SD-TSIA214 Machine Learning for Text Mining

Hidden Markov Models

March 2024

Laurence Likforman-Sulem
Telecom Paris/Institut Polytechnique de Paris
likforman@telecom-paristech.fr

1

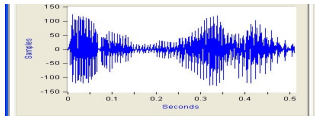
Overview

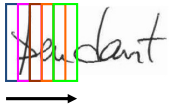
- Part I : Markov Chains
  - stochastic processes, parameters
- Part II : Hidden Markov Models
  - discrete, continuous
- Part III generative models
- Part IV Decoding : Viterbi, Baum-Welch
- Part V Training : Viterbi, Forward-Backward

2

applications

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





THE→ TGE

3

PART I: MARKOV CHAINS

4

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
    - $q_1 \rightarrow q_2 \ldots \rightarrow q_t$ t<= T

- state sequence probability

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

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

$= P(q_1) P(q_2 / q_1) P(q_3 / q_1, q_2) \ldots P(q_T \mid q_1, q_2, \ldots q_{T-1})$

- 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 (bigrams)*
  - $P(q_t | q_1, q_2, \ldots q_{t-1}) = P(q_t | q_{t-1})$
  - $P(q_1, q_2, \ldots q_T) = P(q_1)P(q_2 / q_1)P(q_3 / q_2) \ldots .. P(q_T | q_{T-1})$
  - → transition probabilities between 2 states

Stationary Markov chain

- transition probabilities do not depend on time
  - $P(q_t = j | q_{t-1} = i) = P(q_{t+k} = j | q_{t+k-1} = i) = a_{ij}$
  - a_{ij}= probability to move from state i to state *j*

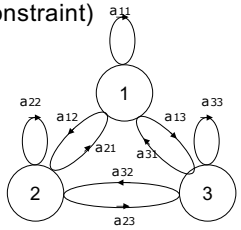
- parameters of a stationary Markov chain model
  - transition probability matrix
    $A=[a_{ij}]$ $i=1,\ldots Q, j=1,\ldots Q$
  - initial probability vector
  - $\Pi = [\pi_i]$ $i=1,\ldots Q$
    $\pi_i = P(q_1 = i)$
  - constraints : $0 <= \pi_i <= 1$ $0 <= a_{ij} <= 1$

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

Model topology : ergodic / left-right

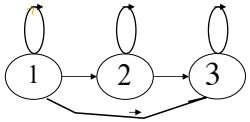
- ergodic model (without constraint)

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



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

$$A = \begin{bmatrix} 0.4 & 0.5 & 0.1 \\ 0 & 0.8 & 0.2 \\ 0 & 0 & 1 \end{bmatrix}$$



9

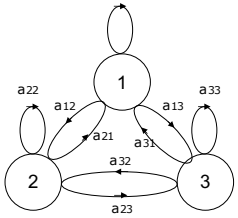
9

Stationary Markov chain model : mini TD

- 3-state Markov chain
  - 1: rainy (r), 2: cloudy (c) , 3: sunny (s)
- observation at t=1: q_1= s , compute the probability to observe the next 7 days the weather (states)

  s s s r r s c s
- *t=1 t=2*
- *ergodic model*

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

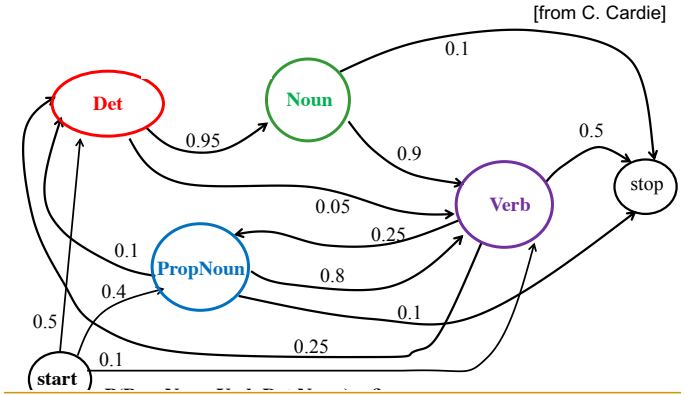


10

10

Mini-TD: POS part-of-speech tagging

[from C. Cardie]



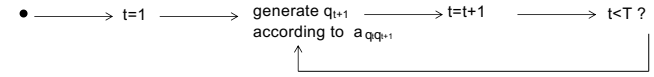
Probability of sequence : Proper name -verb -determiner - name ?

