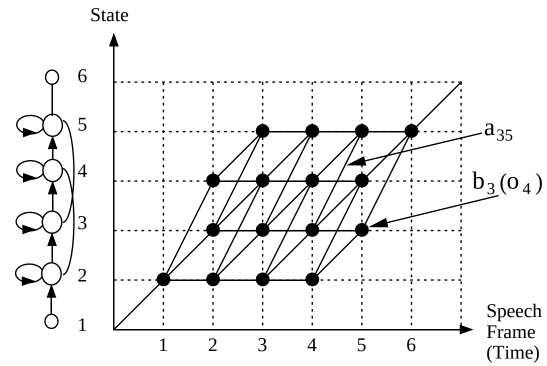
UNIVERSITY OF SURREY
Centre for Vision Speech & Signal Processing
University of Surrey, Guildford GU2 7XH.

HMM Likelihoods

Dr Philip Jackson

- Task 1: computing likelihoods
  - Forward procedure
  - Backward procedure
- Task 2: finding best alignment
  - Viterbi algorithm
  - Trellis diagram

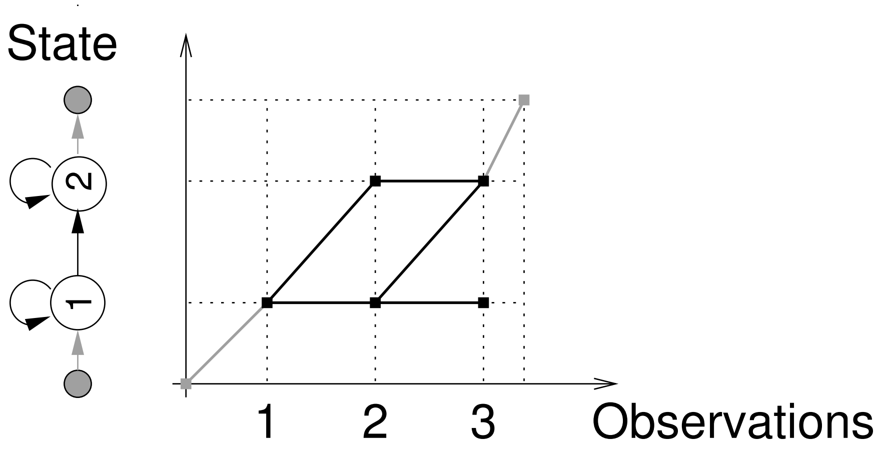


from (Young et al. 1997)

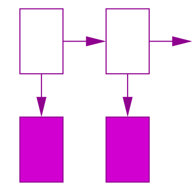
HMM Recognition & Training

Three tasks within HMM framework

1. Compute likelihood of a set of observations for a given model, $P(\mathcal{O}|\lambda)$
2. Decode a test sequence by calculating the most likely path, X^*
3. Optimise pattern templates by training the model parameters, $\Lambda = \{\lambda\}$



Task 1: Computing $P(\mathcal{O}|\lambda)$



So far, we calculated the joint probability of the observations and state sequence, for a given model λ,

$$P(\mathcal{O}, X|\lambda) = P(X|\lambda)\, P(\mathcal{O}|X, \lambda)$$

For the total probability of the observations, we marginalise the state sequence by summing over all possible X:

$$P(\mathcal{O}|\lambda) = \sum_{\text{all } X} P(\mathcal{O}, X|\lambda) = \sum_{\text{all } \boldsymbol{x}_1^T} P(\mathbf{o}_1^T, \boldsymbol{x}_1^T|\lambda) \qquad (1)$$

Now, we define **forward likelihood** for state j as

$$\alpha_t(j) = P(\mathbf{o}_1^t, x_t = j|\lambda) = \sum_{\{\boldsymbol{x}_1^{t-1}, x_t = j\}} P(\mathbf{o}_1^t, \boldsymbol{x}_1^t|\lambda) \qquad (2)$$

and apply the HMM's simplifying assumptions to yield

$$\alpha_t(j) = \sum_{i=1}^{N} \alpha_{t-1}(i)\, P(x_t = j|x_{t-1} = i, \lambda)\, P(o_t|x_t = j, \lambda) \quad (3)$$

as current state x_t depends only on previous state x_{t-1}, and observation o_t on current state (Gold & Morgan, 2000).

