Chapitre II

MODELES DE MARKOV CACHES

Présentation

La reconnaissance automatique des séquences temporelles est une problématique en plein essor. Sous ses différentes formes, elle a déjà donné lieu à une grande variété d'applications comme le traitement automatique de la parole ou le suivi de processus industriels. Par son récent rapprochement au traitement d'images, elle tend désormais à s'ouvrir plus largement à de vastes domaines d'applications : reconnaissance de mouvements, classification de vidéos par le contenu, vision robotique, en bioinformatique et dans le domaine médical tel que le diagnostic médical avec des données puce ADN, et récemment dans la cardiologie, notamment l'analyse et la segmentation du signal ECG.

Ce chapitre, aborde en détail une méthode efficace et très employée pour la classification et la reconnaissance de telles séquences, il s'agit des HMMs.

II.1. INTRODUCTION

La reconnaissance de séquences telle que l'on va étudié dans ce document consiste à extraire un certain nombre de séquences utiles d'une longue observation à partir d'un signal ECG, pour pouvoir reconnaitre les arythmies cardiaques. Il est nécessaire pour notre application de choisir un modèle pour les séquences à reconnaître. Celui-ci doit être composé d'éléments qui peuvent modéliser la séquencialité du système et son processus d'observation. Pour cela, le vocabulaire commun aux différentes applications des divers domaines cités précédemment, définit:

Les états : à un instant donné, la description du système est donnée par un état donné.

Les transitions : ce sont les changements d'état.

Tenant compte de cela, c'est sous la forme d'un graphe qu'un système séquentiel sera le plus clairement représenté. Cette approche est unanime. Les graphes utilisés présentent généralement les états sous forme de places et les transitions sous forme d'arcs séparant les places. Selon la nature stochastique des éléments du graphe, des méthodes statistiques ont été mises en place, offrant de bonnes performances. Ces méthodes présentent en effet l'avantage de pouvoir estimer les paramètres des modèles par apprentissage. Sous réserve de disposer de suffisamment de séquences d'entraînement représentatives des données à traiter ultérieurement, il est ainsi possible de créer un modèle particulièrement bien adapté au système étudié. **[3] + [16]**

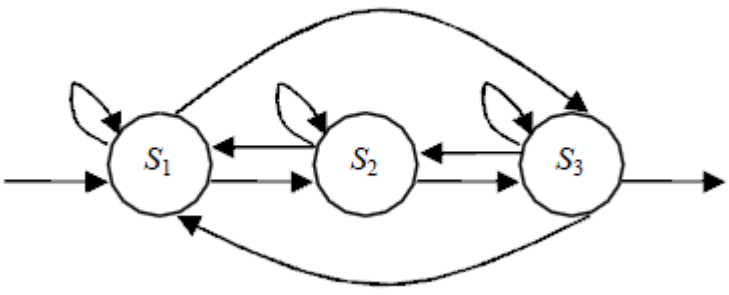


Figure II. 1 : Exemple d'automate probabiliste d'états finis (composé ici de 3 états). Les flèches figurent les transitions possibles entre états.

Les modèles statistiques les plus utilisés pour la reconnaissance de séquences sont les modèles markoviens.

Un modèle markovien d'ordre k considère que l'état du système à un instant t ne dépend que de l'état aux **k** instants précédents. Cela implique la propriété d'indépendance conditionnelle suivante :

$P(q_t / q_1^{t-1}) = P(q_t / q_{t-k}^{t-1})$**, q étant la variable d'état, et** $q_1^{t-1} = \{q_1,...,q_{t-1}\}$**.**

Cette formule sous entend que la séquence passée peut être résumée de façon concise et permet d'alléger fortement le modèle. La plupart des applications repose sur un modèle d'ordre **1**.

Le modèle markovien le plus souvent utilisé est le modèle de Markov caché (Hidden Markov Model ou HMM). Celui-ci repose sur un modèle de Markov d'ordre 1 simulant l'évolution de l'état du système. Ce modèle est caché, c'est à dire que l'on n'a pas accès à la séquence d'état q_1^T proprement dite mais à une séquence d'observations y_t générées par le système à chaque instant. Les observations correspondent aux données du système. **[16] + [17]**

Les HMMs, introduits par Baum Welch dans les années 60, utilisés à partir des années 80 en reconnaissance de la parole, la où ils ont été pleinement exploités, appliqués ensuite à la reconnaissance de texte manuscrit, au traitement d'images, et à la Bioinformatique comme le séquencement de l'ADN. Mais aussi dans bien d'autres applications dans lesquelles apparaît une séquentialité comme la segmentation du signal ECG dans le domaine de la cardiologie. **[18]**

