Automate Stochastique

Partie 2/3

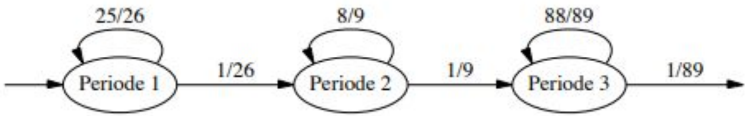
Par *Guillaume Guérard*, Nouvelles Energies

Chaîne de Markov Cachée

OBSERVATION

	Probabilité d'observer un cygne	Probabilité d'observer une oie
Période 1	0.8	0.2
Période 2	0.5	0.5
Période 3	0.1	0.9

ESPACE DISCRET



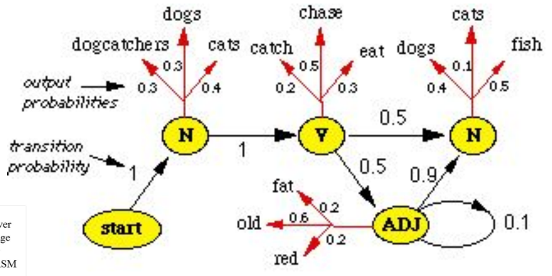
1. Evaluation de la probabilité de l'observation d'un mot
2. Recherche du chemin le plus probable
3. Apprentissage : comment ajuster sa HMM pour maximiser la vraisemblance

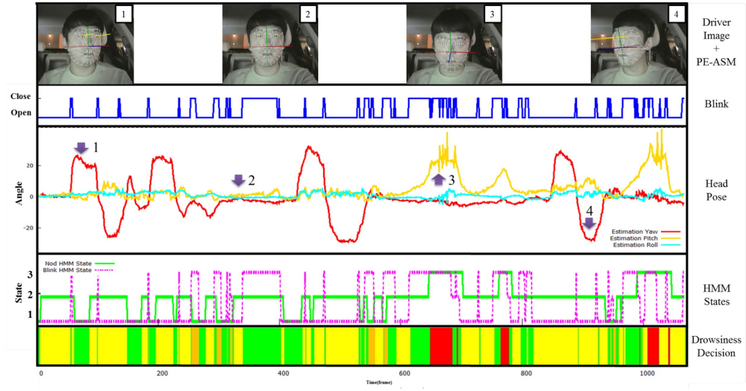
Chaîne de Markov Cachée

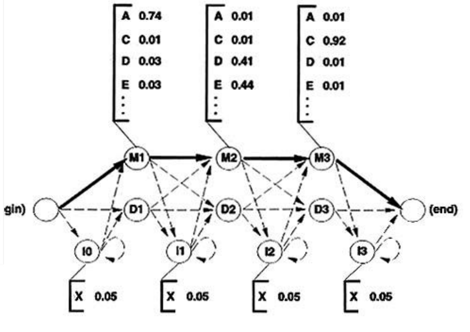
UTILITÉ

Modéliser un processus comprenant différentes étapes selon un ordre précis (mais pas toujours unique)

EXEMPLES







Plus d'information sur <u>smart--grid.net</u>

L'ensemble des cours sont disponible au format Slides et Cours sur le site.

Chaîne de Markov Cachée

QUAND L'UTILISER

Lorsque le processus fonctionne par étape (discrète ou continue) mais que l'on ne connaît que des observations de cette machine.

Le système est considéré autonome.

HMM est une variante des FSM que l'on verra au cours 3/3

CLASSIFICATION

Les HMM sont des techniques de classification comme les ANN, kNN ou SVM.

Les HMM sont grandement utilisés en biologie et bioinformatique (protéines, ADN/ARN, génétique)

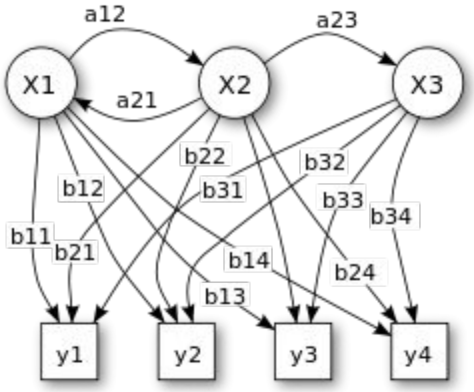
Plus d'information sur smart--grid.net

L'ensemble des cours sont disponible au format Slides et Cours sur le site.

Chaîne de Markov Cachée

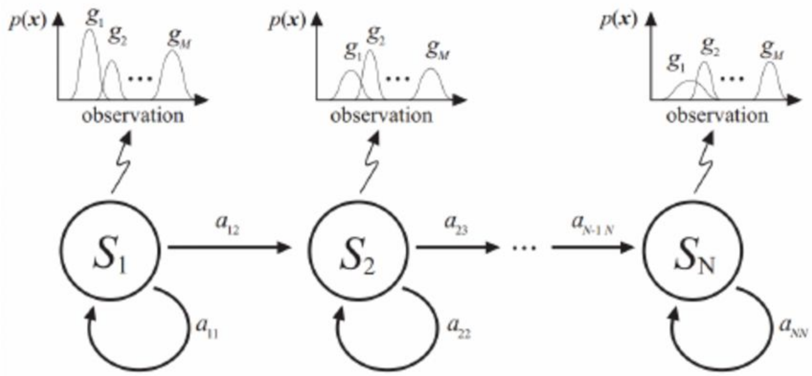
Multinomial HMM

Observations effectuées en temps discret



Gaussian/mixture HMM

Observations effectuées en temps continu



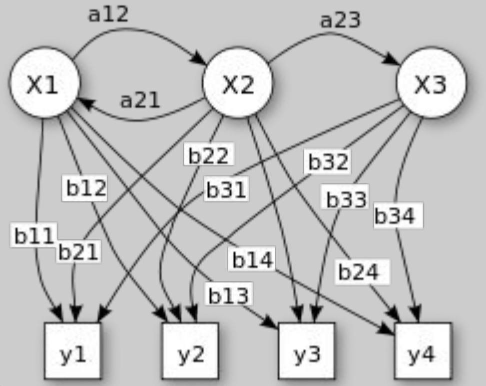
Plus d'information sur [smart--grid.net]

