Moteur de reconnaissance GA/HMM 4

Les modèles de Markov cachés (HMM) sont des outils statistiques permettant de modéliser des phénomènes stochastiques. Ces modèles sont utilisés dans de nombreux domaines (Cappé 2001) tels que la reconnaissance et la synthèse de la parole, la biologie, l'ordonnancement, l'indexation de documents, la reconnaissance d'images, la prédiction de séries temporelles, … Pour pouvoir utiliser ces modèles efficacement, il est nécessaire d'en connaitre les principes.

L'amélioration de l'apprentissage des HMM à l'aide de métaheuristique à base de population est l'objet de ce chapitre. Ce chapitre a donc pour objectif d'établir les principes, les notations utiles et les principaux algorithmes qui constituent la théorie des HMM.

A cet effet, nous commençons ce chapitre en définissant de que sont les HMM leur principes, et nous présentons les algorithmes classiques des HMM : *Forward, Backkward* et de *Viterbi*.

4.1 Modèles de Markov Cachés

4.1.1 Définition

Un modèle HMM est défini comme un ensemble d'états, chacun d'entre eux associé à une distribution de probabilité (en général multidimensionnelle). Les transitions entre les états sont régies par un ensemble de probabilités appelées probabilités de transition Dans un état particulier, un résultat ou observation peut être généré conformément à la distribution de probabilité associée. Par opposition à un modèle de Markov classique ou l'état est directement observable par un observateur externe, dans un modèle HMM, l'état n'est pas directement observable et seulement des variables influencées par l'état le sont. Les états sont donc cachés, d'ou le nom de modèle de Markov caché.

Un HMM (représenté dans la figure 4.1) est défini par :

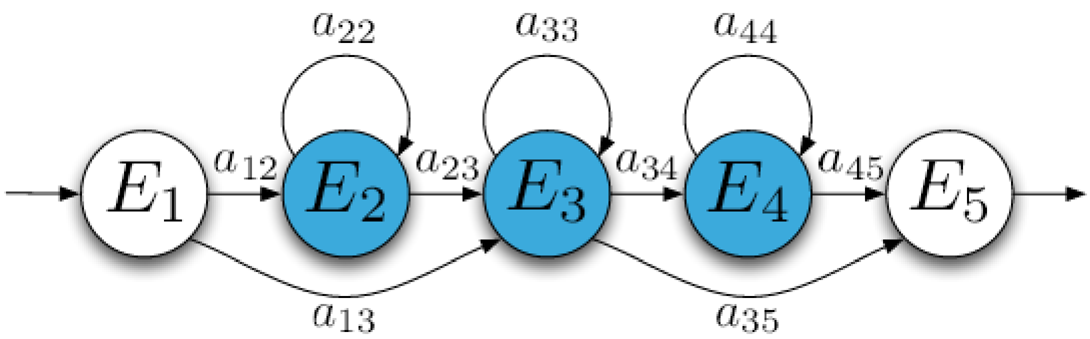

Figure 4.1 – HMM à 5 états dont 3 émetteurs.

- N : le nombre d'états du modèle. Les états seront notés xi pour $1 \leq i \leq N$
- M : le nombre de symboles d'observation. Dans le cas ou les observations sont continues, M est infini. Dans notre notation, les symboles d'observation de l'alphabet sont notés $Y = \{y_j\}$ pour $1 \leq j \leq M$.
- π : le vecteur de probabilités initiales des états. Concernant cet élément, un autre type de HMM utilise des états start et end et non une distribution d'états initiaux. Ce type d'HMM est notamment employé en bioinformatique.
- A : la matrice de transition ou sont définies les probabilités de transition entre les états. Ces probabilités $A = \{a_{ij}\}$ sont définies comme :

$$a_{ij} = p(x_t = i | x_{t-1} = j), 1 \leq i, j \leq N \tag{4.1}$$

avec x_t désigne l'état courant à l'instant t. Les probabilités de transition a_{ij} doivent satisfaire les contraintes stochastiques :

$$a_{ij} \geq 0 \; et \; \sum_{j=1}^{N} a_{ij}, \;\; 1 \leq i, j \leq N \tag{4.2}$$

- B : la matrice de confusion (ou matrice d'observation) contenant les probabilités d'observation (ou probabilités d'émission) B = {bj(k)} associées aux états. Ces probabilités sont définies comme :

$$b_j(k) = p(y_t = v_k | x_t = j), 1 \leq j \leq N, 1 \leq k \leq M \tag{4.3}$$

avec v_k dénote le k^{eme} symbole d'observation dans l'alphabet, et yt le vecteur de paramètres actuel (ou simplement observation actuelle) à l'instant t. Les probabilités d'observation satisfont aussi les contraintes stochastiques. Dans le cas d'observations continues, des densités de probabilités continues sont à utiliser.

