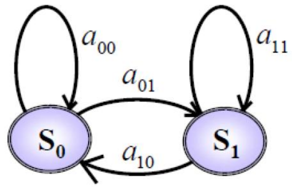
Hidden Markov Models

Contents:

- Markov process, observable Markov models
- Hidden Markov models
- Problem 1: Scoring and evaluation
- Problem 2: Decoding
- Problem 3: Training

Markov model

- stochastic model
- used to model a random system that changes state according to a transition rule that depends only on the current state
- Characterized by a set of N states

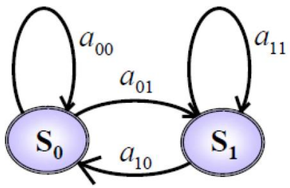


$$S = \{S_0, S_1, \cdots S_N\}$$

and transition probabilities from one state to another a_{ij}

Markov model

- the current state of the system depends only on the previous state, not on the sequence before that
- the state of the system at time t is q_t
- the transition probability depends only on the previous state:



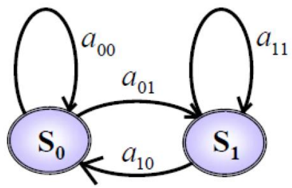
$$P(q_t = S_j \mid q_{t-1} = S_i, q_{t-2} = S_k \cdots) = P(q_t = S_j \mid q_{t-1} = S_i)$$
$$a_{ij} = P(q_t = S_j \mid q_{t-1} = S_i)$$

Markov model properties

$$\mathrm{A} = \begin{bmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{bmatrix}$$

$$a_{ij} \geq 0 \quad \forall\, \mathrm{i, j}$$

$$\sum_{j=0}^{N} a_{ij} = 1 \quad \forall\, \mathrm{i}$$

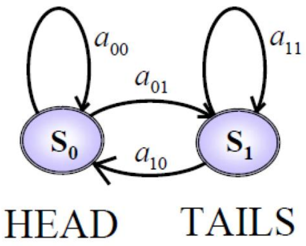


Stochastic matrix:
- each entry is non-negative
- rows sum up to 1

Example 1: Single fair coin observable process

- Observable: the output of the process is a set of states
- Outcomes:
  - Head – State 0
  - Tails – State 1
- Observed outcomes uniquely define a state sequence: HHHTTTHHTT → 0001110011
- Transition probabilities:

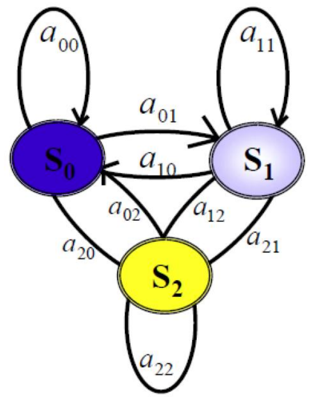
$$A = \begin{bmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{bmatrix}$$



Example 2: Observable Markov model of weather

- Outcomes:
  - State 0 – Rainy
  - State 1 – Cloudy
  - State 2 – Sunny
- State transition probabilities:

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

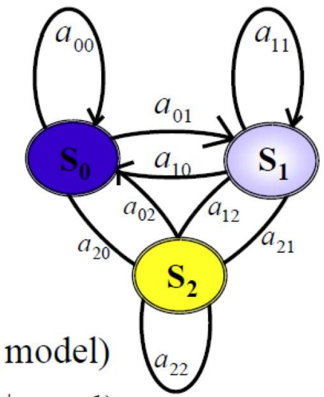


Example 2: Observable Markov model of weather

- What is the probability that the weather for 8 consecutive days is sun, sun, sun, rain, rain, sun, cloudy, sun ?
- Representing the information:
  - **Observation sequence** is:
    O = {sun, sun, sun, rain, rain, sun, cloudy, sun}
  - Corresponds to state sequence:
    S = {2, 2, 2, 0, 0, 2, 1, 2}
  - We need to calculate P(O | model)
    P(O | model) = P(S={2, 2, 2, 0, 0, 2, 1, 2} | model)

Example 2: Observable Markov model of weather

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



π_i : initial state probability

$\pi_i = P(q_1 = i)$

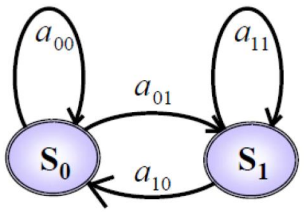
$$P(O\,|\,\text{model}) = P(S = \{2,2,2,0,0,2,1,2\}\,|\,\text{model})$$
$$= P(q_1 = 2)P(q_2 = 2\,|\,q_1 = 2)\cdots P(q_8 = 2\,|\,q_7 = 1)$$
$$= \pi_2 \cdot a_{22} \cdot a_{22} \cdot a_{20} \cdot a_{00} \cdot a_{02} \cdot a_{21} \cdot a_{12}$$
$$= \pi_2 \cdot (0.8)^2 (0.1) \cdot (0.4) \cdot (0.3) \cdot (0.1) \cdot (0.2)$$

