SI221 : Bases de l'Apprentissage

Modèles de Markov Cachés

Octobre 2018

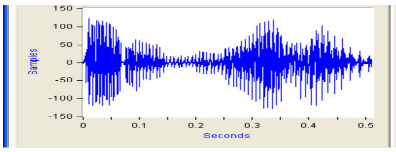
Laurence Likforman-Sulem
Telecom ParisTech/IDS
likforman@telecom-paristech.fr

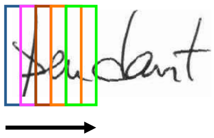
Plan

- Chaînes de Markov
  - modèles stochastiques
  - paramètres
- Modèles de Markov Cachés
  - discrets/continus
  - apprentissage
  - décodage

applications

- HMMs
  - Reconnaissance de la parole (speech recognition)
  - Reconnaissance de l'écriture (handwriting recognition)
  - Reconnaissance d'objets, de visages dans les videos
  - Traitement de la langue-Natural Language Processing-NLP: étiquetage grammatical





THE→ TGE

Modèle stochastique

- processus aléatoire à temps discret
  - ensemble de variables aléatoires q_1, q_2, …., q_T
  - indexées aux instants entiers t=1, 2, ….T
- notation
  - q_t: variable aléatoire d'état observé au temps t
    - notée q(t) ou q_t
    - q(t) prend ses valeurs dans espace fini d'états S
      S={1,2, ….Q}
  - $P(q_t=i)$: probabilité d'observer l'état i au temps t

exemple état: pollution (indice), météo: beau, pluie, nuageux, NLP: fonction des mots d'un texte (verbe,nom, pronom;….)

Modèle stochastique

- évolution du processus
  - état initial q1
  - suite (chaîne) de transitions entre états
    - $q_1 \rightarrow q_2 \ldots \rightarrow q_t \quad t <= T$
- calcul probabilité d'une séquence d'états

  $P(q_1, q_2, \ldots q_T) = P(q_T / q_1, q_2, \ldots q_{T-1}) P(q_1, q_2, \ldots q_{T-1})$

  $= P(q_T | q_1, q_2, \ldots q_{T-1}) P(q_{T-1} | q_1, q_2, \ldots q_{T-2}) P(q_1, q_2, \ldots q_{T-2})$

  $= P(q_1) P(q_2 / q_1) P(q_3 / q_1, q_2) \ \ldots.. \ P(q_T | q_1, q_2, \ldots q_{T-1})$
- modèle: connaître la probabilité de chaque transition+proba initiale $P(q_1)$

Chaîne de Markov à temps discret

- propriété de Markov d'ordre k : dépendance limitée
  - $P(q_t | q_1, q_2, \ldots q_{t-1}) = P(q_t | q_{t-k} \ldots q_{t-1})$
  - k=1 ou 2 en pratique
- *cas k=1*
  - $P(q_t | q_1, q_2, \ldots q_{t-1}) = P(q_t | q_{t-1})$
  - $P(q_1, q_2, \ldots q_T) = P(q_1) P(q_2 / q_1) P(q_3 / q_2) \ \ldots.. \ P(q_T | q_{T-1})$
  - → probabilités de transition entre états

Chaîne de Markov stationnaire

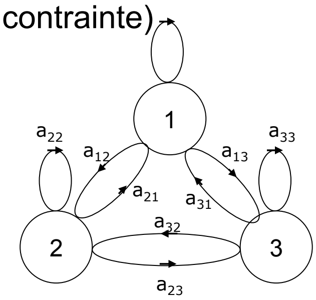
- probabilités de transition ne dépendent pas du temps
  - $P(q_t = j \,/\, q_{t-1} = i) = P(q_{t+k} = j \,/\, q_{t+k-1} = i) = a_{ij}$
  - *a_{ij}= probabilité de passer de l'état i à l'état j*
- définition: modèle d'une chaîne de Markov stationnaire
  - *matrice des probabilités de transitions*
  - $A=[a_{ij}]$ $\quad i=1,\dots Q,\ j=1,\dots Q$
  - vecteur des probabilités initiales
  - $\Pi= [\pi_i]$ $\quad i=1,\dots Q$
  - $\pi_i=P(q_1=i)$
- *contraintes :* $0 <= \pi_i <= 1 \quad 0 <= a_{ij} <= 1$