II.2. DEFINITIONS

Généralement un processus ou modèle stochastique est une suite d'expériences dont le résultat dépend du hasard. En certains temps **0, 1, 2,…, t**, observons un système. Celui-ci peut se trouver dans l'un des états d'une collection finie d'états possibles. L'observation du système dont le résultat (aléatoire) est l'état dans lequel se trouve le système. Un processus stochastique est un phénomène temporel où intervient le hasard, c'est-à-dire une variable aléatoire **X(t)** évoluant en fonction du temps. On peut aussi dire de ce processus qu'il émet des séquences d'états $\mathbf{S = s_1, s_2,…., s_t}$. Chaque séquence est émise avec une probabilité $\mathbf{P(S) = (s_1, s_2,……,s_T)}$. Pour calculer **P(S)**, il faut se donner

la probabilité initiale **P (s_1)** et les probabilités d'être dans un état s_t, connaissant l'évolution antérieure. **[16]**

Un processus stochastique est markovien (ou de Markov) si son évolution est entièrement déterminée par une probabilité initiale et des probabilités de transitions entre états (son évolution ne dépend pas de son passé mais uniquement de son état présent, l'état courant du système contient toute l'information pour prédire son état futur. **[19]**

$$P(X(t_n) \le o_n \mid X(t_{n-1}) = o_{n-1}, \ldots, X(t_1) = o_1) = P(X(t_n) \le o_n \mid X(t_{n-1}) = o_{n-1})$$

Les modèles de Markov Cachés (Hidden Markov Models ou HMM) modélisent des phénomènes aléatoires dont on suppose qu'ils sont composés à un premier niveau d'un processus aléatoire de transition entre des états inobservables (les états cachés) et, à un second niveau, d'un autre processus aléatoire qui, dans chaque état, engendre des valeurs observables (appelées observations).

L'articulation de ces deux niveaux confère aux modèles de type HMM une grande flexibilité et ces modèles basés sur des transitions entre états sont bien adaptés pour rendre compte de processus organisés dans le temps, ce qui explique l'importance considérable de ces modèles. **[3] + [20]**

Un processus aléatoire gère la transition d'état à état, mais l'état du système n'est pas observable (il est « caché »), on ne voit que les émissions de cet état : les observations.

Autrement dit, au temps ***t*** le système est dans l'état q_t (invisible) et émet l'observation O_t (visible). Ces observations O_t peuvent prendre leurs valeurs dans un ensemble fini de valeurs discrètes ou de symboles (par exemple les caractères d'un alphabet fini), ou dans un ensemble continu et infini (fréquence d'un signal, température), dans ce cas le principe reste le même, avec des distributions de probabilité continue. **[18] + [20]**

II.2.1. Les éléments d'un HMM :

Un modèle de Markov caché, est défini par une structure composée d'états, de transitions et par un ensemble de distribution de probabilités sur les transitions. A cette structure proche des automates probabilistes, on adjoint un alphabet et, pour chaque état, une probabilité d'émission des différents symboles de l'alphabet. Un HMM peut donc être défini par le quadruplet **(S, V, A, B)** tel que :

- **S** est un ensemble de N états ;
- **V** est un alphabet de M symboles, ceux-ci pouvant être vectoriels ;
- **A = {a_{ij}} S x S → [0,1]**, la matrice des probabilités de transitions entre les états S_i et S_j. a_{ij} représente la probabilité que le modèle évolue de l'état i vers l'état **j : a_{ij} = A (i, j) = P (q_{t+1} = s_j |q_t = s_i) ∀ i, j ∈ [1,…, N] ∀ t ∈ [1,…, T].** avec : $a_{ij} \geq 0$ ∀ i, j et $\sum_{j=1}^{n} aij = 1$.
- **B = S x V → [0,1]**, une matrice indiquant les probabilités d'émission associées aux états (la matrice des probabilités d'observation des symboles **M**); on note **P (y_t|q_t)** la probabilité d'émettre à l'instant **t** le symbole y_t à partir de l'état q_t. **[16] + [19] + [20]**

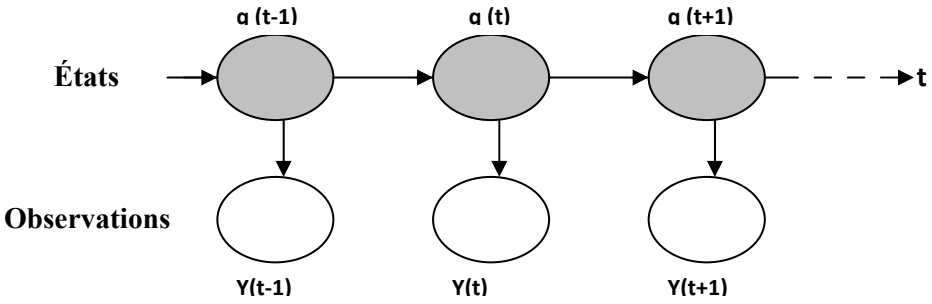


Figure II.2 : Représentation d'un HMM. [16]

Quelques notations utiles :

- **Λ= (A; B; Π)** Un HMM
- ***O*= {O_1,…, O_T}** le vecteur de ***T*** observations émises.
- **M** : La taille de l'alphabet des observations
- **N** : nombre d'états du modèle
- **b_j(k), j ∈[1, n], k ∈[1, M]** Un élément de **B**, elle représente la probabilité que l'on observe le symbole v_k alors que le modèle se trouve dans l'état **j**.
- **T** : la longueur d'une séquence observée.
- **O = O_1,…, O_t,..,O_T** Une séquence observée avec **Ot ∈ V**

- **O [i : j] = O_i… Oj** Une sous-séquence de **O**
- **q_1,…, q_t,… q_T** avec **$q_t \in S$** La suite des états qui a émis une séquence
- **P(O|Λ)** La probabilité que le HMM ait émis la séquence **O**
- **O = O_1,…,O_m** Un ensemble d'apprentissage composé de m séquences
- **P (Λ| O)** La probabilité que l'ensemble de séquences **O** ait été émis par le HMM **Λ**.
- **Π= {π_i}** : Le vecteur des probabilités initiales du HMM.