Hidden Markov models

- Observations are a probabilistic function of state
- The underlying sequence of states is not observable (it is hidden)
- Outputs are independent – observations are dependent only on the state that generated them, not on eachother

Example: Two coins observable Markov process

- States: coins
- Observations:
  - Head
  - Tail
  - Each state can generate each observation with a certain probability
- The observed outcomes do not uniquely define state sequence
- Transition probabilities:

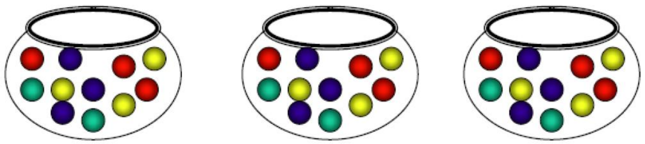


$P(H) = P_1$ $\quad$ $P(H) = P_2$

$P(T) = 1 - P_1$ $\quad$ $P(T) = 1 - P_2$

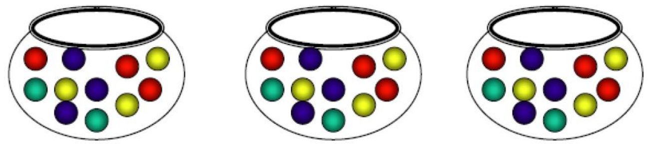
$$A = \begin{bmatrix} a_{00} & 1 - a_{00} \\ 1 - a_{11} & a_{11} \end{bmatrix}$$

Example: urn and ball model



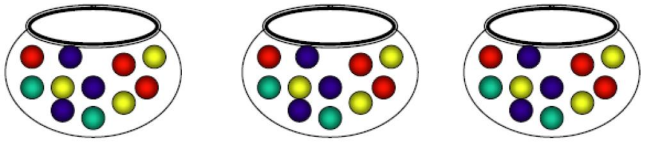
- 3 urns with 4 different color balls
- We do not see the urns, someone is extracting balls from them and tells us the color of each
- Steps:
  1. Select one urn at random
  2. Pick a ball from the urn, tell what color it is
  3. Put ball back to the urn
  4. Select new urn based on a random selection procedure from current urn
  5. Repeat steps 2-4

Example: urn and ball model



- Observations: the colors of the balls
- States: the identity of the urn
- State transitions: the selection process for next urn given current one

Example: urn and ball model



P(R)=0.20	P(R)=0.45	P(R)=0.15
P(G)=0.30	P(G)=0.15	P(G)=0.70
P(B)=0.10	P(B)=0.20	P(B)=0.10
P(Y)=0.40	P(Y)=0.20	P(Y)=0.05

- Urns contain different ratio of colours
- Observation sequence: R B Y Y G B Y G R ..
- The observation sequence (individual colors) do not reveal the state (which urn it comes from)

Discrete symbol observation HMM

- A set of N states — N hidden states

$$S = \{S_0, S_1, \cdots S_N\}$$

- Transition probabilities — Probabilities of states transition

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

- A set of M observation symbols — corresponding observation

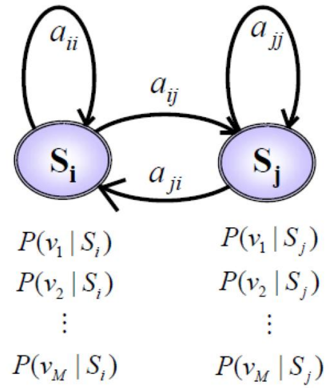
$$V = \{v_1, v_2, \cdots v_M\}$$

- Probability distribution (state j symbol k)

$$b_j(k) = P(o_t = v_k \mid q_t = j)$$

- Initial state distribution

$$\pi = \{\pi_i\} = P(q_1 = i)$$

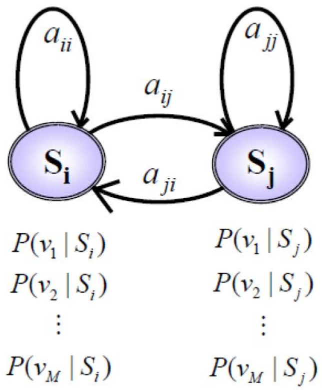


Discrete symbol observation HMM

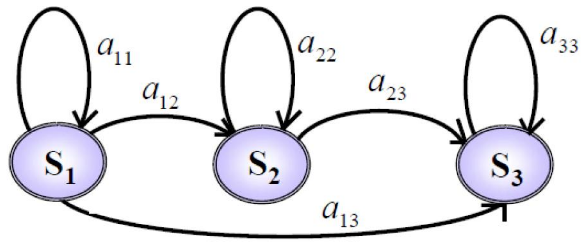
Specification of an HMM:

- Two model parameters, N and M
  - Number of states N
  - Number of symbols M
- Three probability measures A, B, π
  - Transition probability matrix A
  - State probability distribution B
  - Initial state distribution π

$$\lambda = (A, B, \pi)$$



Left-to-right HMM

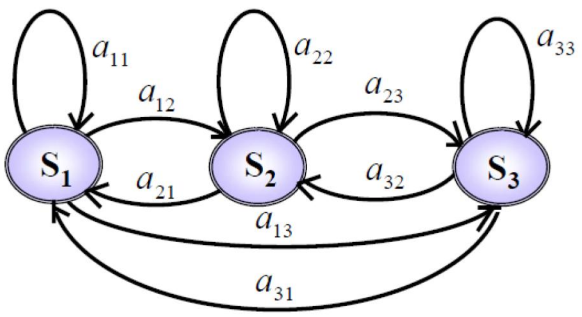


$$\mathrm{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

$$a_{ij} = 0 \qquad j < i$$

$$\pi_i = \begin{cases} 0, & i \neq 1 \\ 1, & i = 1 \end{cases}$$

Ergodic HMM



$$a_{ij} > 0 \qquad \forall i, \forall j$$

HMM as a symbol generator

An HMM with parameters N, M, A, B, and π can generate an observation sequence:

$$O = \{ o_1, o_2, o_3, .. o_T \}$$

1. Choose initial state $q_1 = i$ from the initial state distribution π
2. Set $t=1$
3. Choose $o_T = v_k$ according to distribution $b_j(k)$
4. Transition to state $q_{t+1} = j$ according to state transition probability a_{ij}
5. Set $t=t+1$
6. Repeat from step 3

HMM as a symbol generator

Time t	1	2	3	4	5	6	7	8	…	T
State	q_1	q_2	q_3	q_4	q_5	q_6	q_7	q_8	…	q_T
Observation	o_1	o_2	o_3	o_4	o_5	o_6	o_7	o_8	…	o_T

Decoding — What is most probable hidden states sequence when you have observation sequence?

Think of the HMM as generating the observation sequence as it transitions from state to state

HMM problems

how much likely is that something observable will happen? In other words, what is probability of observation sequence?

- Problem 1: Scoring and evaluation
  - How to compute efficiently the probability of an observation sequence O given the model λ ? (How to calculate P(O| λ))
- Problem 2: Decoding
  - Given an observation sequence O and a model λ, how do we determine the corresponding state sequence q that best explains how the observations were generated?
- Problem 3: Training
  - How to adjust the parameters λ={A,B, π} to maximize the probability of generating a given observation sequence?

    (How to maximize P(O| λ))

Problem 1: Scoring and evaluation

- Given an observation sequence O = { $o_1, o_2, o_3, .. o_T$ }

we want to compute the probability of generating it $P(O|\lambda)$

- We assume a sequence of states q = { $q_1, q_2, q_3, .. q_T$ }
- Decompose the problem by summing over all possible state sequences:

$$P(\mathbf{O}|\lambda) = \sum_{\text{all q}} P(\mathbf{O}|q,\lambda)P(q|\lambda)$$

Problem 1: Scoring and evaluation

- Given an observation sequence O = { $o_1, o_2, o_3, .. o_T$ }

we want to compute the probability of generating it $P(O|\lambda)$

- We assume a sequence of states q = { $q_1, q_2, q_3, .. q_T$ }
- Decompose the problem by summing over all possible state sequences:

$$P(\mathbf{O}|\lambda) = \sum_{\text{all q}} P(\mathbf{O}|q,\lambda)P(q|\lambda)$$

Likelihood of generating the
observed symbol sequence
given the assumed state sequence

Problem 1: Scoring and evaluation

- Given an observation sequence O = { $o_1, o_2, o_3, .. o_T$ }

we want to compute the probability of generating it $P(O|\lambda)$

- We assume a sequence of states q = { $q_1, q_2, q_3, .. q_T$ }
- Decompose the problem by summing over all possible state sequences:

$$P(\mathbf{O}|\lambda) = \sum_{\text{all q}} P(\mathbf{O}|q,\lambda)P(q|\lambda)$$

