HMM Tutorial 2018 | part I   A. Stanic

⊕ For what kind of problems do we use HMMs?

Consider an example

Weather in Lugano can be

$S \in \{Rainy, Cloudy, Sunny\}$
$S \in \{R, C, S\}$

⊕ Therefore, weather takes values from a $\boxed{\text{discrete set of}}$ $\boxed{\text{states.}}$

⊕ Starting probability of a weather is also a number

$\boxed{P(Q_1 = R) = 0.1}$ → initial probabilities

⊕ Transition ~~probability~~ from day to day is also expressed as prob.

$P(Q_{t+1} = Sunny \mid Q_t = Sunny) = 0.6$ → transition probability

⊕ ~~Assume that~~ Assume that there is some guy living in Lugano, and he performs activities based on weather forecast from, again a discrete set of activities (observations)

$A \in \{\cancel{Chat}\ study, Play\}$

⊕ Again, activity is ~~also~~ a probabilistic function of the state (weather) $P(A_t = \cancel{Chat}\ study \mid Q_t = Sunny) = 0.4$ → observation probability

---

⊕ Assume that Matteo's gf doesn't live in Lugano, so she might have only partial knowledge of the weather & Matteo's activities.
→ knows weather, wants to guess activities
→ knows activities, wants to determine most probable weather

⊕ HMM's give us principled way of dealing with following problems

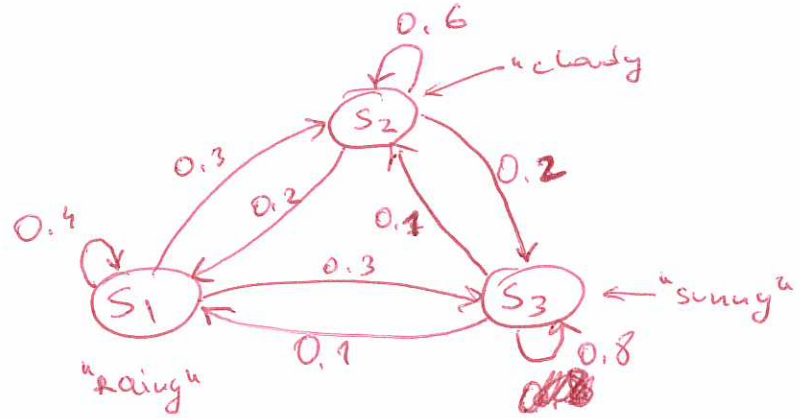
① $P(A_1, \ldots A_T \mid \lambda)$ → Given model λ, compute most probable obs. seq.
→ use simple forward pass

② $P(Q_1 \ldots Q_T \mid A_1 \ldots A_T, \lambda)$ → Given model & observations, compute most probable weather sequence
→ Viterbi algorithm

③ $\arg\max_{\lambda} P(A_1 \ldots A_T \mid \lambda)$ → Given ~~activities~~ observations, find ~~best~~ model λ s.t. this is maximized
Not the best example, better ⊛ → EM algorithm   ⊛ speech recognition

Markov system

Chain with N (in our case 3) states



⊕ The model is said to be in ONE of the states at each time t

model's state at time t → $q_t \in \{S_1, .., S_N\}$

note: discrete time steps

⊕ Between each time step state is choosen randomly therefore the model is stochastic

⊕ Note: probabilities which go out of a state sum to 1.

⊕ What is Markov property:

Assume you observed sequence of states

$S_p, .., S_k, S_i, S_j \quad , p, k, i, j \in \{1, .., N\}$

$$P(q_{t+1} = S_j \mid q_t = S_i, q_{t-1} = S_k, .., q_0 = S_p) = P(q_{t+1} = S_j \mid q_t = S_i)$$

Probability of transitioning to a state depends only on the current state & NOT on past states.

Does not always hold in the real world, but it simplifies the problem & in many cases works quite well.

