement affaiblis par celui de la classe RE. Elle est dans la plupart du temps reconnue comme une classe VY. Les taux de reconnaissance des autres classes sont de même ordre grandeur dans les 4 versions VITERBI-réseau, VITERBI-réseau-3meilleurs, VITERBI-BLOC-MAXDIFF et VITERBI-BLOC-MAXDUREE. Pour les six classes autres que RE, nous remarquons que la classe PS, la moins reconnue avec VITERBI-réseau et VITERBI-réseau-3meilleurs, est bien reconnue avec VITERBI-BLOC-MAXDIFF et surtout avec VITERBI-BLOC-MAXDUREE. En même temps, la classe PV, la moins reconnue avec les versions VITERBI-BLOC, est bien reconnue avec VITERBI-réseau et VITERBI-réseau-3meilleurs. Cette sorte de complémentarité entre les résultats fournis par ces 4 versions peut être utilisée efficacement dans le but d'améliorer le taux de reconnaissance.

Comme nous l'avons déjà dit, l'algorithme VITERBI-frontières est un bon test pour savoir si le corpus d'apprentissage est suffisant. Son taux de reconnaissance (68%) montre que notre corpus de 40 phrases ne permet pas un très bon apprentissage. Ceci s'explique par le nombre d'occurrences de chaque classe que nous avons dans ce corpus :

* 79 occurrences silence
* 68 occurrences PV

	frontières correctes		correc. seg.	Viter. front.	Aphodex			Apho. front.	Viter.	Viterbi - réseau			Viterbi - réseau - 3meilleurs			Viterbi - bloc - maxdiff			Viterbi-bloc-maxdurée		
k	1	15	ps	ps	1	16	ps	ps		2	18	sil	2	18	sil	2	18	sil	2	18	sil
on	16	28	vy	vy	16	28	vy	vy		18	24	vy	18	24	vy	20	26	re	20	28	re
																28	32	ps			
f	29	49	fs	ps	32					26	52	ps	28	50	ps	34	50	fs	30	52	fs
i	50	57	vy	vy		58	fs	vy		54			52	58	vy	52	60	vy	54	58	vy
m	58	70	re	re	60								60			62	74	re	60	74	re
w	71	82	re	vy		78	re	re			80	re		84	re	76	92	vy	76	80	vy
a	83	106	vy	vy	80	110	vy	vy		82	104	vy	84	104	vy	94	104	vy	82	106	vy
k	107	118	ps	pv	110	120	ps	pv		106	118	pv	106	118	pv	106	114	pv	108	118	pv
w	119	125	re	re	120					120			120			116	122	re	120	126	re
a	126	133	vy	vy		132	vy	vy			130	vy		132	vy	124	132	vy	128	132	vy
t	134	146	ps	ps	136	142	ps	ps		132			134			134	148	ps	134	150	ps
					142	148	re	ps													
y	147	157	vy	ps	150	160	vy	ps			152	ps		152	ps	150	160	ps	152	162	ps
p	158	172	ps	ps	162	170	ps	ps		154	174	ps	154	174	ps	162	170	ps	164	174	ps
an	173	207	vy	vy	172	200	vy	vy		176	190	vy	176	192	vy	172	188	vy	176	214	vy
					202	208	re	vy		192			194			190	220	re			
s	208	241	fs	fs	210	242	fs	fs			248	ps		248	ps	222	238	fs	216	240	fs
				73%								53%			60%			67%			67%

Figure 4.8. Reconnaissance de la phrase "Confie-moi à quoi tu penses"

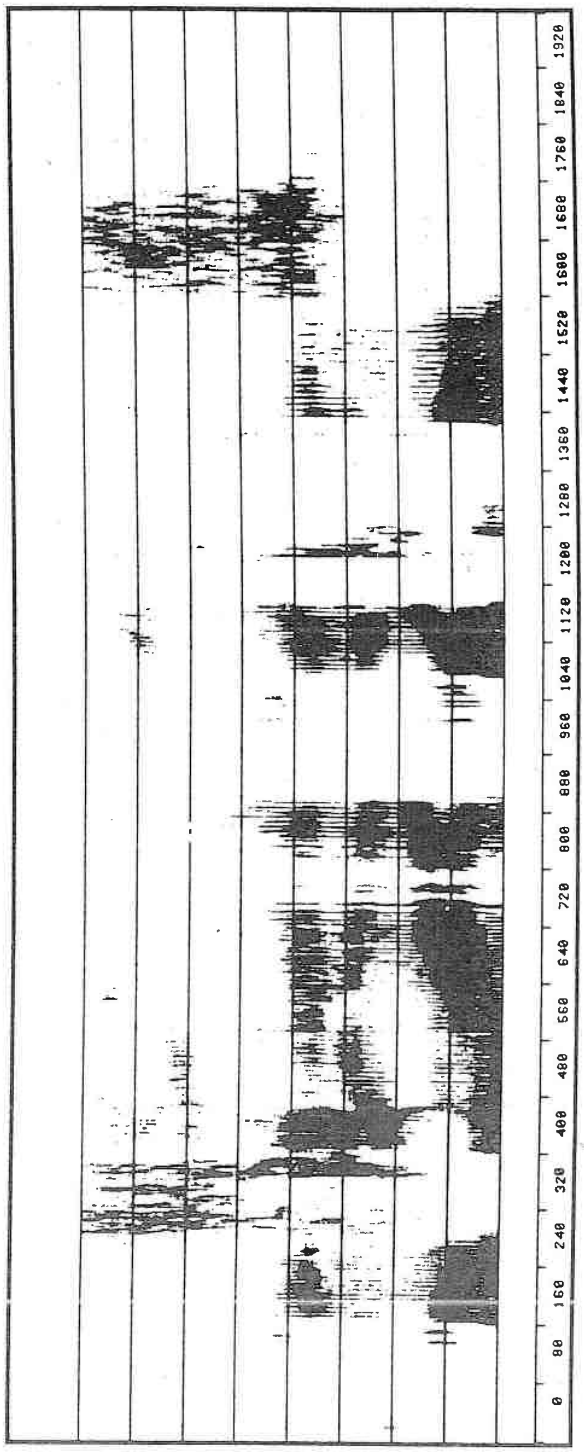


Figure 4.9. Spectrogramme de la phrase "Confie-moi à quoi tu penses"

Version de l'algorithme de Viterbi	DIC 5 (%)	DIC 12 (%)
VITERBI-réseau	59	60
VITERBI-réseau-3meilleurs	60	60
VITERBI-BLOC-MAXDIFF	62	62
VITERBI-BLOC-MAXDUREE	63	64
VITERBI-Frontières	67	68

Figure 4.10. Taux de reconnaissance du corpus test

* 121 occurrences PS
* 34 occurrences FV
* 78 occurrences FS
* 413 occurrences VY
* 229 occurrences RE.

