Machine learning: lecture 16

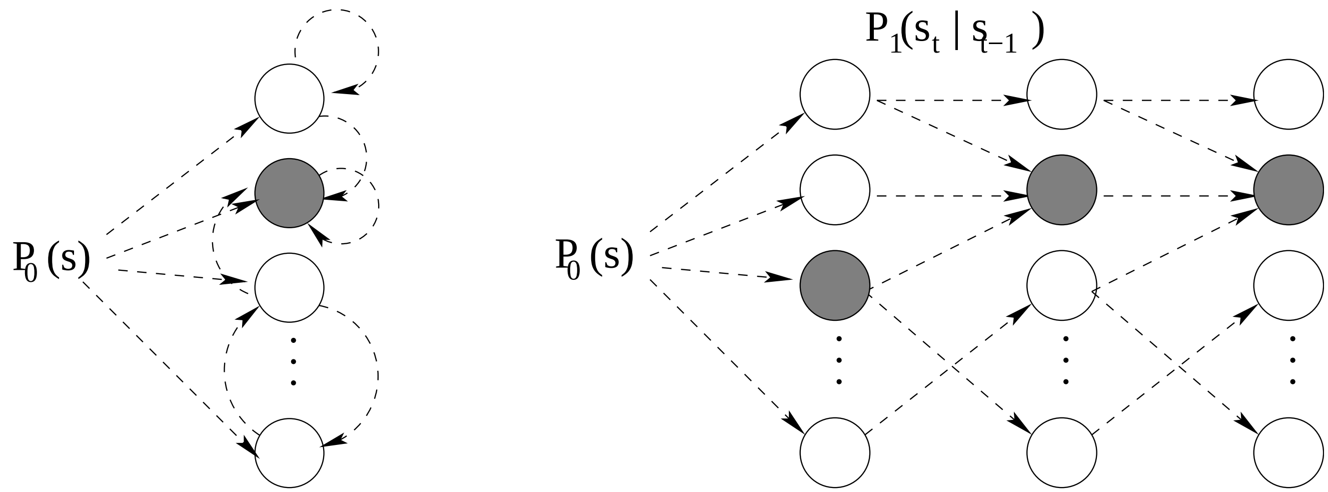
Tommi S. Jaakkola
MIT AI Lab
tommi@ai.mit.edu

Topics

- Structured probability models
  - Markov models
  - Hidden markov models

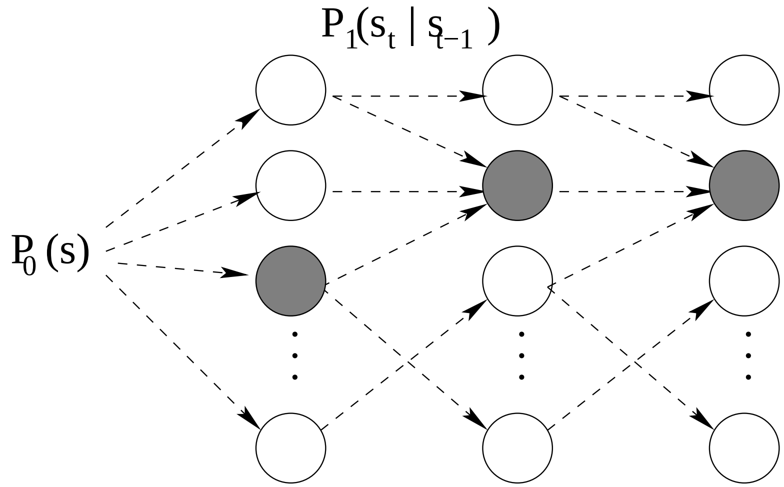
Markov chain: review

- A first order (homogeneous) Markov chain:



- The initial state s_0 is drawn from $P_0(s_0)$. Successive states are drawn from the one step transition probabilities $P_1(s_{t+1}|s_t)$

Markov chain: properties



$$s_0 \rightarrow s_1 \rightarrow s_2 \rightarrow \ldots$$

- If there exists a finite k such that any state i can lead to any other state j after exactly k steps, the markov chain is *ergodic*:

$$P(s_{t+k} = j | s_t = i) > 0 \text{ for all } i, j \text{ and some finite } k$$

Markov chains

- Problems we have to solve
  1. Prediction
  2. Estimation

- **Prediction**: what is the probability distribution over the possible states s_{t+k} at time $t+k$ if we start from $s_t = i$?

$$
\begin{aligned}
& P_1(s_{t+1}|s_t = i) \\
P_2(s_{t+2}|s_t = i) &= \sum_{s_{t+1}} P_1(s_{t+1}|s_t = i)\, P_1(s_{t+2}|s_{t+1}) \\
& \cdots \\
P_k(s_{t+k}|s_t = i) &= \sum_{s_{t+k-1}} P_{k-1}(s_{t+k-1}|s_t = i)\, P_1(s_{t+k}|s_{t+k-1})
\end{aligned}
$$

where $P_k(s'|s)$ is the k-step transition probability matrix.