⊕ Let's write ~~transition~~ probability from S_i to S_j

$$a_{ij} = P(q_{t+1} = S_j \mid q_t = S_i)$$

$\Rightarrow$ defines state transition matrix

$$A = \begin{array}{c} \text{from R} \\ \text{from C} \\ \text{from S} \end{array} \begin{bmatrix} 0.4 & 0.3 & 0.3 \\ 0.2 & 0.6 & 0.2 \\ 0.1 & 0.1 & 0.8 \end{bmatrix}_{N \times N} \longrightarrow \underline{\Sigma = 1}, \; N = 3$$

$$\begin{array}{ccc} \text{to R} & \text{to C} & \text{to S} \end{array}$$

$$\boxed{\sum_{j=1}^{N} a_{ij} = 1}$$

⊕ How to compute prob. of a sequence of states

$$P(q_1 \ldots, q_t) = P(q_t \mid q_{t-1}, \ldots, q_1)\, P(q_{t-1}, \ldots, q_1) \quad \left(\text{Chain rule } P(A,B) = P(A|B)P(B)\right)$$

$$= P(q_t \mid q_{t-1})\, P(q_{t-1}, \ldots, q_1) \quad (\text{Markov property})$$

$$= P(q_t \mid q_{t-1})\, P(q_{t-1} \mid q_{t-2}) \cdots P(q_2 \mid q_1) P(q_1)$$

⊕ To proceed need to define

initial state probabilities $\pi_i = P[q_1 = S_i]$, $i \in \{1, \ldots, N\}$

$$\pi = \{0.1, 0.1, 0.8\}$$

Rainy cloudy sunny

⊕ Example: given state sequence $Q = S, S, S, R, R$

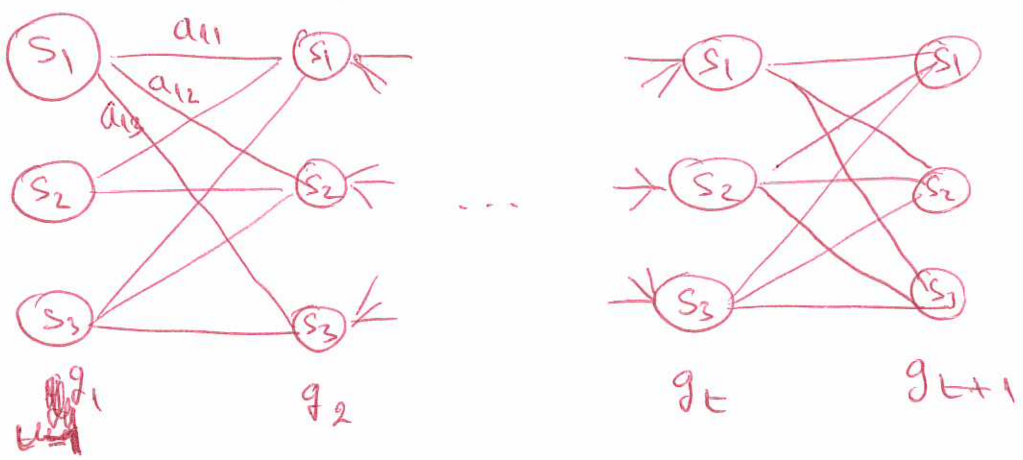
$$P(Q \mid \text{model}) = P(S) \cdot P(S|S) P(S|S) P(R|S) \cdot P(R|R)$$

$$= \underline{0.8} \cdot 0.8 \cdot 0.8 \cdot 0.1 \cdot 0.4 = \ldots$$

(4) Okay, this was pretty simple

How about computing $P(q_t = S_i) = ?$

trellis diagram



2 ways: 1) $P(g_t = S_i) = \sum_{\substack{Q \in \text{path of length } t, \\ \text{ending in } S_i}} p(Q)$ slow, stupid $O(TN^T)$

Explain how this looks on trellis diagram

2) ~~Dynamic~~ Define $p_t(i) = p(g_t = S_i)$ ← our value

Observe that ~~we can compute~~ if we have $p_t(i), \forall i$

we can compute p_{t+1} ~~recursively~~ with dynamic prog.

$$p_{t+1}(j) = \sum_{i=1}^{N} p(g_{t+1} = S_j, g_t = S_i) =$$

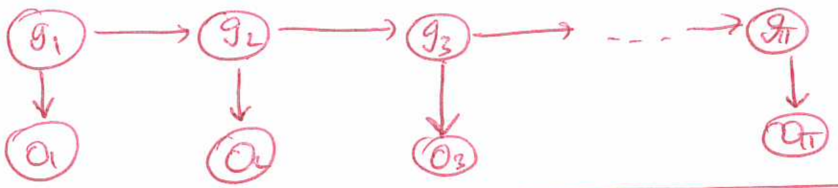
$$= \sum_{i=1}^{N} \underbrace{p(g_{t+1} = S_j \mid g_t = S_i)}_{a_{ij}} \cdot p(g_t = S_i)$$

$$= \sum_{i=1}^{N} a_{ij} \cdot p_t(i) \qquad O(TN^2)$$

Trade computational capacity for memory storage.