Alors que les différentes expérimentations, que nous trouvons dans la littérature, laissent penser qu'un minimum de 400 occurrences de chaque classe est nécessaire pour un bon apprentissage. Cependant, la comparaison des taux de reconnaissance est significative. Les algorithmes VITERBI-BLOC-MAXDIFF et VITERBI-BLOC-MAXDUREE permettent un gain de 3% à 4% par rapport au taux de reconnaissance fourni par VITERBI-réseau. Ce qui est une amélioration très intéressante, surtout que l'amélioration idéale donnée par VITERBI-frontières est de 8%.

Sans passer par l'étape de segmentation, une application directe de ces systèmes pour reconnaître les phonèmes est tout à fait envisageable. D'ailleurs, nous les avons testés avec le même corpus segmenté en 15 classes phonétiques plus fines. Nous constatons que les résultats se dégradent, c'est vraisemblablement à cause de la diminution considérable du nombre d'occurrences servant à apprendre le modèle HMM d'une classe donnée.

	sil	pv	ps	fv	fs	vy	re
sil	78		15			4	
pv		68	5	5		10	10
ps		6	82	5	6		
fv			9	70	19		
fs			7	20	71		
vy	1	7	1	3		82	5
re		13	10		2	48	26

(68.14 %)

Figure 4.11. Matrice de confusion de VITERBI-frontières

	sil	pv	ps	fv	fs	vy	re
sil	67		19			11	
pv		68	5	5		14	6
ps	11	11	62	6	8		
fv			5	66	28		
fs	7		11	14	66		
vy	4	6	3	6	3	71	5
re	6	12	7	3	4	49	18

(59.71 %)

	sil	pv	ps	fv	fs	vy	re
sil	69		16		1	10	1
pv		68	5	5		14	6
ps	11	11	64	5	8		
fv			5	66	28		
fs	7		11	14	66		
vy	3	6	3	6	3	72	5
re	6	12	7	3	4	49	18

(60.42 %)

Figure 4.12. Matrices de confusion de VITERBI-réseau et VITERBI-réseau-3meilleurs

	sil	pv	ps	fv	fs	vy	re
sil	72		16		1	6	2
pv		66	6	7		12	7
ps	9	12	69	4	3		
fv	4		10	67	17		
fs	2		7	19	70		
vy	6	6	5	6	2	69	4
re		13	11	2	3	48	22

(62.14 %)

	sil	pv	ps	fv	fs	vy	re
sil	72		16		1	6	2
pv		65	6	7	1	12	6
ps	4	6	78	5	6		
fv	4		10	67	17		
fs	2		6	19	71		
vy	6	6	6	5	2	69	4
re		13	11	2	2	48	24

(63.71 %)

Figure 4.13. Matrices de confusion de VITERBI-BLOC-MAXDIFF et VITERBI-BLOC-MAXDUREE

	PV	PS	FV	FS	VY	RE
T_1	54.5	56	67	70	60	18
T_2	10.5	18	23.5	17	28	42.5
T_3	16	19.5	4	3.5	9	4
$T_1 + T_2$	65	74	90.5	87	88	60.5
$T_1 + T_3$	70.5	75.5	71	73.5	69	22
$T_1 + T_2 + T_3$	81	93.5	94.5	90.5	97	64.5

Figure 4.14. Tableau de comparaison d'APHODEX et de VITERBI-BLOC-MAXDUREE

4.6.3 Comparaison avec les prétraitements d'APHODEX

Pour des raisons techniques, nous n'avons fait la comparaison avec les prétraitements d'APHODEX qu'avec 8 phrases test au lieu de 17 phrases. D'autre part, la version actuelle d'APHODEX ne prévoit que quatre classes dans les prétraitements : les plosives, les fricatives, les voyelles et les inconnus. Par conséquent une comparaison parfaite n'est pas envisageable, nous avons ainsi considéré que si les plosives voisées ou sourdes sont reconnues comme des plosives alors elles sont bien reconnues. De même, pour les fricatives voisées ou sourdes si elles sont reconnues comme des fricatives.

Pour dresser un tabeau de comparaison [figure 4.14] entre les résultats fournis par VITERBI-BLOC-MAXDUREE et par les prétraitements d'APHODEX, nous avons utilisé les notations suivantes :

T_1 : Taux de reconnaissance des classes reconnues par les deux systèmes.

T_2 : Taux de reconnaissance des classes reconnues par APHODEX et non par VITERBI-BLOC-MAXDUREE.

T_3 : Taux de reconnaissance des classes reconnues par VITERBI-BLOC-MAXDUREE et non par APHODEX.

$T_1 + T_2$: Taux de reconnaissance des classes reconnues par le système APHODEX.

$T_1 + T_3$: Taux de reconnaissance des classes reconnues par VITERBI-BLOC-MAXDUREE .

$T_1+T_2+T_3$: Taux de reconnaissance d'une intéraction parfaite entre les deux systémes.

Les résultats de ce tableau 4.14 montrent une large complémentarité entre les deux approches. L'étude d'une stratégie d'interaction des deux systèmes s'impose alors.

4.7 Conclusion

Pour conclure ce chapitre, insistons sur le fait qu'un bon apprentissage et un bon critère de comparaison feront de VITERBI-BLOC un très bon algorithme de reconnaissance de la parole continue. Nous pensons aussi qu'au lieu d'utiliser une matrice durée discrète C, des distributions de probabilités continues (Gaussiennes par exemple) améliorerons les résultats de l'algorithme VITERBI-BLOC-MAXDUREE.

L'algorithme VITERBI-BLOC est un algorithme ouvert à l'intervention d'autres processeurs pour améliorer l'identification des segments pendant le décodage. C'est une voie de recherche qui semble très intéressante.

Conclusion

Des travaux intensifs sur la reconnaissance automatique de la parole utilisant les modèles stochastiques ont été réalisés durant les cinq dernières années. L'application des modèles markoviens cachés du premier ordre a conduit à des résultats impréssionnants dans le domaine de la reconnaissance de mots isolés, de mots enchaînés et de la parole continue.

L'objectif de ce travail est de montrer que l'apport des modèles markoviens cachés à la reconnaissance automatique de la parole est d'autant plus important qu'on mène des réflexions fondamentales sur les modèles markoviens eux-mêmes et sur la façon de les appliquer.

Notre travail consiste en deux améliorations significatives des modèles markoviens cahés :

* La modélisation des modèles markoviens cachés du second ordre.
* Une nouvelle distance de Viterbi à optimalité locale.

Nous avons développé une nouvelle formulation de l'algorithme de Baum-Welch et une extension de l'algorithme de Viterbi, qui rendent les modèles markoviens cachés du second ordre efficaces en calcul pour des applications en temps réel. Nous avons pu montrer, à travers les résultats d'un reconnaisseur de mots isolés, qu'il y avait une nette diminution du taux d'erreurs de reconnaissance avec le second ordre par rapport au premier ordre. Il serait très utile d'appliquer ces modèles du second ordre à la reconnaissance de mots enchaînés ou à celle de la parole continue. L'extension à des modèles markoviens cachés d'ordre plus élevé a été aussi discutée. La généralisation à un ordre supérieur quelconque de ces modèles HMM, ouvre une voie de recherche sur le choix de l'ordre des HMMs permettant une bonne reconnaissance tout en gardant l'efficacité de calcul, et sur la possibilité de réaliser un système général intégrant des différents ordres HMMs.