Forward procedure

To calculate **forward likelihood**, $\alpha_t(i) = P(\mathbf{o}_1^t, x_t = i|\lambda)$:

1. Initialise at $t = 1$,

   $\alpha_1(i) = \pi_i \, b_i(o_1)$ for $1 \le i \le N$

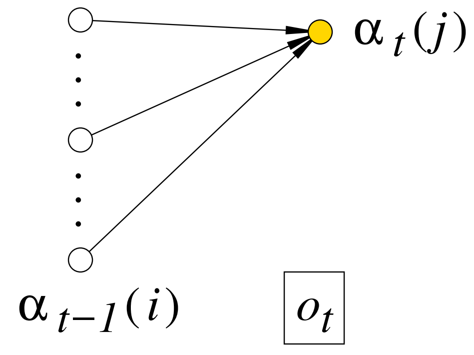
2. Recur for $t = \{2, 3, \ldots, T\}$,

   $$\alpha_t(j) = \left[\sum_{i=1}^{N} \alpha_{t-1}(i)\, a_{ij}\right] b_j(o_t) \qquad \text{for } 1 \le j \le N \tag{4}$$

3. Finalise,

   $P(\mathcal{O}|\lambda) = \sum_{i=1}^{N} \alpha_T(i)\, \eta_i$

Thus Task 1 is solved efficiently by recursion.



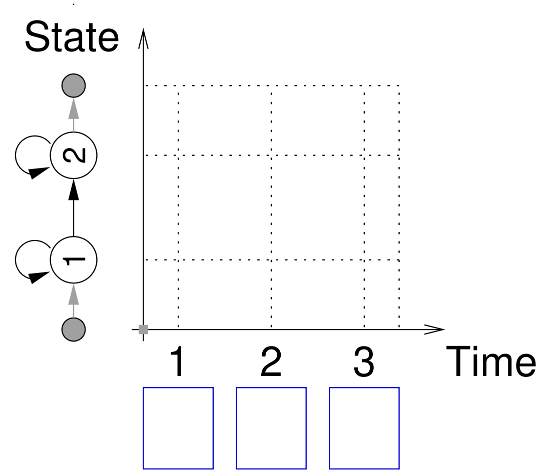
Forward procedure example

state transition matrix

$$A = \left[\begin{array}{c|cc|c} 0 & 1 & 0 & 0 \\ \hline 0 & 0.8 & 0.2 & 0 \\ 0 & 0 & 0.6 & 0.4 \\ \hline 0 & 0 & 0 & 0 \end{array}\right]$$

output matrix

$$B = \begin{bmatrix} 0.5 & 0.2 & 0.3 \\ 0 & 0.9 & 0.1 \end{bmatrix}$$



$$\alpha_1(1) =$$

$$\alpha_1(2) =$$

$$\alpha_2(1) =$$

$$\alpha_2(2) =$$

$$\alpha_3(1) =$$

$$\alpha_3(2) =$$

$$P(\mathcal{O}|\lambda) =$$

Task 1: Backward procedure

We define **backward likelihood**, $\beta_t(i) = P(\mathbf{o}_{t+1}^T | x_t = i, \lambda)$, and calculate:

1. Initialise at $t = T$,

   $\beta_T(i) = \eta_i$ for $1 \leq i \leq N$

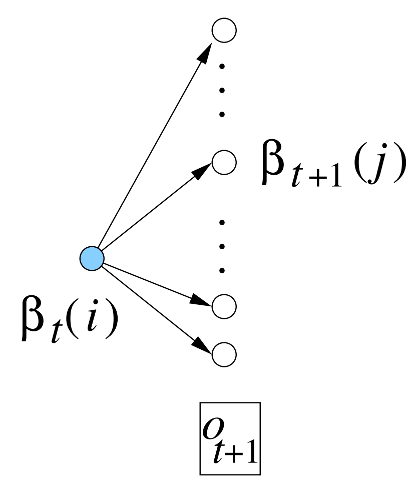
2. Recur for $t = \{T-1, T-2, \ldots, 1\}$,

   $$\beta_t(i) = \sum_{j=1}^{N} a_{ij}\, b_j(o_{t+1})\, \beta_{t+1}(j) \qquad \text{for } 1 \leq i \leq N \tag{5}$$

3. Finalise,

   $P(\mathcal{O}|\lambda) = \sum_{i=1}^{N} \pi_i\, b_i(o_1)\, \beta_1(i)$

This equivalently computes $P(\mathcal{O}|\lambda)$.