Let's extend this to HMMs

- in Markov system observed directly the states
- in HMMs ~~we don't observe the states~~ set of observations ~~are different~~ is different than set of states (which is hidden)



observations noisy
state hidden

2 → $P(O_t = \text{study} \mid q_t = \text{rain}) = 0.8$

in our example

1
$q_t \in \{\text{rainy}, \text{cloudy}, \text{sunny}\}$
$O_t \in \{\text{study}, \text{play}\}$

- HMM: model $\lambda = (N, M, \pi, A, B)$
  1. $N : \{S_1, \dots, S_N\}$ ← number of states
  2. $M : \{O_1, \dots, O_M\}$ ← number of observations note: often $M \neq N$
  3. $\pi_i = P(q_1 = S_i) \to \sum_{i=1}^{N} \pi_i = 1$ ← initial state probabilities
  4. ~~$A = \{a_{ij}\}$~~ $a_{ij} = P(q_{t+1} = S_j \mid q_t = S_i)$

     $$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1N} \\ \vdots & & & \\ a_{N1} & & \cdots & a_{NN} \end{bmatrix}_{N \times N}$$

     ↑ state transition probabilities
  5. $$B = \begin{bmatrix} b_1(1) & b_1(2) & \cdots & b_1(M) \\ \vdots & & & \\ b_N(1) & b_N(2) & \cdots & b_N(M) \end{bmatrix}_{N \times M}$$

     ← observation symbol probability distribution

     $$b_i(k) = P(O_t = k \mid q_t = S_i)$$

~~Sometimes~~ Often written as $\lambda = (\pi, A, B)$

~~3 problems are:~~ ~~① $P(O|\lambda)$ efficiently compute observation sequence~~

~~② $P(Q|O,\lambda)$ optimal state sequence given O~~

~~③ $\arg\max_{\lambda} P(O|\lambda)$ adjust model parameters to maximize $P(O|\lambda)$~~

(F) Let's look at the first problem:

Compute probabilities of observation sequence given model λ

$$P(O|\lambda) = P(O_1 \ldots O_T|\lambda) = \sum_{Q \in \mathrm{len}\, T} P(O \land Q|\lambda)$$

$$= \sum_{Q \in \mathrm{len}\, T} P(O|Q,\lambda)\, P(Q|\lambda)$$

$$P(O|Q,\lambda) = \prod_{t=1}^{T} P(O_t | q_t, \lambda) =$$

$$= b_{q_1}(O_1) \cdot b_{q_2}(O_2) \cdot \ldots \cdot b_{q_T}(O_T)$$

$$P(Q|\lambda) = \pi_{q_1} \cdot a_{q_1 q_2} \cdot a_{q_2 q_3} \cdot \ldots \cdot a_{q_{T-1} q_T}$$

$$\Rightarrow P(O|\lambda) = \sum_{Q \in \mathrm{len}\, T} P(O|Q,\lambda) P(Q|\lambda) =$$

$$= \sum_{q_1, q_2, \ldots, q_T} \pi_{q_1} \underline{b_{q_1}(O_1)} \cdot a_{q_1 q_2} \cdot \underline{b_{q_2}(O_2)} \cdot \ldots \cdot a_{q_{T-1} q_T} \underline{b_{q_T}(O_T)}$$

$O(TN^T)$

Similar to what we had for normal Markov chain, just that we have one additional component, observation probabilities.