Enfin, nous avons proposé une nouvelle stratégie d'utilisation de l'algorithme de Viterbi dans la phase de reconnaissance pour la parole continue. Elle est basée sur une procédure de comparaison d'optimum locaux calculés par l'algorithme de Viterbi dans une fénêtre de trames de largeur fixe et à distance fixe pour les différents modèles HMM. Cette stratégie, par bloc, a donné de meilleurs résultats que les versions classiques de l'algorithme de Viterbi. Elle laisse entrevoir

des résultats encourageants et permet surtout l'utilisation de traits acoustiques dans le processus de reconnaissance. En effet, le point faible des HMMs réside dans le fait qu'ils forment un système fermé par rapport à l'introduction des connaissances spécifiques au domaine de la parole. L'algorithme VITERBI-BLOC représente une bonne voie pour remédier à cet inconvénient. En effet, par le biais des choix des critères de comparaison, il permettra une forte intéraction avec d'autres processeurs pour confirmer ou infirmer un segment décodé. La recherche dans cette voie s'impose, surtout qu'au CRIN le système APHODEX fournit de bons outils pour commencer cette recherche.

Enfin, une très bonne continuation de ce travail, sera la réalisation d'un système pour la reconnaissance de la parole continue intégrant :

* Les HMMs du second ordre,
* L'algorithme VITERBI-BLOC,
* Des techniques à bases de connaissances.

Par exemple, la réalisation d'un système utilisant l'extension de l'algorithme de VITERBI-BLOC au second ordre en interaction avec d'autres processeurs ou sources de connaissance, grâce à la stratégie locale de cet algorithme.

Bibliographie

[Anigbogu 89] J. C. Anigbogu. Un système stochastique de reconnaissance de caractères imprimés multipolices. *DEA Informatique, CRIN 89-R-127. Uni. Nancy I*, 1989.

[Ariki 84] Y. Ariki, K. Maenubu, et T. Sakai. Speaker-independent word recognition in connected speech on the basis of phoneme recognition. *Information Sciences 33*, pages 31–61, 1984.

[Atal 76] B. S. Atal. A pattern recognition system approach to voiced-unvoiced-silence classification with applications to speech recognition. *IEEE Trans. on ASSP*, 24(3), 1976.

[Averbuch 86] A. Averbuch, L. R. Bahl, R. Bakis, P. F. Brown, A. Cole, G. Daggett, S. K. Das, K. Davies, S. V. DeGennaro, P. V. de Souza, E. A. Epstein, D. Fraleigh, F. Jelinek, S. Katz, B. L. Lewis, R. L. Mercer, A. J. Nadas, D Nahamoo, M. A. Picheny, G. Shichman, et P. Spinelli. An IBM-PC based large-vocabulary isolated-utterance speech recognizer. *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, pages 53–56, Tokyo, 1986.

[Averbuch 87] A. Averbuch, L. R. Bahl, R. Bakis, P. F. Brown, A. Cole, G. Daggett, S. K. Das, K. Davies, S. V. DeGennaro, P. V. de Souza, E. A. Epstein, D. Fraleigh, F. Jelinek, J. Moorhead, B. L. Lewis, R. L. Mercer, A. J. Nadas, D Nahamoo, M. A. Picheny, G. Shichman, et P. Spinelli. Experiments with the Tangora 20,000 word speech recognizer. *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, pages 701–704, Dallas, 1987.

[Bahl 74] L. R. Bahl, J. Cocke, F. Jelinek, et J. Raviv. Optimal decoding of linear codes for minimizing symbol error rate. *IEEE Trans. Inform. Theory IT-20*, 275–320, 1974.

[Bahl 75] L. R. Bahl et F. Jelinek. Decoding for channels with insertions, deletions and substitutions, with applications to speech recognition. *IEEE Trans. Inform. Theory, IT-21*, 404–411, July 1975.

[Bahl 76] L. R. Bahl, J. K. Baker, P. S. Cohen, N. R. Dixon, F. Jelinek, R. L. Mercer, et H. F. Silverman. Preliminary results on the performance of a system for the automatic recognition of continuous speech. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pages 425–429, 1976.

[Bahl 79] L. R. Bahl, R. Bakis, P. S. Cohen, A. G. Cole, F. Jelinek, B. L. Lewis, et R. L. Mercer. Recognition results with several experimental acoustic processors. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pages 249–251, 1979.

[Bahl 80] L. R. Bahl, R. Bakis, P. S. Cohen, A. G. Cole, F. Jelinek, B. L. Lewis, et R. L. Mercer. Further results on the recognition of a continuously read natural corpus. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pages 872–875, 1980.

[Bahl 83] L. R. Bahl, F. Jelinek, et R. L. Mercer. A maximum likelihood approach to continuous speech recognition. *IEEE Trans. PAMI.*, PAMI-5(2):179–190, 1983.

[Bahl 86] L. R. Bahl, P. F. Brown, P.V. de Souza, et R. L. Mercer. Maximum mutual information estimation of hidden Markov models parameters for speech recognition. *IEEE-IECEJ-ASJ ICASSP*, pages 49–52, 1986.

[Bahl 87] L. Bahl, P. Brown, P. de Souza, et R. Mercer. Estimating HMM parameters so as to maximise speech recognition accuracy. *Research Report RC-13121. IBM TJ Waston Research Center*, 9/10/1987.

[Bahl 88] L. R. Bahl, P. F. Brown, P. V. de Souza, et R. L. Mercer. A new algorithm for the estimation of hidden Markov model parameters. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pages 493–496, New York, 1988.

[Bahl 89] L. R. Bahl, R. Bakis, J. Bellegarda, P. F. Brown, D. Burshtein, S. K. Das, P. V. de Souza, P. S. Gopalakrishnan, F. Jelinek, D. Kanevsky, R. L. Mercer, A. J. Nadas, D. Nahamoo, et M. A. Picheny. Large vocabulary natural langage continuous speech recognition. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pages 465–467, Glasgow, Scotland, 1989.

[Baker 75a] J. K. Baker. The DRAGON system - An overview. *IEEE Trans. Acoust., Speech, Signal Processing*, ASSP-23(1):24–29, 1975.

[Baker 75b] J. K. Baker. *Stochastic Modeling as a Means of Automatic Speech Recognition.* Thèse de Doctorat, Carnegie-Mellon University, April 1975.

[Baker 75c] J. K. Baker. Stochastic modeling for automatic speech understanding. D. R. Reddy, éditeur, *Speech Recognition*, Academic Press, New York, 1975.

