Lecture 11: Hidden Markov Models

Cognitive Systems - Machine Learning

Cognitive Systems, Applied Computer Science, Bamberg University

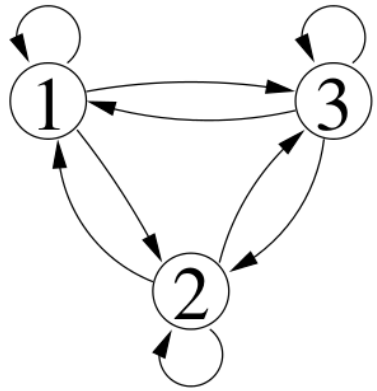
slides by **Dr. Philip Jackson**
Centre for Vision, Speech & Signal Processing
University of Surrey, UK

last change December 22, 2014

Outline

Introduction to Markov models

We can model stochastic sequences using a Markov chain, e.g., the state topology of an ergodic Markov model:



For 1st-order Markov chains, probability of state occupation depends only on the previous step (Rabiner, 1989):

$$P(x_t = j | x_{t-1} = i, x_{t-2} = h, \ldots) \approx P(x_t = j | x_{t-1} = i)$$

So, if we assume the RHS of eq. 5 is independent of time, we can write the **state-transition probabilities**

$$a_{ij} = P(x_t = j | x_{t-1} = i), \qquad 1 \leq i, j \leq N$$

with the properties

$$a_{ij} \geq 0 \quad \textit{and} \quad \sum_{j=1}^{N} a_{ij} = 1 \qquad 1 \leq i, j \leq N$$

Dr. Philip Jackson: Lecture slides for *Hidden Markov Models*

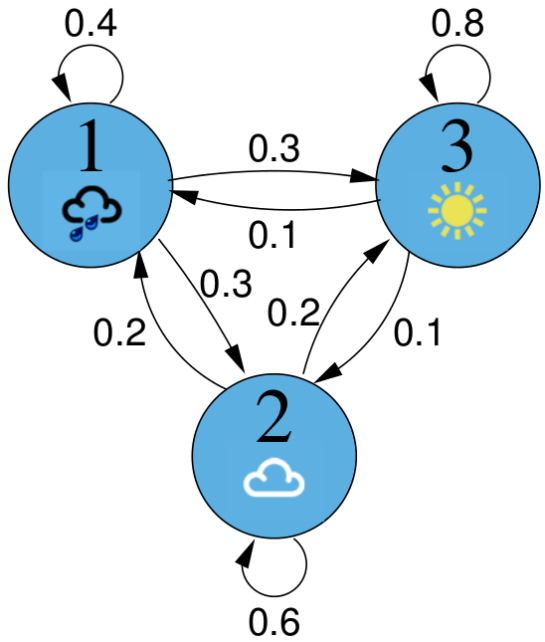
Weather prediction example

Let us represent the state of the weather by a 1st-order, ergodic Markov model, $\mathcal{M}$:

state 1: raining

state 2: cloudy

state 3: sunny



with state-transition probabilities expressed in matrix form:

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

Dr. Philip Jackson: Lecture slides for *Hidden Markov Models*

Weather predictor probability calculation

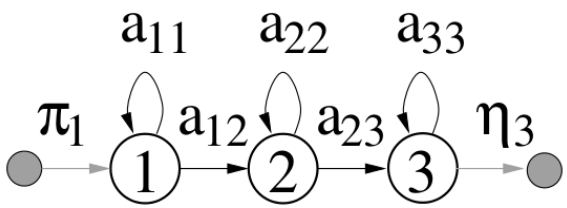
Given today's weather, what is the probability of directly observing the sequence of weather states "rain-sun-sun" with model $\mathcal{M}$?