Smarter way to calculate this

Forward algorithm

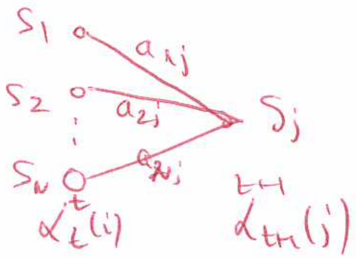
Define $\alpha_t(i) = P(O_1 \ldots O_t \wedge q_t = S_i \mid \lambda)$

"we have observed $O_1, \ldots, O_t$ & ended up in state S_i"

① initialization

$$\alpha_1(i) = P(O_1 \wedge q_1 = S_i) =$$
$$= P(O_1 \mid q_1 = S_i)\, P(q_1 = S_i)$$
$$= b_i(O_1) \cdot \pi_i$$

② update ~~induction~~ step



$P(A \mid B) = P(A)$

$P(A, B) = P(A) \cdot P(B)$

$P(A, B) = P(A \mid B) \cdot P(B)$

$$\alpha_{t+1}(j) = P(O_1, \ldots O_t O_{t+1} \wedge q_{t+1} = S_j)$$
$$= \sum_{i=1}^{N} P(O_1 \ldots O_t \wedge q_t = S_i \wedge O_{t+1} \wedge q_{t+1} = S_j)$$

$P(A, B) = P(A \mid B) \cdot P(B)$ ~~$P(A \mid B, C) = P \ldots$~~ $\rightarrow$
$$= \sum_{i=1}^{N} P(O_{t+1}, q_{t+1} = S_j \mid O_1 \ldots O_t \wedge q_t = S_i)\, P(O_1 \ldots O_t \wedge q_t = S_i)$$

$P(A \mid B, C) = P(A \mid C)$ if $P(A \mid B) = P(A)$ $\rightarrow$
$$= \sum_{i=1}^{N} P(O_{t+1}, q_{t+1} = S_j \mid q_t = S_i) \cdot \alpha_t(i)$$
$$= \sum_{i=1}^{N} P(O_{t+1} \mid q_{t+1} = S_j)\, P(q_{t+1} = S_j \mid q_t = S_i) \cdot \alpha_t(i)$$
$$= \sum_{i=1}^{N} a_{ij} \cdot b_j(O_{t+1}) \cdot \alpha_t(i)$$

③ termination $P(O_1 \ldots O_T \mid \lambda) = \sum_{i=1}^{N} P(O_1 \ldots O_T \wedge q_T = S_i \mid \lambda)$
$$= \sum_{i=1}^{N} \alpha_T(i) \quad \sim \; O(NT^2)$$

④ How to calculate
$$P(q_t = S_i \mid O_1 \ldots O_t) = \frac{P(O_1 \ldots O_t \wedge q_t = S_i)}{P(O)} = \frac{\alpha_t(i)}{\sum_{j=1}^{N} \alpha_t(j)}$$

Problem 2: Given observations, find the most likely state sequence Q

$$\max_Q P(Q \mid O, \lambda)$$

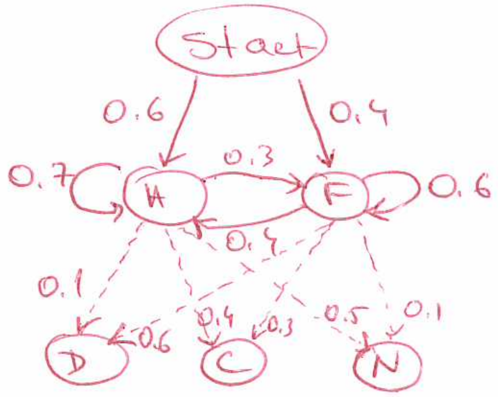
Viterbi algorithm Let's explain it on an example, & later give general definition

Assume 2 states: states $\in \{H, F\}$ (H = Healthy, F = Fever)

$O \in \{\text{Cold}, \text{Normal}, \text{Dizzy}\}$
C N D

$$A = \begin{array}{c} \text{Healthy} \\ \text{Fever} \end{array} \begin{pmatrix} 0.7 & 0.3 \\ 0.4 & 0.6 \end{pmatrix}$$