L'ensemble des cours sont disponible au format Slides et Cours sur le site.

Chaîne de Markov Cachée

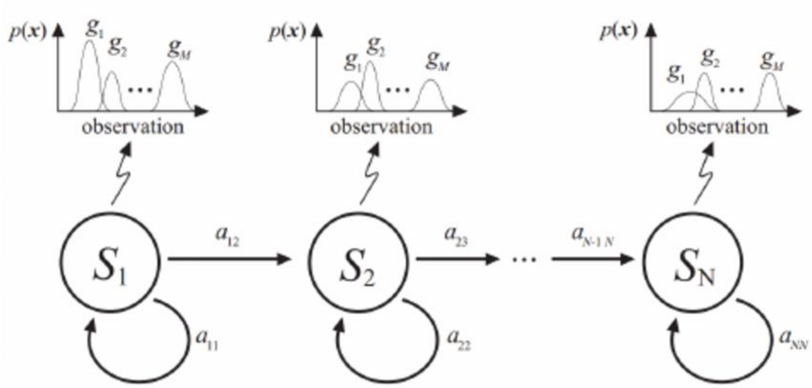
Multinomial HMM

Observations effectuées en temps discret



Gaussian/mixture HMM

Observations effectuées en temps continu



Plus d'information sur [smart--grid.net]()

L'ensemble des cours sont disponible au format Slides et Cours sur le site.

Plan

1. HMM
2. Evaluation
3. Recherche
4. Apprentissage

1

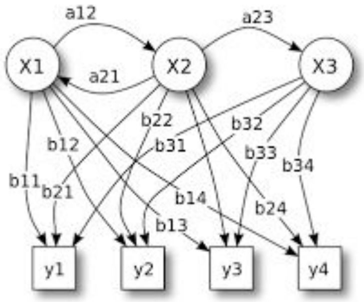
Chaîne à états cachés

I'm not gonna hurt ya. I'm not gonna hurt ya. I'm just gonna bash your brains in. I'm just gonna bash em right the fuck in ! I said, I'm not gonna hurt ya. I'm just going to bash your brains in. Ha ha ha.

Théorie des HMM

Une séquence = deux suites de V.A.

- Une cachée, notée Q(1 : T) où q(i) est un état du modèle.
- Une observable, notée O(1 : T) où o(i) est une lettre de l'alphabet (dit des symboles observables).
- A chaque état est associé les probabilités vers un autre état
- A chaque état est associé les probabilités pour un ensemble de symboles observables



Théorie des HMM

Définition d'une HMM $\Lambda=(A,B,\pi)$

- Etats où se trouve la HMM à l'instant t $q(t) \in S = \{s1, s2, \ldots sn\}$
- Symboles observable dans chaque état. L'alphabet est noté $V = \{v1, v2, \ldots, vM\}$. $O(t)$ est le symbole observé à l'instant t.
- Matrice A de probabilité de transition dans la HMM

$$a_{ij} = A(i,j) = \mathbf{P}(q_{t+1} = s_j \mid q_t = s_i) \ \forall i,j \in [1 \ldots n] \ \forall t \in [1 \ldots T]$$

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

Théorie des HMM

Définition d'une HMM Λ=(A,B,π) --suite--

- Matrice B de probabilités d'observation des symboles dans chacun des états $b_j(k) = \mathbf{P}(O_t = v_k \mid q_t = s_j) \quad 1 \leq j \leq n, \; 1 \leq k \leq M$

$$b_j(k) \geq 0 \quad \forall j, k \qquad \text{et :} \qquad \sum_{k=1}^{M} b_j(k) = 1$$

- Vecteur de probabilités initiales

$$\pi_i = \mathbf{P}(q_1 = s_i) \quad 1 \leq i \leq n$$

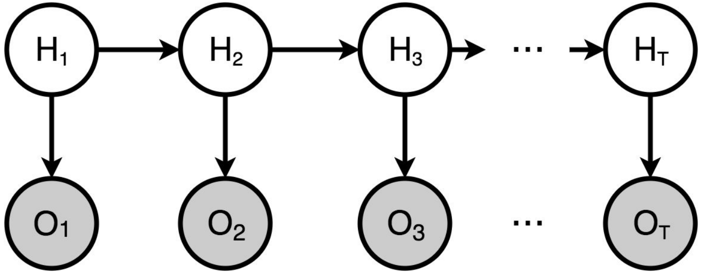
$$\pi_i \geq 0 \quad \forall i \qquad \text{et :} \qquad \sum_{i=1}^{n} \pi_i = 1$$

- Des états finaux (au moins un)