Likelihood of generating the
observed symbol sequence
given the assumed state sequence

How likely it is for the
system to go through the
given sequence of states

Problem 1: Scoring and evaluation

- Probability of the observation sequence given the state sequence:

$$P(\mathbf{O}|q,\lambda) = \prod_{t=1}^{T} p(\mathbf{o}_t|q_t,\lambda) = b_{q_1}(\mathbf{o}_1)\cdot b_{q_2}(\mathbf{o}_2)\cdots b_{q_T}(\mathbf{o}_T)$$

- Probability of the state sequence:

$$P(q|\lambda) = \pi_{q_1}(a_{q_1q_2})\cdot(a_{q_2q_3})\cdots(a_{q_{T-1}q_T})$$

- Using the chain rule:

$$P(\mathbf{O}|\lambda) = \sum_{\text{all q}} P(\mathbf{O}|q,\lambda)P(q|\lambda)$$

$$= \sum_{\text{all q}} \pi_{q_1} b_{q_1}(o_1) a_{q_1q_2} b_{q_2}(o_2)\cdots a_{q_{T-1}q_T} b_{q_T}(o_T)$$

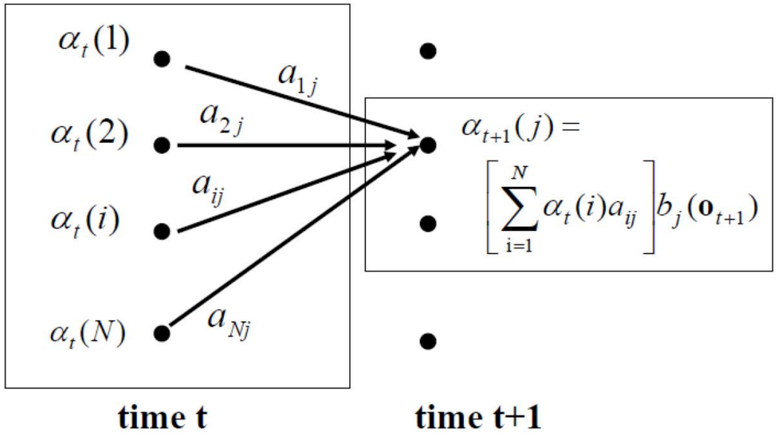
Not practical to compute!

Forward algorithm

- Define the probability of seeing observations o_1 to o_T, and ending in state i, given HMM λ:

$$\alpha_t(i) = P(\mathbf{o}_1\mathbf{o}_2 \ldots \mathbf{o}_t, q_t = i \mid \lambda)$$

Forward algorithm



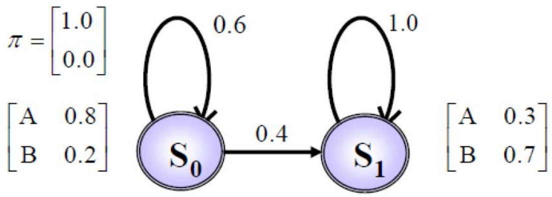
Forward algorithm

- Define the probability of seeing observations o_1 to o_T, and ending in state i, given HMM λ:

$$\alpha_t(i) = P(\mathbf{o}_1\mathbf{o}_2 \ldots \mathbf{o}_t, q_t = i \mid \lambda)$$

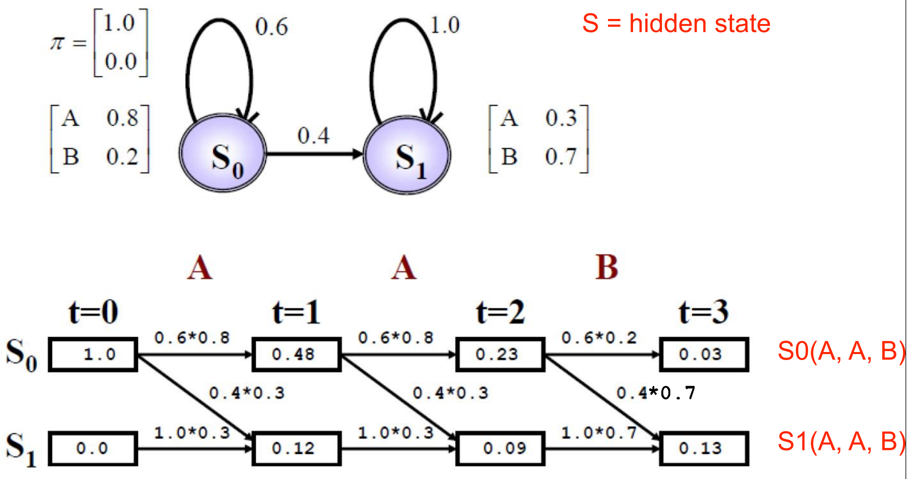
- Initialization: $\alpha_0(i) = \pi_i$
- Induction: $\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(\mathbf{o}_{t+1})$
- Termination: $P(\mathbf{O} \mid \lambda) = \sum_{i=1}^{N} \alpha_T(i)$