$$\sum_{i=1}^{Q} \pi_i = 1 \qquad \sum_{j=1}^{Q} a_{ij} = 1$$

topologie du modèle: ergodique / gauche droite

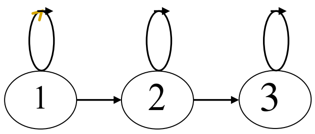
- modèle ergodique (sans contrainte)

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



- modèle gauche droite (contrainte: transitions i→ j ≥ i)
-

$$A= \begin{bmatrix} 0.4 & 0.6 & 0 \\ 0 & 0.8 & 0.2 \\ 0 & 0 & 1 \end{bmatrix}$$

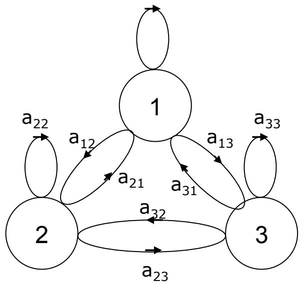


Chaîne de Markov stationnaire: mini TD

- Soit une chaîne à 3 états
  - 1: pluie (r), 2: nuages (c) , 3: soleil (s)
- on observe q_1= s , quelle est la probabilité d'observer pendant les 7 jours suivants les temps (états)

  s   s s r r s c s
- *t=1   t=2*
- *modèle ergodique*

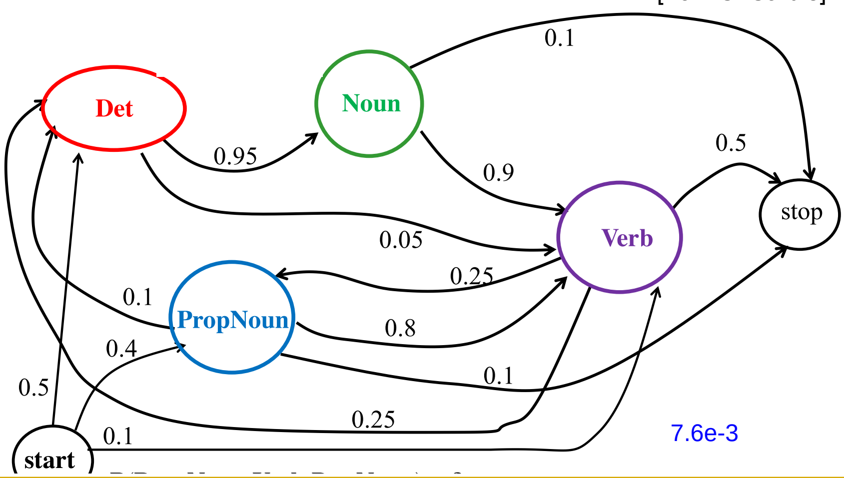
$$A = \begin{bmatrix} 0.4 & 0.3 & 0.3 \\ 0.2 & 0.6 & 0.2 \\ 0.1 & 0.1 & 0.8 \end{bmatrix}$$



solution: 1.536e-4

Mini-TD: POS part-of-speech tagging

[from C. Cardie]



7.6e-3

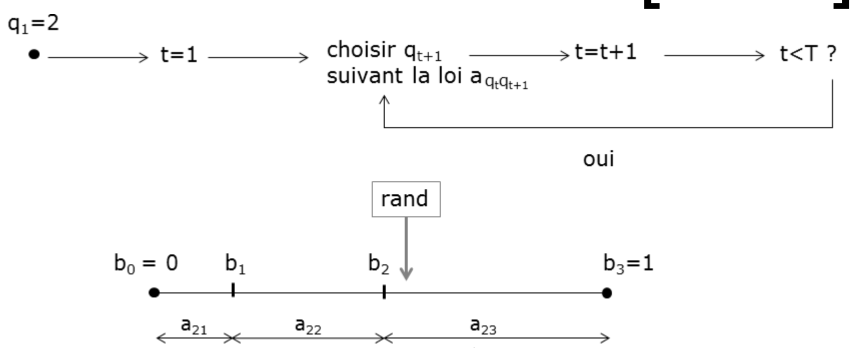
Probabilité de la séquence: Nom propre-verbe -déterminant – Nom ?