$$A = \begin{array}{c} \\ rain \\ cloud \\ sun \end{array} \begin{array}{c} \begin{array}{ccc} rain & cloud & sun \end{array} \\ \begin{bmatrix} 0.4 & 0.3 & 0.3 \\ 0.2 & 0.6 & 0.2 \\ 0.1 & 0.1 & 0.8 \end{bmatrix} \end{array}$$

$$\begin{aligned} P(X|\mathcal{M}) &= P(X = \langle 1,3,3 \rangle |\mathcal{M}) \\ &= P(x_1 = rain|today) \times P(x_2 = sun|x_1 = rain) \\ &\quad \times P(x_3 = sun|x_2 = sun) \\ &= a_{21} a_{13} a_{33} \\ &= 0.2 \times 0.3 \times 0.8 \\ &= 0.048 \end{aligned}$$

Start and end of a stochastic state sequence

Null states deal with the start and end of sequences, as in the state topology of this left-right Markov model:



Entry probabilities at $t = 1$ for each state i are defined

$$\pi_i = P(x_1 = i) \qquad 1 \leq i \leq N$$

with the properties $\pi_i \geq 0$ for $1 \leq i \leq N$ and $\sum_{i=1}^{N} \pi_i = 1$.

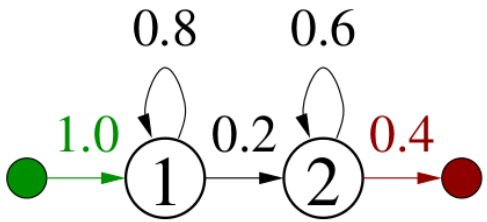
Exit probabilities at $t = T$ are similarly defined

$$\eta_i = P(x_T = i) \qquad 1 \leq i \leq N$$

with the properties $\eta_i \geq 0$ and $\eta_i + \sum_{j=1}^{N} a_{ij} = 1$ for $1 \leq i \leq N$

Example: probability of MM state sequence

Consider the state topology



state transition probabilities

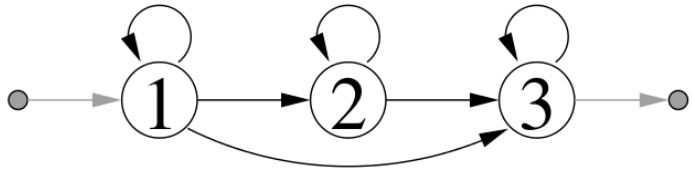
$$A = \begin{array}{c|ccc} 0 & 1 & 0 & 0 \\ \hline 0 & 0.8 & 0.2 & 0 \\ 0 & 0 & 0.6 & 0.4 \\ \hline 0 & 0 & 0 & 0 \end{array}$$

The probability of state sequence $X = \langle 1, 2, 2 \rangle$ is

$$\begin{aligned} P(X|\mathcal{M}) &= \pi_1 a_{12} a_{22} \eta_2 \\ &= 1 \times 0.2 \times 0.6 \times 0.4 \\ &= 0.048 \end{aligned}$$

Summary of Markov models

State topology diagram:



Entry probabilities $\pi = \{\pi_i\} = [\ 1 \quad 0 \quad 0\ ]$ and exit probabilities $\eta = \{\eta_i\} = [\ 0 \quad 0 \quad 0.2\ ]^T$ are combined with state transition probabilities in complete *A* matrix:

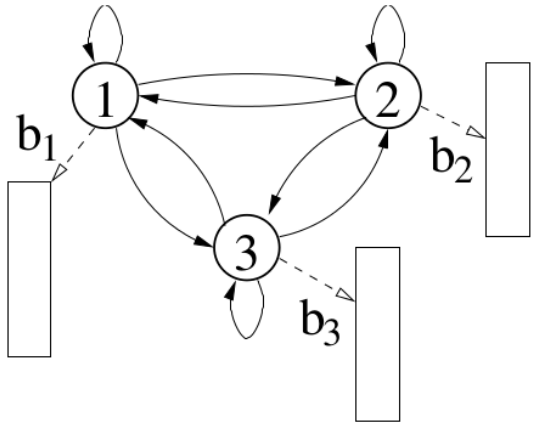
$$A = \left[\begin{array}{c|ccc|c} 0 & 1 & 0 & 0 & 0 \\ \hline 0 & 0.6 & 0.3 & 0.1 & 0 \\ 0 & 0 & 0.9 & 0.1 & 0 \\ 0 & 0 & 0 & 0.8 & 0.2 \\ \hline 0 & 0 & 0 & 0 & 0 \end{array}\right]$$