Forward algorithm example



- Given this HMM with discrete observations A and B, what is the probability of generating the sequence {A,A,B}?
- We need to calculate P(O = {A,A,B} | λ)

Forward algorithm example



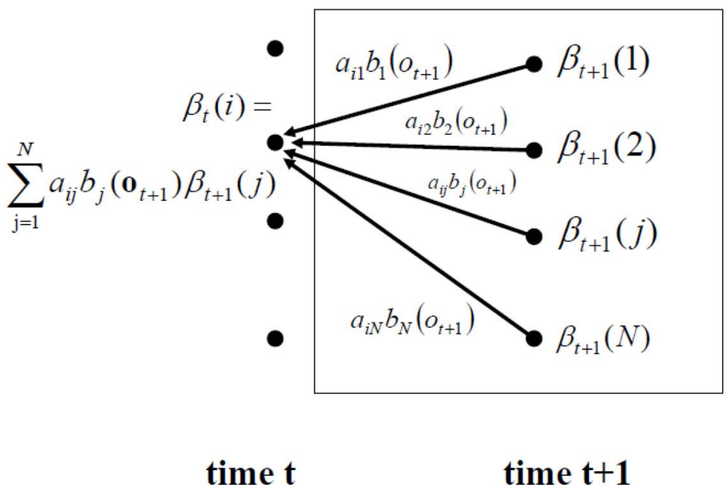
P(O = {A,A,B} | λ } = 0.03 + 0.13 = 0.16

Backward algorithm

- Define the probability of seeing observations o_{t+1} to o_T, given state *i* at time *t* and HMM λ:

$$\beta_t(i) = P(\mathbf{o}_{t+1}\mathbf{o}_{t+2}\ldots\mathbf{o}_T, q_t = i \mid \lambda)$$

Backward algorithm



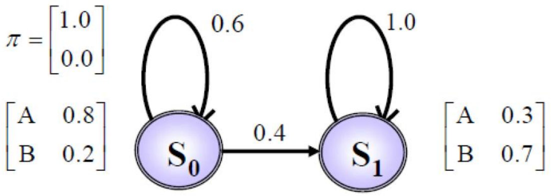
Backward algorithm

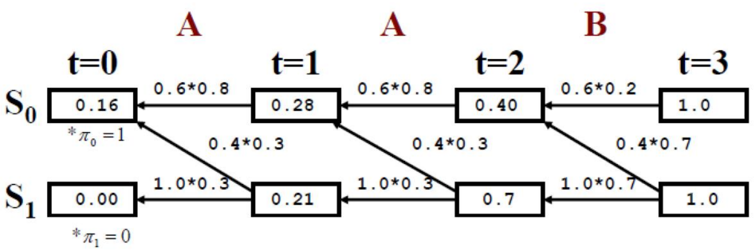
- Define the probability of seeing observations o_{t+1} to o_T, given state *i* at time *t* and HMM λ:

$$\beta_t(i) = P(\mathbf{o}_{t+1}\mathbf{o}_{t+2}\ldots\mathbf{o}_T, q_t = i \mid \lambda)$$

- Initialization: $\beta_T(i) = 1$
- Induction: $\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(\mathbf{o}_{t+1}) \beta_{t+1}(j)$ $\quad t = T-1, T-2, \ldots, 1$ $\quad 1 \le i \le N$
- Termination: $P(\mathbf{O} \mid \lambda) = \sum_{i=1}^{N} \pi_i \beta_0(i)$