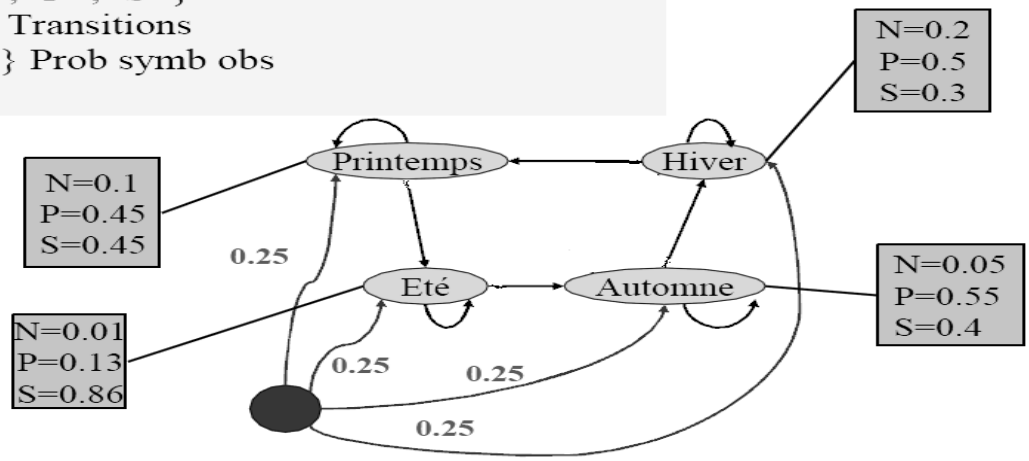
  $\pi_i = P(q1 = Si), 1 \leq i \leq N$

Le vecteur **π** gère les probabilités de transitions depuis un état d'entrée virtuel où se trouve le système en $t = 0$. Pour tout état **i**, π_i est la probabilité que l'état de départ du HMM soit l'état **i** :

π_i = P (q1 = si) 1 ≤ i ≤ n. avec : $\pi_i \geq 0$ ∀ i et $\sum_{i=1}^{n} \pi i = 1$ [19] + [21]

II.2.2. Un exemple de HMM [20]

S={' Printemps ', 'Eté ', ' Automne ', 'Hiver '}
Σ={' N ', 'P ', 'S '}
A={$a_{i,j}$} Transitions
B={$b_j(.)$} Prob symb obs



II.3. GENERATION D'UNE SEQUENCE PAR UN HMM

Un HMM peut être vu comme un processus permettant d'engendrer une séquence; inversement, on peut considérer une séquence comme une suite d'observations sur un HMM en fonctionnement.

En se plaçant du premier point de vue, la génération d'une séquence peut se décrire par l'algorithme suivant qui est une procédure itérative gérée par des tirages aléatoires. **[19] + [20]**

Algorithme : les étapes d'une séquence générée par un HMM

t ←1

Choisir l'état initial $q_1 = s_i$ avec la probabilité π_i

Tant que t ≤ T faire

Choisir l'observation $o_t = v_k$ avec la probabilité $b_i(k)$

Passer à l'état suivant $q_{t+1} = s_j$ avec la probabilité a_{ij}

t ←t + 1

fin tant que

[19] + [20]

Notons ici qu'une séquence donnée peut en général être engendrée de plusieurs façons distinctes par un HMM. Il a été montré que plusieurs séquences d'état peuvent engendrer la même observation. Il serait alors intéressant de calculer la probabilité d'émission de cette observation pour chacun des chemins possibles, le calcul de ces différentes probabilités est l'un des principaux problèmes des HMM. C'est le sujet que nous allons développer au paragraphe suivant.

II.4. LES TROIS PROBLEMES DES HMMS

Comme on vient de le voir, les HMM permettent de modéliser des séquences d'observations discrètes ou continues. Ils permettent de résoudre principalement trois grands problèmes:

- ✓ ***L'apprentissage:*** étant donné un ensemble de séquences, déterminer les paramètres d'un modèle de Markov caché d'architecture fixée pour maximiser les probabilités d'émission de ces séquences.
- ✓ ***L'évaluation d'une séquence:*** étant donné un modèle de Markov caché et une séquence, déterminer quelle est la probabilité d'émission de cette séquence suivant ce modèle.
- ✓ ***La recherche du chemin le plus probable:*** étant donné un modèle de Markov caché et une séquence, déterminer la suite d'états qui maximise la probabilité d'observation de cette séquence. **[19] + [20] + [22]**

II.4.1. Apprentissage

Afin d'effectuer la reconnaissance, il faut avoir un modèle de la séquence à reconnaître que l'on pourra ensuite comparer aux séquences inconnues. Pour construire ce modèle, on pourrait utiliser les connaissances a priori dont on dispose sur le système. Celles-ci sont généralement insuffisantes pour donner des résultats convaincants. Nous allons donc plutôt faire appel à une méthode d'apprentissage statistique. Celle-ci va permettre de modifier les probabilités des différentes transitions du HMM afin de le rapprocher du modèle recherché.