Probability of a given state sequence *X*:

$$P(X|\mathcal{M}) = \pi_{x_1} \left(\prod_{t=2}^{T} a_{x_{t-1} x_t} \right) \eta_{x_T}$$

Introduction to Hidden Markov Models

Hidden Markov Models (HMMs) use a Markov chain to model stochastic state sequences which emit stochastic observations, e.g., the state topology of an ergodic HMM:



Probability of state i generating a **discrete** observation o_t, which has one of a finite set of values, $k \in 1..K$, is

$$b_i(k) = P(o_t = k | x_t = i).$$

Dr. Philip Jackson: Lecture slides for *Hidden Markov Models*

Parameters of a discrete HMM, λ

State transition probabilities,
$A = \{\pi_j, a_{ij}, \eta_i\} = \{P(x_t = j | x_{t-1} = i)\}$ for $1 \leq i, j \leq N$
where N is the number of states

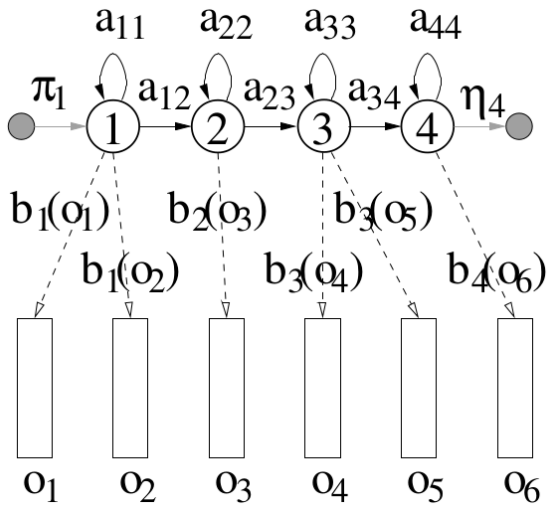
Discrete output probabilities,
$B = \{b_i(k)\} = \{P(o_t = k | x_t = i)\}$ for $1 \leq i \leq N$
$1 \leq k \leq K$

where K is the number of distinct observations.

HMM Probability Calculation

The joint likelihood of state and observation sequences is

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

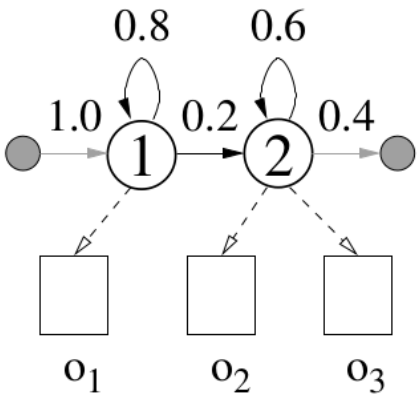


The state sequence $X = \langle 1, 1, 2, 3, 3, 4 \rangle$ produces the observations $\mathcal{O} = \langle o_1, o_2, \ldots, o_6 \rangle$:

$$\begin{aligned} P(X|\lambda) &= \pi_1 a_{11} a_{12} a_{23} a_{33} a_{34} \eta_4 \\ P(\mathcal{O}|X, \lambda) &= b_1(o_1) b_1(o_2) b_2(o_3) b_3(o_4) b_3(o_5) b_4(o_6) \\ P(\mathcal{O}, X|\lambda) &= \pi_{x_1} b_{x_1}(o_1) \left(\prod_{t=2}^{T} a_{x_{t-1} x_t} b_{x_t}(o_t) \right) \eta_{x_T} \end{aligned}$$