Backward algorithm example





Problem 1: Scoring and evaluation

- Solution: two ways of calculating P(O| λ)
  - Forward algorithm

$$P(\mathbf{O} \mid \lambda) = \sum_{i=1}^{N} \alpha_T(i)$$

  - Backward algorithm

$$P(\mathbf{O} \mid \lambda) = \sum_{i=1}^{N} \pi_i \beta_0(i)$$

Problem 2: Decoding

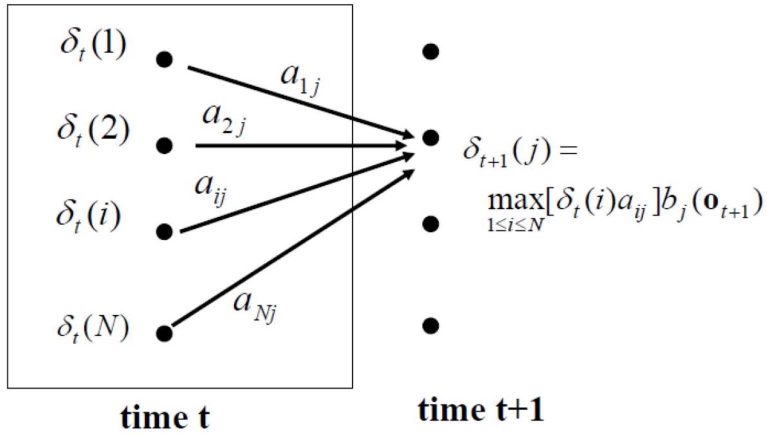
- Given an observation sequence O = { o_1, o_2, o_3, .. o_T }, and a model λ, how do we find the best sequence of states q = { q_1, q_2, q_3, .. q_T } which maximizes P(O,q| λ)?
- Define the highest probably state sequence that accounts for observations o_1 to o_t and ends in state i at time t:

$$\delta_t(i) = \max_{q_1 q_2 \cdots q_{t-1}} P(q_1 q_2 \cdots q_{t-1}, q_t = i, \mathbf{o}_1 \mathbf{o}_2 \ldots \mathbf{o}_t \mid \lambda)$$

- At next transition:

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

Viterbi algorithm



Viterbi algorithm

- Initialization:

$$\delta_1(i) = \pi_i b_i(\mathbf{o}_1)$$

$$\psi_1(i) = 0$$

- Recursion:

$$\delta_t(j) = \max_{1 \le i \le N}[\delta_{t-1}(i) a_{ij}] b_j(\mathbf{o}_t)$$

$$\psi_t(j) = \arg\max_{1 \le i \le N}[\delta_{t-1}(i) a_{ij}]$$

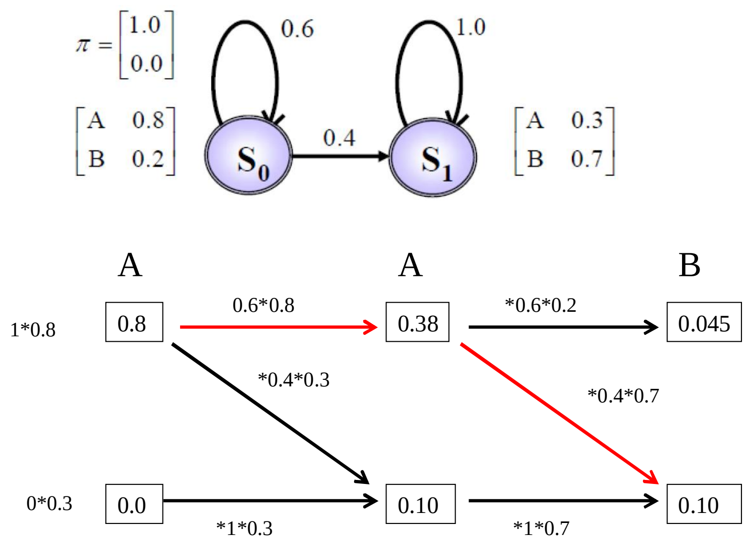
- Termination:

$$P^* = \max_{1 \le i \le N}\left[\delta_T(i)\right]$$

$$q_T^* = \arg\max_{1 \le i \le N}\left[\delta_T(i)\right]$$

- Path back-tracing: $q_t^* = \psi_{t+1}(q_{t+1}^*)$

Viterbi algorithm example



Viterbi algorithm in log-domain

$$\tilde{\pi}_i = \log(\pi_i)$$

$$\tilde{b}_j(o_t) = \log(b_j(o_t))$$

$$\tilde{a}_{ij} = \log(a_{ij})$$

- Same steps are followed in log-domain:

$$\delta_1(i) = \pi_i b_i(\mathbf{o}_1) \longrightarrow \tilde{\delta}_1(i) = \tilde{\pi}_i + \tilde{b}_i(\mathbf{o}_1)$$

