Introduction to graphical models :

Part II Hidden Markov Models

June 2018

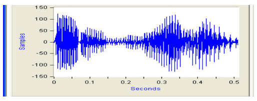
Laurence Likforman-Sulem
IMT/Telecom ParisTech/University Paris-Saclay
likforman@telecom-paristech.fr

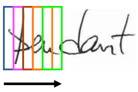
Overview

- Part I : Introduction to Graphical Models
- Part II : Hidden Markov Models
  - discretes, continuous
  - generative models
  - Decoding : Viterbi, Baum-Welch
  - Training : Viterbi, Forward-Backward

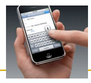
applications

- HMMs
  - speech, handwriting recognition
  - face recognition in videos
  - Natural Language Processing (NLP)
    - Part-of-speech tagging
    - lexical correction

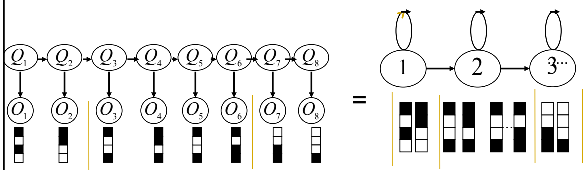




THE→ TGE



HMM= special case of DBN



- HMM: Hidden Markov Model
- DBN: tree
- 1 state variable + 1 observation variable at each time step t

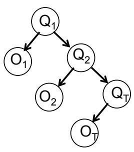
$(Q_t)_{1 \le t \le T}$: state variable (hidden)

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

joint probability

- factorization provided by network

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



joint probability

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

- state sequence probability
- $P(q_1...q_t...q_T)$
- probability of observation sequence given the state sequence

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

Stochastic process

- set of random variables $q_1, q_2, \ldots, q_T$
- indexed at time t=1, 2, ….T

- notation
  - q_t: random state variable at time t
    q(t) or q_t
  - values of q(t) belong to finite set S
    S={1,2, ….Q}
  - $P(q_t=i)$: probability for observing state i at time t

states may be : pollution indexes, météo (sunny, rainy, cloudy), word tags (verb,name, pronoun….)(NLP)

Stochastic process

- evolution of process
  - from initial state q_1
  - chain of state transitions
    - $q_1 \rightarrow q_2 \ldots \rightarrow q_t \quad t \le T$
- state sequence probability
  $P(q_1, q_2, \ldots q_T) = P(q_T / q_1, q_2, \ldots q_{T-1}) P(q_1, q_2, \ldots q_{T-1})$
  $= P(q_T | q_1, q_2, \ldots q_{T-1}) P(q_{T-1} | q_1, q_2, \ldots q_{T-2}) P(q_1, q_2, \ldots q_{T-2})$
  $= P(q_1)P(q_2 / q_1)P(q_3 / q_1,q_2) \ \ldots\ldots \ P(q_T | q_1, q_2, \ldots q_{T-1})$
- model: transition probabilities +initial state probability $P(q_1)$

Markov chain (discrete time)

- Markov property (order k): limits dependencies
  - $P(q_t| q_1, q_2, \ldots q_{t-1}) = P(q_t| q_{t-k} \ldots q_{t-1})$
  - k=1 or 2
- *case k=1*
  - $P(q_t | q_1, q_2, \ldots q_{t-1}) = P(q_t| q_{t-1})$
  - $P(q_1, q_2, \ldots q_T) = P(q_1)P(q_2 / q_1)P(q_3 / q_2) \ \ldots\ldots \ P(q_T | q_{T-1})$
  - → state transition probabilities

Stationary Markov chain

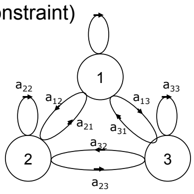
- transition probabilities do not depend on time
  - $P(q_t = j \mid q_{t-1} = i) = P(q_{t+k} = j \mid q_{t+k-1} = i) = a_{ij}$
  - a_{ij}= probability to move from state i to state *j*
- model for a stationary Markov chain
  - transition probability matrix
    $A=[a_{ij}] \qquad i=1,\ldots Q, \ j=1,\ldots.Q$
  - initial probability vector
  - $\Pi = [\pi_i] \qquad i=1,\ldots Q$
    $\pi_i = P(q_1 = i)$
  - constrains : $0 \le \pi_i \le 1 \quad 0 \le a_{ij} \le 1$