$$\Pi = \{\overset{H}{0.6}, \overset{F}{0.4}\}$$

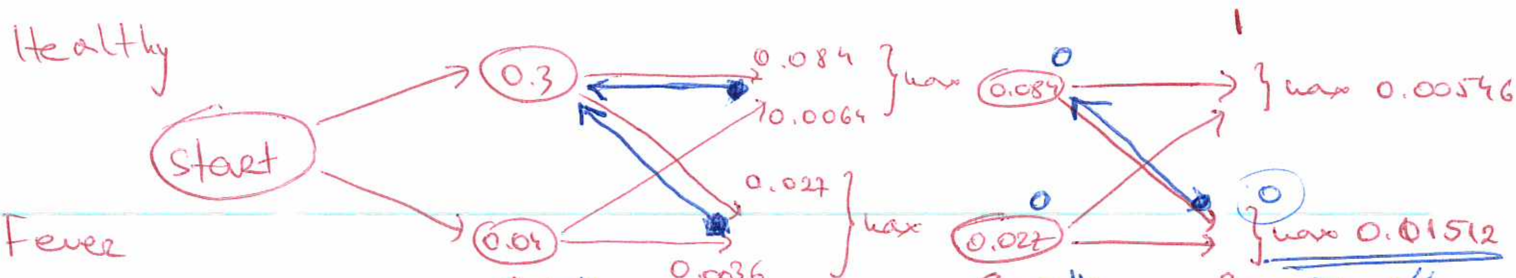


Assume you observe a sequence N, C, D.
Find most probable sequence of states.

→ ① could find most probable individual states
② or most probable pairs / triplets of states

$$\max_{g_1 g_2 g_3} P(g_1, g_2, g_3, O_1 = N, O_2 = C, O_3 = D \mid \lambda)$$

If you look at all possible set of states & find most prob path, again you have many calculations.



$g_1 = H$ $g_2 = H$ $g_3 = F$

g_1: $P(g_1, O_1 = N) = P(O_1 = N \mid g_1) \cdot P(g_1) = b(O_1) \cdot \Pi =$ if $g_1 = H$: $0.5 \cdot 0.6 = 0.3$; if $g_1 = F$: $0.1 \cdot 0.4 = 0.04$

(Slides: $\delta_1 a_{g_1}(g_2)\, b_{g_2}(O_2)$)

g_2: $P(g_1 = H, g_2 = H, O_2 = C) = P(g_1 = H) \cdot P(g_2 = H \mid g_1 = H) \cdot P(O_2 = C \mid g_2 = H) = 0.3 \cdot 0.7 \cdot 0.4$

g_1	g_2	O_2	g_1	g_2	g_1	O_2	g_2	
F	H	C	F	H	F	C	H	$= 0.04 \cdot 0.4 \cdot 0.4$
			H	F	H	C	F	$= 0.3 \cdot 0.3 \cdot 0.3$
			F	F	F	C	F	$= 0.04 \cdot 0.3 \cdot 0.3$

In case of a long sequence

⇒ lots of factors ⇒ smaller & smaller numbers

⇒ $\boxed{\text{convert to log probabilities \& sum!}}$

$$\log P(q_1)P(q_2|q_1)P(O_2|q_2) = \log P(q_1) + \log P(q_2|q_1) + \log P(O_2|q_2)$$

Viterbi formula

$$\delta_t(i) = \max_{q_1 \dots q_{t-1}} P(q_1 \dots q_{t-1} \wedge q_t = S_i \wedge O_1 \dots O_T)$$

= the maximum probability of ending up in state S_i at time t and producing $O_1 \dots O_T$

$$mpp(i) = \operatorname*{argmax}_{q_1, \dots, q_{t-1}} \delta_t(i)$$

$$\delta_1(j) = \pi_i b_i(O_1)$$

$$\delta_t(j) = \max_{1 \le i \le N} [\delta_{t-1}(i) a_{ij}] \, b_j(O_t) \qquad t \in [2,T], \; j \in [1,N]$$

$$\psi_t(j) = \operatorname*{argmax}_{1 \le i \le N} [\delta_{t-1}(i) a_{ij}] \qquad t \in [2,T], \; j \in [1,N]$$

Q: why don't we have b here?

↳ b/c it wouldn't change anything as we look for arg, not value!

Keeps track of argument which maximizes $\delta_t(j)$

Once we come to the end, we look at maximal probability, take ~~the~~ the state which generated this probability as last, and do backtracking based on ψ_t's

Q: how is this different from the forward pass?!

Difference to forward pass: keeps track of backtrack variables that lead to maximal probability.