Générer une séquence d'états

- on part de l'état $q_1 = 2$
- générer séquence d'états de longueur T

suivant chaîne de Markov (matrice A)

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



$$F_{Q_{t+1}|q_t=i}(j) = P(Q_{t+1} \le j | q_t = i, \lambda) = \sum_{k=1}^{j} a_{ik}$$

générer une séquence d'états: mini-TD

- on donne
- générer séquence d'états de longueur T

suivant chaîne de Markov (matrice A)

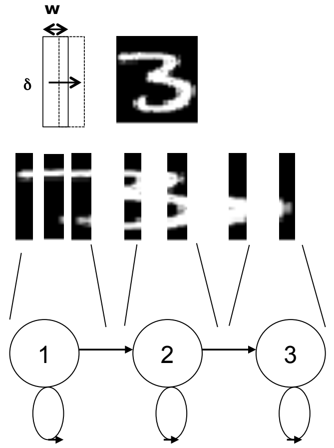
- π=[0.35 0.65]

$$A = \begin{bmatrix} 0.35 & 0.65 \\ 0.2 & 0.8 \end{bmatrix}$$

- *on tire les nombres aléatoires suivants:*
- *u1= 0.92 (q1)*
- *u2= 0.31*
- *u3= 0.1*
- *u4=0.4*
- *u5=0.01*

Modèles de Markov Cachés

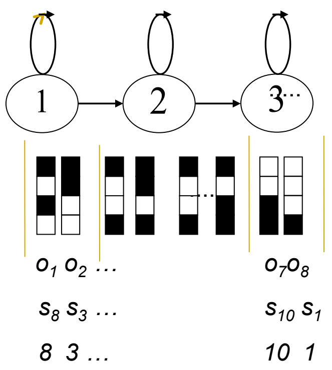
- une classe de forme
  - modèle λ
- combinaison de 2 processus stochastiques
  - un observé
  - un caché
- on n'observe pas la séquence d'états
  $q= q_1\, q_2 \ldots q_T$
- on observe la séquence d'observations
  $o=o_1\, o_2 \ldots o_T$
- les observations sont générées (émises) par les états



HMMs discrets

- ensemble de Q états discrets {1,2,..Q}
- ensemble de N symboles discrets
  - $\{s_1, s_2, s_3, \ldots s_N\} \rightarrow \{1,2,3,..,N\}$
- on observe $o=o_1\, o_2\, o_t \ldots o_T$
  - $o = s_8\, s_3\, s_{13}\, s_6\, s_8\, s_5\, s_{10}\, s_1$
  - o = 8 3 13 6 8 5 10 1
- o correspond à séquence d'états (cachés)
  - $q=q_1\, q_2\, q_t \ldots q_T$
  - q= 1 1 2 2 2 2 3 3

Q=3, N=16,



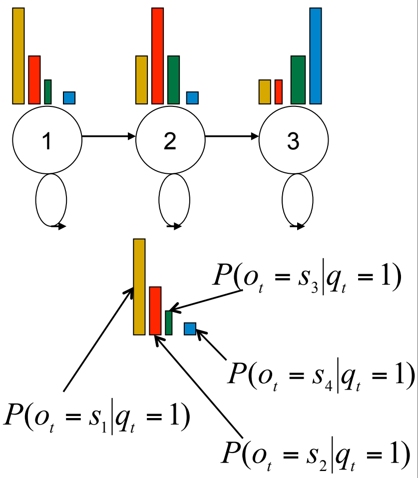
HMMs discrets

- HMM λ discret est défini par
  - π vecteur probabilités initiales
  - A: matrice transition
  - B : matrice des probabilités discrètes d'observation des symboles (dans les états)