Étant donné un modèle de Markov caché d'architecture fixée, l'apprentissage vise à déterminer ses paramètres (matrice de probabilités de transitions, matrice de probabilités d'émission et matrice de probabilités initiales). Cet apprentissage se fait par une approche rigoureuse qui consiste à chercher les paramètres **λ** de **Λ** qui maximisent :

$$P(\mathbf{O}|\Lambda) = \prod_{k=1}^{K} P\left(Y^k \middle| \Lambda\right) \qquad \textbf{[19] + [20]}$$

Il faut en effet que **Λ** ait une probabilité maximale d'émettre les séquences d'apprentissage. Cet entraînement, qui suit le principe du maximum de vraisemblance, s'effectue suivant l'algorithme d'entraînement de **Baum-Welch**.

Principe :

Supposons disposer d'un ensemble de séquences $\mathbf{O} = \{\mathbf{O}^1, \ldots, \mathbf{O}^m\}$ dont l'élément courant est noté $\mathbf{O}^k$. Le but de l'apprentissage est de déterminer les paramètres d'un HMM d'architecture fixée: $\mathbf{\Lambda = (A, B, \pi)}$, qui maximisent la probabilité $\mathbf{P(O|\Lambda)}$. Comme on suppose les séquences d'apprentissage tirées indépendamment, on cherche donc à maximiser :

$$P(\mathrm{O}|\Lambda) = \prod_{k=1}^{m} P\left(O^k \middle| \Lambda\right)$$

L'idée est d'utiliser une procédure de réestimation qui affine le modèle petit à petit selon les étapes suivantes :

- Choisir un ensemble initial $\mathbf{\Lambda_0}$ de paramètres;
- Calculer $\mathbf{\Lambda_1}$ à partir de $\mathbf{\Lambda_0}$, puis $\mathbf{\Lambda_2}$ à partir de $\mathbf{\Lambda_1}$, etc.
- Répéter ce processus jusqu'à un critère de fin.
- Pour chaque étape **p** d'apprentissage, on dispose de $\mathbf{\Lambda_p}$ et on cherche un $\mathbf{\Lambda_{p+1}}$ qui doit vérifier : $P(\mathrm{O}|\Lambda_{p+1}) \geq P(\mathrm{O}|\Lambda_p)$

Soit: $\boxed{\prod_{k=1}^{m} P\left(O^k \middle| \Lambda_{p+1}\right) \geq \prod_{k=1}^{m} P\left(O^k \middle| \Lambda_p\right)}$ **[19] + [20]**

$\mathbf{\Lambda_{p+1}}$ doit donc améliorer la probabilité de l'émission des observations de l'ensemble d'apprentissage. Pour calculer $\mathbf{\Lambda_{p+1}}$ à partir de $\mathbf{\Lambda_p}$,on fait un comptage de l'utilisation des transitions A et des distributions **B** et π du modèle $\mathbf{\Lambda_p}$ quand il produit l'ensemble **O**. Si cet ensemble est assez important, ces fréquences fournissent de bonnes approximations a posteriori des distributions de probabilités **A**, **B** et **Λ** et sont utilisables alors comme paramètres du modèle $\mathbf{\Lambda_{p+1}}$ pour l'itération suivante.

La méthode d'apprentissage consiste donc dans ce cas à regarder comment se comporte le modèle défini par $\mathbf{\Lambda_p}$ sur **O**, à réestimer ses paramètres à partir des mesures prises sur **O**, puis à recommencer cette réestimation jusqu'à obtenir une convergence. Dans les calculs qui suivent, on verra apparaître en indice supérieur la lettre k quand il faudra faire référence à la séquence d'apprentissage concernée. L'indice **p**, qui compte les passes d'apprentissage, sera omis : on partira d'un modèle noté simplement **Λ** et on calculera celui qui s'en déduit. **[19] + [20]**

Les formules de réestimation :

On définit $\boxed{\xi_t^k(i,j)}$ comme la probabilité, Etant donnés une phrase O^k et un HMM **Λ**, que ce soit l'état S_i qui ait émis la lettre de rang t de O^k l'état S_i qui ait émis celle de rang **t+1**. Donc :

$$\boxed{\xi_t^k(i,j) = p(q_t = s_i, q_{t+1} = s_j \middle| o^K, \Lambda)}$$

Ce qui se réécrit : $$\boxed{\xi_t^k(i,j) = \frac{P(q_t = s_i, q_{t+1} = s_j, O^K \middle| \Lambda)}{P(O^K \middle| \Lambda)}}$$