Example: probability of HMM state sequence

Consider state topology,



state transition matrix, and

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

output probabilities

$$B = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} = \begin{array}{c} \begin{matrix} \text{R} & \text{G} & \text{B} \end{matrix} \\ \begin{bmatrix} 0.5 & 0.2 & 0.3 \\ 0 & 0.9 & 0.1 \end{bmatrix} \end{array}$$

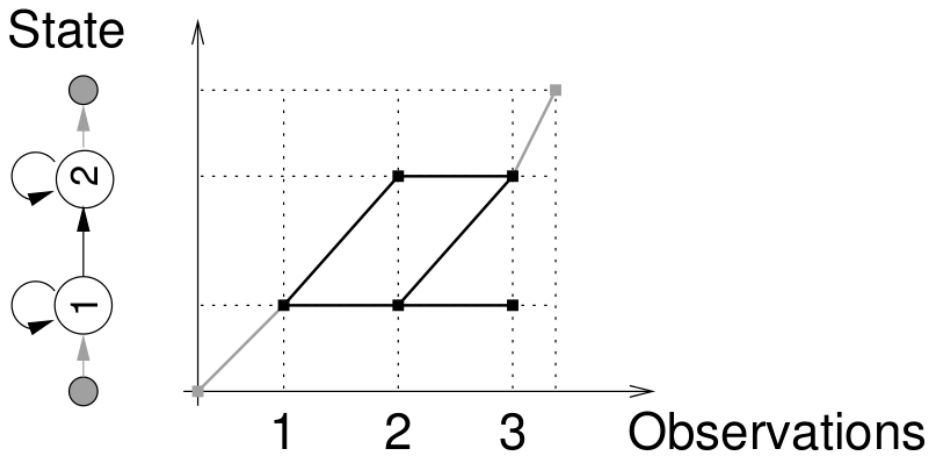
What is the probability of observations $\mathcal{O} = \langle R, G, G \rangle$ with state sequence $X = \langle 1, 2, 2 \rangle$?

$$\begin{aligned} P(\mathcal{O}, X|\lambda) &= P(X|\lambda)P(\mathcal{O}|X, \lambda) \\ &= \pi_1 b_1(o_1) a_{12} b_2(o_2) a_{22} b_2(o_3) \eta_2 \\ &= \mathbf{1} \times \mathbf{0.5} \times \mathbf{0.2} \times \mathbf{0.9} \times \mathbf{0.6} \times \mathbf{0.9} \times \mathbf{0.4} \\ &= \mathbf{0.01944} \approx \mathbf{0.02} \end{aligned}$$

HMM Recognition & Training

Three tasks within HMM framework

1. Compute likelihood of a set of observations with 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 parameters in the models



Dr. Philip Jackson: Lecture slides for *Hidden Markov Models*

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

- So far, we calculated the joint probability of 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

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

Alternate Notation

$$\mathcal{O} = \mathbf{o}_1^T = \mathbf{o}_t^1.\mathbf{o}_{t+1}^T \qquad 1 \leq t < T$$

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

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

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

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

- considering the (unknown) previous state x_{t-1} this can be reformulated:

$$\begin{aligned}
\alpha_t(j) &= \sum_{\mathbf{x}_1^{t-2}, x_{t-1}, x_t = j} P(\mathbf{o}_1^t, \mathbf{x}_1^t|\lambda) \\
&= \sum_{\mathbf{x}_1^{t-2}, x_{t-1}} P(\mathbf{o}_1^{t-1}, \mathbf{x}_1^{t-1}|\lambda) a_{x_{t-1}j} b_j(o_t) \\
&= \sum_{i=1}^{N} \sum_{\mathbf{x}_1^{t-2}, x_{t-1} = i} P(\mathbf{o}_1^{t-1}, \mathbf{x}_1^{t-1}|\lambda) a_{ij} b_j(o_t) \\
&= \sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij} b_j(o_t)
\end{aligned}$$

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 $1 \leq i \leq N$
  2. Recur for $t = \{2, 3, \ldots, T\}$,
     $\alpha_t(j) = [\sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij}] b_j(o_t)$ for $1 \leq j \leq N$
  3. Finalise,
     $P(\mathcal{O}|\lambda) = \sum_{i=1}^{N} \alpha_T(i) \eta_i$

- Thus, we can solve Task 1 efficiently by recursion.