11

11

generate a state sequence

generate q_1 according to π



oui

ex: $q_6=3$

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

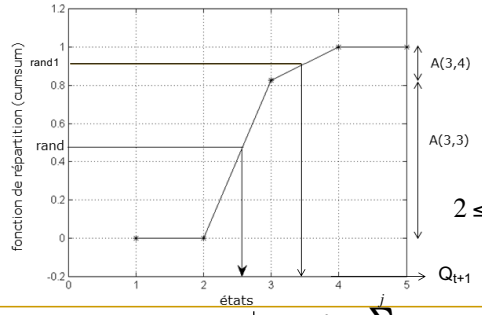
→ $q_7=4$

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

12

generating random samples: inverse of cumulative distribution function

$$A = \begin{bmatrix} 0.1 & 0.9 & 0 & 0 \\ 0 & 0.15 & 0.85 & 0 \\ 0 & 0 & 0.81 & 0.19 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



$$2 \le F^{-1}(rand) \le 3$$

$$F_{Q_{t+1}|q_t=i}(j) = P(Q_{t+1} \le j \,|\, q_t = i, \lambda) = \sum_{k=1}^{j} a_{ik}$$ (here q_t=3)

13

générer une séquence d'états: mini-TD

- generate a length T=5 state sequence

  according to Markov chain model parameters (matrix A, vector pi)

- π=[0.3 0.7] A= $\begin{bmatrix} 0.35 & 0.65 \\ 0.2 & 0.8 \end{bmatrix}$
- *random numbers sampled at each time step are :*
- *u1= 0.92 (q1)*
- *u2= 0.31*
- *u3= 0.1*
- *u4=0.4*
- *u5=0.01*

14

PART II: HIDDEN MARKOV MODELS

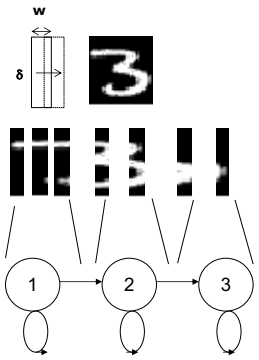
15

15

HMMs for pattern recognition

- one model for each class c
  - modèle λ_c
- 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)

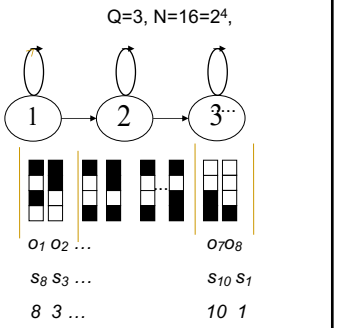
  o=$o_1\ o_2\ \ldots o_T$
- observations are generated by the states



16

discrete HMMs

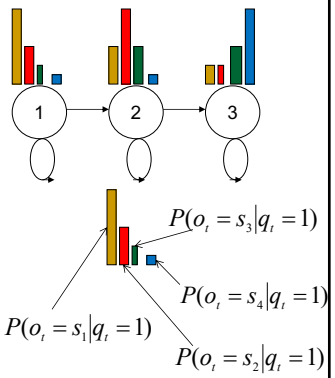
- set of Q discrete states {1,2,..Q}
- set of N observed symbols
  - $\{s_1, s_2, s_3, ,\ldots s_N\} \rightarrow \{1,2,3,..,N\}$
- observation sequence o
  - $o=o_1\ o_2\ o_t \ldots o_T$
  - $o = s_8\ s_3\ s_{13}\ s_6\ s_8\ s_5\ s_{10}\ s_1$
  - o = 8 3 13 6 8 5 10 1
- q corresponds to hidden state sequence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



17

discrete HMMs

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



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

$$A = \{a_{ij}\}_{1 \le i,j \le Q} \; a_{ij} = P(q_t = j \,|\, q_{t-1} = i)$$

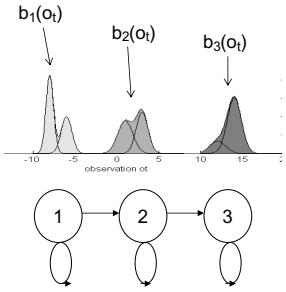
$$B = \{b_{ki}\}_{\substack{1 \le k \le N \\ 1 \le i \le Q}} \; b_{ki} = P(o_t = s_k \,|\, q_t = i)$$

18

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)