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

Model topology : ergodic / left-right

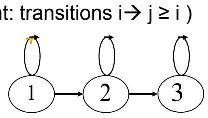
- ergodic model (without constraint)

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



- left-right model (constraint: transitions i→ j ≥ i)

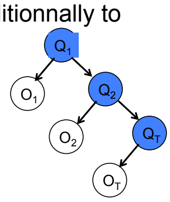
$$A = \begin{bmatrix} 0.4 & 0.6 & 0 \\ 0 & 0.8 & 0.2 \\ 0 & 0 & 1 \end{bmatrix}$$



HMM modeling

- observations are independent conditionnally to states

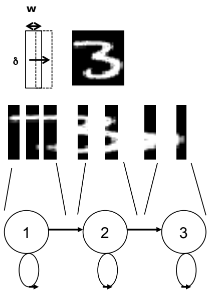
$$P(o_1,..o_t...o_T | q_1...q_t...q_T, \lambda) = \prod_{t=1}^{T} P(o_t | q_t, \lambda)$$



- parameters
  - observation probabilities $P(o_t \mid q_t)$
  - transition probabilities
    $P(q_t = j \mid q_{t-1} = i)$ and $P(q_1=i)$

HMMs for pattern recognition

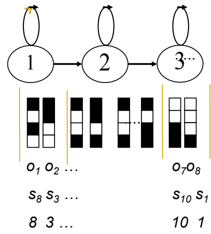
- one model for each class
  - modèle λ
- combinaison of 2 stochastic processes
  - one observed
  - one hidden
- state sequence q hidden
- $q = q_1\ q_2\ \ldots q_T$
- observation sequence o (discrete or continuous)
  $o = o_1\ o_2\ \ldots o_T$
- observations are generated by the states



discrete HMMs

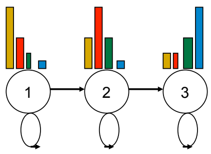
- set of Q discrete states {1,2,..Q}
- set of N observed symbols
  - $\{s_1, s_2, s_3, \ldots s_N\} \rightarrow \{1,2,3,..,N\}$
- observation sequence
  - $o = o_1\ o_2\ o_t \ldots o_T$
  - $o = s_8\ s_3\ s_{13}\ s_6\ s_8\ s_5\ s_{10}\ s_1$
  - o = 8 3 13 6 8 5 10 1
- o corresponds to hidden state sequence
  - $q = q_1\ q_2\ q_t \ldots q_T$
  - q= 1 1 2 2 2 2 3 3

Q=3, N=16=2^4,

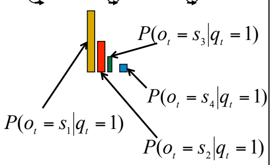


discrete HMMs

- a discrete HMM λ is defined by:
- π initial probability vector
- A: transition matrix
- B : observation probability matrix (probability of observing each symbol in each state)



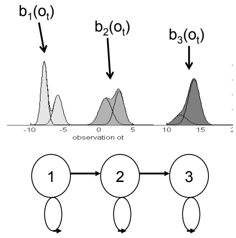
$$\pi = (\pi_1, \pi_2, \ldots \pi_Q) \quad \pi_i = P(q_1 = i)$$
$$A = \{a_{ij}\} = P(q_t = j \mid q_{t-1} = i)$$
$$B = \{b_{ki}\} = P(o_t = s_k \mid q_t = i)$$



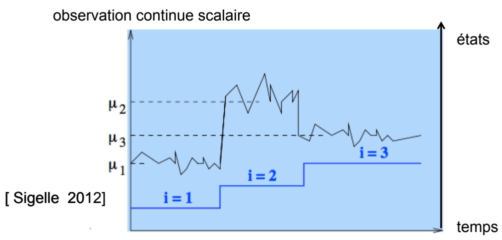
Continuous Markov models

- λ: continuous HMM defined by
- π initial probability vector
- A: state transition matrix
- probability density function (pdf)
- $b_i(o_t)$: probability of observing o_t in state i, i=1,..Q