Par définition des fonctions forward-backward, on en déduit :

$$\boxed{\xi_t^k(i,j) = \frac{\alpha_t^k(i) a_{ij} b_j(O_{t+1}^k) \beta_{t+1}^k(j)}{P(O^k \middle| \Lambda)}}$$

On définit aussi la quantité $\gamma_t^k(i)$ comme la probabilité que la lettre de rang t de la phrase O^k soit émise par l'état S_j.

$$\gamma_t^k(i) = P(q_t = s_i | o^k, \Lambda)$$

Soit :

$$\gamma_t^k(i) = \sum_{j=1}^{n} P(q_t = s_i, q_{t+1} = s_j | o^k, \Lambda) = \frac{j = \sum_{j=1}^{n} P(q_t = s_i, q_{t+1} = s_j | o^k, \Lambda)}{P(O^k | \Lambda)}$$

On a la relation : $\gamma_t^k(i) = \sum_{j=1}^{n} \xi_t^k(i,j) = \frac{\alpha_t^k(i)\beta_t^k(i)}{P(O^k|\Lambda)}$ **[19] + [20]**

Remarque : Les calculs de **α** et **β** seront présentés par la suite dans la section (1.5.2).

Le nouveau modèle HMM se calcule à partir de l'ancien en réestimant **π, A** et **B** par comptage sur la base d'apprentissage. On mesure les fréquences :

- $\overline{a_{ij}} = \frac{Nombre\ \ de\ fois\ où\ la\ transition\ \ de\ Si\ à\ Sj\ a\ été\ utilisée}{Nombre\ \ de\ transition\ s\ éffectuées\ \ à\ \ partir\ \ deSi}$
- $\overline{b_{j(k)}} = \frac{Nombre\ \ de\ fois\ où\ le\ HMM\ \ s'est\ trouvé\ \ dans\ l'état\ \ Sj\ en\ observant\ \ Uk}{Nombre\ \ de\ fois\ où\ le\ HMM\ \ s'est\ trouvé\ \ dans\ l'état\ \ Sj}$
- $\overline{\pi_i} = \frac{Nombre\ \ de\ fois\ où\ le\ HMM\ \ s'est\ trouvé\ dans\ l'état\ Si\ en\ émmet\ le\ 1^{er}\ \ symbole\ d'une\ \ phrase}{Nombre\ \ de\ fois\ où\ le\ HMM\ \ a\ émis\ le\ \ premier\ \ symbole\ d'une\ \ phrase}$

Compte tenu de ces définitions : $\overline{\pi_i} = \frac{1}{N}\sum_{k=1}^{N}\gamma_1^k(i)$

$$\overline{a_{ij}} = \frac{\sum_{k=1}^{N}\left(\sum_{t=1}^{|O^k|-1}\xi_t^k(i,j)\right)}{\sum_{k=1}^{N}\sum_{t=1}^{|O^k|-1}\gamma_t^k(i)} \qquad \overline{b}_j(l) = \frac{\sum_{k=1}^{N}\sum_{t=1\ avec O_t^k=v_l}^{|O^k|-1}\gamma_t^k(j)}{\sum_{k=1}^{N}\sum_{t=1}^{|O^k|-1}\gamma_t^k(j)}$$

La suite des modèles construits par l'algorithme de Baum-Welsh vérifie la relation cherchée : **P (O $|\Lambda_{p+1}$) ≥ P (O j $|\Lambda_p$). [19] + [20]**

Remarque:

- Le choix du modèle initial influe sur les résultats ; par exemple, si certaines valeurs de **A** et **B** sont égales à **0** au départ, elles le resteront jusqu'à la fin de l'apprentissage.
- L'algorithme converge vers des valeurs de paramètres qui assurent un maximum local de **P (O|Λ)**. Il est donc important, si l'on veut être aussi prés que possible du minimum global, de bien choisir la structure et l'initialisation.
- Le nombre d'itérations est fixé empiriquement. L'expérience prouve que, si le point précédent a été correctement traité, la stabilisation des paramètres ne correspond pas à un sur apprentissage: il n'y a donc en général pas besoin de contrôler la convergence par un ensemble de validation. Mais cette possibilité est évidemment toujours à disposition. **[19] + [20]**

Algorithme : Algorithme de Baum-Welch

Fixer des valeurs initiales (A, B, π)

On définit le HMM de départ comme $\Lambda_0 = (A_0, B, \pi)$.

p $\leftarrow$0

Tant que la convergence n'est pas réalisée faire

On possède le HMM Λ_p.

On calcule pour ce modèle, sur l'ensemble d'apprentissage, les valeurs :

$$\xi(i,j),\ \gamma_t(i) \qquad 1 \le i,j \le n \quad 1 \le t \le T-1$$

On en déduit $\bar{\pi}, \bar{A}, \bar{B}$ défini par en utilisant les formules de réestimation.