Used also in telecommunications to decode a sequence

States 00, 01, 10, 11 → noisy channel → O_1, O_2, O_3, O_4

3rd problem of HMMs

Given obs. O

Find λ s.t $\max_{\lambda} P(O|\lambda)$

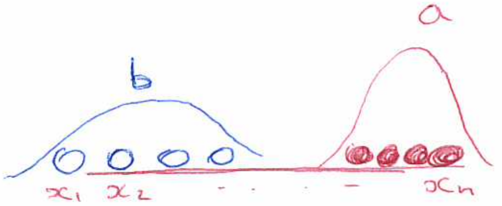
↳ no known way to analytically compute such model

↳ no optimal way of estimating model parameters

Approximate ways to chose λ s.t $\max_{\lambda} P(O|\lambda)$: ① iterative procedure EM (Baum-Welch algo)

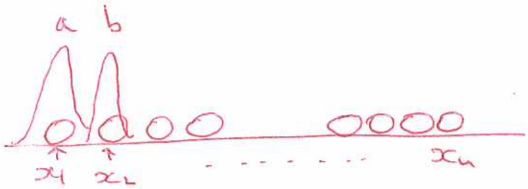
② gradient descent

1-D EM algorithm



+ Labeled → easy to fit the two distributions
+ ~~[illegible]~~ have distributions → easy to classify

Not clear what to do if ~~data~~ is unlabeled!



$P(a|x_1) > P(b|x_1)$ "belongs more to a"

$P(a|x_2) < P(b|x_2)$ "belongs more to b"

To solve this issue you can use iterative EM procedure

EM algorithm used in:

+ Example here: Gaussian Mixture model
+ k-means clustering (special case of GMMs) ← covariances diagonal equal & such
  ↘ minimize pairwise distance of points in the same cluster
  ↓ initialize mean pts → minimize $\sum_{x, x_j \in S_i} \|x - \mu_i\|^2$ this distance to μ_i
+ $\boxed{\text{don't mix with k-nearest neighbour}}$! ↳ supervised !
+ HMMs

1D - EM

① initialize randomly $\mu_a, \mu_b; \sigma_a, \sigma_b, P(a), P(b)$

means and variances of the Gaussians

Priors (numbers): how likely is a data point to belong to classes a and b

for each of the "clusters"

② $\underline{\text{E-step}}$: $\boxed{\text{compute the belief for each } x_i \text{ using our current Gaussian param}}$

where is each datapoint expected to belong?

$$a_i = P(a|x_i) = \frac{P(x_i|a) \cdot P(a)}{P(x_i|a) \cdot P(a) + P(x_i|b) \cdot P(b)}$$

As we have assumed we are working with Gaussian dist:

$$P(x_i|a) = \frac{1}{\sqrt{2\pi\sigma_a^2}} e^{-\frac{(x_i-\mu_a)^2}{2\sigma_a^2}}$$

$$b_i = 1 - a_i$$

③ $\underline{\text{M-step}}$: Re-estimate the Gaussian parameters using <u>all</u> data points

$$\left.\begin{aligned} \mu_a &= \frac{a_1 \cdot x_1 + \dots + a_n x_n}{a_1 + \dots + a_n} \\ \sigma_a &= \sqrt{\frac{\sum_{i=1}^{n} (x_i - \mu_i)^2 \cdot a_i}{\sum_{i=1}^{n} a_i}} \\ P(a) &= \frac{a_1 + \dots + a_n}{n} \\ P(b) &= 1 - P(a) \end{aligned}\right\} \text{analytical solutions to MLE}$$

MLE $\quad \theta_{MLE} = \arg\max_\theta P(X|\theta) = \arg\max_\theta \log P(X|\theta) = \arg\max_\theta \sum_i \log P(x_i|\theta)$

maximum likelihood estimate $\quad \theta = \{\mu_a, \mu_b, \sigma_a, \sigma_b, P(a), P(b)\}$

can optimize it also via GD

It is just a special case of MAP (Maximum a posteriori estimate)

We can see this through Bayes theorem

posterior $\rightarrow$ $P(\theta|X) = \frac{P(X|\theta) P(\theta)}{\underbrace{P(X)}_{\text{const}}}$ $\leftarrow$ prior, likelihood $\propto P(X|\theta) P(\theta)$ proportional

$\Rightarrow \quad \theta_{MAP} = \arg\max_\theta \sum_i \log P(x_i|\theta) \cdot P(\theta)$

if prior is uniform $P(\theta) = \text{const} \Rightarrow \theta_{MAP} = \underbrace{\arg\max_\theta \sum_i \log P(x_i|\theta)}_{= \theta_{MLE}}$