(Gaussian or Gaussian mixture)



modèle d'observations Gaussien



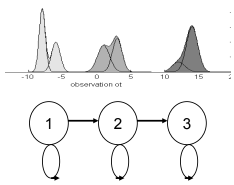
$$P(o_t \ / \ q_t = i, \ \lambda) = \frac{1}{\sqrt{2\pi}\sigma_i} \exp - \frac{(o_t - \mu_i)^2}{2\sigma_i{}^2}$$

modèle: inclut μi et σi, i=1,2, 3

mélange de gaussiennes

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

observations continues (scalaires ou vectorielles)



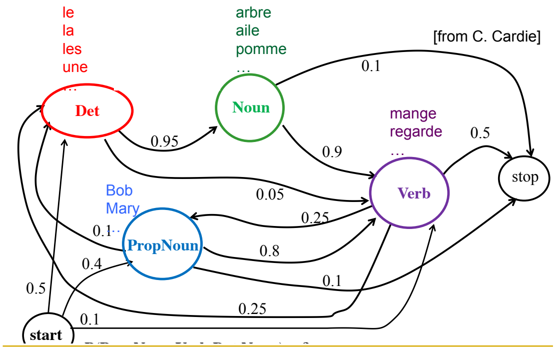
c_{ik}: poids de la kième loi gausssienne du mélange de M gaussiennes, associée à l'état i

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

example HMM : tagging

- observations: words
- observation sequence : sequence of words
- hidden states (tags) : name, pronoun, verb, etc.…
- model:
  - state transitions : tag bi-grams
  - observation probabilities according to tags (states) P(« the » I verb) , P(« the » I pronoun) etc.…

Mini-TD: POS part-of-speech tagging

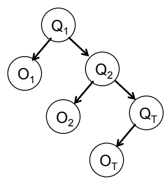


- HMM : generating state and observation sequences

generating an observation sequence

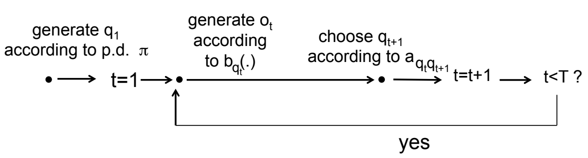
- fondamental assumption
- (conditional) independence of observations given states

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

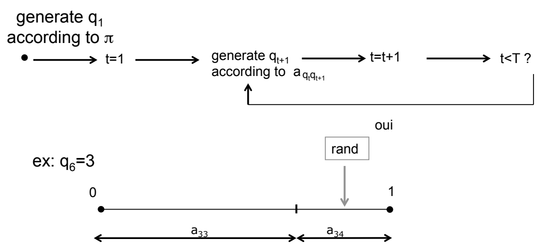


generating an observation sequence

- generating an observation sequence of length T
  - genererate a sequence of hidden states.
  - from each state, generate one observation.



step 1 : generate a state sequence



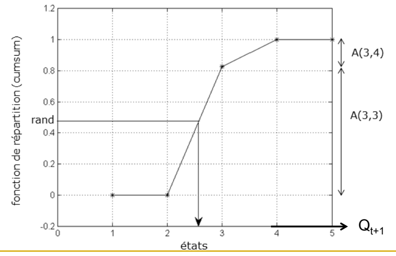
$a_{31}=0;\ a_{32}=0;$

→ $q_7=4$

$$\pi = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad A = \begin{bmatrix} 0.1 & 0.9 & 0 & 0 \\ 0 & 0.15 & 0.85 & 0 \\ 0 & 0 & 0.8 & 0.2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

generating random samples: inverse of cumulative distribution function

$$A = \begin{bmatrix} 0.1 & 0.9 & 0 & 0 \\ 0 & 0.15 & 0.85 & 0 \\ 0 & 0 & 0.8 & 0.2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



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

- HMM for pattern recognition

HMM models for pattern recognition

- each class is represented by an HMM model λ_m
- for a given observation sequence (pattern) $o = o_1\, o_2 \ldots o_T$ compute likelihood of model λ

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

- assign the pattern to class $\hat{m}$ such as:

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

computing likelihood: Viterbi algo.

- for observation séquence $o = o_1, \ldots o_T$

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

- instead of summing over all state sequences, search for the optimal state sequence :

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

- then estimate likelihood by :

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

decoding with Viterbi algorithm

- $\delta_t(i)$: proba. (joint) of best partial state sequence ending at t on state i and corresponding to the partial observation sequence $o_1 \ldots o_t$.