Pour dénoter un modèle HMM le triplet λ = (π, A, B) est généralement utilisé. Il est important de noter que chaque probabilité dans la matrice de transition (de confusion) est indépendante du temps. En d'autres termes, les matrices ne changent pas dans le temps quand le système évolue. En pratique, ceci est l'une des suppositions les plus discutables des modèles de Markov à propos des processus réels.

Dans la théorie des HMMs, des hypothèses sont faites pour une docibilité mathématique et informatique :

- Hypothèse markovienne : concernant la définition des éléments de la matrice de transition A, la probabilité de transition vers un état ne dépend que de l'état actuel et non des états rencontrés précédemment. Ainsi, la séquence des états constitue une chaîne de Markov simple.
- Hypothèse de stationnarité : comme nous l'avons déjà évoqué, la matrice des probabilités de transition est indépendante de l'actuel temps, dans lequel les transitions prennent place.

  Mathématiquement :

$$p\left(x_{t_1+1}=j \middle| x_{t_1}=i\right)=p\left(x_{t_2+1}=j \middle| x_{t_2}=i\right) pour\ tout\ t_1\ et\ t_2, \qquad (4.4)$$

- Hypothèse d'indépendance des sorties (observations) : l'observation courante est statiquement indépendante des observations précédentes. Mathématiquement, cette hypothèse peut être formulée pour un HMM λ par :

$$p(Y|x_1,x_2,\dots,x_t,\lambda)=\prod_{t=1}^{T} p(y_t|x_t,\lambda). \qquad (4.5)$$

4.1.2 Utilisation et algorithmes

Une fois qu'un système est décrit comme un HMM, trois problèmes doivent être résolus. Les deux premiers sont des problèmes qu'on peut associer à la reconnaissance : détermination de la probabilité d'une séquence observée étant donné un HMM (c'est le problème de l'évaluation); et, étant donné un modèle HMM et une séquence d'observations, déterminer quelle séquence d'états cachés dans le modèle est la plus probable (c'est le problème de décodage). Le troisième problème est la génération d'un HMM étant donné une séquence d'observations (c'est le problème d'apprentissage).

4.1.2.1 Evaluation et l'algorithme de Forward

Ce problème se pose notamment quand nous avons, par exemple, plusieurs HMMs décrivant différents systèmes, et une séquence d'observations. Nous voulons ainsi connaître quel est le HMM ayant la plus forte probabilité d'avoir généré cette séquence. En d'autres termes, pour un modèle λ = (π, A, B) et une séquence d'observations Y = y1, y2, ..., yT, nous avons à calculer la probabilité $P(Y|\lambda)$. Un calcul de cette probabilité implique un nombre d'opérations de l'ordre de N^T. Heureusement, une autre méthode, ayant une complexité inférieure, existe. Cette méthode utilise une variable intermédiaire appelée variable "avant" ou forward; d'ou le nom de l'algorithme Forward (ou "avant").

Algorithme Forward : Cet algorithme est utilisé pour calculer la probabilité d'une séquence d'observation de longueur T :

$$\boldsymbol{Y = y_1, y_2, \dots, y_T} \tag{4.6}$$

avec chaque y est un élément de l'ensemble observable. La variable intermédiaire $\alpha_t(i)$ est définie comme la probabilité de la séquence d'observation partielle $\boldsymbol{Y^t = y_1, y_2, \dots, y_t\ t \leq T}$, qui se termine à l'état i. Les probabilités intermédiaires (ou partielles) sont calculées de manière récursive en calculant premièrement ces probabilités pour tous les états à t = 1.

$$\alpha_1(j) = \pi(j).\, b_j(1), pour\ 1 \leq j \leq N \tag{4.7}$$

Ensuite, pour chaque instant, t = 2, ..., T, les probabilités partielles sont calculées pour chaque état par la relation récursive suivante :

$$\alpha_{t+1}(j) = \sum_{i=1}^{N} \big(\alpha_t(i) a_{ij}\big) b_j(t),\ \ pour\ 1 \leq j \leq N,\ \ 1 \leq t \leq T-1 \tag{4.8}$$