[Bakis 76] R. Bakis. Continuous speech recognition via centisecond acoustic states. *J. Acoustical Society Am. 59 Supp. 1*, 1976.

[Baum 66] L. E. Baum et T. Petrie. Statistical inference for probabilistic functions of finite state Markov chains. *Ann. Math. Stat. 37*, 1554–1563, 1966.

[Baum 67] L. E. Baum et J. A. Eagon. An inequality with applications to statistical estimation for probabilistic functions of Markov

processes and to a model for ecology. *Bull. Amer. Math. Soc. 73*, 360–363, 1967.

[Baum 68] L. E. Baum et G. R. Sell. Growth transformations for functions on manifolds. *Pac. J. Math., 27*, 211–227, 1968.

[Baum 70] L. E. Baum, T. Petrie, G. Soules, et N. Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *Ann. Math. Stat.*, 41(1):164–171, 1970.

[Baum 72] L. E. Baum. An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes. *Inequalities*, 3:1–8, 1972.

[Bellman 57] R. Bellman. *Dynamic Programming.* Princeton, N.F., Univ. Press, Princeton, 1957.

[Billi 82] R. Billi. Vector quantification and Markov source models applied to speech recognition. *Proc. Int. Conf. on Acoustics, Speech and Signal Processing 1982*, pages 574–577, 1982.

[Billingsley 61] P. Billingsley. Statistical inference for Markov processes. *Univ. of Chicago Press*, Chicago, 1961.

[Boe 88] L.J Boé et J.S. Liénard. La communication parlée est-elle une science ? *Actes des* $17^{èmes}$ *Journées d'Etudes sur la Parole*, pages 79–92, Nancy, 1988.

[Bourlard 86] H. Bourlard et C. J. Wellekens. Connected speech recognition by statistical methods. *Eurasip short course on speech recognition*, Brussel, feb. 1986.

[Boyer 87a] A. Boyer. Application des techniques de programmation dynamique et de quantification vectorielle à la reconnaissance des mots isolés et des mots enchaînés. *Thèse de Doct. Univ. de NANCY 1*, 1987.

[Boyer 87b] A. Boyer, J. Di Martino, et J. P. Haton. DTW and QV in isolated and connected word recognition. *European conference on speech technology*, Edimburg, 1987.

[Boyer 88] A. Boyer, J. Di Martino, P. Divoux, J. P. Haton, J. F. Mari, et K. Smaili. Statistical methods in multi-speaker automatic speech recognition. *Fourth International Symposium on Applied Stochastic Models and Data Analysis*, Nancy, Dec. 1988.

[Breant 82] M. Breant. Méthode de segmentation automatique en phonèmes d'une phrase donnée. *Thèse de Docteur-ingénieur. Uni. P. Sab. de Toulouse*, Octobre 1982.

[Briant 84] N. Briant et B. Flocon. Syril: système temps réel de reconnaissance de mots indépendants du locuteur. *Actes du* $4^{ème}$ *congrès RFIA, AFCET-INRIA*, Paris, Janvier 1984.

[Bridle 82] J. S. Bridle, M. D. Brown, et R. M. Chamberlain. An algorithm for connected word recognition. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pages 899–902, mai 1982.

[Brown 87] P. F. Brown. The acoustic-modeling problem in automatic speech recognition. *PhD Dissertation. Carnegie Mellon University*, May 1987.

[Burton 84] D. K. Burton, J. T. Buck, et J. E. Shore. Parameter selection for isolated word recognition using vector quantization. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, San Diego, 1984.

[Buzo 80] A. Buzo, A. H. Gray, R. M. Gray, et J. D. Markel. Speech coding based upon vector quantification. *IEEE Trans. Acoust., Speech, Signal Processing*, ASSP-28(5):562–574, 1980.

[Caelen 78] J. Caelen et G. Perennou. Indices et traits acoustiques dans un système de reconnaissance de la parole continue, quelques résultats. *GALF, 9^e JEP*, Lannion, 1978.

[Calliope 89] Calliope. *La parole et son traitement automatique.* Masson, Paris, 1989.

[Cave 80] R. L. Cave et L. P. Neuwirth. Hidden Markov Models for English. *Proc. of the Symposium on the Applications of Hidden Markov Models to Text and Speech, IDA-CRD*, pages 16–56, Princeton, NJ, 1980.

[Cerf 86] H. Cerf, A. M. Derouault, M. El Beze, B. Mérialdo, et S. Soudoplatoff. Reconnaissance de la parole par des modèles markoviens: application aux grands vocabulaires. *15^e JEP-SFA*, 1986.

[Charpillet 84] F. Charpillet, J.P. Haton, et J. M. Pierrel. Apport de la programmation dynamique en reconnaissance automaique de la parole continue. *Actes du 4^ème congrès RFIA, AFCET-INRIA*, Paris, Janvier 1984.

[Chow 86] Y. L. Chow, R. M. Schwartz, S. Roucos, O. A. Kimball, P. J. Price, G. F. Kubala, M. O. Dunham, M. A. Krasner, et J. Makhoul. The role of word-dependent coarticulatory effects in a phoneme-based speech recognition system. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pages 1593–1596, Tokyo, 1986.

[Chow 87] Y. L. Chow, M. O. Dunham, O. A. Kimball, M. A. Krasner, G. F. Kubala, J. Makhoul, P. J. Price, S. Roucos, et R. M. Schwartz. BYBLOS: the BBN continuous speech recognition system. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pages 89–92, Dallas, 1987.

[Codogno 87] M. Codogno et L. Fissore. Duration modelling in finite state automata for speech recognition and fast speaker adaptation. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pages 1269–1272, Dallas, 1987.

[Combescure 81] P. Combescure. Vingt listes de dix phrases phonétiquement équilibrées. *Revue d'Acoustique*, 14(56), 1981.

[Cook 88] A. E. Cook. Experimental evaluation of algorithms for connected speech recognition using hidden Markov models. *7^th FASE Symposium*, Edinburg, 1988.

[Cover 84] T. M. Cover. An algorithm for maximizing expected log investment return. *IEEE Trans. Inform. Theory IT-30*, 369–373, 1984.

[Cravero 84] M. Cravero, L. Fissore, R. Pieraccini, et C. Scagliola. Syntax driven recognition of connected words by Markov models. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, page paper 35.5, San Diego, 1984.

[Cravero 86] M. Cravero, R. Pieraccini, et F. Raineri. Definition and evaluation of phonetic units for speech recognition by hidden Markov models. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, Tokyo, 1986.

[Crystal 86] T. H. Crystal et A. S. House. Characterisation and modelling of speech-segment durations. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pages 2791–2794, Tokyo, 1986.

[Davis 52] K. H. Davis, R. Biddulph, et S. Balashek. Automatic recognition of spoken digits. *JASA, 24*, 637–642, 1952.

[Dempster 77] A. P. Dempster, N. M. Laird, et D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J. R. Stat. Soc. B 39*, 1–38, 1977.