Théorie des HMM

Définition d'une HMM $\Lambda=(A,B,\pi)$ --fin--

- Modèle Gauche-Droite

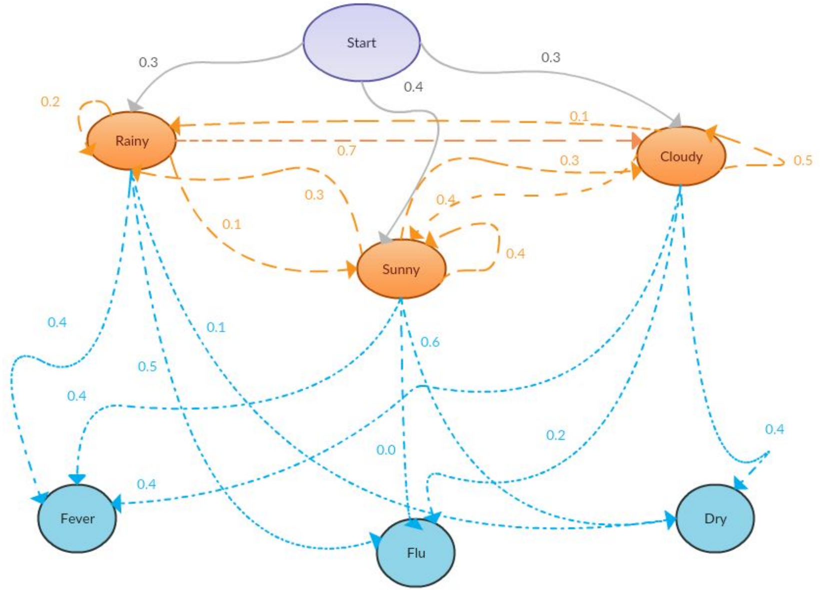


Liaison boucle ou vers un état à droite (direct ou éloigné)

Théorie des HMM

Définition d’une HMM $\Lambda=(A,B,\pi)$ --fin--

- Modèle ergodique



La chaîne cachée est ergodique

Théorie des HMM

Marche aléatoire

- Dans une HMM les observations sont les informations utiles pour l'utilisation. Générer une marche aléatoire revient à générer une séquence d'état observable

1. Choisir un état initial d'après le vecteur initial
2. Choisir l'observation du temps t d'après B
3. Choisir l'état pour le temps t+1 d'après A
4. Revenir à 4

Une séquence donnée peut être généré de plusieurs façons.

Théorie des HMM

Les trois problématiques des HMM

1. Evaluation de la probabilité de l'observation d'une séquence

Étant donné la séquence d'observations O et une HMM $\Lambda=(A,B,\pi)$, comment évaluer la probabilité d'observation $P(O|\Lambda)$?

Théorie des HMM

Les trois problématiques des HMM

2. Recherche du chemin le plus probable

Étant donné la séquence d'observations O et une HMM $\Lambda=(A,B,\pi)$, , comment trouver une suite d'états Q = q1, q2,...qT qui maximise la probabilité d'observation de la séquence ?

Théorie des HMM

Les trois problématiques des HMM

3. Apprentissage

Comment ajuster les paramètres (A, B, π) d'un Hmm Λ pour maximiser la vraisemblance $$\mathsf{P}(\mathcal{O} \mid \Lambda) = \prod_{O \in \mathcal{O}} \mathsf{P}(O \mid \Lambda)$$

à partir d'un ensemble O de séquences d'apprentissage ?

2 Evaluation

Lorsqu'on entraîne plusieurs modèles de HMM par apprentissage Bayesienne, il est nécessaire de les évaluations. Avant tout, il faut savoir comment évaluer la probabilité qu'une phrase soit émise par une HMM.

Evaluation

Remarquons d'abord que la probabilité de la suite d'observations O, étant donné le modèle Λ, est égale à la somme sur toutes les suites d'états possibles Q des probabilités conjointes de O et de Q.

$$\mathbf{P}(O \mid \Lambda) = \sum_{Q} \mathbf{P}(O, Q \mid \Lambda) = \sum_{Q} \mathbf{P}(O \mid Q, \Lambda)\mathbf{P}(Q \mid \Lambda)$$

Evaluation

On réécrit les deux termes :

$$\mathbf{P}(Q \mid \Lambda) = \pi_{q_1} a_{q_1 q_2} a_{q_2 q_3} \cdots a_{q_{T-1} q_T}$$

$$\mathbf{P}(O \mid Q, \Lambda) = b_{q_1}(O_1) b_{q_2}(O_2) \ldots b_{q_T}(O_T)$$

La formule est donc la suivante

$$\mathbf{P}(O \mid \Lambda) = \sum_{Q = q_1, q_2, \ldots, q_T} \pi_{q_1} b_{q_1}(O_1) a_{q_1 q_2} b_{q_2}(O_2) \ldots a_{q_{T-1} q_T} b_{q_T}(O_T)$$

Cela revient à énumérer toutes les séquences d'états de longueur T, donc O(nˆT) … Ce qui est bien trop long !