Markov chain: estimation

- We need to estimate the initial state distribution $P_0(s_0)$ and the transition probabilities $P_1(s'|s)$
- Estimation from L observed sequences of different lengths

$$
\begin{array}{l}
s_0^{(1)} \rightarrow s_1^{(1)} \rightarrow s_2^{(1)} \rightarrow \ldots \rightarrow s_{n_1}^{(1)} \\
\ldots \\
s_0^{(L)} \rightarrow s_1^{(L)} \rightarrow s_2^{(L)} \rightarrow \ldots \rightarrow s_{n_L}^{(L)}
\end{array}
$$

Maximum likelihood estimates (observed fractions)

$$\hat{P}_0(s_0 = i) \;=\; \frac{1}{L}\sum_{l=1}^{L} \delta(s_0^{(l)}, i)$$

where $\delta(x, y) = 1$ if $x = y$ and zero otherwise

Markov chain: estimation

$$s_0^{(1)} \rightarrow s_1^{(1)} \rightarrow s_2^{(1)} \rightarrow \ldots \rightarrow s_{n_1}^{(1)}$$

$$\ldots$$

$$s_0^{(L)} \rightarrow s_1^{(L)} \rightarrow s_2^{(L)} \rightarrow \ldots \rightarrow s_{n_L}^{(L)}$$

- The transition probabilities are obtained as observed fractions of transitions out of a specific state

  Joint estimate over successive states

$$\hat{P}_{s,s'}(s=i, s'=j) \;\; = \;\; \frac{1}{(\sum_{l=1}^{L} n_l)} \sum_{l=1}^{L} \sum_{t=0}^{n_l - 1} \delta(s_t^{(l)}, i)\delta(s_{t+1}^{(l)}, j)$$

  and the transition probability estimates

$$\hat{P}_1(s'=j|s=i) \;\; = \;\; \frac{\hat{P}_{s,s'}(s=i, s'=j)}{\sum_k \hat{P}_{s,s'}(s=i, s'=k)}$$

Markov chain: estimation

- Can we simply estimate Markov chains from a single long sequence?

$$s_0 \longrightarrow s_1 \longrightarrow s_2 \longrightarrow \ldots \longrightarrow s_n$$

  - Ergodicity?
  - What about the initial state distribution $\hat{P}_0(s_0)$?

Clustering by dynamics

- We can cluster time course signals by means of comparing their dynamics, where the dynamics is captured by a Markov chain model
  - system behavior monitoring (anomaly detection)
  - biosequencies

  etc.
- There are still many ways of using the Markov chain models for clustering (e.g., what is the clustering metric?)
- The approach we follow here is to derive a criterion for determining whether two (or more) sequences should be in the same cluster

Cluster criterion

- How can we tell whether two arbitrary sequences

$$S^{(1)} = \{s_0^{(1)}, \ldots, s_{n_1}^{(1)}\} \text{ and } S^{(2)} = \{s_0^{(2)}, \ldots, s_{n_2}^{(2)}\}$$

  should be in the same cluster?

- We can compare (approximate) description lengths of either encoding the sequencies separately or jointly

$$\mathsf{DL}^{(1)} + \mathsf{DL}^{(2)} \gtrless \mathsf{DL}^{(1+2)}$$

  where $\mathsf{DL}^{(1+2)}$ uses the same Markov chain for both sequencies while $\mathsf{DL}^{(1)}$ and $\mathsf{DL}^{(2)}$ are based on different models.