[Derouault 86] A. M. Derouault et B. Mérialdo. Natural langage modeling for phoneme-to-text transcription. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):742–749, November 1986.

[Derouault 87] A. M. Derouault. Context-dependent phonetic Markov models for large vocabulary speech recognition. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, Dallas, 1987.

[Derouault 89] A. M. Derouault et B. Mérialdo. Improving speech recognition accuracy with contextual phonemes and MMI training. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, Glasgow,Scotland, 1989.

[Divoux 85] P. Divoux et J. P. Haton. Classification hiérarchique et reconnaissance multilocuteur. *Actes du 5ème congrès RFIA, AFCET-INRIA*, Grenoble, 1985.

[Divoux 88] P. Divoux. Mimule: Un système de reconnaissance de mots isolés multilocuteurs utilisant les techniques de classification. *Thèse de Doct. Univ. de NANCY 1*, Mars 1988.

[Dours 88] C. Dours et G. Pérennou. The role of intermediary scores in word-spotting. *Seventh FASE Symposium*, Edinburgh, 1988.

[Dours 89] C. Dours. Contribution à l'étude du décodage acoustico-phonétique pour la reconnaissance automatique de la parole. *Thèse de Doct. Univ. Paul Sabatier de Toulouse*, 1989.

[Dreyfus 49] J. G. Dreyfus. Sonograph and sound mechanics. *JASA, 22*, 731–739, 1949.

[Ephraim 87] Y. Ephraim, A. Dembo, et L. R. Rabiner. A minimum discrimination information approach for hidden Markov modeling. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pages 25–28, Dallas, 1987.

[Esin 77] N. A. Esin et M. I. Shlesinger. Synthesis of a probabilistic finite-state grammar describing a given set of sequences. *Kibernetika*, pages 116–120, 1977.

[Euler 88] S. Euler et D. Wolf. Continuous hidden Markov models in speaker independent isolated word recognition. *Proc. 4^e EUSIPCO*, pages 1185–1188, Grenoble, France, september 1988.

[Falaschi 88] A. Falaschi. Phonetic recognition of natural speech by non stationary Markov models. *Rapport Technique,88D012, ENST, Dep. Signal. Paris*, 1988.

[Faray 79] R. F. H. Faray. Word-level recognition of cursive script. *IEEE Trans. on Computers*, C-28(No 2):172–175, 1979.

[Feller 58] W. Feller. *An Introduction to Probability Theory and Its Applications.* John Wiley, 2nd Edition, Vol I, New York, 1958.

[Ferguson 80a] J. D. Ferguson. Hidden Markov analysis: an introduction. *Proc. of the Symp. on the Applications of Hidden Markov Models to Text and Speech, IDA-CRD*, pages 8–15, Princeton,NJ, 1980.

[Ferguson 80b] J. D. Ferguson. Variable duration models for speech. *Proc. of the Symp. on the Applications of Hidden Markov Models to Text and Speech, IDA-CRD*, pages 143–179, Princeton,NJ, 1980.

[Flocon 84] B. Flocon et Ph. Lockwood. Système de reconnaissance de mots isolés multilocuteurs pour un vocabulaire de 130 mots. intégration dans un poste de travail. 13^e *JEP,GALF*, Bruxelles, 1984.

[Fohr 86] D. Fohr. APHODEX : Un système expert en décodage acoustico-phonétique de la parole continue. *Thèse de Doct. Univ. de NANCY 1*, 1986.

[Forney 73] G. D. Forney. The Viterbi algorithm. *Proc. IEEE*, pages 268–278, Mar 1973.

[Gagnoulet 82] C. Gagnoulet et M. Couvrat. Séraphine: a connected word speech recognition system. *Proc. IEEE ICASSP*, page 887, Mai 1982.

[Gauvain 82] J. L. Gauvain et J. J. Mariani. A method for connected word recognition and word spotting on a microprocessor. *Proc. IEEE ICASSP*, page 891, 1982.

[Gauvain 83] J. L. Gauvain et J. Mariani. Mozart: Un système de reconnaissance globale de parole continue. 11^e *I.C.A.*, Paris, 1983.

[Gilloux 84] M. Gilloux, G. Mercier, et C. Tarridec. Un système expert pour la reconnaissance de la parole. *Colloque international d'intelligence artificielle*, pages 30–40, Marseille, Octobre 1984.

[Gourinda 88] A. Gourinda. Codage et reconnaissance de la parole par quantification vectorielle. *Thèse de Doct. Univ. de NANCY 1*, 1988.

[Gray 84] R. M. Gray. Vector quantization. *IEEE ASSP Magazine*, 4–29, April 1984.

[Groc 80] B. Groc et D. Tuffelli. A continuous speech recognition system for data-base consultation. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pages 896–899, 1980.

[Hartley 58] H. O. Hartley. Maximum likelihood estimation from incomplete data. *Biometrics*, 14:174–194, 1958.

[Haton 76] J. P. Haton et J. M. Pierrel. Organization and operation of a connected speech understanding system, at lexical, syntactical and semantical levels. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, 1976.

[Haton 82] J. P. Haton. Speech recognition and understanding. *Proc. IEEE* 6^{th} *Int. Conf. on Pattern Recognition, Invited paper*, pages 570–581, Munich, 1982.

[Haton 84] J.P. Haton et M. Lazrek. Segmentation et identification des phonèmes dans un système de reconnaissance de la parole continue. *Actes du* $4^{ème}$ *congrès RFIA, AFCET-INRIA*, pages 5–21, Paris, Janvier 1984.

[Haton 85] J. P. Haton. Intelligence artificielle en compréhension de la parole : État des recherches et comparaison avec la vision par ordinateur. *TSI*, 4(3):265–287, 1985.

[Haton 87] J. P. Haton, N. Carbonell, D. Fohr, J. F. Mari, et A. Kriouile. Interaction between stochastic modeling and knowledge-based techniques in acoustic-phonetic decoding of speech. *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, pages 868–871, Dallas, 1987.

[He 88] Y. He. Extended Viterbi algorithm for second order hidden Markov process. *Proc. IEEE* 9^{th} *Int. Conf. on Pattern Recognition*, pages 718–720, Rome,Italy, 1988.

[Huang 88] X. D. Huang et M. A. Jack. Semi-continuous hidden Markov models in isolated word recognition. *Proc. IEEE* 9^{th} *Int. Conf. on Pattern Recognition*, pages 406–408, Rome,Italy, 1988.

[Jambu 78] M. Jambu. *Classification Automatique pour l'analyse des données.* Volume I : Méthodes et Algorithmes, Dunod, 1978.

[Jelinek 69] F. Jelinek. A fast sequential decoding algorithm using a stack. *IBM Journal of Research and Development*, 13:675–685, November 1969.

[Jelinek 75] F. Jelinek, L. R. Bahl, et R. L. Mercer. Design of a linguistic statistical decoder for the recognition of continuous speech. *IEEE Trans. IT*, IT-21(No 3), May 1975.