Le HMM courant est désormais $\Lambda_p = (\bar{\pi}, \bar{A}, \bar{B})$.

p$\leftarrow$ p + 1

Fin tant que

Exemple : [19] + [20]

En partant du HMM $\mathbf{\Lambda_0}$ défini par les paramètres suivants :

$$A = \begin{pmatrix} 0.45 & 0.35 & 0.20 \\ 0.10 & 0.50 & 0.40 \\ 0.15 & 0.25 & 0.60 \end{pmatrix} \qquad B = \begin{pmatrix} 1.0 & 0.0 \\ 0.5 & 0.5 \\ 0.0 & 1.0 \end{pmatrix} \qquad \pi = \begin{pmatrix} 0.5 \\ 0.3 \\ 0.2 \end{pmatrix}$$

On peut calculer que, s'il émet sur l'alphabet à deux lettres **V = {a, b}**, on a:

P (a b b a a $|\Lambda_0$) = 0.0278

Si on prend comme ensemble d'apprentissage cette seule phrase, l'application de l'algorithme de **Baum-Welsh** doit augmenter sa probabilité de reconnaissance.

Après une réestimation, on trouve le HMM $\mathbf{\Lambda_1}$:

$$A = \begin{pmatrix} 0.346 & 0.365 & 0.289 \\ 0.159 & 0.514 & 0.327 \\ 0.377 & 0.259 & 0.364 \end{pmatrix} \qquad B = \begin{pmatrix} 1.0 & 0.0 \\ 0.631 & 0.369 \\ 0.0 & 1.0 \end{pmatrix} \qquad \pi = \begin{pmatrix} 0.656 \\ 0.344 \\ 0.0 \end{pmatrix}$$

P (a b b a a | Λ_1)=0.0529

Après 15 itérations :

$$A = \begin{pmatrix} 0.0 & 0.0 & 1.0 \\ 0.212 & 0.788 & 0.0 \\ 0.0 & 0.515 & 0.485 \end{pmatrix} \quad B = \begin{pmatrix} 1.0 & 0.0 \\ 0.969 & 0.031 \\ 0.0 & 1.0 \end{pmatrix} \quad \pi = \begin{pmatrix} 1.0 \\ 0.0 \\ 0.0 \end{pmatrix}$$

$$P(a\ b\ b\ a\ a \mid \Lambda_{15}) = 0.2474$$

Après cent cinquante itérations, la convergence est réalisée. La Figure II.5 et II.6 décrivent les résultats, que l'on peut donner aussi sous la forme suivante :

$$A = \begin{pmatrix} 0.0 & 0.0 & 1.0 \\ 0.18 & 0.82 & 0.0 \\ 0.0 & 0.5 & 0.5 \end{pmatrix} \quad B = \begin{pmatrix} 1.0 & 0.0 \\ 1.0 & 0.0 \\ 0.0 & 1.0 \end{pmatrix} \quad \pi = \begin{pmatrix} 1.0 \\ 0.0 \\ 0.0 \end{pmatrix}$$

$$P(a\ b\ b\ a\ a \mid \Lambda_{150}) = 0.2500$$

Etat	**1**	**2**	**3**
P(a)	**1**	**1**	**0**
P(b)	**0**	**0**	**1**

Figure II.3 : la matrice B de cet HMM

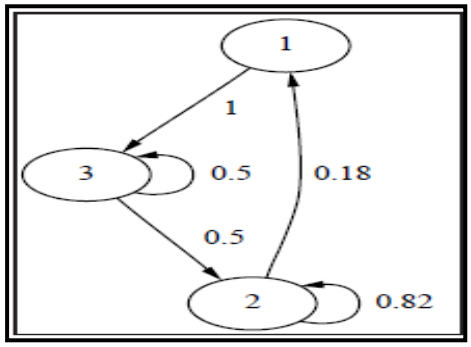


Figure II.4: Le HMM entrainé sur une seule phrase, après convergence. Le seul état initial possible est l'état 1. [19] + [20]

II.4.2. Évaluation de la probabilité d'observation d'une séquence

Le problème est de calculer la probabilité d'observation de la séquence d'observation étant donner un HMM $P(O|\Lambda)$. Il existe plusieurs techniques pour évaluer cette probabilité d'observation d'une séquence de longueur t :

- ✓ **L'évaluation directe**: Remarquons d'abord que la probabilité de la suite d'observations **O**, étant donné le modèle **Λ** est égale à la somme sur tous les suites d'états possibles **Q** des probabilités conjointes de **O** et de **Q**, donc

P(O|Λ) doit être évaluée pour toutes les séquences d'états possibles. Dans ce cas, il faut énumérer toutes les suites d'états de longueur **t** ce qui entraîne un coût en **$2O(n^t)$.**

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

- Pour une séquence d'états donnée :

$$P(O|Q,\lambda) = \sum_{i=1}^{n} \mathbf{b_{q1}(O_1)\, b_{q2}(O2)\ldots b_{qt}(O_t)}$$
$$P(Q|\lambda) = \pi_{\mathbf{q1}}\, \mathbf{a_{q1q2}\, a_{q2q3}\ldots a_{qt\,-\,1qt}}$$