Cluster criterion cont'd

- Approximate description lengths:

$$\begin{aligned}
\mathsf{DL}^{(1)} + \mathsf{DL}^{(2)} &= -\log P(S^{(1)}|\hat{\theta}_1) + \frac{d}{2}\log(n_1) \\
&\quad -\log P(S^{(2)}|\hat{\theta}_2) + \frac{d}{2}\log(n_2) \\
\mathsf{DL}^{(1+2)} &= -\log P(S^{(1)}|\hat{\theta}) - \log P(S^{(2)}|\hat{\theta}) \\
&\quad +\frac{d}{2}\log(n_1 + n_2)
\end{aligned}$$

  where the maximum likelihood parameter estimates $\hat{\theta}_1$, $\hat{\theta}_2$, and $\hat{\theta}$ each include the initial state distribution and the transition probabilities; $d = 3$ for binary sequences.

- We are essentially testing here whether the two sequencies have the same first order Markov dynamics

Simple example

- Four binary sequences of length 50:

```
1.   00100110010001010000001000011101101010100...
2.   01011111101001101010000010000000101011001...
3.   1101011000000110110010001101111101011101...
4.   11010101111010111101111011011011010001 01...
```

Evaluations:

$$
\begin{aligned}
\mathsf{DL}^{(1)} + \mathsf{DL}^{(2)} - \mathsf{DL}^{(1+2)} &= 6.6\,\mathsf{bits} \\
\mathsf{DL}^{(1+2)} + \mathsf{DL}^{(3+4)} - \mathsf{DL}^{(1+2+3+4)} &= -0.9\,\mathsf{bits}
\end{aligned}
$$

Agglomerative hierarchical clustering with Euclidean distance would give $(((2,3),4),1)$

Beyond Markov chains

- Potential problems with using Markov chains
  - if the state is continuous
  - if we cannot fully determine what the current state is (e.g., due to noisy observations)
  - if the state is an abstraction and never directly observable
- We need to augment the markov chain with a model that relates the states to observables

Hidden Markov models

- A hidden Markov model (HMM) is model where we generate a sequence of outputs in addition to the Markov state sequence

$$\begin{array}{ccccccc} s_0 & \rightarrow & s_1 & \rightarrow & s_2 & \rightarrow & \ldots \\ \downarrow & & \downarrow & & \downarrow & & \\ \mathbf{x}_0 & & \mathbf{x}_1 & & \mathbf{x}_2 & & \end{array}$$

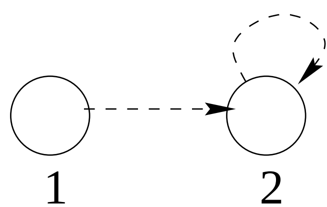
  - number of states m
  - initial state distribution $P_0(s_0)$
  - state transition model $P_1(s_{t+1}|s_t)$
  - output model $P_o(\mathbf{x}_t|s_t)$ (discrete or continuous)

- This is a *latent variable model* in the sense that we will only observe the outputs $\{\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_n\}$; the state sequence remains "hidden"

HMM example

- Two states 1 and 2; observations are tosses of unbiased coins

$$P_0(s=1)=0.5,\;\; P_0(s=2)=0.5$$
$$P_1(s'=1|s=1)=0,\;\; P_1(s'=2|s=1)=1$$
$$P_1(s'=1|s=2)=0,\;\; P_1(s'=2|s=2)=1$$
$$P_o(x=\text{heads}|s=1)=0.5,\;\; P_o(x=\text{tails}|s=1)=0.5$$
$$P_o(x=\text{heads}|s=2)=0.5,\;\; P_o(x=\text{tails}|s=2)=0.5$$



- This model is *unidentifiable* in the sense that the particular hidden state Markov chain has no effect on the observations