Evaluation

Fonctions Forward-Backward

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 qi au temps t ,puis, l'émission de la fin de l'observation O(t+1:T) sachant que l'on part de qi au temps t:

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

Où αt(i) est la probabilité d'émettre le début O(1:t) et d'aboutir à qi à l'instant t, et βt(i) est la probabilité d'émettre la fin O(t+1:T) sachant que l'on part de qi à l'instant t. Le calcul de α s'effectue avec t croissant tandis que le calcul de β est réalisé avec t

Evaluation

Forward-Backward : calcul de α

$$\alpha_t(i) = \mathsf{P}(O_1 O_2 \ldots O_t,\ q_t = s_i \mid \Lambda)$$

On suppose connaître la suite d'observation O(1:t) pour émettre le début de l'observation O(1:t+1) et aboutir dans l'état sj au temps t+1, on doit nécessairement être dans l'un des états si à l'instant t (O(n^2T))

début
  pour $i = 1, n$ **faire** $\alpha_1(i) \leftarrow \pi_i b_i(O_1)$
  $t \leftarrow 1$
  tant que $t < T$ **faire**
    $j \leftarrow 1$
    tant que $j \leq n$ **faire**
      $\alpha_{t+1}(j) \leftarrow [\sum_{i=1}^{n} \alpha_t(i) a_{ij}] b_j(O_{t+1})$
      $j \leftarrow j + 1$
    fin tant que
    $t \leftarrow t + 1$
  fin tant que
  $\mathsf{P}(O \mid \Lambda) \leftarrow \sum_{i=1}^{n} \alpha_T(i)$
fin

Evaluation

Forward-Backward : calcul de β

On suppose connaître la suite d'observation O(t:T) pour émettre le début de l'observation O(t-1:T) et venir de l'état si au temps t-1, on doit nécessairement être dans l'un des états si à l'instant t (O(n²T))

début
- **pour** $i = 1, n$ **faire** $\beta_T(i) \leftarrow 1$
- $t \leftarrow T - 1$
- **tant que** $t \geq 1$ **faire**
  - $i \leftarrow 1$
  - **tant que** $i \leq n$ **faire**
    - $\beta_t(i) \leftarrow \sum_{j=1}^{n} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)$
    - $i \leftarrow i + 1$
  - **fin tant que**
  - $t \leftarrow t - 1$
- **fin tant que**
- $\mathsf{P}(O \mid \Lambda) \leftarrow \sum_{i=1}^{n} \beta_1(i)$

fin

Evaluation

Forward-Backward : finalité

La probabilité d'observation d'une séquence est obtenue en prenant les valeurs de α et de β à un instant t quelconque :

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

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

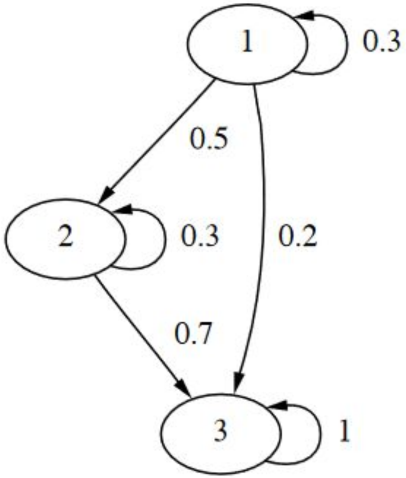
$$\mathbf{P}(O \mid \Lambda) = \sum_{i=1}^{n} \alpha_T(i) = \sum_{i=1}^{n} \pi_i \beta_0(i)$$

Evaluation

Forward-Backward : Exemple

$$A = \begin{pmatrix} 0.3 & 0.5 & 0.2 \\ 0 & 0.3 & 0.7 \\ 0 & 0 & 1 \end{pmatrix} \quad B = \begin{pmatrix} 1 & 0 \\ 0.5 & 0.5 \\ 0 & 1 \end{pmatrix} \quad \pi = \begin{pmatrix} 0.6 \\ 0.4 \\ 0 \end{pmatrix}$$

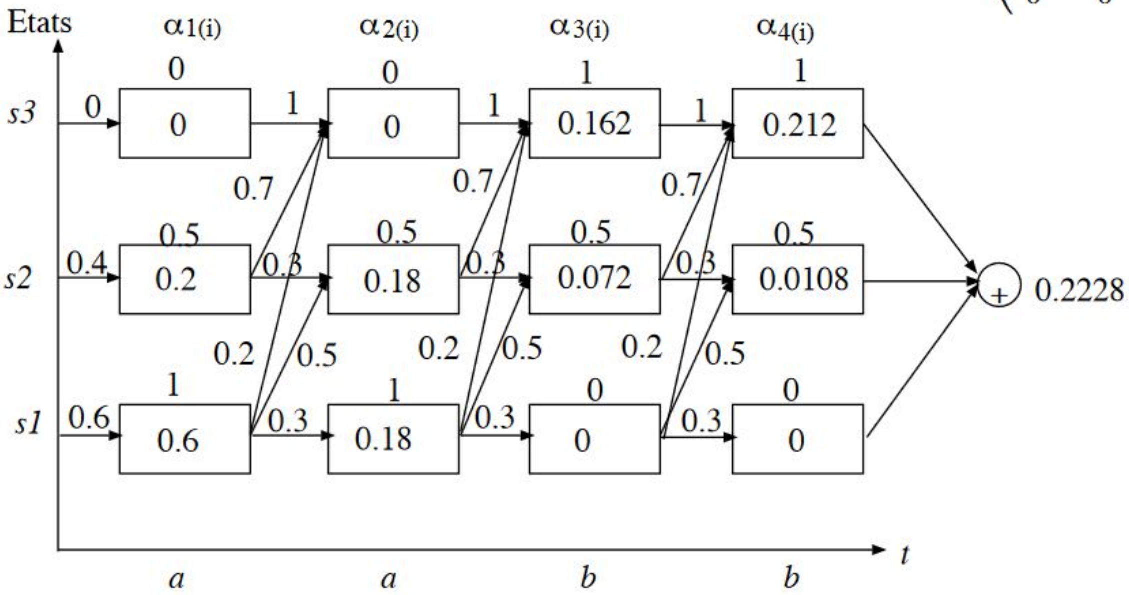
$$\begin{aligned}
\alpha_1(1) &= \pi_1 b_1(a) = 0.6 \times 1 = 0.6 \\
\alpha_1(2) &= \pi_2 b_2(a) = 0.4 \times 0.5 = 0.2 \\
\alpha_1(3) &= \pi_3 b_3(a) = 0 \times 0 = 0 \\
\alpha_2(1) &= (\alpha_1(1) a_{11} + \alpha_1(2) a_{21} + \alpha_1(3) a_{31}) b_1(a) \\
&= (0.6 \times 0.3 + 0.2 \times 0 + 0 \times 0) \times 1 \\
&= (0.18) \times 1 = 0.18 \\
\alpha_2(2) &= (\alpha_1(1) a_{12} + \alpha_1(2) a_{22} + \alpha_1(3) a_{32}) b_2(a) \\
&= (0.6 \times 0.5 + 0.2 \times 0.3 + 0 \times 0) \times 0.5 \\
&= (0.36) \times 0.5 = 0.18 \\
\ldots & \quad \ldots \\
\mathbf{P}(a\ a\ b\ b \mid \Lambda) &= \textstyle\sum_{q_i} \alpha_4(i) = 0.2228
\end{aligned}$$