- Alors : $$P(\mathbf{O}|\lambda) = \sum_{q1q2\ldots qT} \pi_{q1} a_{q1q2} b_{q1}(o_1) a_{q2q3} b_{q2}(o_2) a_{q2q3}\ldots a_{qT\,-\,1qT} b_{qT}(o_T)$$

- Avec T observations et N états dans le modèle :
  - N^T possibles séquences d'états.
  - Approximativement $2TN^T$ opérations requises.
  - Pour T=100 et un HMM à 5 états≈$200*5^{100}$ opérations. **[19] + [20]**

✓ **L'évaluation par les fonctions forward- backward** : C'est l'algorithme le plus efficace pour traiter ce problème. Son coût est en **$O(n^2T)$**. Dans cette approche, on remarque que l'observation peut se faire en deux temps : d'abord, l'émission du début de l'observation **O (1 : t)** en aboutissant à l'état $\mathbf{q_i}$ au temps **t**, puis, l'émission de la fin de l'observation **O(t +1 : T)** sachant que l'on part de $\mathbf{q_i}$ au temps **t**. Ceci pose, la probabilité de l'observation est donc égale à : **[19] + [20] + [23]**

$$P(\mathbf{O}|\Lambda) = \sum_{i=1}^{n} \alpha_t(i)\beta_t(i)$$

Où α_t**(i)** est la probabilité d'émettre le début **O (1 : t)** et d'aboutir à q_i à l'instant t, et B_t **(i)** est la probabilité d'émettre la fin **O(t+1 :T)** sachant que l'on part de q_i à l'instant t. le calcul de **α** se fait avec **t** croissant tandis que le calcul de **β** se fait avec **t** décroissant, d'où l'appellation **forward-backward**.

- **le calcule de α :** On a : $a_t(i) = P(O_1O_2 \ldots \ldots O_t, q_t = s_i | \Lambda)$

α_t **(i)** ce calcule par l'algorithme 13.2 qui exprime que pour émettre le début de l'observation **O(t+1 :T)** et aboutir dans l'état S_j au temps **t+1**, on doit nécessairement être dans l'un des états S_i à l'instant **t**. cette remarque permet d'exprimer α_{t+1}**(j)** en fonction des α_t **(i)** et d'utiliser un algorithme de programmation dynamique pour le calcul les α_t **(i)** pour tout **i**, puis des α_{t+1} **(i)** pour tout **i**, etc.

Ce calcul a une complexité en **θ (n² T). [19] + [20] + [23]**

- **Le calcul de β**

De manière analogue, β_t **(i)** se calcule par l'algorithme 13.2.

Le calcul de β est lui aussi en **θ (n² T).**

$\alpha_t(i) = P(O_1O_2 \ldots O_t, q_t = i | \lambda)$

1. **Initialisation :** $\alpha_1(i) = \pi i b i(O_1),\ 1 \le i \le N$

2. **Induction :** $\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a i_j\right] b_j(O_{t+1}),\ \ 1 \le t \le T-1 \quad 1 \le j \le N$

3. **Terminaison :** $P(\mathbf{O}|\lambda) = \sum_{i=1}^{N} \alpha_T(i)$ **[19] + [20] + [23]**

Algorithme : calcul de la fonction forward α

pour i=1, n faire
$\alpha_T(i) \leftarrow \pi_i b_i(O_1)$
fin pour
t$\leftarrow 1$
tant que t <T faire
j $\leftarrow 1$
tant que j $\leq$ **n faire**

$$\alpha_{t+1}(j) \leftarrow \left[\sum_{i=1}^{n} \alpha_t a_{ij}\right] b_j(O_{t+1})$$

j $\leftarrow j+1$
fin tant que
t$\leftarrow$ **t-+1**
fin tant que

$$P(O|\Lambda) \leftarrow \sum_{i=1}^{n} \alpha_T(i)$$

[19] + [20]

Algorithme : calcul de la fonction backward β

pour i=1, n faire
$\beta_T(i) \leftarrow 1$
t$\leftarrow$T
tant que t > 1 faire
j $\leftarrow 1$
tant que j $\leq$ **n faire**

$$\beta_t(i) \leftarrow \sum_{j=1}^{n} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)$$

j $\leftarrow j-1$
fin tant que
t$\leftarrow$ **t- 1**
fin tant que

$$P(O|\Lambda) \leftarrow \sum_{i=1}^{n} \beta_T(i)$$

✓ **Le calcul de la probabilité d'observation:**

Finalement, la probabilité d'observation d'une séquence est obtenue en prenant les valeurs de α et de β à un instant t quelconque: $\boxed{P(O|\Lambda) \leftarrow \sum_{i=1}^{n} \alpha_T(i)\beta_T(i)}$.