HMM example: biased coins

- Two states 1 and 2; outputs are tosses of *biased* coins

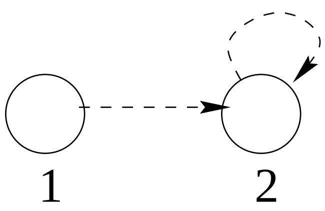
$$P_0(s=1) = 0.5, \;\; P_0(s=2) = 0.5$$

$$P_1(s'=1|s=1) = 0, \;\; P_1(s'=2|s=1) = 1$$

$$P_1(s'=1|s=2) = 0, \;\; P_1(s'=2|s=2) = 1$$

$$P_o(x=\text{heads}|s=1) = 0.25, \;\; P_o(x=\text{tails}|s=1) = 0.75$$

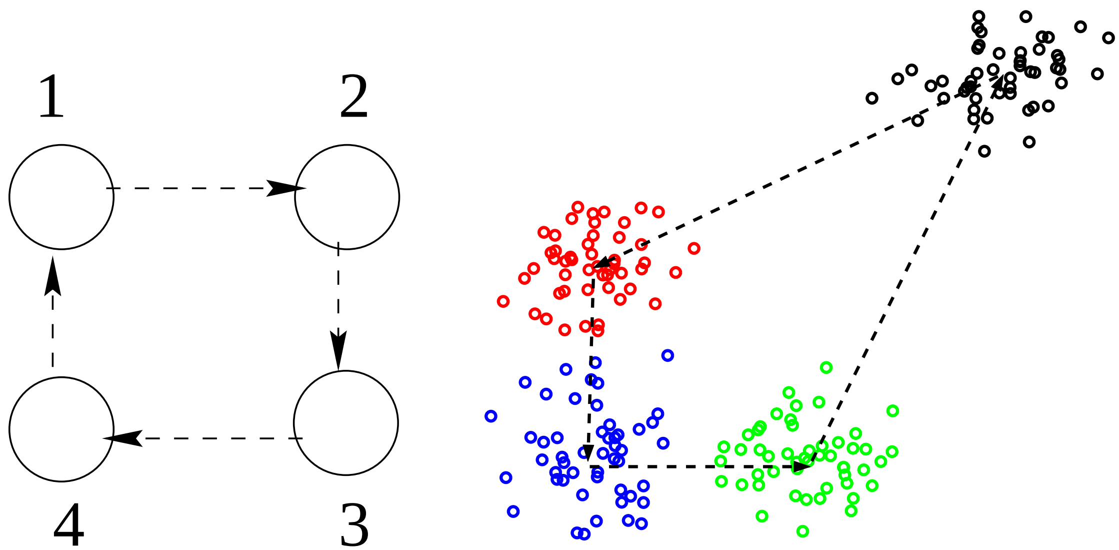
$$P_o(x=\text{heads}|s=2) = 0.75, \;\; P_o(x=\text{tails}|s=2) = 0.25$$



- What type of output sequences do we get from this HMM model?

HMM example

- Continuous output model: $\mathbf{x} = [x_1, x_2]$, $P_o(\mathbf{x}|s)$ is a Gaussian with mean and covariance depending on the underlying state s. Each state is initially equally likely.



- How does this compare to a mixture of four Gaussians model?

HMM problems

- There are several problems we have to solve
  1. How do we evaluate the probability that our model generated the observation sequence $\{\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_n\}$?
     - forward-backward algorithm
  2. How do we uncover the most likely hidden state sequence corresponding to these observations?
     - dynamic programming
  3. How do we adapt the parameters of the HMM to better account for the observations?
     - the EM-algorithm

Probability of observed data

- In principle computing the probability of the observed sequence involves summing over exponentially many possible hidden state sequences

$$P(\mathbf{x}_0, \ldots, \mathbf{x}_n) =$$

$$\sum_{s_0,\ldots,s_n} \overbrace{P_0(s_0)P_1(\mathbf{x}_0|s_0)\ldots P_1(s_n|s_{n-1})P_o(\mathbf{x}_n|s_n)}^{\text{Prob. of obs. and a hidden state sequence}}$$

$$\begin{array}{ccccccccc}
s_0 & \rightarrow & s_1 & \rightarrow & s_2 & \rightarrow \ldots \rightarrow & s_{n-1} & \rightarrow & s_n \\
\downarrow & & \downarrow & & \downarrow & & \downarrow & & \downarrow \\
\mathbf{x}_0 & & \mathbf{x}_1 & & \mathbf{x}_2 & & \mathbf{x}_{n-1} & & \mathbf{x}_n
\end{array}$$

- We can, however, exploit the structure of the model to evaluate the probability much more efficiently

Forward-backward algorithm

$$\begin{array}{ccccccccc} s_0 & \longrightarrow & s_1 & \longrightarrow & s_2 & \longrightarrow \ldots \longrightarrow & s_{n-1} & \longrightarrow & s_n \\ \downarrow & & \downarrow & & \downarrow & & \downarrow & & \downarrow \\ \mathbf{x}_0 & & \mathbf{x}_1 & & \mathbf{x}_2 & & \mathbf{x}_{n-1} & & \mathbf{x}_n \end{array}$$