Avec cette relation, nous pouvons alors calculer la probabilité intermédiaire à l'instant T pour chaque état j, $\alpha_T(j)$. Et finalement, la somme de toutes les probabilités partielles à l'instant T fournit la probabilité requise :

$$p(Y|\lambda) = \sum_{i=1}^{N} \alpha_T(i) \tag{4.9}$$

Pour récapituler, chaque probabilité partielle (à l'instant t > 2) est calculée à partir de tous les états précédents. De façon similaire, nous pouvons définir une variable « arrière » ou backward $\beta_t(i)$ comme la probabilité de la séquence d'observation partielle $\boldsymbol{y_{t+1}, y_{t+2}, \dots, y_T}$, étant donné que l'état courant est i. Pour calculer les$\beta_t(i)$, il existe aussi, comme pour les$\alpha_t(i)$, une relation récursive :

$$\beta_t(i) = \sum_{j=1}^{N} \beta_{t+1}(j) a_{ij}\, b_j(t+1),\ \ pour\ 1 \leq i \leq N,\ \ 1 \leq t \leq T-1 \qquad (4.10)$$

Avec

$$\beta_T(i) = 1, pour\ 1 \leq i \leq N. \qquad (4.11)$$

Si nous cherchions un lien entre les deux variables intermédiaires $\beta_t(i)$ et $\alpha_t(i)$, nous pouvons remarquer que :

$$\alpha_t(i)\beta_t(i) = p(Y, y_t = i|\lambda), pour\ 1 \leq i \leq N, 1 \leq t \leq T. \qquad (4.12)$$

Ainsi, la somme de ce produit donne une autre façon pour calculer la probabilité$p(Y|\lambda)$, tout en utilisant les probabilités forward et backward :

$$p(Y|\lambda) = \sum_{i=1}^{N} p(Y, y_t = i|\lambda) = \sum_{i=1}^{N} \alpha_t(i)\beta_t(i),\ \ pour\ 1 \leq t \leq T \qquad (4.13)$$

4.1.2.2 Décodage et l'algorithme de Viterbi

Le problème du décodage se pose quand, étant donné une série d'observations, nous avons à trouver la séquence la plus probable des états cachés d'un modèle HMM. Ce problème est d'autant plus intéressant que dans plusieurs cas, les états cachés du HMM représentent quelque chose de non observable directement. Pour déterminer la séquence des états cachés la plus probable, étant donné une séquence d'observations,$\boldsymbol{Y} = \boldsymbol{y_1}, \boldsymbol{y_2}, \dots, \boldsymbol{y_T}$ et un HMM λ = (π, A, B), l'algorithme de Viterbi est le plus utilisé. Dans cette méthode, la séquence complète des états avec le maximum de vraisemblance est trouvée.

Algorithme de Viterbi : L'algorithme peut se résumer formellement de la façon suivante :

– Pour chacun des états, calcul par récurrence de la variable intermédiaire :

$$\delta_t(i) = \max\ \ p(x_1, x_2, \dots, x_{t-1}, x_t = i, y_1, y_2, \dots, y_{t-1}|\lambda) \qquad (4.14)$$

Le maximum étant calculé sur toutes les séquences d'´etats possibles$x_1, x_2 \dots, x_{t-1}$. Ce calcul se fait de manière récursive en deux étapes :

- Initialisation :

$$\delta_1(j) = \pi(j). b_j(1), pour\ 1 \leq j \leq N \qquad (4.15)$$

- Relation récursive :

$$\delta_{t+1}(j) = b_j(t+1)\{max_{1 \le j \le N}\ \delta_t(i)a_{ij}\}, pour 1 \le j \le N, 1 \le t \le T-1 \quad (4.16)$$

– Calcul de $\delta_T(i)$, $1 \le j \le N$, en utilisant cette dernière récursion et en retenant toujours un pointeur sur l'état « élu » dans une opération de maximisation.

– Détermination de l'état final du système (t = T) le plus probable :

$$i_t = argmax_{1 \le j \le N}\ (\delta_T(i)) \quad (4.17)$$

– Suivi du chemin le plus probable en revenant en arrière, soit : Si on note :

$$\emptyset_t(i) = argmax_{1 \le j \le N}\ (\delta_{t-1}(j)) \quad (4.18)$$

la séquence d'état la plus probable peut être trouvée par :

$$i_t = \emptyset_{t+1}(i_{t+1}) \quad (4.19)$$

Et en fin, la séquence $i_1, i_2, \dots, i_T$ est la séquence la plus probable des états cachés pour la séquence d'observation considérée.