$$\pi = (\pi_1, \pi_2, \ldots \pi_Q) \quad \pi_i = P(q_1 = i)$$

$$A = \{a_{ij}\} = P(q_t = j | q_{t-1} = i)$$

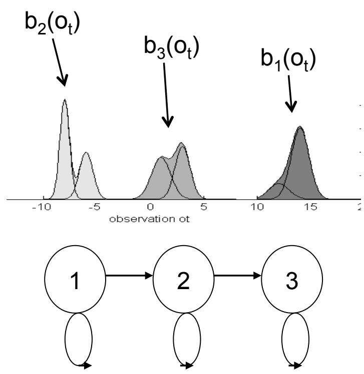
$$B = \{b_{ki}\} = P(o_t = s_k | q_t = i)$$



modèles de Markov cachés continus

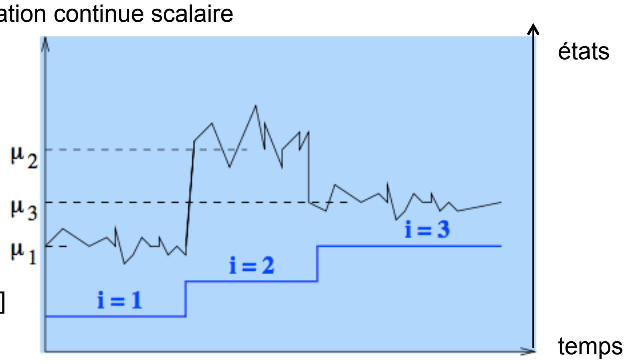
- HMM λ continu défini par :
- π vecteur de probabilités initiales
- A: matrice de transition entre états
- $b_i(o_t)$: densité de probabilité des observations dans état i, i=1,..Q

→ gaussienne ou mélange gaussiennes



modèle d'observations Gaussien

observation continue scalaire



[Sigelle 2012]

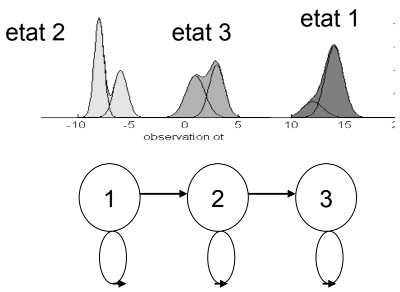
$$P(o_t \ / \ q_t = i, \ \lambda) = \frac{1}{\sqrt{2\pi}\sigma_i} \exp - \frac{(o_t - \mu_i)^2}{2\sigma_i{}^2}$$

modèle: inclut μi et σi, i=1,2, 3

mélange de gaussiennes

$$b_i(o_t) = \sum_{k=1}^{M} c_{ik} \mathcal{N}(o_t; \Sigma_{ik}, \mu_{ik}) \quad \forall i = 1, ...Q.$$

observations continues (scalaires ou vectorielles)

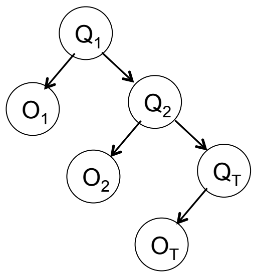


c_{ik}: poids de la kième loi gausssienne
du mélange
de M gaussiennes,
associée à l'état i

modèle λ: inclut c_{ik}, μ_{ik} et Σ_{ik}, i=1,2, 3 et k=1,..M

hypothèses fondamentales

- indépendance des observations conditionnellement aux états



$$P(o_1,..o_t...o_T|q_1...q_t...q_T,\lambda) = \prod_{t=1}^{T} P(o_t|q_t,\lambda)$$