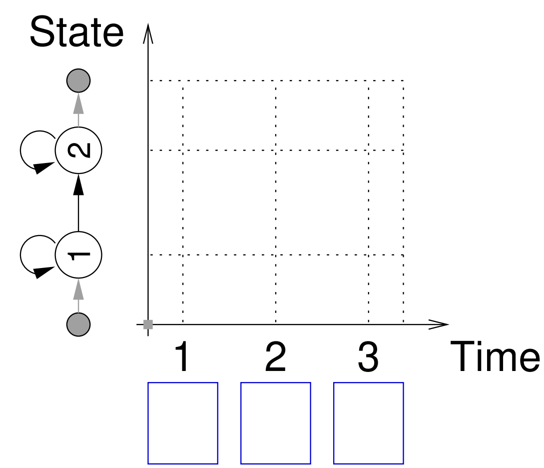
Backward procedure example

state transition matrix

$$A = \left[\begin{array}{c|cc|c} 0 & 1 & 0 & 0 \\ \hline 0 & 0.8 & 0.2 & 0 \\ 0 & 0 & 0.6 & 0.4 \\ \hline 0 & 0 & 0 & 0 \end{array}\right]$$

output matrix

$$B = \begin{bmatrix} 0.5 & 0.2 & 0.3 \\ 0 & 0.9 & 0.1 \end{bmatrix}$$



$$\beta_3(1) \;=$$

$$\beta_3(2) \;=$$

$$\beta_2(1) \;=$$

$$\beta_2(2) \;=$$

$$\beta_1(1) \;=$$

$$\beta_1(2) \;=$$

$$P(\mathcal{O}|\lambda) \;=$$

Task 2: finding the best path

Given observations $\mathcal{O} = \{o_1, \ldots, o_T\}$, find the HMM state sequence $X = \{x_1, \ldots, x_T\}$ that has greatest likelihood

$$X^* = \arg\max_X P(\mathcal{O}, X|\lambda), \tag{6}$$

where

$$\begin{aligned} P(\mathcal{O}, X|\lambda) &= P(\mathcal{O}|X, \lambda)P(X|\lambda) \\ &= \left(\prod_{t=1}^{T} a_{x_{t-1}x_t} b_{x_t}(o_t) \right) \eta_{x_T} \end{aligned} \tag{7}$$

Viterbi algorithm is an inductive method to find optimal state sequence X^* efficiently, similar to forward procedure. It computes **maximum cumulative likelihood** $\delta_t(j)$ up to current time t for each state j:

$$\delta_t(j) = \max_{\{\boldsymbol{x}_1^{t-1}, x_t = j\}} P(\mathbf{o}_1^t, \boldsymbol{x}_1^{t-1}, x_t = j|\lambda) \tag{8}$$

Viterbi algorithm

To compute the **maximum cumulative likelihood**, $\delta_t(i)$:

1. Initialise at $t = 1$,
   $\delta_1(i) = \pi_i b_i(o_1)$
   $\psi_1(i) = 0$ for $1 \leq i \leq N$

2. Recur for $t = \{2, 3, \ldots, T\}$,
   $$\delta_t(j) = \max_i \left[\delta_{t-1}(i) a_{ij}\right] b_j(o_t) \quad (9)$$
   $$\psi_t(j) = \arg\max_i \left[\delta_{t-1}(i) a_{ij}\right] \quad \text{for } 1 \leq j \leq N$$

3. Finalise,
   $P(\mathcal{O}, X^*|\lambda) = \max_i \left[\delta_T(i)\eta_i\right]$
   $x_T^* = \arg\max_i \left[\delta_T(i)\eta_i\right]$

4. Trace back,
   for $t = \{T, T-1, \ldots, 2\}$,
   $x_{t-1}^* = \psi_t\left(x_t^*\right)$, and
   $X^* = \{x_1^*, x_2^*, \ldots, x_T^*\}$