---

④ repeat 2 & 3 until converged

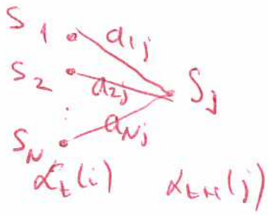
<u>Notes</u>:
- ⊕ can lead to local maxima, no guaranteed convergence in general case
- ⊕ in practice you typically make several runs with different initializations
- ⊕ Question also how many clusters to use

Baum-Welch algorithm (EM for HMMs)

We need 4 variables: $\alpha_t(i), \beta_t(i), \gamma_t(i), \xi_t(i)$

① Previously we have considered

$$\alpha_t(i) = P(O_1 \dots O_t, q_t = S_i)$$



which can be computed recursively as

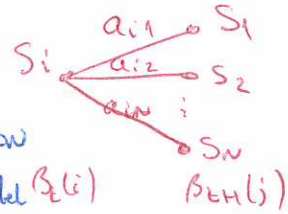
$$\alpha_1(i) = \pi_i \cdot b_i(O_1)$$

$$\alpha_{t+1}(j) = \sum_{i=1}^{N} \alpha_t(i) \cdot a_{ij} \cdot b_j(O_{t+1})$$

---

② Similarly, we introduce backward variable

$$\beta_t(i) = P(O_{t+1} \dots O_T, q_t = S_i)$$



PROBABILITY OF THE PARTIAL OBSERVATION SEQUENCE, GIVEN THE STATE S_i @ t & model $\beta_t(i)$

$$\beta_T(i) = 1$$

$$\beta_t(i) = \sum_{j=1}^{N} \beta_{t+1}(j) \cdot a_{ij} \cdot b_j(O_{t+1})$$

---

③ Probability of being in state S_i @ time t, given O

$$\gamma_t(i) = P(q_t = S_i | O) = \frac{P(q_t = S_i, O_1 \dots O_T)}{P(O_1 \dots O_T)} = \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i)\beta_t(i)}$$

Note $\sum_i \gamma_t(i) = 1$ — for sure we are in some state S_i

Most likely state @ time t: $q_t = \arg\max_i \gamma_t(i)$.

What is the problem with this?

+ ~~Only~~ maximizes individual states, not the path as a whole
+ In the resulting path there could be impossible transitions (if $a_{ij} = 0$)

Solutions:
+ could observe pairs of consecutive states
+ or triplets, and so on...
+ Viterbi algorithm gives us the most prob. path.

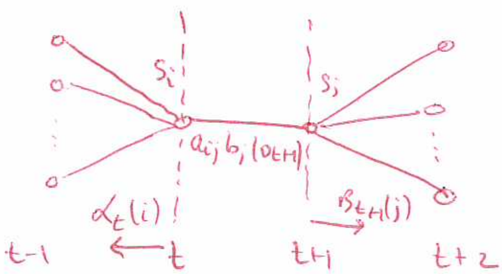
---

④ Probability of being in state S_i at time t & S_j at t+1

$$\xi(i,j) = P(q_t = S_i, q_{t+1} = S_j | O)$$

$$= \frac{P(q_t = S_i, q_{t+1} = S_j, O_1 \dots O_T)}{P(O_1 \dots O_T)} =$$

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



Note: $\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j)$

$\sum_{t=1}^{T-1} \gamma_t(i)$ = expected number of ~~transitions~~ from S_i

$\sum_{t=1}^{T-1} \xi_t(i,j)$ = expected number of transitions from S_i to S_j

Baum-Welch algo

1. Start with initial $\lambda = (\Pi, A, B)$
2. E-step: Calculate $\gamma_t(i)$, $\xi(i,j)$
3. M-step: Determine new model as

   - $\Pi_i = \gamma_1(i)$ ← expected number of times in state S_i

   - $$a_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} = \frac{\text{expected number of transitions from } S_i \text{ to } S_j}{\text{expected number of } \cancel{\text{transitions}} \text{ times in } S_i}$$

   - $$b_j(k) = \frac{\sum_{\substack{t=1 \\ s.t. O_t=k}}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)} = \frac{\text{expected number of times in } S_j \text{ \& observed symbol } k}{\text{expected number of times in } S_j}$$

4. Repeat 2 & 3 until converged