Cependant, on utilise le plus souvent les valeurs obtenues pour deux cas particuliers (t=0) ou (t=T), ce qui donne :

$$\boxed{P(O|\Lambda) \leftarrow \sum_{i=1}^{n} \alpha_T(i) = \sum_{i=1}^{n} \pi_i \beta_0(i)}$$ **[19] + [20]**

II.4.3. Décodage : Le calcul du chemin optimal: l'algorithme de Viterbi

Il s'agit de déterminer le meilleur chemin correspondant à l'observation, c'est-à-dire de trouver dans le modèle Λ la meilleure suite d'états Q, qui maximise la quantité :

P (Q, O|Λ)

Pour trouver **Q=($q_1, q_2, \ldots\ldots\ldots, q_r$)** pour une séquence d'observations **O($O_1, O_2, \ldots\ldots, O_r$),** on définit la variable intermédiaire $\delta_t(i)$ comme la probabilité du meilleur chemin amenant à l'état s_i à l'instant t, en étant guidé par les t premières observations :

$$\delta_t(i) = \mathrm{Max}_{q_1,q_2,\ldots,q_{t-1}} P(q_1, q_2, \ldots\ldots, q_t = s_i, O_1, O_2, \ldots\ldots, O_t|\Lambda)$$

Par récurrence, on calcule :

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

En gardant trace, lors du calcul, de la suite d'états qui donne le meilleur chemin amenant à l'état s_i à **t** dans un tableau ψ**.**

On utilise une variante de la programmation dynamique, l'algorithme de ***Viterbi*** pour formaliser cette récurrence. Il fournit en sortie la valeur P^* de la probabilité de l'émission de la séquence par la meilleure suite d'états $(q_1^*, \ldots\ldots, q_T^*)$.

La fonction ***Argmax*** permet de mémoriser l'indice **i**, entre 1 et n, avec lequel on atteint le maximum des quantités $(\delta_{t-1}(i)a_{ij})$. Le coût des opérations est également en **θ(n²T)** **[19] + [20] +[24]**

Algorithme : Algorithme de Viterbi

pour i=1, n faire
δ_1 (i) $\leftarrow \pi_i b_i(O_1)$

ψ_1(i)$\leftarrow 0$

fin pour
t$\leftarrow 2$
tant que t <T faire
j $\leftarrow 1$

tant que j $\leq$ **n faire**

$$\delta_t(j) \leftarrow Max\left[\delta_{t-1}(i)a_{ij}\right]b_j(O_t) \qquad 1 \le i \le n$$
$$\psi_t(j) \leftarrow ArgMax\left[\delta_{t-1}(i)a_{ij}\right] \qquad 1 \le i \le n$$

j $\leftarrow j+1$

fin tant que

$$P^* \leftarrow Max\left[\delta_T(i)\right] \qquad 1 \le i \le n$$
$$q_T^*(j) \leftarrow ArgMax\left[\delta_{t-1}(i)\right] \quad 1 \le i \le n$$

t$\leftarrow$ **T**
fin tant que

tant que t $\geq$**1 faire**

$$q_t^* \leftarrow \psi_t + 1(q_{t+1}^*)$$

t $\leftarrow t-1$

fin tant que

II.5.UTILISATION DES HMMS POUR LA CLASSIFICATION DE SEQUENCES

Les bases des modèles de Markov cachés étant posées, nous allons maintenant pouvoir les utiliser pour la reconnaissance de séquences.

On veut classifier des séquences en un nombre ***nc*** de catégories. Pour cela, on crée ***nc*** HMMs et on entraîne chacun d'entre eux avec un ensemble de séquences d'apprentissage représentatif d'une classe donnée. **[16] + [25]**

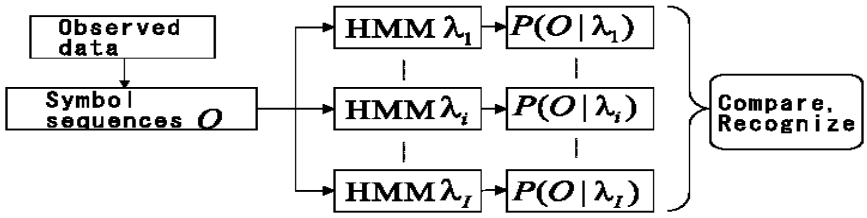


Figure II.5 : Organigramme de la méthode de classification de séquences par les HMMs. [16]

II.6. Les avantages et les inconvénients des HMMs : [20]

Avantages	**Inconvénients**
- Base mathématique solide pour comprendre son fonctionnement. - Variabilité de la forme. - Alignement temporel incorporé systématiquement. - Reconnaissance réalisée par un simple calcul de probabilité cumulée. - Décision globale sans obligation d'utiliser des seuils. - Séparation franche entre données et algorithmes.	- Le choix à priori de la typologie des modèles (nombre d'états, transitions autorisées et règles de transitions). - Dégradation des performances si l'apprentissage n'est pas suffisant.

Résumé

Nous avons présenté dans ce chapitre les principes des HMM, c'est une méthode qui a été utilisée, au début pour la prévision et puis elle a été appliquée dans le domaine de la reconnaissance et de la classification en général.

Nous avons défini les notions de base des HMMs comme les états cachés, l'alphabet des symboles, la matrice de transition et la matrice d'émission qui sont les éléments brique de ces modèles. Dans le chapitre suivant nous implémentons un classifieur des arythmies cardiaques en se basant sur le principe de cette technique.