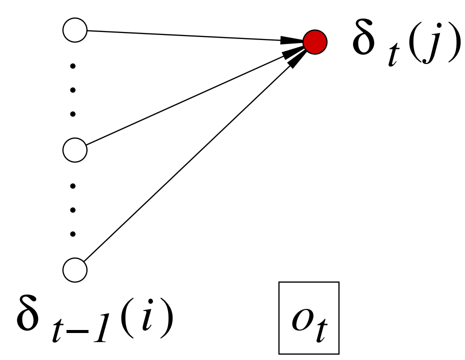
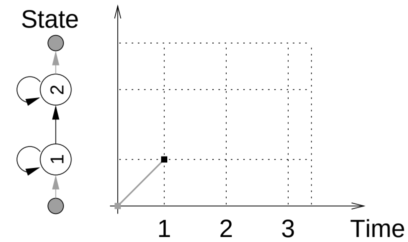
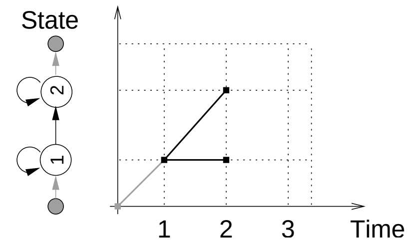


Illustration of the Viterbi algorithm

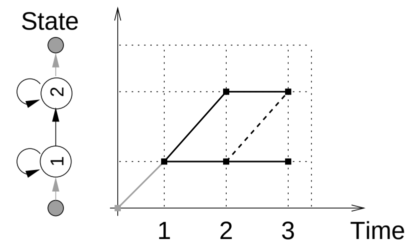
1. Initialise,
   $\delta_1(i) = \pi_i b_i(o_1)$
   $\psi_1(i) = 0$



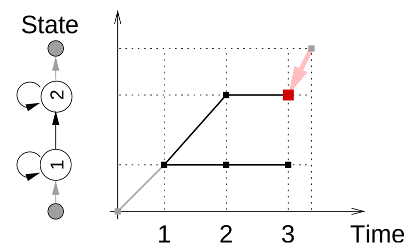
2. Recur for $t = 2$,
   $\delta_2(j) = \max_i \left[\delta_1(i) a_{ij}\right] b_j(o_2)$
   $\psi_2(j) = \arg\max_i \left[\delta_1(i) a_{ij}\right]$



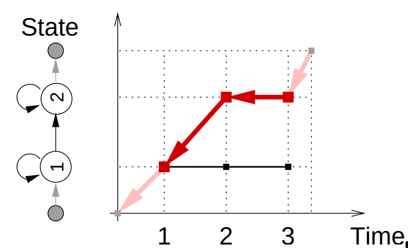
   Recur for $t = 3$,
   $\delta_3(j) = \max_i \left[\delta_2(i) a_{ij}\right] b_j(o_3)$
   $\psi_3(j) = \arg\max_i \left[\delta_2(i) a_{ij}\right]$



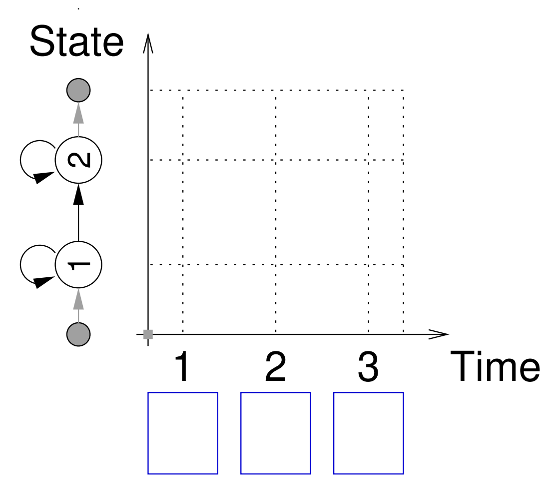
3. Finalise,
   $P(\mathcal{O}, X^* | \lambda) = \max_i \left[\delta_3(i) \eta_i\right]$
   $x_3^* = \arg\max_i \left[\delta_3(i) \eta_i\right]$



4. Trace back for $t = \{3..2\}$,
   $x_2^* = \psi_3\left(x_3^*\right)$
   $x_1^* = \psi_2\left(x_2^*\right)$
   $X^* = \{x_1^*, x_2^*, x_3^*\}$