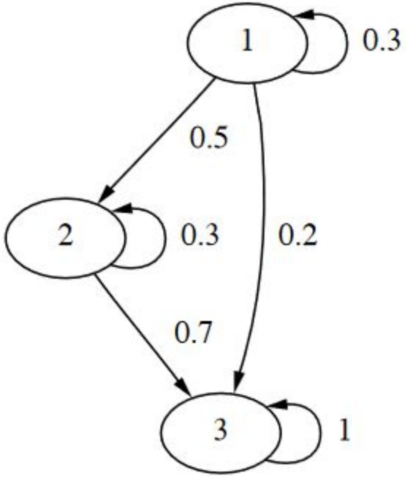


Evaluation

Forward-Backward : Exemple

$$A = \begin{pmatrix} 0.3 & 0.5 & 0.2 \\ 0 & 0.3 & 0.7 \\ 0 & 0 & 1 \end{pmatrix} \quad B = \begin{pmatrix} 1 & 0 \\ 0.5 & 0.5 \\ 0 & 1 \end{pmatrix} \quad \pi = \begin{pmatrix} 0.6 \\ 0.4 \\ 0 \end{pmatrix}$$





3

Recherche du chemin optimal

Il s'agit maintenant de déterminer le meilleur chemin correspondant à l'observation.

Calcul du chemin optimal

On cherche la meilleure séquence d’états Q qui maximise la quantité : $\mathbf{P}(Q, O \mid \Lambda)$

Cela revient à énumérer toutes les séquences d’états de longueur T, donc O(nˆT) … Ce qui est bien trop long !

Calcul du chemin optimal

Algorithme de Viterbi

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

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

En $O(n^2T)$

Ψ contient l'indice du chemin optimal

début
- **pour** $i = 1, n$ **faire**
  - $\delta_1(i) \leftarrow \pi_i b_i(O_1)$
  - $\psi_1(i) \leftarrow 0$
- **fin pour**
- $t \leftarrow 2$
- **tant que** $t \leq T-1$ **faire**
  - $j \leftarrow 1$
  - **tant que** $j \leq n$ **faire**
    - $\delta_t(j) \leftarrow \mathrm{Max}_{1\leq i\leq n}\, [\delta_{t-1}(i)\, a_{ij}]\, b_j(O_t)$
    - $\psi_t(j) \leftarrow \mathrm{ArgMax}_{1\leq i\leq n}\, [\delta_{t-1}(i)\, a_{ij}]$
    - $j \leftarrow j+1$
  - **fin tant que**
  - $t \leftarrow t+1$
- **fin tant que**
- $\mathbf{P}^* \leftarrow \mathrm{Max}_{1\leq i\leq n}\, [\delta_T(i)]$
- $q_T^* \leftarrow \mathrm{ArgMax}_{1\leq i\leq n}\, [\delta_T(i)]$
- $t \leftarrow T$
- **tant que** $t \geq 1$ **faire**
  - $q_t^* \leftarrow \psi_{t+1}(q_{t+1}^*)$
  - $t \leftarrow t-1$
- **fin tant que**

fin

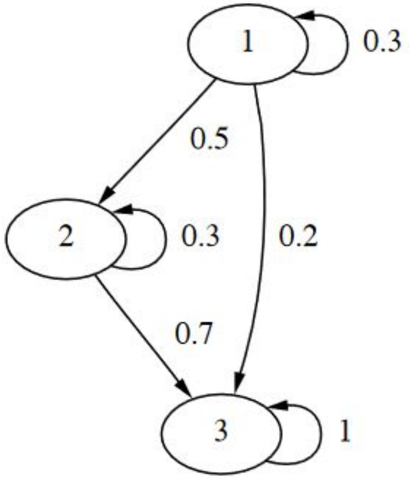
Calcul du chemin optimal

Algorithme de Viterbi : Exemple

$$A = \begin{pmatrix} 0.3 & 0.5 & 0.2 \\ 0 & 0.3 & 0.7 \\ 0 & 0 & 1 \end{pmatrix} \quad B = \begin{pmatrix} 1 & 0 \\ 0.5 & 0.5 \\ 0 & 1 \end{pmatrix} \quad \pi = \begin{pmatrix} 0.6 \\ 0.4 \\ 0 \end{pmatrix}$$