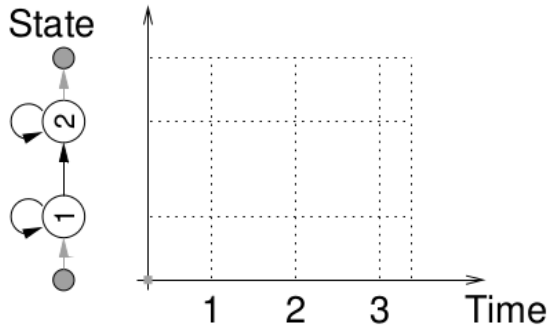
Worked example of the forward procedure

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

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



$$\alpha_1(1) = \mathbf{1} \times \mathbf{0.5} = \mathbf{0.5}$$

$$\alpha_1(2) = \mathbf{0} \times \mathbf{0} = \mathbf{0}$$

$$\alpha_2(1) = \mathbf{0.5} \times \mathbf{0.8} \times \mathbf{0.2} = \mathbf{0.08}$$

$$\alpha_2(2) = \mathbf{0.5} \times \mathbf{0.2} \times \mathbf{0.9} = \mathbf{0.09}$$

$$\alpha_3(1) = \mathbf{0.08} \times \mathbf{0.8} \times \mathbf{0.2} = \mathbf{0.0128}$$

$$\alpha_3(2) = (\mathbf{0.08} \times \mathbf{0.2} + \mathbf{0.09} \times \mathbf{0.6}) \times \mathbf{0.9} = \mathbf{0.063}$$

$$P(\mathcal{O}|\lambda) = \mathbf{0.063} \times \mathbf{0.4} = \mathbf{0.0252}$$

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. Recure for $t = \{T-1, T-2, \ldots, 1\}$,
   $\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)$ for $1 \leq i \leq N$
3. Finalise,
   $P(\mathcal{O}|\lambda) = \sum_{i=1}^{N} \pi_i b_i(o_1) \beta_1(i)$

- This is an equivalent way of computing $P(\mathcal{O}|\lambda)$ recursively.

Finding the best path

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

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

where

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

$$= \pi_{x_1} b_{x_1}(o_1) \left(\prod_{t=2}^{T} a_{x_{t-1} x_t} b_{x_t}(o_t) \right) \eta_{x_T}$$

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

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

Dr. Philip Jackson: Lecture slides for *Hidden Markov Models*

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 $t = \{2, 3, \ldots, T\}$,
   $\delta_t(j) = \max_i[\delta_{t-1}(i)a_{ij}]b_j(o_t)$
   $\psi_t(j) = \arg\max_i[\delta_{t-1}(i)a_{ij}]$ for $1 \leq j \leq N$
3. Finalise,
   $P(\mathcal{O}, X^*|\lambda) = \max_i[\delta_T(i)\eta_i]$
   $x_T^* = \arg\max_i[\delta_T(i)\eta_i]$
4. Trace back, for $t = \{T, T-1, \ldots, 2\}$,