Viterbi algorithm example



state transition matrix

$$A = \left[\begin{array}{c|cc|c} 0 & 1 & 0 & 0 \\ \hline 0 & 0.8 & 0.2 & 0 \\ 0 & 0 & 0.6 & 0.4 \\ \hline 0 & 0 & 0 & 0 \end{array}\right]$$

output matrix

$$B = \begin{bmatrix} 0.5 & 0.2 & 0.3 \\ 0 & 0.9 & 0.1 \end{bmatrix}$$

$\delta_1(1) =$ $\qquad\qquad$ $\psi_1(1) = 0$

$\delta_1(2) =$ $\qquad\qquad$ $\psi_1(2) = 0$

$\delta_2(1) =$ $\qquad\qquad$ $\psi_2(1) =$

$\delta_2(2) =$ $\qquad\qquad$ $\psi_2(2) =$

$\delta_3(1) =$ $\qquad\qquad$ $\psi_3(1) =$

$\delta_3(2) =$ $\qquad\qquad$ $\psi_3(2) =$

$P(\mathcal{O}, X^*|\lambda) =$ $\qquad\qquad$ $X^* = \{ \qquad\qquad \}$

Practical reformulation of the optimisation

Recall the likelihood calculation, eq. 7,

$$
\begin{aligned}
P(\mathcal{O}, X|\lambda) &= P(\mathcal{O}|X,\lambda)P(X|\lambda) \\
&= \left(\prod_{t=1}^{T} a_{x_{t-1}x_t}\, b_{x_t}(o_t) \right) \eta_{x_T}
\end{aligned}
$$

Taking the logarithm of both sides gives

$$
Q(X) = \left[\sum_{t=1}^{T} \left(\ln a_{x_{t-1}x_t} + \ln b_{x_t}(o_t) \right) + \ln \eta_{x_T} \right] \qquad (10)
$$

where the best path has the maximum log-likelihood

$$
Q^* = \max_X Q(X) \qquad (11)
$$

Since the log function is monotonic, eq. 6 becomes

$$
X^* = \arg\max_X Q(X) \qquad (12)
$$

Reformulated Viterbi algorithm

To compute **maximum cumulative log-likelihood**, $\ln \delta_t(i)$:

1. Initially at $t = 1$,
   $\ln \delta_1(i) = \ln \pi_i + \ln b_i(o_1)$
   $\psi_1(i) = 0$ for $1 \leq i \leq N$;

2. For $t = \{2, 3, \ldots, T\}$,
   $\ln \delta_t(j) = \max_i \left[\ln \delta_{t-1}(i) + \ln a_{ij}\right] + \ln b_j(o_t)$
   $\psi_t(j) = \arg\max_i \left[\ln \delta_{t-1}(i) + \ln a_{ij}\right]$
   for $1 \leq j \leq N$;

3. Finally,
   $Q^* = \max_i \left[\ln \delta_T(i) + \ln \eta_i\right]$
   $x_T^* = \arg\max_i \left[\ln \delta_T(i) + \ln \eta_i\right]$;

4. Trace back, for $t = \{T, T-1, \ldots, 2\}$,

$$x_{t-1}^* = \psi_t\left(x_t^*\right), \quad \text{and} \quad X^* = \{x_1^*, x_2^*, \ldots, x_T^*\} \tag{13}$$

HMM likelihoods summary

- Computing likelihoods, $P(\mathcal{O}|\lambda)$
  - Trellis diagrams
  - forward procedure
    to calculate $\alpha_t(i)$
  - backward procedure
    to calculate $\beta_t(i)$

- Finding the best state sequence
  - Viterbi algorithm
    to calculate Q^* and X^*

Homework

- Complete worked examples:
  - forward procedure
  - backward procedure
  - Viterbi algorithm

Next week in machine learning

- Task 3: training the parameters in the models $\Lambda = \{\lambda\}$
  - Forward-backward algorithm
  - Baum-Welch re-estimation

Further reading

L. R. Rabiner. *A tutorial on HMM and selected applications in speech recognition*. In *Proc. IEEE*, Vol. 77, No. 2, pp. 257–286, 1989.

B. Gold & N. Morgan, *Speech and Audio Signal Processing*, New York: Wiley, pp.346–347, 2000 [0-471-35154-7].

B. Gold, N. Morgan & D. Ellis, *Speech and Audio Signal Processing*, 2nd ed. (hardback), New York: Wiley, 2011 [0-470-19536-3].