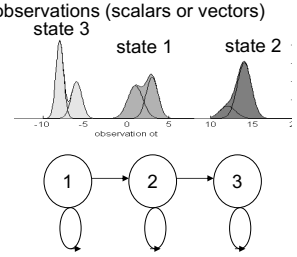


19

Gaussian Mixtures

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

continuous observations (scalars or vectors)



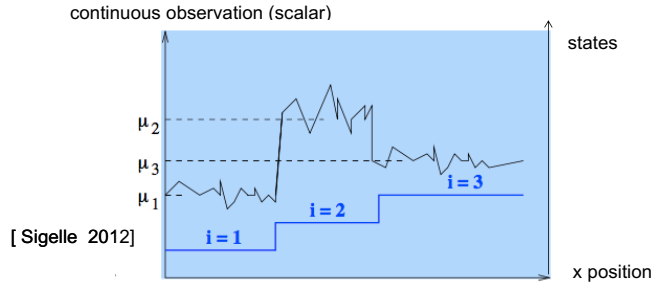
Mixture of M gaussians, associated to state i

c_{ik}: weight of kth gaussian distribution.

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

20

Gaussian model



[Sigelle 2012]

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

model: includes μi and σi, i=1,2, 3

Laurence Likforman-Telecom ParisTech

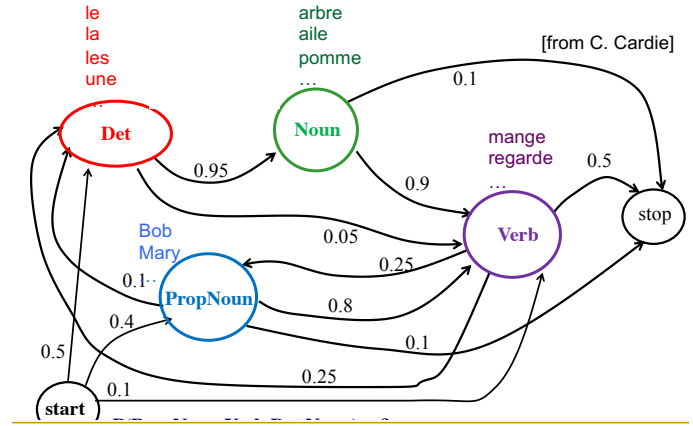
21

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.…

22

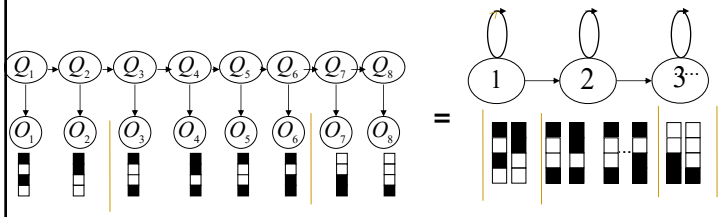
POS part-of-speech tagging



23

23

HMM= special case of DBN



- HMM: Hidden Markov Model
- DBN: Dynamic Bayesian Network (here a tree)
- 1 state variable + 1 observation variable at each time step

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

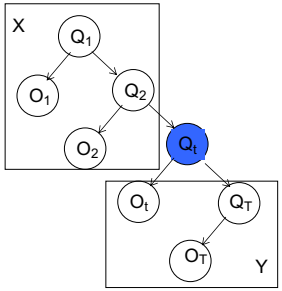
$(O_t)_{1 \le t \le T}$: observed variable « generated» by state variable

24

24

independence of set of nodes

- set of nodes X and set of nodes Y are independent given (observed) node Qt

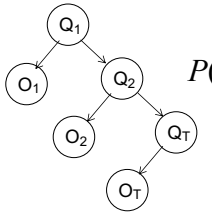


25

25

conditional independence of variables

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



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

- stationary Markov chain (state transitions)

$$P(q_1, q_2, \ldots q_T)= P(q_1)P(q_2 / q_1)P(q_3 / q_2) \ \ldots..\ P(q_T \mid q_{T-1})$$