- chaîne de Markov stationnaire (transitions entre états)

$$P(q_1, q_2, \ldots q_T) = P(q_1)P(q_2/q_1)\underbrace{P(q_3/q_2)}_{a_{q2\ q3}} \;\ldots\ldots\; P(q_T/q_{T-1})$$

hypothèses fondamentales

- probabilité jointe pour une séquence d'observations et un chemin d'états

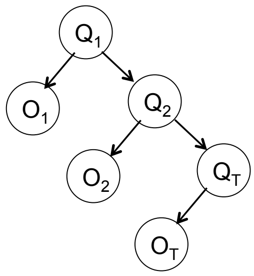
$$P(o_1,..o_t...o_T,q_1...q_t...q_T|\lambda) = \pi_{q_1} b_{q_1}(o_1) \prod_{t=2}^{T} a_{q_{t-1},q_t} P(o_t|q_t,\lambda)$$

$$= \pi_{q_1} b_{q_1}(o_1) \prod_{t=2}^{T} a_{q_{t-1},q_t} b_{q_t}(o_t)$$

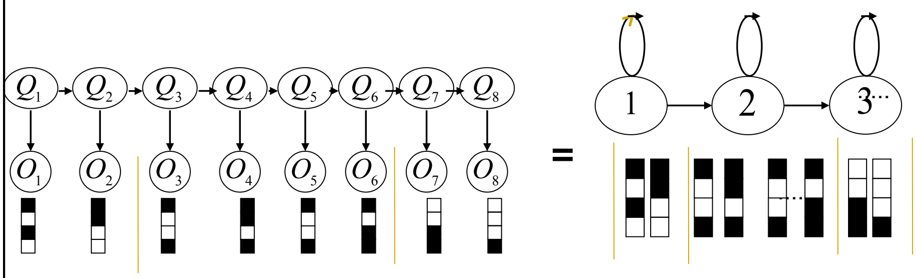
$$= P(o_1,..o_t...o_T|q_1...q_t...q_T,\lambda) P(q_1...q_t...q_T)$$

HMM / réseau bayésien

- un HMM est un cas particulier de réseau Bayésien
- les variables d'observations sont indépendantes connaissant leur variable parent (état)



HMM= cas particulier de DBN



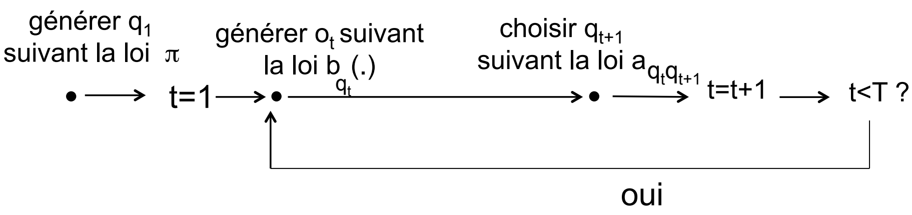
- HMM: Hidden Markov Model
- RBD: réseau Bayésien Dynamique de type arbre
- 1 state variable + 1 observation variable at each time step t

$(Q_t)_{1 \le t \le T}$: state variable (hidden)

$(O_t)_{1 \le t \le T}$: observation variable generated by state variable

générer une séquence d'observations

- générer la séquence d'états q1,….qT, puis générer la séquence observations à partir de chaque état
- ou générer q1 puis o1 (q1→ o1); générer q2 à partir de q1 (q1→ q2), puis o2 (q2→ o2), etc…



générer une séquence d'états: mini-TD

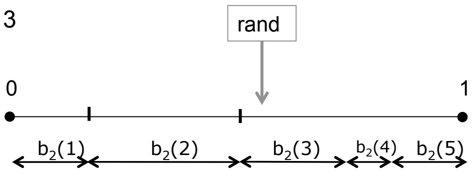
- on donne
- générer séquence d'états de longueur T

suivant chaîne de Markov (matrice A)

- π=[0.35 0.65] A= $\begin{bmatrix} 0.35 & 0.65 \\ 0.2 & 0.8 \end{bmatrix}$
- *on tire les nombres aléatoires suivants:*
- *u1= 0.92 (q1)*
- *u2= 0.31*
- *u3= 0.1*
- *u4=0.4*
- *u5=0.01*

étape 2 : générer les observations (discrètes)