- Forward (predictive) probabilities $\alpha_t(i)$:

$$\begin{aligned} \alpha_t(i) &= P(\mathbf{x}_0, \ldots, \mathbf{x}_t, s_t = i) \\ \frac{\alpha_t(i)}{\sum_j \alpha_t(j)} &= P(s_t = i | \mathbf{x}_0, \ldots, \mathbf{x}_t) \end{aligned}$$

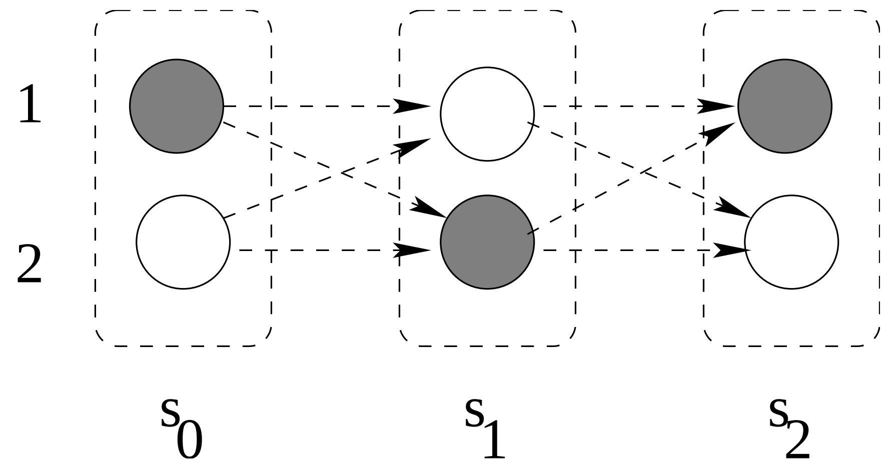
- Backward propabilities $\beta_t(i)$:

$$\beta_t(i) = P(\mathbf{x}_{t+1}, \ldots, \mathbf{x}_n | s_t = i)$$

(evidence about the current state from future observations)

- Both can be updated *recursively*

Recursive forward updates

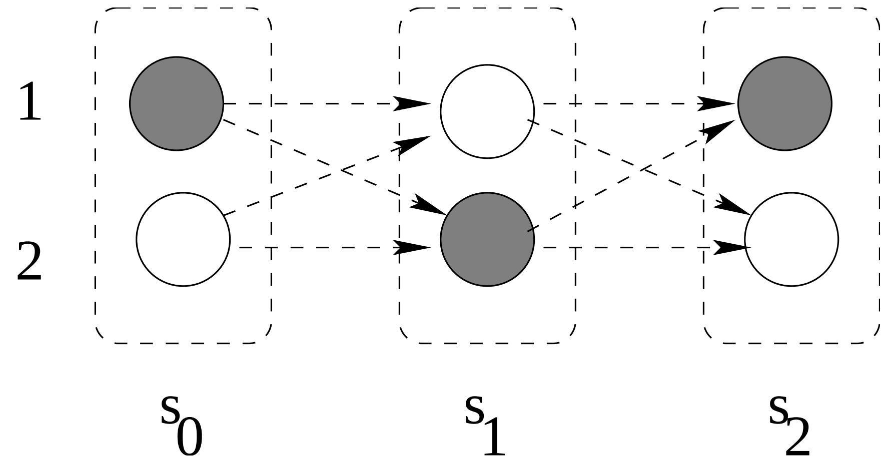


$\mathbf{x}_0 = heads$, $\mathbf{x}_1 = tails$, $\mathbf{x}_2 = heads$

- Forward recursion: $\alpha_t(i) = P(\mathbf{x}_0, \ldots, \mathbf{x}_t, s_t = i)$

$$\begin{aligned}
\alpha_0(1) &= P_0(1)\, P_o(heads|1) \\
\alpha_0(2) &= P_0(2)\, P_o(heads|2) \\
\alpha_1(1) &= \left[\alpha_0(1)P_1(1|1) + \alpha_0(2)P_1(1|2)\right] P_o(tails|1) \\
\alpha_1(2) &= \left[\alpha_0(1)P_1(2|1) + \alpha_0(2)P_1(2|2)\right] P_o(tails|2)
\end{aligned}$$