[Jelinek 76] F. Jelinek, R. L. Mercer, et L. R. Bahl. Continuous speech recognition: statistical methods. *C.S.R. group, IBM T.J.*, 1976.

[Jelinek 80] F. Jelinek et R. L. Mercer. Interpolated estimation of Markov source parameters fromm sparse data. *Pattern recogni-*

tion in pratice, E.S.Gelsema et L.N.Kanals,Eds Amsterdam, pages 381–402, The Netherlands: North Holland, 1980.

[Jouvet 86] D. Jouvet, J. Monné, et D. Dubois. A new network-based speaker-independent connected-word recognition system. *IEEE-ICASSP*, pages 1109–1112, Tokyo, 1986.

[Jouvet 87] D. Jouvet. Reconnaissance de mots connectés indépendamment du locuteur par des méthodes statistiques. *AFCET-INRIA*, 1987.

[Juang 85a] B. H. Juang et L. R. Rabiner. Mixture autoregressive hidden Markov models for speech signals. *IEEE-ASSP-33*, pages 1404–1413, 1985.

[Juang 85b] B. H. Juang, L. R. Rabiner, S. E. Levinson, et M. M. Sondhi. Recent developments in the application of hidden Markov models to speaker-independent isolated word recognition. *IEEE-ICASSP*, pages 9–12, 1985.

[Juang 86] B. H. Juang et L. R. Rabiner. Mixture autoregressive hidden Markov models for speaker independent isolated word recognition. *IEEE-ICASSP*, pages 41–44, Tokyo, 1986.

[Katz 87] S. M. Katz. Estimation of probabilities from sparse data for the langage model component of a speech recognizer. *IEEE-ASSP-35*, pages 400–401, 1987.

[Klatt 80] O. H. Klatt. Scriber and lafs: two new approaches to speech. Wayne A. Lea, éditeur, *Trends in speech recognition*, Prentice-Hall, 1980.

[Kordi 87] K. Kordi, C. Xydeas, et M. Holt. Printed character recognition using Markov models. *Actes du* $11^{ème}$ *Colloque GRETSI*, pages 507–509, Nice, Juin 1987.

[Kriouile 86] A. Kriouile. Segmentation de la parole continue par une méthode stochastique. *DEA Informatique, CRIN, Uni. Nancy I*, 1986.

[Kriouile 90a] A. Kriouile, J. F. Mari, et J. P. Haton. L'algorithme viterbi-bloc pour la reconnaissance de la parole continue. 18^{e} *JEP-SFA*, Montréal, 1990.

[Kriouile 90b] A. Kriouile, J. F. Mari, et J. P. Haton. Some improvements in speech recognition algorithms based on HMM. *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, Albuquerque, New Mexico, 1990.

[Kubala 88] F. Kubala, Y. Chow, A. Derr, M. Feng, O. Kimball, J. Makhoul, P. Price, J. Rohlicek, S. Roucos, R. Schwartz, et J. Vandegrift. Continuous speech recognition results of the BYBLOS system on the DARPA 1000-word resource management database. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pages 291–294, New York, 1988.

[Kundu 88] A. Kundu, Y. He, et P. Bahl. Recognition of handwritten word: first and second order hidden Markov model based approach. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition, Ann Arbor*, Michigan, June 1988.

[Lea 70] W. A. Lea. Evaluating speech-recognition work. *J. Acoust. Soc. Am.*, 47:1612–1614(L), 1970.

[Lee 88a] K. F. Lee et H. W. Hon. Large-vocabulary speaker-independent continuous speech recognition using HMM. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pages 123–126, New York, 1988.

[Lee 88b] K.F. Lee. *Large-Vocabulary Speaker-Independant Continuous Speech Recognition: The SPHINX System, PhD thesis.* CMU-CS-88-148, Carnegie-Mellon University, April 1988.

[Levinson 78] S. E. Levinson. The effects of syntactic analysis on word recognition accuracy. *Bell Sys. Tech. J.*, 57:1627–1644, 1978.

[Levinson 80] S. E. Levinson et K. L. Shipley. A conversational mode airline information and reservation system using speech input and output. *Bell Sys. Tech. J.*, 59(1):119–137, 1980.

[Levinson 83a] S. E. Levinson, L. R. Rabiner, et M. M. Sondhi. An introduction to the application of the theory of probabilistic functions of a Markov process to automatic speech recognition. *The Bell System Technical Journal*, 62(4), 1983.

[Levinson 83b] S. E. Levinson, L. R. Rabiner, et M. M. Sondhi. Speaker independent isolated digit recognition using hidden Markov models. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pages 1049–1052, 1983.

[Levinson 86] S. E. Levinson. Continuously variable duration hidden Markov models for automatic speech recognition. *Comp.Speech and Lang*, 1:29–46, 1986.

[Levinson 87] S. E. Levinson. Continuous speech recognition by means of acoustic/phonetic classification obtained from a hidden Markov model. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pages 93–96, Dallas, 1987.

[Levinson 88] S. E. Levinson, A. Ljolje, et L. G. Miller. Large vocabulary speech recognition using a hidden Markov model for acoustic/phonetic classification. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pages 505–508, New York, 1988.

[Linde 80] Y. Linde, A. Buzo, et R. M. Gary. An algorithm for the vector quantizer design. *IEEE. Trans. on Communication*, com. 28(1), Jan. 1980.

[Liporace 82] L. R. Liporace. Maximum likelihood estimation for multivariate observations of Markov sources. *IEEE Trans. Inform. Theory IT-28*, 729–734, 1982.

[Lochschmidt 82] B. Lochschmidt. Acoustic-phonetic analysis based on an articulatory. J. P. Haton, éditeur, *Automatic Speech Analysis and Recognition*, Reidel, 1982.

[Lowerre 80] B. Lowerre et R. Reddy. The HARPY speech understanding system. W. A. Lea, éditeur, *Trends in speech recognition*, pages 340–360, Prentice-Hall Signal processing series, 1980.

[Mao 90] W.D. Mao et S.Y. Kung. An object recognition system using stochastic knowledge source and VLSI parallel architecture.

Proc. Int. Conf. on PATTERN RECOGNITION, pages 832–836, June, 1990.

[Mari 85a] J. F. Mari. Reconnaissance de mots enchainés à l'aide des modèles markoviens discrets. *Actes du 5ème congrès RFIA, AFCET-INRIA*, Grenoble, Novembre 1985.

[Mari 85b] J. F. Mari et S. Roucos. Speaker independent connected digit recognition using hidden Markov models. *Speech tec.*, pages 22–24, New York, April 1985.

[Mariani 78] J. J. Mariani et J. S. Lienard. Esope 0: un programme de compréhension automatique de la parole procédant par prédiction-vérification aux niveaux phonétiques, lexical et syntaxique. *Congrès ARCET-IRIA Reconnaissance des Formes et Traitement des Images*, Paris, 1978.