$$\begin{aligned}
\delta_1(1) &= \pi_1 b_1(a) = 0.6 \times 1 = 0.6 & \psi_1(1) = 0, \\
\delta_1(2) &= \pi_2 b_2(a) = 0.4 \times 0.5 = 0.2 & \psi_1(2) = 0, \\
\delta_1(3) &= \pi_3 b_3(a) = 0 \times 0 = 0 & \psi_1(3) = 0, \\
\delta_2(1) &= max_{1 \leq i \leq n}(\delta_1(i) a_{i1}) b_1(a) & \\
&= max \begin{Bmatrix} \delta_1(1) a_{11} \\ \delta_1(2) a_{21} \\ \delta_1(3) a_{31} \end{Bmatrix} \times b_1(a) & \\
&= max \begin{Bmatrix} \underline{0.6 \times 0.3} \\ 0 \\ 0 \end{Bmatrix} \times 1 = 0.18 & \psi_2(1) = 1,
\end{aligned}$$

$\ldots$

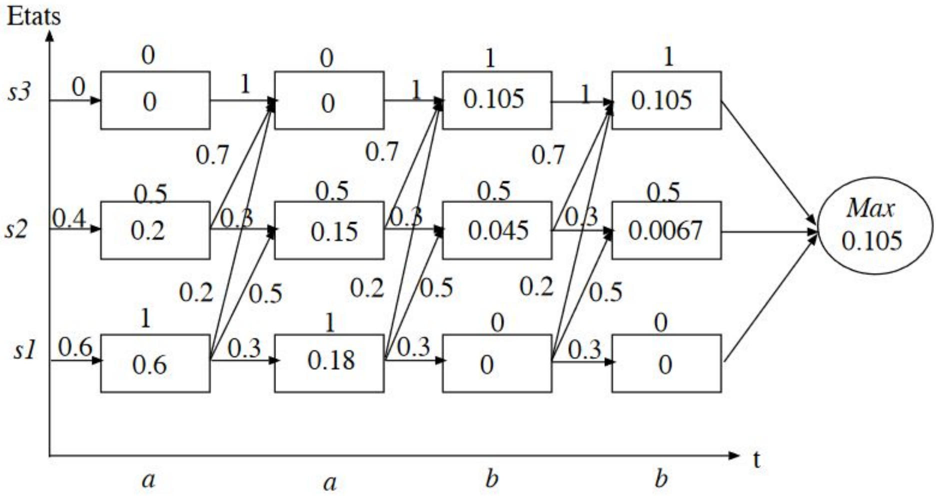


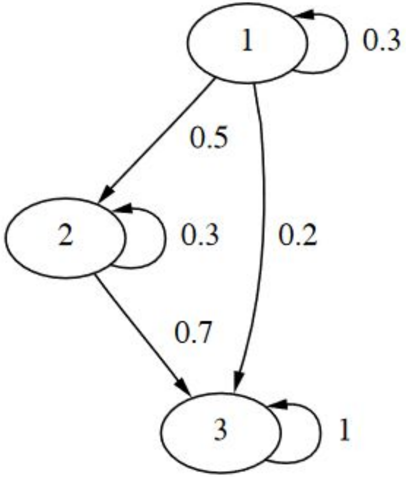
Calcul du chemin optimal

Algorithme de Viterbi : Exemple

$$
\begin{array}{llll}
\underline{\psi_1(1) = 0} & \psi_2(1) = 1 & \psi_3(1) = 1 & \psi_4(1) = 1 \\
\psi_1(2) = 0 & \underline{\psi_2(2) = 1} & \psi_3(2) = 1 & \psi_4(2) = 2 \\
\psi_1(3) = 0 & \psi_2(3) = 2 & \underline{\psi_3(3) = 2} & \underline{\psi_4(3) = 3},
\end{array}
$$

$$
A = \begin{pmatrix} 0.3 & 0.5 & 0.2 \\ 0 & 0.3 & 0.7 \\ 0 & 0 & 1 \end{pmatrix} \quad B = \begin{pmatrix} 1 & 0 \\ 0.5 & 0.5 \\ 0 & 1 \end{pmatrix} \quad \pi = \begin{pmatrix} 0.6 \\ 0.4 \\ 0 \end{pmatrix}
$$





4 Apprentissage

Disposant de séquences d'observations, on souhaite déterminer les paramètres d'une HMM qui maximise la probabilité d'avoir ces séquences d'observations.

Apprentissage

Comme on suppose les séquences d'apprentissages tirées indépendamment, on cherche donc à maximiser :

$$\mathbf{P}(\mathcal{O} \mid \Lambda) = \prod_{k=1}^{m} \mathbf{P}(O^k \mid \Lambda)$$

Apprentissage

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

- choisir un ensemble initial Λ0 de paramètres
- calculer Λ1 à partir de Λ0,puis Λ2 à partir de Λ1, etc.
- répéter ce processus jusqu'à un critère de fin.