- séquence états
  - q1= 1; q_2= 1; q_3= 1; q_4= 2; q_5=2; q_6=3;…….
- générer l'observation à t=4
  - q_4=2;
  - → o_4= 3



algorithme de décodage de Viterbi

- calcul de la vraisemblance
- séquence observation $o=o_1,\ldots o_T$

$$P(o\,|\lambda) = \sum_q P(o,q|\lambda)$$

- au lieu de sommer sur toutes les séquences d'états, recherche de la séquence optimale :

$$\hat{q} = \arg\max_q P(q,o|\lambda)$$

- puis estimer la vraisemblance par :

$$P(o\,|\lambda) \approx P(o,\hat{q}|\lambda)$$

HMM pour la reconnaissance des formes

- chaque clase m est modélisée par un modèle λ_m
- calcul de la vraisemblance du modèle λ_m pour une séquence d'observations $o=o_1\ldots\ldots o_T$ extraite d'une forme

$$P(o_1,..o_t...o_T|\lambda_m)$$

- attribution de la forme à la classe $\hat{m}$ telle que :

$$\hat{m} = \arg\max_m P(o_1,..o_t...o_T|\lambda_m)$$

27

HMM pour étiquetage morpho-syntaxique

- observations: mots
- séquence d'observation : suite de mots
- états cachés: Nom, pronom, verbe, etc.…
- modèle
  - probabilités de transitions entre éléments grammaticaux, bi-grams (tags)
  - probabilités d'observer les mots pour un élément grammatical donné (tag)
  - P(« le » I verbe) , P(« le » I pronom) etc.…

28

algorithme de décodage de Viterbi

- calcul de la vraisemblance
- séquence observation $o=o_1,\ldots o_T$

$$P(o|\lambda) = \sum_q P(o,q|\lambda)$$

- au lieu de sommer sur toutes les séquences d'états, recherche de la séquence optimale :

$$\hat{q} = \arg\max_q P(q,o|\lambda)$$

- puis estimer la vraisemblance par :

$$P(o|\lambda) \approx P(o,\hat{q}|\lambda)$$

Apprentissage en données complètes

- pour chaque modèle λ, estimer les paramètres
- base d'apprentissage
  - L séquences d'observation $o^{(l)}$, l=1….L
  - + séquences d'états associées
- séquence $o=o_1\ldots.o_T$ associée à séquence d'états $q=q_1\ldots..q_T$

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} 1_{\{q_t=i,q_{t+1}=j\}}}{\sum_{t=1}^{T-1} 1_{\{q_t=i\}}} \qquad \hat{b}_i(s_k) = \frac{\sum_{t=1}^{T} 1_{\{o_t=s_k,q_t=i\}}}{\sum_{t=1}^{T} 1_{\{q_t=i\}}}$$

Apprentissage en données complètes

- sur la base d'apprentissage totale

$$\hat{a}_{ij} = \frac{\sum_{l=1}^{L} \sum_{t=1}^{T(l)-1} 1_{\{q_t^{(l)}=i, q_{t+1}^{(l)}=j\}}}{\sum_{l=1}^{L} \sum_{t=1}^{T(l)-1} 1_{\{q_t^{(l)}=i\}}}$$

$$\hat{b}_i(s_k) = \frac{\sum_{l=1}^{L} \sum_{t=1}^{T(l)} 1_{\{o_t^{(l)}=s_k, q_t^{(l)}=i\}}}{\sum_{l=1}^{L} \sum_{t=1}^{T(l)} 1_{\{q_t^{(l)}=i\}}}$$

Apprentissage en données incomplètes

- estimer les paramètres, modèle λ
- on a une base d'apprentissage
  - L séquences d'observation $o^{(l)}$, l=1…L
- plus difficile (pas connaissance des états cachés)
- algorithme apprentissage
  - Baum-Welch
  - de Viterbi
  - basés sur EM

calcul de la vraisemblance

- algorithme de décodage de Viterbi
  pour séquence observation o=o_1,…o_T

$$P(o|\lambda) = \sum_q P(o,q|\lambda)$$

- au lieu de sommer sur toutes les séquences d'états, on ne considère que la séquence d'état optimale :

$$\hat{q} = \arg\max_q P(q,o|\lambda)$$

- puis on estime la vraisemblance du modèle par :

$$P(o|\lambda) \approx P(o,\hat{q}|\lambda)$$

décodage : algorithme de Viterbi

- $\delta_t(i)$: proba. (jointe) meilleure séquence partielle d'états aboutissant à l'état i au temps t et correspondant à la séquence partielle d'observations $o_1 \ldots o_t$.