26

factorization

- joint probability for an observation sequence and a state path

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

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

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

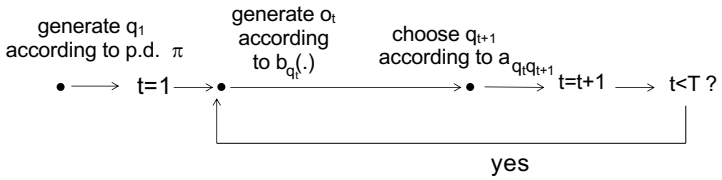
27

- **Part 3 : generative HMM model**

28

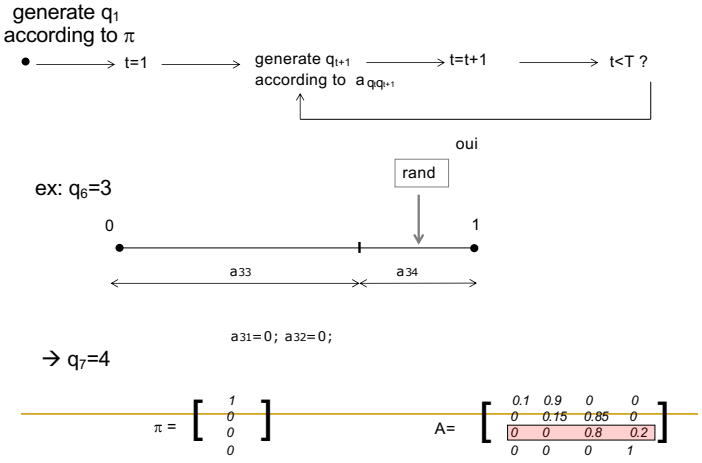
generating an observation sequence

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



29

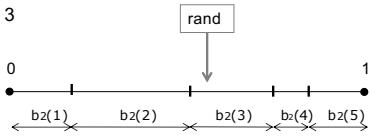
step 1 : generate a state sequence



30

step 2 : generate observations (discrete)

- state sequence
  - q1= 1; q_2= 1; q_3= 1; q_4= 2; q_5=2; q_6=3;…….
- generate observation at t=4
  - q_4=2;
  - → o_4= 3



- N=5 (number of symbols)
- Q=4 (nomber of states)

states

$$B=\begin{bmatrix} 0.3 & 0. & 0.1 & 0.2 \\ 0.1 & 0.7 & 0.2 & 0.1 \\ 0.5 & 0.1 & 0.2 & 0.1 \\ 0.1 & 0.1 & 0.3 & 0.3 \\ 0 & 0.1 & 0.2 & 0.3 \end{bmatrix}$$

observations

31

Part 4 : HMM Decoding

32

HMM models for pattern recognition

- each class *m* 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 each model λ_m

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

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

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

33

application of Viterbi decoding to POS

- POS tagging (Part of Speech)
- Compute the optimal hidden state sequence
- « Bob mange la pomme »
  → 'Proper Name' 'Verb' 'determinant' 'Noun'

34

computing likelihood: Viterbi algo.

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

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

summing over all state sequences

- instead, search for the optimal state sequence

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

- then estimate likelihood by : $P(o\,|\lambda\,) \approx P(o,\hat{q}|\lambda)$

35

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 $o_1\dots o_t$.

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

- recurrence

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

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

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

36

Viterbi decoding algorithm

- 1st column: Initialization

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

- columns 2 to T : recursion

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

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

- termination

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

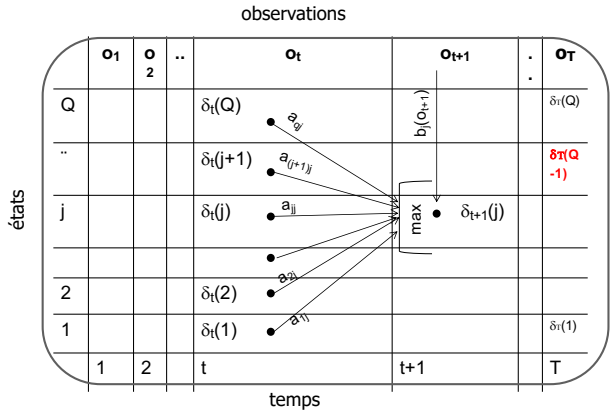
$$\hat{q}_T = \arg\max_j \delta_T(j)$$

- backtrack

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

37

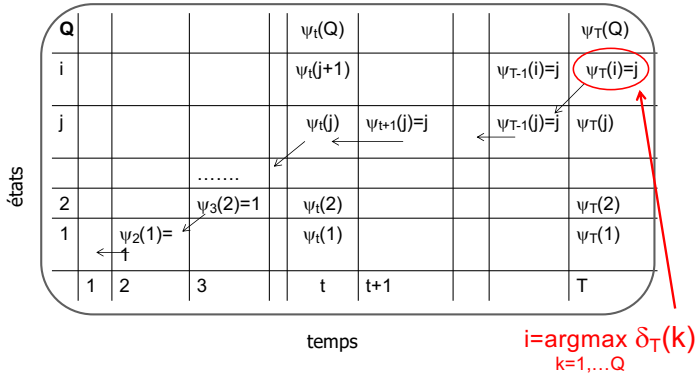
calcul des deltas



Laurence Likforman-Telecom Paris 38

38

backtrack



Mini TD

HMM model λ whose parameters are :