[Mariani 87] J. J. Mariani. Hamlet: A prototype of voice activated typewriter. *Proceedings of European Conference on Speech Technology*, pages 222–225, Edinburgh, U.K., September, 1987.

[Markov 13] A. A. Markov. An example of statistical investigation in the text of 'eugene onyegin' illustrating coupling of 'tests' in chains. *Proc. Acad. Sci. St. Petersburg VI Ser. 7*, pages 153–162, 1913.

[Martino 85] J. Di Martino. DTW algorithms for isolated and connected word recognition. R. de Mori et C. Y. Suen, éditeurs, *New Systems and Architectures for Automatic Speech Recognition and Synthesis*, Springer verlag, Heidelberg, 1985.

[Martino 87] J. Di Martino. A multi-level machine for continuous speech recognition. *Proc. IJCAI*, Milan, 1987.

[Mercier 77] G. Mercier et al. A multipurpose speech understanding system. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, 1977.

[Merialdo 87] B. Mérialdo, A. M. Derouault, M. El Beze, et S. Soudoplatoff. Reconnaissance de parole avec un très grand vocabulaires. 16e *JEP-SFA*, 1987.

[Merialdo 88a] B. Mérialdo. Apprentissage des modèles markoviens par maximum d'information mutuelle. 17e *JEP-SFA*, 1988.

[Merialdo 88b] B. Mérialdo. Multi-level decoding for very-large-size-dictionary speech recognition. *IBM Journal of Research and Development*, 32(2), March 1988.

[Merialdo 88c] B. Mérialdo. Phonetic recognition using Hidden Markov Models and Maximum Mutual Information training. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pages 111–114, New York, 1988.

[Myers 81] C. Myers et L. R. Rabiner. A level building dynamic time warping algorithm for connected word recognition. *IEEE Trans. Acoust., Speech, Signal Processing*, ASSP-29(2):284–296, June 1981.

[Myers 82] C. S. Myers et S. E. Levinson. Speaker-independent connected word recognition using a syntax-directed dynamic pro-

gramming procedure. *IEEE Trans. Acoust., Speech, Signal Processing*, pages 561–565, Aug. 1982.

[Nakagawa 88] S. Nakagawa et Y. Hashimoto. A method for continuous speech segmentation using HMM. *Proc. IEEE* 9^{th} *Int. Conf. on Pattern Recognition*, pages 960–962, Rome,Italy, 1988.

[Obrecht 85] R. André Obrecht. Segmentation automatique du signal de parole, sans reconnaissance. *Thèse de Troisième Cycle. Uni. Rennes I*, 1985.

[Ott 77] J. Ott. Counting methods (EM algorithm) in human pedigree analysis; linkage and segregation analysis. *Ann. Human Genetics*, 40:443–454, 1977.

[Paul 85] D. B. Paul. Training of HMM recognisers by simulated annealing. *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, pages 13–16, 1985.

[Perennou 82] G. Pérennou. The ARIAL II speech recognition system. J. P. Haton, éditeur, *Automatic Speech Analysis and Recognition*, pages 269–275, Reidel, 1982.

[Petrie 69] T. Petrie. Probabilistic functions of finite state Markov chains. *Ann. Math. Stat. 40*, 97–115, 1969.

[Pierce 69] J. Pierce. Wither speech recognition? *JASA, 47*, 1969.

[Pierrel 82] J. M. Pierrel. Utilisation des contraintes linguistiques en compréhension de la parole continue : le système myrtille II. *TSI*, 1(5):403–421, 1982.

[Ponting 88] K. M. Ponting. A statistical approach to the determination of hidden Markov models structure. 7^{th} *FASE Symposium*, Edinburgh, 1988.

[Poritz 86] A. B. Poritz et A. G. Richter. On hidden Markov models in isolated word recognition. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pages 705–708, Tokyo, 1986.

[Poritz 88] A. B. Poritz. Hidden Markov Models: A Guided Tour. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pages 7–13, New York, 1988.

[Rabiner 79] L. R. Rabiner. Clustering techniques. *J. Acoust. Soc. Amer.*, 66:663–673, 1979.

[Rabiner 83] L. R. Rabiner, S. E. Levison, et M. M. Sondhi. On the application of vector quantization and hidden Markov models to speaker-independent isolated word recognition. *Bell system tech. J. 62*, 1075–1105, 1983.

[Rabiner 84a] L. R. Rabiner. On the application of energy contours to the recognition of connected word sequences. *Bell Sys. Tech. J.*, 63:1981–1995, Nov 1984.

[Rabiner 84b] L. R. Rabiner, S. E. Levinson, et M. M. Sondhi. On the use of hidden Markov models for speaker independent recognition of isolated words from a medium size vocabulary. *Bell Sys. Tech. J.*, 63:627–642, April 1984.

[Rabiner 85] L. R. Rabiner et S. E. Levinson. A speaker-independent, syntax-directed, connected word recognition system based on

hidden Markov models and Level Building. *IEEE-ASSP-33, No 3*, pages 561–573, June 1985.

[Rabiner 86a] L. R. Rabiner et B. H. Juang. An introduction to hidden Markov models. *IEEE ASSP Magazine*, 4–16, Jan 1986.

[Rabiner 86b] L. R. Rabiner, B. H. Juang, S. E. Levinson, et M. M. Sondhi. Recognition of isolated digits using hidden Markov models with continuous mixture densities. *Bell Sys. Tech. J.*, 64(6):1211–1222, July-Aug 1986.

[Rabiner 86c] L. R. Rabiner, J. G. Wilpon, et B. H. Juang. A model-based connected-digit recognition system using either hidden Markov models or templates. *Computer Speech and Langage*, 1(2):167–197, Dec. 1986.

[Rabiner 87] L. R. Rabiner, J. G. Wilpon, et B. H. Juang. A performance evaluation of a connected digit recognizer. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, Dallas, 1987.

[Rabiner 88] L. R. Rabiner. Mathematical foundations of hidden Markov models. H. Niemann, M. Lang, et G. Sagerer, éditeurs, *Recent Advances in Speech Understanding and Dialog Systems*, pages 183–205, Springer Verlag, 1988.

[Rabiner 89] L. R. Rabiner, C. H. Lee, B. H. Juang, et J. G. Wilpon. HMM clustering for connected word recognition. *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, pages 405–408, Glasgow, Scotland, 1989.

[Rohlicek 89] J. R. Rohlicek, W. Russell, S. Roucos, et H. Gish. Continuous hidden Markov modeling for speaker independent word spotting. *Proc. Int. Conf. on Acoustics, Speech and Signal Processing*, 1989.

[Rosemberg 87] A. E. Rosemberg et A. M. Colla. A connected speech recognition system based on spotting diphone-like segments-preliminary results. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pages 85–87, Dallas, 1987.