Viterbi algorithm in log-domain

- Initialization:

$$\tilde{\delta}_1(i) = \tilde{\pi}_i + \tilde{b}_i(\mathbf{o}_1)$$

$$\psi_1(i) = 0$$

- Recursion:

$$\delta_t(j) = \max_{1 \le i \le N}[\tilde{\delta}_{t-1}(i) + \tilde{a}_{ij}] + \tilde{b}_j(\mathbf{o}_t)$$

$$\psi_t(j) = \arg\max_{1 \le i \le N}[\tilde{\delta}_{t-1}(i) + \tilde{a}_{ij}]$$

- Termination:

$$\tilde{P}^* = \max_{1 \le i \le N}\left[\tilde{\delta}_T(i)\right]$$

$$q_T^* = \arg\max_{1 \le i \le N}\left[\tilde{\delta}_T(i)\right]$$

- Path backtracking: $q_t^* = \psi_{t+1}(q_{t+1}^*)$

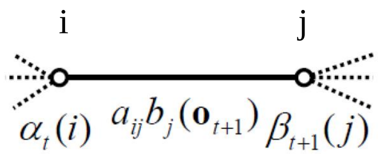
Problem 3: Training

- How do we tune λ to maximize P(O| λ) ?
  - No efficient algorithm to find global optimum
- Baum-Welch algorithm (forward-backward algorithm)
  - Iterative algorithm to find a local optimum
  - Compute probabilities using current model
  - Refine model parameters based on computed values

Forward-backward algorithm

- Define the probability of being in state *i* at time *t* and in state *j* at time *t+1*, given the model and the sequence

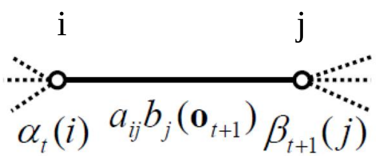
$$\xi_t(i,j) = P(q_t = i, q_{t+1} = j \mid \mathbf{O}, \lambda)$$



Forward-backward algorithm

- Define the probability of being in state *i* at time *t* and in state *j* at time *t+1*, given the model and the sequence

$$\xi_t(i,j) = \frac{\alpha_t(i) a_{ij} b_j(\mathbf{o}_{t+1}) \beta_{t+1}(j)}{P(\mathbf{O} \mid \lambda)} = \frac{\alpha_t(i) a_{ij} b_j(\mathbf{o}_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^{N} \alpha_t(i) \beta_t(i)}$$



Forward-backward algorithm

- More definitions, based on $\xi_t(i,j)$

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j)$$ Probability of being in state i at time t

$$\sum_{t=1}^{T-1} \gamma_t(i)$$ Expected number of transitions from state i in O

$$\sum_{t=1}^{T-1} \xi_t(i,j)$$ Expected number of transitions from state i to state j in O

Computing the model parameters

- Initial state occupancy probability is the **expected number of times in state *i* at time *t=1***

$$\overline{\pi}_i = \gamma_1(i)$$

Computing the model parameters

- transition probability from state i to state j

$$\overline{a}_{ij} = \frac{\text{expected number of transitions from state } i \text{ to state } j}{\text{expected number of transitions from state } i}$$

$$= \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}$$

Computing the model parameters

- Probability of observing symbol *k* in state *j*

$$\overline{b}_j(k) = \frac{\text{expected number of times in state } j \text{ and observing kth symbol}}{\text{expected number times in state } j}$$

$$= \frac{\sum_{\substack{t=1 \\ \mathbf{o}_t = v_k}}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)} = \frac{\sum_{\substack{t=1 \\ \mathbf{o}_t = v_k}}^{T} \alpha_t(j)\beta_t(j)}{\sum_{t=1}^{T} \alpha_t(j)\beta_t(j)}$$

Forward-backward algorithm iterations

1. Initialize $\lambda = (\mathbf{A}, \mathbf{B}, \pi)$
2. Compute α, β and ξ
3. Estimate $\overline{\lambda} = (\overline{\mathbf{A}}, \overline{\mathbf{B}}, \overline{\pi})$
4. Replace λ with $\overline{\lambda}$
5. Repeat from step 2 until convergence

Constraints:

$$\sum_{i=1}^{N} \overline{\pi}_i = 1$$

$$\sum_{j=1}^{N} \overline{a}_{ij} = 1 \qquad 1 \le i \le N$$

$$\sum_{k=1}^{M} \overline{b}_j(k) = 1 \qquad 1 \le j \le N$$

It can be shown that $\mathrm{P}(\mathbf{O} \mid \overline{\lambda}) > \mathrm{P}(\mathbf{O} \mid \lambda)$ unless $\overline{\lambda} = \lambda$