$$A = \begin{bmatrix} 0.3 & 0.5 & 0.2 \\ 0 & 0.3 & 0.7 \\ 0 & 0 & 1 \end{bmatrix} \qquad \pi = \begin{bmatrix} 0.6 \\ 0.4 \\ 0 \end{bmatrix}$$

$$B = \underbrace{\begin{bmatrix} 1 & 0.5 & 0 \\ 0 & 0.5 & 1 \end{bmatrix}}_{\text{states}} \Big\} \text{observations}$$

compute likelihood $P(aabb|\lambda)$

forward-backward variables

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

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

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

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

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

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

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

$a^2 + b^2 = c^2$ t(i)

forward-backward variables

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

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

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

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

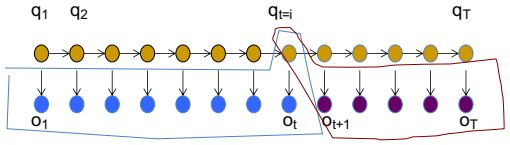
$$= \beta_t(i)\alpha_t(i)$$

$\alpha_t(i)$ forward variable

$\beta_t(i)$ backward variable

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

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

43

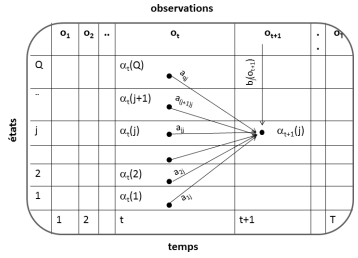
forward-backward decoding

- exact computation of likelihood P(o| model): Baum-Welch
- based on forward and/or backward variables

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

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

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



44

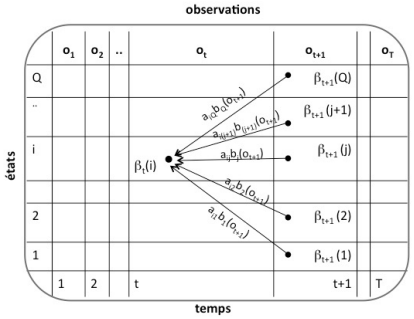
forward-backward decoding : backward algorithm

- calcul exact de la vraisemblance P(o| modele): Baum-Welch

$$\beta_T(i) = 1$$

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

$$P(O|\lambda) = \sum_{j=1}^{Q} \beta_1(j) \pi_j b_j(o_1)$$



45

other variables: γ and ξ

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

- $\gamma_t(i)$: a posteriori probability that observation o_t is in state i
- can be used for decoding (local) : $\hat{q}_t = \arg\max_j \gamma_t(j)$
- soft alignment of observation séquence O to states
- permits to compute state occupation counts

46

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)

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

47

PART 5: PARAMETER ESTIMATION

48

training : complete data

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

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

49

training : complete data

- whole training dataset

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

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

50

training : complete data

continuous HMM, one-dimensional gaussian

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

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

51

training : incomplete data

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

52

training : incomplete data

- Viterbi training
  - observation sequence o
  - parameters initialized
    - for instance observations equally distributed into states
  - decoding with Viterbi decoding algorithm
    → optimal state sequence q*
  - case «complete data» : (o,q*)
  - iterate

53

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

54

Baum-Welch training

one-dimensional gaussian,

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

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

55

Forward-backward EM algorithm: expectation maximization

1) Initialization of A and B and π

2) Iterate

E-step

compute $\gamma_t(i)$ and $\xi_t(i,j)$ for all t ,i and j

M-step

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

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

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

3) Return A, B, π

56

algorithm EM: expectation maximization

iterative algorithm
1) initialization
2) compute $\alpha, \beta, \gamma, \xi,$
3) update parameters

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

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

57

conclusion

- Markov chains
- HMMs
  - 2 random processes, one observed, one hidden
  - special case of Bayesian network
- Generative approach for sequence modelling
  - emit observations which are conditionnaly independant
- Decoding
  - Viterbi decoding algorithm
  - Baum-Welch decoding algorithm
- Training
  - with complete data
  - with incomplete data
    - algorithm EM (Viterbi, Baum-Welch)

58

conclusion (cont.)

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

59

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.

60