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

Dr. Philip Jackson: Lecture slides for *Hidden Markov Models*

Illustration of the Viterbi algorithm

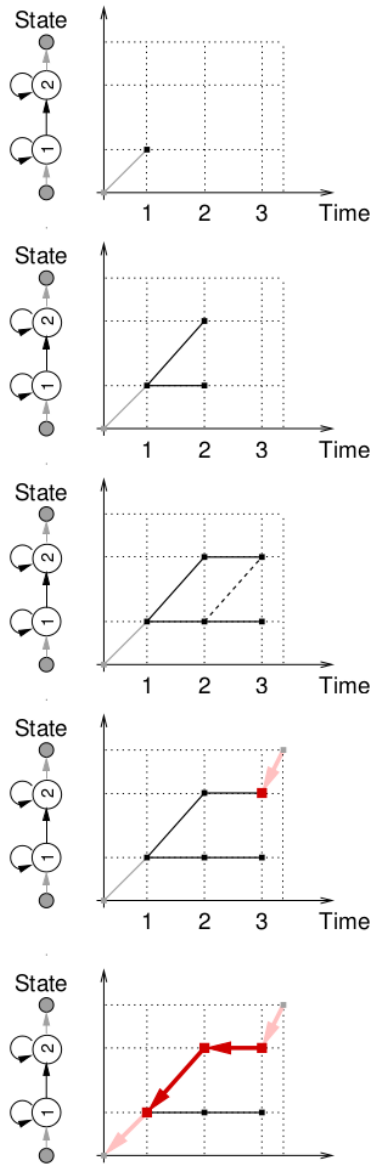
1. Initialise,
$$\delta_1(i) = \pi_i b_i(o_1)$$
$$\psi_1(i) = 0$$

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 $t = \{3..2\}$,
$$x_2^* = \psi_3(x_3^*)$$
$$x_1^* = \psi_2(x_2^*)$$
$$X^* = \{x_1^*, x_2^*, x_3^*\}$$



Dr. Philip Jackson: Lecture slides for *Hidden Markov Models*

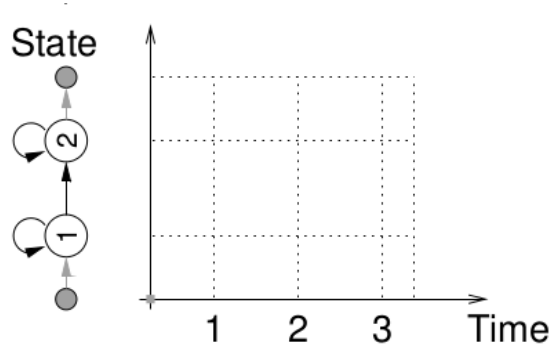
Worked example of the Viterbi algorithm

state transition matrix

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

output matrix

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



$$\delta_1(1) = 1 \times 0.5 = 0.5 \qquad \psi_1(1) = 0$$
$$\delta_1(2) = 0 \times 0 = 0 \qquad \psi_1(2) = 0$$

$$\delta_2(1) = 0.5 \times 0.8 \times 0.2 = 0.08 \qquad \psi_2(1) = 1$$
$$\delta_2(2) = 0.5 \times 0.2 \times 0.9 = 0.09 \qquad \psi_2(2) = 1$$

$$\delta_3(1) = 0.08 \times 0.8 \times 0.2 = 0.0128 \qquad \psi_3(1) = 1$$
$$\delta_3(2) = max(0.08 \times 0.2, 0.09 \times 0.6) \times 0.9 = 0.0486 \qquad \psi_3(2) = 2$$

$$P(\mathcal{O}, X^*|\delta) = 0.0486 \times 0.4 = 0.01944$$

Maximum likelihood training

In general, we want to find the value of some model parameter c that is most likely to give our set of training data $\mathcal{O}_{train}$.

This maximum likelihood (ML) estimate $\hat{c}$ is obtained by setting the derivative of $P(\mathcal{O}_{train}|c)$ w.r.t. c equal to zero, which is equivalent to:

$$\frac{\partial ln(P(\mathcal{O}_{train}|\hat{c})}{\partial c} = 0$$

Solving this **likelihood equation** tells us how to optimise the model parameters in training.

Re-estimating the parameters of the model λ

As a preliminary approach, let us use the optimal path X^* computed by the Viterbi algorithm with some initial model parameters $\lambda = \{A, B\}$. In so doing, we approximate the total likelihood of the observations:

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

Using X^*, we make a hard binary decision about the state occupation, $q_t(i) \in \{0, 1\}$, and train the parameters of our model accordingly.

Viterbi training (hard state assignment)

Model parameters can be re-estimated using the Viterbi alignment to assign observations to states (a.k.a. segmental k-means training).