Pour chaque étape p d'apprentissage, on dispose de Λp et on cherche un Λp+1 qui doit vérifier :

$$\mathbf{P}(\mathcal{O} \mid \Lambda_{p+1}) \geq \mathbf{P}(\mathcal{O} \mid \Lambda_p)$$

$$\prod_{k=1}^{m} \mathbf{P}(O^k \mid \Lambda_{p+1}) \ \geq \ \prod_{k=1}^{m} \mathbf{P}(O^k \mid \Lambda_p)$$

Apprentissage

Ré-estimation par la séquence d'observations k

On définit $\xi_t^k(i,j)$ comme la probabilité, étant donnés une phrase Ok et une Hmm Λ, que ce soit l'état si qui ait émis la lettre de rang t de Ok et l'état sj qui ait émis celle de rang t+1. Donc :

$$\xi_t^k(i,j) = \mathbf{P}(q_t = s_i,\ q_{t+1} = s_j \ \mid O^k, \Lambda)$$

$$\xi_t^k(i,j) = \frac{\mathbf{P}(q_t = s_i,\ q_{t+1} = s_j,\ O^k \mid \Lambda)}{\mathbf{P}(O^k \mid \Lambda)}$$

Apprentissage

Ré-estimation par la séquence d'observations k

On réécrit par les fonctions Backward-Forward :

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

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

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

$$\gamma_t^k(i) = \sum_{j=1}^{n} \xi_t(i,j) \frac{\alpha_t^k(i) \beta_t^k(i)}{\mathbf{P}(O^k \mid \Lambda)}$$

Apprentissage

Ré-estimation par la séquence d'observations k : Baum

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

$$\bar{a}_{ij} = \frac{\text{nombre de fois où la transition de } s_i \text{ à } s_j \text{ a été utilisée}}{\text{nombre de transitions effectuées à partir de } s_i}$$

$$\bar{b}_j(l) = \frac{\text{nombre de fois où le H}\textsc{mm}\text{ s'est trouvé dans l'état } s_j \text{ en observant } v_l}{\text{nombre de fois où le H}\textsc{mm}\text{ s'est trouvé dans l'état } s_j}$$

$$\bar{\pi}_i = \frac{\text{nombre de fois où le H}\textsc{mm}\text{ s'est trouvé dans l'état } s_i \ldots}{\text{nombre de fois où le H}\textsc{mm} \ldots}$$

$$\frac{\ldots \text{ en émettant le premier symbole d'une phrase}}{\ldots \text{ a émis le premier symbole d'une phrase}}$$

Apprentissage

Ré-estimation par la séquence d'observations k : Baum (algo EM)

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

$$\overline{\pi}_i = \frac{1}{m}\sum_{k=1}^{m}\gamma_1^k(i)$$

$$\overline{a}_{ij} = \frac{\displaystyle\sum_{k=1}^{m}\sum_{t=1}^{|O^k|-1}\xi_t^k(i,j)}{\displaystyle\sum_{k=1}^{N}\sum_{t=1}^{|O^k|-1}\gamma_t^k(i)}$$

$$\overline{b}_j(l) = \frac{\displaystyle\sum_{k=1}^{m}\sum_{\substack{t=1\\ \text{avec } O_t^k=v_l}}^{|O^k|-1}\gamma_t^k(j)}{\displaystyle\sum_{k=1}^{m}\sum_{t=1}^{|O^k|-1}\gamma_t^k(j)}$$

Apprentissage

Ré-estimation par la séquence d'observations k : Baum-Welch

L'algorithme de mise-à-jour de la HMM est le suivant :

début

Fixer des valeurs initiales (A, B, π)

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

$p \leftarrow 0$

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 :

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

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

Le HMM courant est désormais défini par $\Lambda_{p+1} = (\overline{\pi}, \overline{A}, \overline{B})$

$p \leftarrow p + 1$

fin tant que

fin

Apprentissage

Ré-estimation par la séquence d'observations k : Baum-Welch

- 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 $P(O|\Lambda)$. Il est donc important, si l'on veut être aussi près que possible du minimum global (pour la vraisemblance), de bien choisir la structure et l'initialisation.
- Le nombre d'itérations est fixé empiriquement. Il n'y a généralement pas de surapprentissage.

Apprentissage

Exemple 1

Hmm Λ0 défini par les paramètres suivants sur un alphabet {a,b}

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

On cherche à apprendre sur le mot abbaa

$$\mathbf{P}(a\ b\ b\ a\ a \mid \Lambda_0) = 0.0278$$

Apprentissage

Exemple 2

Hmm Λ0 défini par les paramètres suivants sur un alphabet {a,b}

Après 1 ré-estimation

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

$$\mathbf{P}(a\ b\ b\ a\ a \mid \Lambda_1) = 0.0529$$

Après 15 ré-estimation

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

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

Après 150 ré-estimation

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

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

Apprentissage

Exemple 2

Attention : nombre d'état caché trop grand = apprentissage par coeur

Convergence du mot abbaa dans un HMM à 5 états

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

$$\mathbf{P}(a\ b\ b\ a\ a \mid \Lambda) = 1.0$$

Apprentissage

Exemple 3

Attention : données d'apprentissage non homogène, convergence vers un optimum local

$$A = \begin{pmatrix} 0.40 & 0.60 \\ 0.52 & 0.48 \end{pmatrix} \quad B = \begin{pmatrix} 0.49 & 0.51 \\ 0.40 & .60 \end{pmatrix} \quad \pi = \begin{pmatrix} 0.31 \\ 0.69 \end{pmatrix}$$