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

- récurrence

$$P(q_1 q_2 \ldots q_t = i, q_{t+1} = j, o_1 o_2 \ldots o_t o_{t+1}|\lambda)$$
$$= P(o_{t+1}, q_{t+1} = j | o_1 \ldots o_t, q_1 \ldots q_t = i, \lambda) P(o_1 \ldots o_t, q_1 \ldots q_t = i|\lambda)$$
$$= P(o_{t+1}|q_{t+1} = j, \lambda) P(q_{t+1} = j | q_t = i, \lambda) P(o_1 \ldots o_t, q_1 \ldots q_t = i|\lambda)$$
$$\max_i P(q_1 q_2 \ldots q_t = i, q_{t+1} = j, o_1 o_2 \ldots o_t o_{t+1}|\lambda) = \max_i b_j(o_{t+1}) a_{ij} P(q_1 q_2 \ldots q_t = i, o_1 o_2 \ldots o_t|\lambda)$$

$$\boxed{\delta_{t+1}(j) = \max_i b_j(o_{t+1}) a_{ij} \delta_t(i) = b_j(o_{t+1}) \max_i a_{ij} \delta_t(i)}$$

$$P(o,\hat{q}) = \max_j \delta_T(j)$$

algorithme de décodage de Viterbi

- 1ere colonne: Initialisation

$$\delta_1(i) = P(q_1 = i, o_1) = b_i(o_1)\pi_i \quad i = 1,\ldots Q$$

- colonnes 2 à T : récursion

$$\delta_{t+1}(j) = b_j(o_{t+1}) \max_i a_{ij}\delta_t(i) \quad t = 1,\ldots \mathrm{T}-1, \mathrm{j} = 1,\ldots \mathrm{Q}$$

$$\varphi_{t+1}(j) = \arg\max_i a_{ij}\delta_t(i)$$ sauvegarde meilleur chemin (état précédent)