State-transition probabilities,

$$\hat{a}_{ij} = \frac{\sum_{t=2}^{T} q_{t-1}(i) q_t(j)}{\sum_{t=1}^{T} q_t(i)} \qquad \text{for } 1 \leq i, j \leq N$$

where state indicator $q_t(i) = \begin{cases} 1 & \text{for } i = x_t \\ 0 & \text{otherwise} \end{cases}$

Discrete output probabilities,

$$\hat{b}_j(k) = \frac{\sum_{t=1}^{T} q_t(j) \omega_t(k)}{\sum_{t=1}^{T} q_t(j)} \qquad \text{for } 1 \leq j \leq N \text{ and } 1 \leq k \leq K$$

where event indicator $\omega_t(k) = \begin{cases} 1 & \text{for } k = o_t \\ 0 & \text{otherwise} \end{cases}$

Dr. Philip Jackson: Lecture slides for *Hidden Markov Models*

Maximum likelihood training by EM

Baum-Welch re-estimation (occupation)

Yet, the hidden state occupation is not known with absolute certainty. So, the expectation maximisation (EM) method optimises the model parameters based on soft assignment of observations to states via the **occupation likelihood**,

$$\gamma_t(i) = P(x_t = i|\mathcal{O}, \lambda)$$

relaxing the assumption previously made two slides before

We can rearrange, using Bayes theorem, to obtain

$$\gamma_t(i) = \frac{P(\mathcal{O}, x_t = i|\lambda)}{P(\mathcal{O}|\lambda)}$$

$$= \frac{\alpha_t(i)\beta_t(i)}{P(\mathcal{O}|\lambda)}$$

where α_t, β_t and $P(\mathcal{O}|\lambda)$ are computed by the forward and backward procedures.

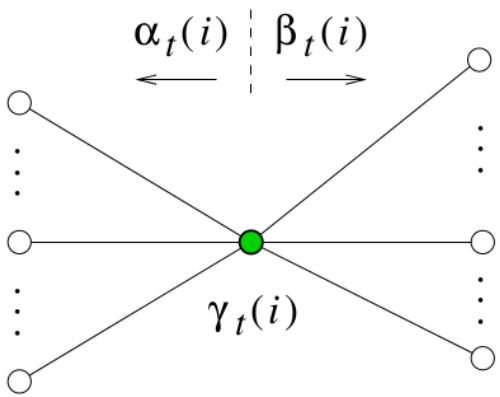
Baum-Welch re-estimation (transition)

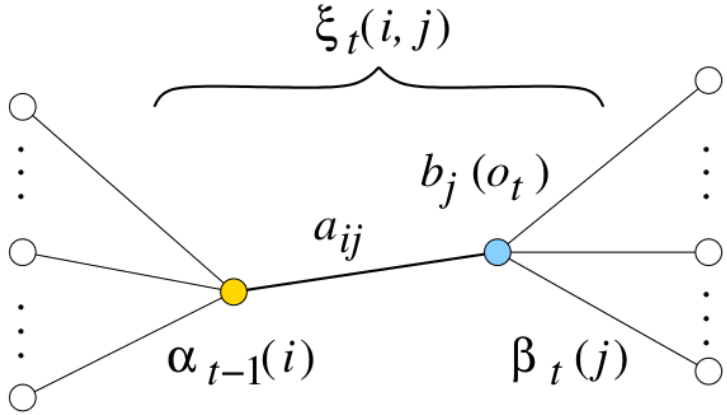
Similarly, we also define a **transition likelihood**,

$$\xi_t(i,j) = P(x_{t-1} = i, x_t = j|\mathcal{O}, \lambda)$$

$$= \frac{P(\mathcal{O}, x_{t-1} = i, x_t = j|\lambda)}{P(\mathcal{O}|\lambda)}$$

$$= \frac{\alpha_{t-1}(i)\, a_{ij} b_j(o_t)\, \beta_t(j)}{P(\mathcal{O}|\lambda)}$$