[Ruske 81] G. Ruske et T. Schotola. The efficiency of demisyllabe segmentation in the recognition of spoken words. *Proc. Int. Conf. on Acoustics, Speech and Signal Processing*, 1981.

[Russel 85] M. J. Russel et R. K. Moore. Explicit models for automatic speech recognition. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pages 5–8, Tampa, Florida, 1985.

[Russel 86] M. J. Russel et A. E. Cook. Experiments in speaker-dependent isolated digit recognition using hidden Markov models. *Proc. Inst. of Acoust. 8, part 7*, pages 291–298, 1986.

[Russel 87] M. J. Russel et A. E. Cook. Experimental evaluation of duration modelling techniques for automatic speech recognition. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pages 2376–2379, Dallas, 1987.

[Russel 88] M. J. Russel. Statistical modelling of state duration correlations in hidden Markov models. 7^{th} *FASE Symposium*, Edinburgh, 1988.

[Sakoe 71] H. Sakoe et S. Chiba. A dynamic programming approach to continuous speech recognition. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, Budapest, Hungary, 1971.

[Sakoe 78] H. Sakoe et S. Chiba. Dynamic programming algorithm optimisation for spoken word recognition. *IEEE Trans. Acoust., Speech, Signal Processing*, 43–49, February 1978.

[Sakoe 79] H. Sakoe. Two-level DP-matching – a dynamic programming – based pattern matching algorithm for connected word recognition. *IEEE Trans. Acoust., Speech, Signal Processing*, ASSP-27(6):588–595, 1979.

[Sambur 76] M. R. Sambur et L. R. Rabiner. A statistical decision approach to the recognition of connected digits. *IEEE Trans. Acoust. Speech, and Signal Proc. ASP-24*, Dec. 1976.

[Sauter 84] L. Sauter. RAPACE : un système de reconnaissance analytique de parole continue. *Congrès AFCET RFIA*, page 89, Paris, 1984.

[Schwartz 84] R. Schwartz, Y.L. Chow, S. Roucos, M. Krasner, et J. Makhoul. Improved hidden Markov modeling of phonemes for continuous speech recognition. *Proc. Int. Conf. on Acoustics, Speech and Signal Processing*, 1984.

[Schwartz 85] R. Schwartz, Y. Chow, O. Kimball, S. Roucos, M. Krasner, et J. Makhoul. Context-dependent modeling for acoustic-phonetic recognition of continuous speech. *Proc. Int. Conf. on Acoustics, Speech and Signal Processing*, pages 1205–1208, 1985.

[Shannon 48] C. C. Shannon. A mathematical theory of communications. *Bell Sys. Tech. J. 27*, 379–423, 623–656, 1948.

[Shannon 51] C. C. Shannon. Prediction and entropy of printed english. *Bell Sys. Tech. J. 30*, 50–64, 1951.

[Stern 86] P.E. Stern, M. Ezkénazi, et D. Memmi. An expert system for speech spectrogram reading. *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, pages 1193–1196, Tokyo, 1986.

[Sugawara 85] K. Sugawara, M. Nishimura, K. Toshioka, M. Okochi, et T. Kaneko. Isolated word recognition using hidden Markov models. *Proc. Int. Conf. on Acoustics, Speech and Signal Processing*, pages 1–4, 1985.

[Sugawara 86] K. Sugawara, M. Nishimura, et A. Kuroda. Speaker adaptation for a hidden Markov model. *IEEE-IECEJ-ASJ ICASSP*, pages 2667–2670, 1986.

[Tishby 88] N. Tishby. Information theoretic factorization of speaker and language in hidden Markov models, with application to speaker recognition. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pages 87–90, New York, 1988.

[Vardi 85] Y. Vardi, L. A. Shepp, et L. Kauffman. A statistical model for positron emission tomography. *J. Am. Stat. Assn.*, 80:8–37, 1985.

[Viterbi 67] A. J. Viterbi. Error bounds for convolutional codes and asymtotically optimum decoding algorithm. *IEEE Trans on Information Theory*, IT-13:260–269, April 1967.

[Wellekens 86] C. J. Wellekens. Global connected digit recognition using Baum-Welch algorithm. *IEEE-ICASSP*, pages 1081–1084, Tokyo, 1986.

[Wellekens 87] C. J. Wellekens. Explicit time correlation in hidden Markov models for speech recognition. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pages 384–386, Dallas, 1987.

[White 78] G. M. White. Dynamic programming, the Viterbi algorithm, and low cost speech recognition. *Proc. IEEE. Int. Conf. Acoust., Speech, Signal Processing*, 413–417, 1978.

NOM DE L'ETUDIANT : KRIOUILE Abdelaziz

NATURE DE LA THESE : "La reconnaissance automatique de la parole et les modèles markoviens cachés, modèle du second ordre et distance de Viterbi à optimalité locale."

VU, APPROUVE ET PERMIS D'IMPRIMER

NANCY, le 23 SEP. 1990 n° 1899

LE PRESIDENT DE L'UNIVERSITE DE NANCY I

Résumé

Des travaux intensifs sur la reconnaissance automatique de la parole utilisant les modèles stochastiques ont été réalisés durant les cinq dernières années. L'application des modèles markoviens cachés (HMM) du premier ordre a conduit à des résultats impréssionnants dans le domaine de la reconnaissance de mots isolés et de la parole continue.

Notre objectif était de montrer que l'apport des modèles markoviens cachés à la reconnaissance automatique de la parole est d'autant plus important qu'on mène des réflexions fondamentales sur les modèles markoviens eux même et sur la façon de les appliquer.

Notre travail consiste en deux améliorations significatives des modèles markoviens :

- La modélisation des HMMs du second ordre.
- Une nouvelle distance de Viterbi à optimalité locale.

Nous avons développé une nouvelle formulation de l'algorithme de Baum-Welch et une extension de l'algorithme de Viterbi, qui rendent les modèles markoviens cachés du second ordre efficaces en calcul pour des applications en temps réel. Nous avons pu montrer, à travers les résultats d'un reconnaisseur de mots isolés, qu'il y avait une nette amélioration du taux de reconnaissance avec le second ordre par rapport au premier ordre. L'extension à des HMMs d'ordre plus élevé a été aussi discutée.

Enfin, nous avons proposé une nouvelle stratégie d'utilisation de l'algorithme de Viterbi dans la phase de reconnaissance de la parole continue. Elle est basée sur une procédure de comparaison d'optimums locaux calculés par l'algorithme de Viterbi dans une fenêtre de trames de largeur fixe et à distance fixe pour les différents modèles HMM. Cette stratégie, par bloc, a donné de meilleurs résultats que les versions classiques de l'algorithme de Viterbi. Elle laisse entrevoir des résultats encourageants et permet une forte interaction avec d'autres processeurs acoustiques pour confirmer ou infirmer un segment décodé.

Mots clés :

Reconnaissance automatique de la parole
Modèles markoviens cachés du premier et du second ordre
Algorithme de Viterbi
Algorithme Baum-Welch