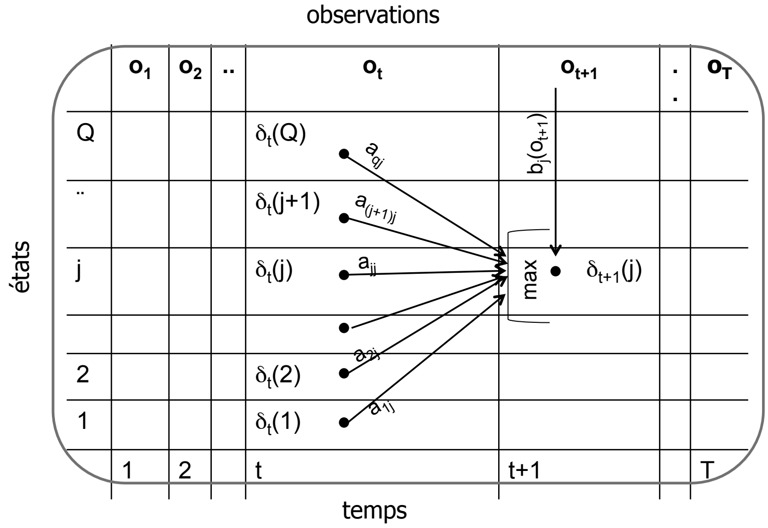
- terminaison

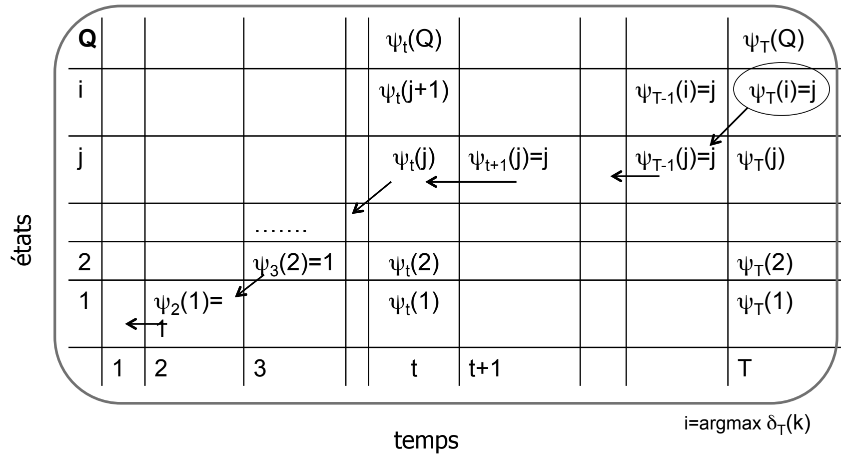
$$P(o,\hat{q}) = \max_j \delta_T(j)$$

$$\hat{q}_T = \arg\max_j \delta_T(j)$$

- backtrack

$$\hat{q}_t = \varphi(\hat{q}_{t+1}) \quad t = T-1, T-2, \ldots 1$$





variables forward-backward

$$P(o|\lambda) = \sum_i P(o, q_t = i|\lambda)$$

$$P(o, q_t = i|\lambda) = P(o_1 \ldots o_t, q_t = i, o_{t+1} \ldots o_T|\lambda)$$

$$= P(o_{t+1} \ldots o_T|o_1 \ldots o_t, q_t = i, \lambda) P(o_1 \ldots o_t, q_t = i|\lambda)$$

$$= \underbrace{P(o_{t+1} \ldots o_T|q_t = i, \lambda)}_{\beta_t(i)} \underbrace{P(o_1 \ldots o_t, q_t = i|\lambda)}_{\alpha_t(i)}$$

$$= \beta_t(i)\alpha_t(i)$$

$\beta_t(i)$: variable backward (analogue à λ)

$\alpha_t(i)$: variable forward (analogue à π)

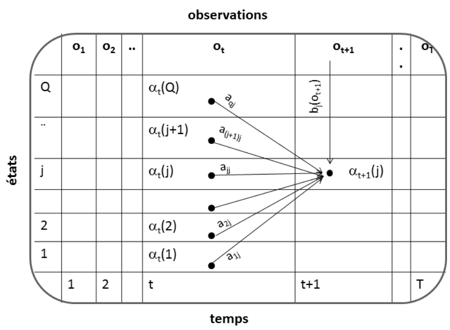
algorithme de décodage forward-backward

- calcul exact de la vraisemblance P(o| modele): Baum-Welch
- basé sur les variables forward et/ou backward

$$\alpha_1(j) = b_j(o_1)\pi_j$$

$$\alpha_{t+1}(i) = b_j(o_{t+1})\sum_{j=1}^{Q}\alpha_t(j)a_{ij}$$

$$P(o|\lambda) = \sum_{j=1}^{Q}\alpha_T(j)$$



Mini TD

$$A = \begin{bmatrix} 0.3 & 0.5 & 0.2 \\ 0 & 0.3 & 0.3 \\ 0 & 0 & 1 \end{bmatrix} \qquad \pi = \begin{bmatrix} 0.6 \\ 0.4 \\ 0 \end{bmatrix}$$

$$B = \begin{bmatrix} 1 & 0 \\ 0.5 & 0.5 \\ 0 & 1 \end{bmatrix}$$

calculer P(aabb)

conclusion

- chaînes de Markov
- modèles de Markov Cachés
  - apprentissage cas discret et données complètes
  - décodage de Viterbi
  - lien entre réseaux bayésiens dynamiques et HMMs
- données incomplètes

  algorithme EM (Viterbi, Baum-Welch)

références

- M. Sigelle, Bases de la Reconnaissance des Formes: Chaînes de Markov et Modèles de Markov Cachés, chapitre 7, Polycopié Telecom ParisTech, 2012.
- L. Likforman-Sulem, E. Barney Smith, Reconnaissance des Formes: théorie et pratique sous matlab, Ellipses, TechnoSup, 2013.
- L. Rabiner, A tutorial on Hidden Markov Models and selected applications in Speech Recognition, proc. of the IEEE, 1989.