Trellis depiction of (left) occupation and (right) transition likelihoods

Baum-Welch training (soft state assignment)

State-transition probabilities,

$$\hat{a}_{ij} = \frac{\sum_{t=2}^{T} \xi_t(i,j)}{\sum_{t=1}^{T} \gamma_t(i)} \qquad \text{for } 1 \leq i, j \leq N$$

Discrete output probabilities,

$$\hat{b}_j(k) = \frac{\sum_{t=1}^{T} \gamma_t(j)\omega_t(k)}{\sum_{t=1}^{T} \gamma_t(j)} \qquad \text{for } 1 \leq j \leq N \text{ and } 1 \leq k \leq K$$

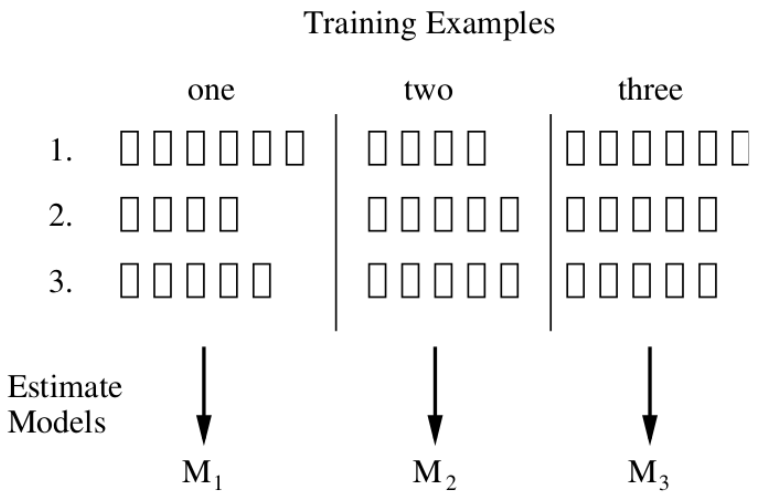
Re-estimation increases the likelihood over the training data for the new model $\hat{\lambda}$

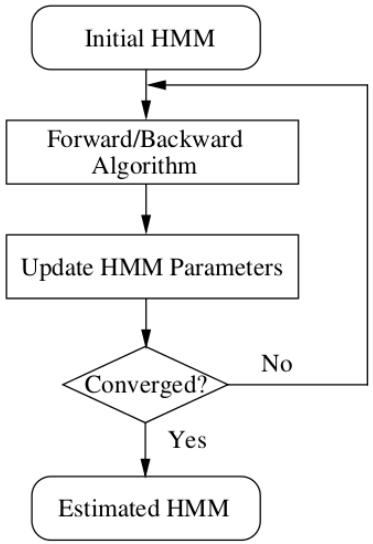
$$P(\mathcal{O}_{train}|\hat{\lambda}) \geq P(\mathcal{O}_{train}|\lambda)$$

although it does not guarantee a global maximum.

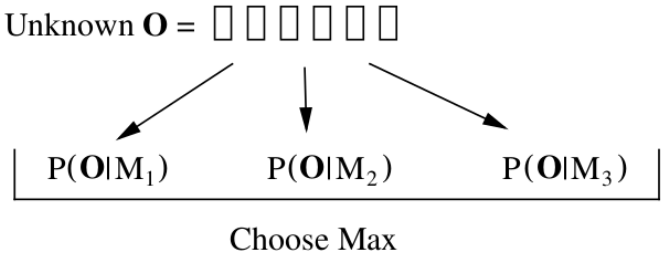
Use of HMMs with training and test data

(a) Training





(b) Recognition



Isolated word training and recognition (Young et al., 2009)

Learning Terminology

Hidden Markov Models

Supervised Learning	unsupervised learning

Approaches:

Concept / Classification	Policy Learning
symbolic	**statistical** / neuronal network
inductive	analytical

Learning Strategy:	**learning from examples**
Data:	**sequences of discrete observations**
Target Values:	**probabilities for classes**