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

- recurrence

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

$$\max_{q1q2..qt} P(q_1 q_2 \ldots q_t = i, q_{t+1} = j, o_1 o_2 \ldots o_t o_{t+1} | \lambda) = \max_i b_j(o_{t+1}) a_{ij} \delta_t(i)$$

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

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

Viterbi decoding algorithm

- 1st column: Initialization

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

- columns 2 to T : recursion

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

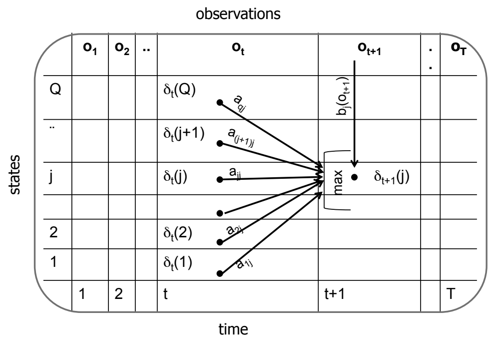
$$\varphi_{t+1}(j) = \arg\max_i a_{ij} \delta_t(i)$$ save best path (preceding state)

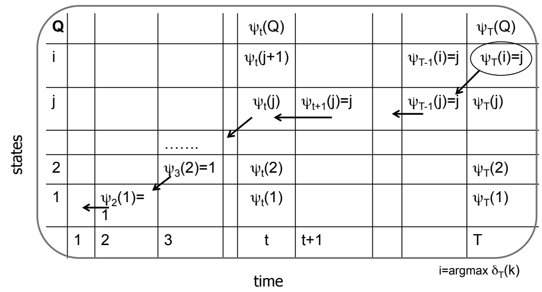
- termination $P(o, \hat{q}) = \max_j \delta_T(j)$

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

- backtrack

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





variables forward-backward

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

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

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

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

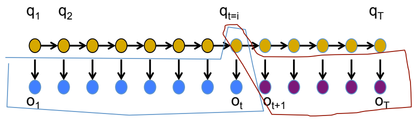
$$= \beta_t(i)\alpha_t(i)$$

$\beta_t(i)$: backward variable (similar *to* λ)

$\alpha_t(i)$: forward variable (similar to π)

forward-backward variables

$$P(o \,|\lambda) = \sum_i P(o, q_t = i|\lambda) = \sum_{i=1}^{Q} \alpha_t(i)\beta_t(i)$$



$\beta_t(i)$: variable backward

$\alpha_t(i)$: variable forward

$$P(o|\lambda) = \sum_{i=1}^{Q} \alpha_t(i)\beta_t(i)$$

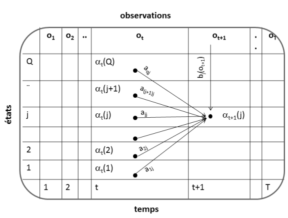
algorithme de décodage forward-backward

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

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

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

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



other variables: γ and ξ

$$\gamma_t(i) = P(q_t = i|O) = \frac{\alpha_t(i)\beta_t(i)}{P(O)} = \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=1}^{Q} \alpha_t(j)\beta_t(j)} \qquad i = 1, \ldots, Q$$

- $\gamma_t(i)$: a posteriori probability that observation o_t is in state i
- soft alignment of observation séquence O to states
- permit to compute state occupation counts

other variables: γ et ξ

$$\xi_t(i,j) = P(q_t = i, q_{t+1} = j | o, \lambda) = \frac{P(o, q_t = i, q_{t+1} = j|\lambda)}{P(o|\lambda)}$$

- $\xi_t(i,j)$: probability that observation o_t is in state i and observation o_{t+1} is in state j, given whole sequence o (2-state formula)

$$\xi_t(i,j) = P(q_t = i, q_{t+1} = j | o, \lambda) = \frac{\beta_{t+1}(j) b_j(o_{t+1}) a_{ij} \alpha_t(i)}{\sum_{k=1}^{Q} \alpha_t(k)\beta_t(k)}$$