4.1.2.3 Apprentissage

Le troisième, et le plus difficile, problème associé aux HMMs est de prendre une séquence connue d'observations pour représenter un ensemble d'états cachés, et d'obtenir le HMM λ = (π, A, B) qui est le modèle le plus probable décrivant ce qui est observé. En d'autres termes, dans plusieurs cas d'applications, le problème de l'apprentissage concerne la façon avec laquelle les paramètres du HMM sont ajustés, étant donné un ensemble d'observations (appelé ensemble d'apprentissage). Les paramètres du HMM à optimiser peuvent être différents d'une application à l'autre. De ce fait, il peut y avoir divers critères d'optimisation pour l'apprentissage, chacun d'entre eux étant choisi selon l'application considérée. Parmi ces critères, nous trouvons le critère du maximum de vraisemblance et de l'Information Maximum Mutuelle (MMI pour Maximum Mutual Information). Nous nous contentons ici de décrire un seul algorithme permettant de générer les paramètres d'un HMM à partir d'une séquence d'observations. Il s'agit de l'algorithme de Baum-Welch avec un critère de maximum de vraisemblance. Cet algorithme est aussi connu sous le nom de *Forward-Backward.*

- **Algorithme de Forward-backward :** Cet algorithme est utilisé quand les matrices A et B d'un HMM ne sont pas directement mesurables, comme c'est souvent le cas dans plusieurs applications réelles. Plus formellement, on considère une unique séquence d'observation $Y =$

$\boldsymbol{y_1}, \boldsymbol{y_2}, \dots, \boldsymbol{y_T}$. Notre but est de trouver les paramètres λ= (A, B) qui maximisent la probabilité de générer Y avec le modèle. Formellement, les calculs doivent maximiser la quantité :

$$Q(\lambda, \bar{\lambda}) = \sum_x p(x|Y, \lambda) \log\{p(Y, x, \bar{\lambda})\} \tag{4.20}$$

ou x désigne un état donné et $\bar{\lambda}$ le modèle estimé. Pour décrire l'algorithme nous avons à définir deux variables intermédiaires :$-\varepsilon_t(i,j) = p(x_t = i, x_{t+1} = j|Y, \lambda)$: la probabilité d'être dans l'état i à l'instant t et dans l'état j à l'instant t+1.$-\gamma_t(i) = p(x_t = i|Y, \lambda)$: la probabilité d'être dans l'état i à l'instant t étant donné la séquence d'observation et le modèle HMM. Ces deux variables peuvent être exprimées en fonction des variables forward, $\alpha_t(i)$ et backward, $\beta_t(i)$ définies précédemment. Pour résumer, l'algorithme peut être décrit de la façon suivante :

Initialisation : Des paramètres arbitraires pour le modèle sont choisis ; entre autre, les valeurs de π sont choisies aléatoirement tandis que les variables A et B sont initialisées. Par exemple, les valeurs de A sont fixées à priori et celles de B sont initialisées par une quantification vectorielle.

Itération :

- Les variables A et B sont placées à leurs valeurs de pseudo-comptes.
- Calcul des variables $\alpha_t(i)$ et $\beta_t(i)$ pour chaque état i, en utilisant respectivement les algorithmes forward et backward.
- En déduire les variables $\varepsilon_t(i,j)$ et $\gamma_t(i)$ en utilisant les expressions suivantes qui les lient aux variables forward et backward :

$$\varepsilon_t(i,j) = \frac{\alpha_t(i) a_{ij} \beta_{t+1}(j) b_j(t+1)}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_t(i) a_{ij} \beta_{t+1}(j) b_j(t+1)} \tag{4.21}$$

et

$$\gamma_t(i) = \frac{\alpha_t(i) \beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i) \beta_t(i)} \tag{4.22}$$

De ces deux expressions, il facile de remarquer que :

$$\gamma_t(i) = \sum_{j=1}^{N} \varepsilon_t(i,j) \tag{4.23}$$

– L'étape suivante consiste à actualiser les paramètres du HMM en utilisant ce qu'on appelle les *formules de ré-estimation* :

$$\bar{\pi} = \gamma_1(i), pour\ 1 \leq i \leq N \tag{4.24}$$

$$\overline{a_{ij}} = \frac{\sum_{t=1}^{T} \varepsilon_t(i,j)}{\sum_{t=1}^{T} \gamma_t(i)}\ pour\ 1 \leq i \leq N, 1 \leq j \leq N \tag{4.25}$$

$$\overline{b_j}(k) = \frac{\sum_{t=1}^{T} \varepsilon_t(i,j)}{\sum_{t=1}^{T} \gamma_t(i)}\ pour\ 1 \leq i \leq N, 1 \leq k \leq M \tag{4.26}$$

L'algorithme est arrêté si le changement de la log-vraisemblance est inférieur à un seuil prédéfini ou si le nombre maximum d'itération est atteint.