Recursive forward updates cont'd



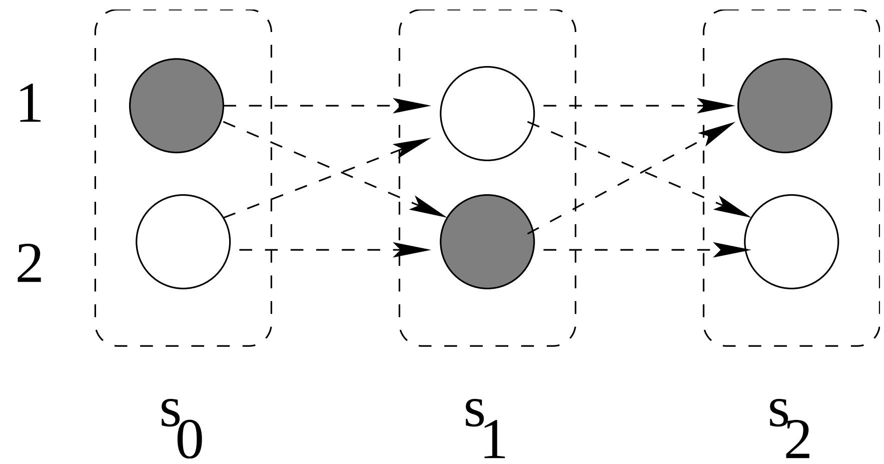
$$\mathbf{x}_0 = heads,\ \mathbf{x}_1 = tails,\ \mathbf{x}_2 = heads$$

- Recursive updates for $\alpha_t(i) = P(\mathbf{x}_0, \ldots, \mathbf{x}_t, s_t = i)$:

$$\alpha_0(i) = P_0(s_0 = i)\, P_o(\mathbf{x}_0 | s_0 = i)$$

$$\alpha_t(i) = \Big[\sum_j \alpha_{t-1}(j)\, P_1(s_t = i | s_{t-1} = j)\Big]\, P_o(\mathbf{x}_t | s_t = i)$$

Recursive backward updates

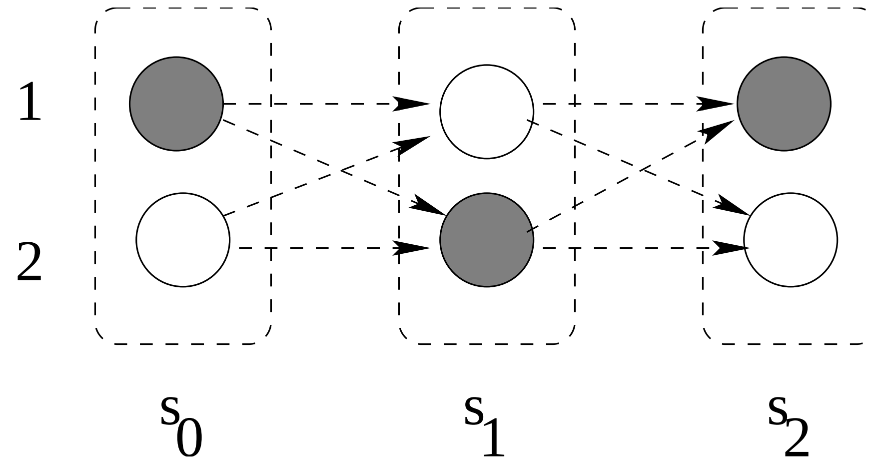


$$\mathbf{x}_0 = heads,\ \mathbf{x}_1 = tails,\ \mathbf{x}_2 = heads$$

- Backward recursion: $\beta_t(i) = P(\mathbf{x}_{t+1}, \ldots, \mathbf{x}_n | s_t = i)$

$$\begin{aligned}
\beta_2(1) &= 1 \\
\beta_2(2) &= 1 \\
\beta_1(1) &= P_1(1|1)P_o(heads|1)\beta_2(1) + P_1(2|1)P_o(heads|2)\beta_2(2) \\
\beta_1(2) &= P_1(1|2)P_o(heads|1)\beta_2(1) + P_1(2|2)P_o(heads|2)\beta_2(2)
\end{aligned}$$

Recursive backward updates cont'd



$\mathbf{x}_0 = heads,\ \mathbf{x}_1 = tails,\ \mathbf{x}_2 = heads$

- Recursive updates for $\beta_t(i) = P(\mathbf{x}_{t+1}, \ldots, \mathbf{x}_n | s_t = i)$:

$$\begin{aligned} \beta_n(i) &= 1 \\ \beta_{t-1}(i) &= \sum_j P_1(s_t = j | s_{t-1} = i) P_o(\mathbf{x}_t | s_t = j)\, \beta_t(j) \end{aligned}$$