PART III: PARAMETER ESTIMATION

training : complete data

- for each model λ, estimate HMM parameters
- training database
  - L observation sequences $o^{(l)}$, l=1….L
  - + associated state sequences
- sequence $o=o_1 \ldots o_T$ associated to state sequence $q=q_1 \ldots q_T$

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

training : complete data

- whole training database

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

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

training : complete data

continuous HMM, one-dimensional gaussian

$$\widehat{\mu_i} = \frac{\sum_{l=1}^{L} \sum_{t=1}^{T^{(l)}} o_t^{(l)} \, \mathbb{1}_{q_t^{(l)}=i}}{\sum_{l=1}^{L} \sum_{t=1}^{T^{(l)}} \mathbb{1}_{q_t^{(l)}=i}}$$

$$\widehat{(\sigma_i)^2} = \frac{\sum_{l=1}^{L} \sum_{t=1}^{T^{(l)}} \left(o_t^{(l)} - \widehat{\mu_i}\right)^2 \mathbb{1}_{q_t^{(l)}=i}}{\sum_{l=1}^{L} \sum_{t=1}^{T^{(l)}} \mathbb{1}_{q_t^{(l)}=i}}$$

training : incomplete data

- training database
  - L observation sequences $o^{(l)}$, l=1…L
- no knowledge about states
  - more difficult
- training algorithm
  - Baum-Welch
  - Viterbi

training : incomplete data

- Viterbi training
  - observation sequence o
  - decoding with Viterbi decoding algorithm
    - → optimal state sequence q*
  - case «complete data» : (o,q*)

Baum-Welch training

$$\hat{\pi}_i = \frac{\sum_{l=1}^{L} \gamma_1^{(l)}(i)}{L}$$

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

$$\hat{b}_i(s_k) = \frac{\sum_{l=1}^{L} \sum_{t=1 \; et \; o_t^{(l)}=s_k}^{T(l)} \gamma_t^{(l)}(i)}{\sum_{l=1}^{L} \sum_{t=1}^{T(l)} \gamma_t^{(l)}(i)}$$

discrete HMM

Baum-Welch training

one-dimensional gaussian,

$$\widehat{\mu_i} = \frac{\sum_{t=1}^{T} o_t \gamma_t(i)}{\sum_{t=1}^{T} \gamma_t(i)}$$

$$\widehat{(\sigma_i)^2} = \frac{\sum_{t=1}^{T} (o_t - \widehat{\mu_i})^2 \; \gamma_t(i)}{\sum_{t=1}^{T} \gamma_t(i)}$$

algorithm EM: expectation maximization

iterative algorithm
1) initialization
2) compute α, β, γ, ξ,
3) update parameters

$$a_{ij}^{(\mathbf{n+1})} = \frac{\sum_{l=1}^{L} \sum_{t=1}^{T^{(l)}-1} P(q_t^{(l)} = i, \; q_{t+1}^{(l)} = j \; / \; o^{(l)}, \lambda^{(\mathbf{n})})}{\sum_{l=1}^{L} \sum_{t=1}^{T^{(l)}-1} P(q_t^{(l)} = i \; / \; o^{(l)}, \lambda^{(\mathbf{n})})}$$

$$\mu_i^{(\mathbf{n+1})} = \frac{\sum_{l=1}^{L} \sum_{t=1}^{T^{(l)}} o_t^{(l)} \; P(q_t^{(l)} = i \; / \; o^{(l)}, \lambda^{(\mathbf{n})})}{\sum_{l=1}^{L} \sum_{t=1}^{T^{(l)}} P(q_t^{(l)} = i \; / \; o^{(l)}, \lambda^{(\mathbf{n})})}$$

conclusion

- HMMs
  - special case of Bayesian network
- hidden state sequence
  - emit observations which are conditionnaly independant
  - generative approach for sequence modelling
- decoding
  - Viterbi decoding algorithm
  - Baum-Welch decoding algorithm
- training
  - with complete data
  - with incomplete data
    - algorithm EM (Viterbi, Baum-Welch)

conclusion (cont.)

- for pattern recognition
  - deep learning approaches ouptperform HMM approaches
    - convolutional networks, Recurrent networks
  - However : training data must be large

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.
- Rabiner A tutorial on hidden Markov models and selected applications in speech recognition, Proceedings of the IEEE, Volume: 77 Issue: 2, 1989.