4.1.3 Différents types de modèles HMM

Depuis le début de cette section, nous avons traité en général le modèle HMM en supposant qu'il est caractérisé par une matrice de transition des états pleine ; c'est-à-dire que les transitions peuvent s'effectuer à partir de n'importe quel état vers n'importe quel autre état. On parle ici de modèle ergodique. Un tel modèle est défini comme un HMM tel que tous les états sont accessibles à partir de n'importe quel autre état. Pour certaines applications, il est demandé d'imposer certaines contraintes sur la matrice de transition ; ce qui rend le modèle non ergodique.

Dans ce sens, la littérature nous donne deux exemples types de modèles non-ergodique largement employés (Rabiner and Juang 1993). Ces deux modèles sont appelés gauche-droite du fait que la séquence des états produisant la séquence d'observations doit toujours avancer de l'état le plus à gauche à l'état le plus à droite. Ils diffèrent par le fait qu'un est un simple gauche-droite dans lequel il y a qu'un seul chemin à travers les états, et l'autre est un parallèle gauche-droite dans lequel il y a plusieurs chemins. Un modèle gauche-droite (parallèle ou simple) impose une structure temporelle ordonnée pour le HMM dans laquelle l'état numéroté avec un numéro inférieur précède toujours l'état avec un numéro supérieur. La figure 4.2 illustre les trois structures HMM.

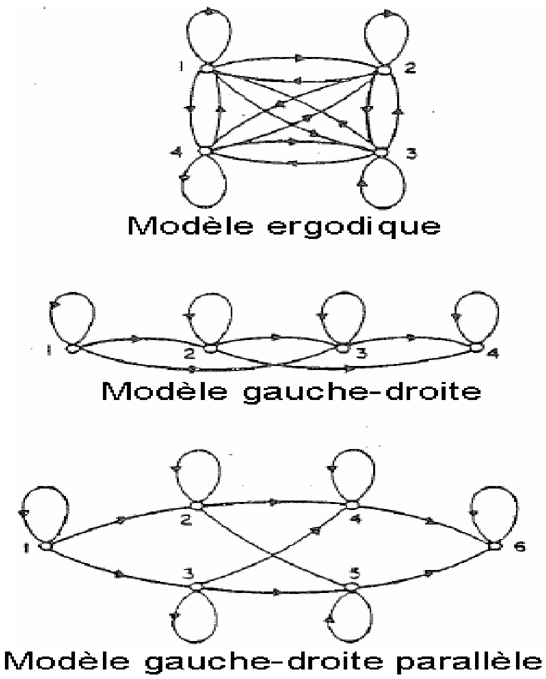

Figure 4.2 – Trois types distincts de modèles HMM. Illustration avec un exemple de HMM à 4 état (d'après Rabiner et Juang 1993).

4.1.4 Résumé

Le modèle de Markov caché est un outil statistique qui peut être défini quand les états d'un processus ne sont pas directement observables, mais sont indirectement et probabilistiquement observables comme un autre ensemble d'états. De tels modèles, appliqués dans des processus réels, imposent de résoudre trois problèmes :

- Evaluation : avec quelle probabilité un modèle donné génère-t-il une séquence d'observations donnée. L'algorithme forward résout efficacement ce problème.
- Décodage : quelle est la séquence d'états cachés la plus probable qui génère une séquence d'observations. L'algorithme de Viterbi résout ce problème.
- Apprentissage : comment optimiser (apprendre) les paramètres d'un modèle HMM à partir d'un échantillon donné de séquences d'observations. Ce problème peut être résolu en utilisant l'algorithme *forward-backward*.

Enfin, il est à noter un défaut habituel des modèles HMM qui concerne la sur-simplification associée à l'hypothèse markovienne ; c'est-à-dire qu'un état dépend seulement de ses prédécesseurs directs et que cette dépendance est indépendante du temps. Cependant,