Mixture Gaussian PDFs

- Probability distribution of the state is a gaussian mixture

$$b_j(\mathbf{o_t}) = \sum_{k=1}^{M} c_{jk} \mathrm{N}(\mathbf{o_t}, \mu_{jk}, \Sigma_{jk}) \qquad \sum_{k=1}^{M} c_{jk} = 1$$

$$c_{jk} \geq 0 \qquad 1 \leq \mathrm{k} \leq \mathrm{M}$$

- Probability of being in state *j* at time *t* with the mixture component *k* accounting for the observation o_t is:

$$\gamma_t(j,k) = \left[\frac{\alpha_t(j)\beta_t(j)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)} \right] \left[\frac{c_{jk} \mathrm{N}(\mathbf{o}_t, \mu_{jk}, \Sigma_{jk})}{\sum_{m=1}^{M} c_{jm} \mathrm{N}(\mathbf{o}_t, \mu_{jm}, \Sigma_{jm})} \right]$$

Parameter update equations for GMM PDF

- Mixture weight and mean:

$$\bar{c}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k)}{\sum_{t=1}^{T} \sum_{k'=1}^{M} \gamma_t(j,k')} \qquad \bar{\mu}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k) \cdot \mathbf{o}_t}{\sum_{t=1}^{T} \gamma_t(j,k)}$$

- Transition matrix elements a_{ij} get updates same way as in the case of discrete symbols
- Covariance matrix:

$$\bar{\Sigma}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k) \cdot (\mathbf{o}_t - \bar{\mu}_{jk})(\mathbf{o}_t - \bar{\mu}_{jk})'}{\sum_{t=1}^{T} \gamma_t(j,k)}$$

Multiple observation sequences

- Variability in producing each sound unit is modeled by estimating HMM parameters from multiple examples of speech, collected form different speakers

- Assume *K* training observation sequences

$$\mathbf{O} = [\mathbf{O}^{(1)}, \mathbf{O}^{(2)}, ..., \mathbf{O}^{(K)}]$$

$$\mathbf{O}^{(k)} = \{\mathbf{o}_1^k, \mathbf{o}_2^k, ... \mathbf{o}_{T_k}^k\}$$

$$\mathrm{P_k} = \mathrm{P}(\mathbf{O}^{(k)} \mid \lambda)$$

Parameter update for multiple observations

$$\bar{a}_{ij} = \frac{\sum_{k=1}^{K} \frac{1}{P_k} \sum_{t=1}^{T_k - 1} \alpha_t^k(i) a_{ij} b_j(\mathbf{o_{t+1}^{(k)}}) \beta_{t+1}^k(j)}{\sum_{k=1}^{K} \frac{1}{P_k} \sum_{t=1}^{T_k - 1} \alpha_t^k(i) \beta_t^k(i)}$$

$$\bar{b}_j(l) = \frac{\sum_{k=1}^{K} \frac{1}{P_k} \sum_{\substack{t=1 \\ s.t. O_t = v_l}}^{T_k - 1} \alpha_t^k(i) \beta_t^k(i)}{\sum_{k=1}^{K} \frac{1}{P_k} \sum_{t=1}^{T_k - 1} \alpha_t^k(i) \beta_t^k(i)}$$

One-state HMM with M component GMM

- Observation probability is a GMM with M components

$$b(\mathbf{o_t}) = \sum_{k=1}^{M} w_k b_k(\mathbf{o_t}, \mu_k, \Sigma_k)$$

- Probability that o_t is generated by the *k*th component

$$P(k \mid o_t, \lambda) = \frac{w_k b_k(o_t)}{\sum_{k=1}^{M} w_k b_k(o_t)}$$

Update equations for one-state HMM

Updated (new) parameter estimates

$$\boxed{\overline{w}_k} = \frac{1}{T}\sum_{t=1}^{T} \boxed{P(k \mid o_t, \lambda)}$$

$$\boxed{\overline{u}_k} = \frac{\sum_{t=1}^{T} P(k \mid o_t, \lambda) \cdot o_t}{\sum_{t=1}^{T} P(k \mid o_t, \lambda)}$$

$$\boxed{\overline{\Sigma}_k} = \frac{\sum_{t=1}^{T} P(k \mid o_t, \lambda) \cdot (o_t)^2}{\sum_{t=1}^{T} P(k \mid o_t, \lambda)} - (\overline{u}_k)^2$$

This term is computed using model parameters from previous algorithm iteration.