Apprentissage sur le premier set puis le deuxième set

$$\mathcal{O}_1 = \{aaabb, abaabbb, aaababb, aabab, ab\}$$

$$\mathcal{O}_2 = \{bbbaa, babbaa, bbbabaa, bbabba, bbaa\}$$

Apprentissage

Exemple 3

Attention : données d'apprentissage non homogène, convergence vers un optimum local

$$A = \begin{pmatrix} 0.40 & 0.60 \\ 0.52 & 0.48 \end{pmatrix} \quad B = \begin{pmatrix} 0.49 & 0.51 \\ 0.40 & .60 \end{pmatrix} \quad \pi = \begin{pmatrix} 0.31 \\ 0.69 \end{pmatrix}$$

$$\mathcal{O}_1 = \{aaabb, abaabbb, aaababb, aabab, ab\}$$

$$\mathcal{O}_2 = \{bbbaa, babbaa, bbbabaa, bbabba, bbaa\}$$

Apprentissage sur le premier set puis le deuxième set

$$A = \begin{pmatrix} 1.0 & 0.0 \\ 0.69 & 0.31 \end{pmatrix} \quad B = \begin{pmatrix} 0.36 & 0.64 \\ 1.0 & 0.0 \end{pmatrix} \quad \pi = \begin{pmatrix} 0.0 \\ 1.0 \end{pmatrix}$$

$$A = \begin{pmatrix} 1.0 & 0.0 \\ 0.60 & 0.40 \end{pmatrix} \quad B = \begin{pmatrix} 0.65 & 0.34 \\ 0.0 & 1.0 \end{pmatrix} \quad \pi = \begin{pmatrix} 0.0 \\ 1.0 \end{pmatrix}$$

$$\mathbf{P}(a\ a\ b\ a\ b\ b\ b \mid \Lambda_1) = 0.0437$$

$$\mathbf{P}(a\ a\ b\ a\ b\ b\ b \mid \Lambda_2) = 0.000$$

$$\mathbf{P}(b\ b\ a\ b\ a\ a\ a \mid \Lambda_1) = 0.000$$

$$\mathbf{P}(b\ b\ a\ b\ a\ a\ a \mid \Lambda_2) = 0.0434$$

Conclusion et Python

MonSieuR G R1 ComPri !

Python

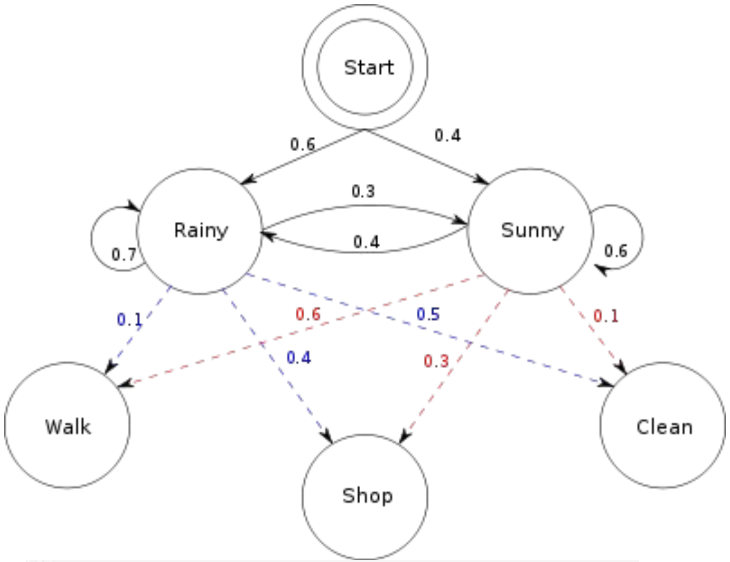
Bibliothèque sklearn (hmm)

Modèle multinomialHMM = en temps discret

Python

Viterbi (meilleur chemin d'après l'observation)

```
import numpy as np
from sklearn import hmm

states = ["Rainy", "Sunny"]
n_states = len(states)

observations = ["walk", "shop", "clean"]
n_observations = len(observations)

start_probability = np.array([0.6, 0.4])

transition_probability = np.array([
  [0.7, 0.3],
  [0.4, 0.6]
])

emission_probability = np.array([
  [0.1, 0.4, 0.5],
  [0.6, 0.3, 0.1]
])

model = hmm.MultinomialHMM(n_components=n_states)
model._set_startprob(start_probability)
model._set_transmat(transition_probability)
model._set_emissionprob(emission_probability)

# predict a sequence of hidden states based on visible states
bob_says = [0, 2, 1, 1, 2, 0]
logprob, alice_hears = model.decode(bob_says, algorithm="viterbi")
print "Bob says:", ", ".join(map(lambda x: observations[x], bob_says))
print "Alice hears:", ", ".join(map(lambda x: states[x], alice_hears))
```




```
1 Bob says: walk, clean, shop, shop, clean, walk
2 Alice hears: Sunny, Rainy, Rainy, Rainy, Rainy, Sunny
```

Théorie des HMM

Comment choisir sa HMM ?

- Réseau Bayesien

http://mlg.eng.cam.ac.uk/zoubin/papers/ijprai.pdf

http://thesis.univ-biskra.dz/2288/5/Chapitre%203.pdf

http://www2.agroparistech.fr/ufr-info/membres/