CSE 150A-250A AI: Probabilistic Models

Lecture 12

Fall 2025

Trevor Bonjour
Department of Computer Science and Engineering
University of California, San Diego

Slides adapted from previous versions of the course (Prof. Lawrence, Prof. Alvarado, Prof Berg-Kirkpatrick)

Agenda

Review

Viterbi algorithm

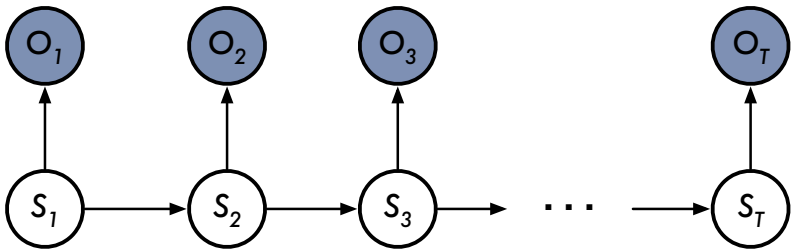
Learning in HMMs

Backward Algorithm

Review

Hidden Markov models

- Belief network



observations $o_t \in \{1, 2, \ldots, m\}$

states $s_t \in \{1, 2, \ldots, n\}$

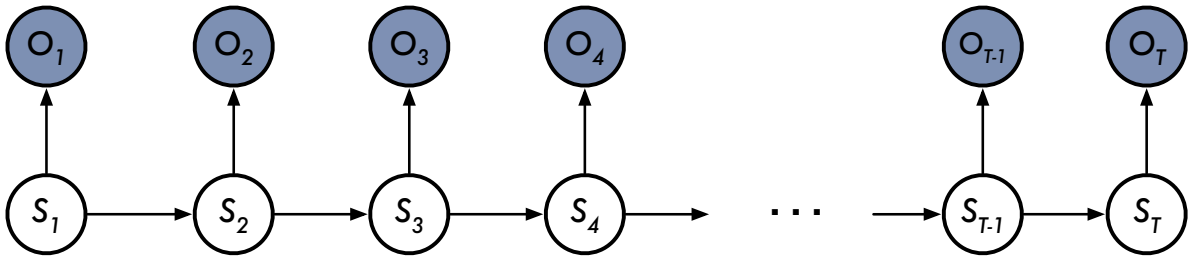
- Parameters

$$
\begin{aligned}
a_{ij} &= P(S_{t+1} = j | S_t = i) && \text{transition matrix} \\
b_{ik} &= P(O_t = k | S_t = i) && \text{emission matrix} \\
\pi_i &= P(S_1 = i) && \text{initial state distribution}
\end{aligned}
$$

- Notation

  Sometimes we'll write $b_i(k) = b_{ik}$ to avoid double subscripts.

Key computations in HMMs



Inference

1. How to compute the likelihood $P(o_1, o_2, \ldots, o_T)$?
2. How to compute the most likely hidden states $\textbf{argmax}_{\vec{s}} P(\vec{s}|\vec{o})$?
3. How to update beliefs by computing $P(s_t|o_1, o_2, \ldots, o_t)$?

Learning

How to estimate parameters $\{\pi_i, a_{ij}, b_{ik}\}$ that maximize the log-likelihood of observed sequences?

Forward Algorithm

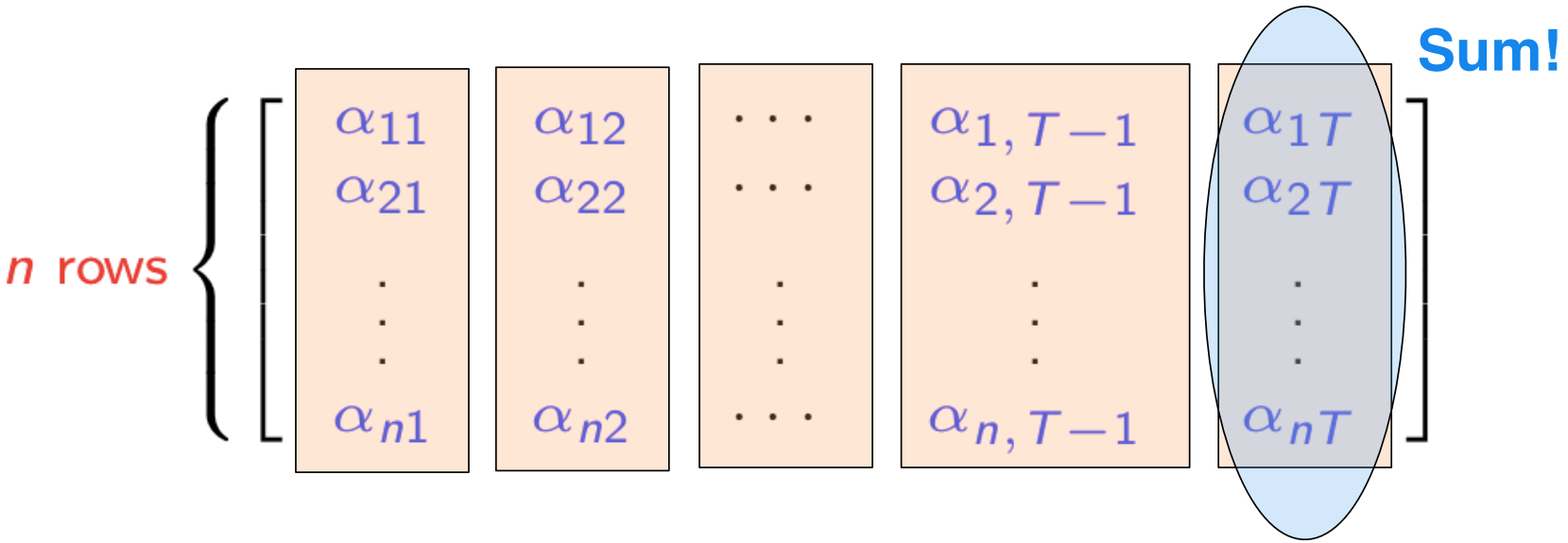
For a particular sequence of observations $\{o_1, o_2, \ldots, o_T\}$, define the matrix with elements:

$$\alpha_{it} = P(o_1, o_2, \ldots, o_t, S_t = i) \quad \begin{matrix} \\ n \text{ rows} \end{matrix} \left\{ \begin{bmatrix} \alpha_{11} & \alpha_{12} & \cdots & \alpha_{1,T-1} & \alpha_{1T} \\ \alpha_{21} & \alpha_{22} & \cdots & \alpha_{2,T-1} & \alpha_{2T} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \alpha_{n1} & \alpha_{n2} & \cdots & \alpha_{n,T-1} & \alpha_{nT} \end{bmatrix} \right.$$

The forward algorithm fills in the matrix of α_{it} elements one column at a time:

$$\begin{aligned} \alpha_{i1} &= \pi_i \, b_i(o_1) \\ \alpha_{j,t+1} &= \sum_{i=1}^{n} \alpha_{it} \, a_{ij} \, b_j(o_{t+1}) \end{aligned}$$

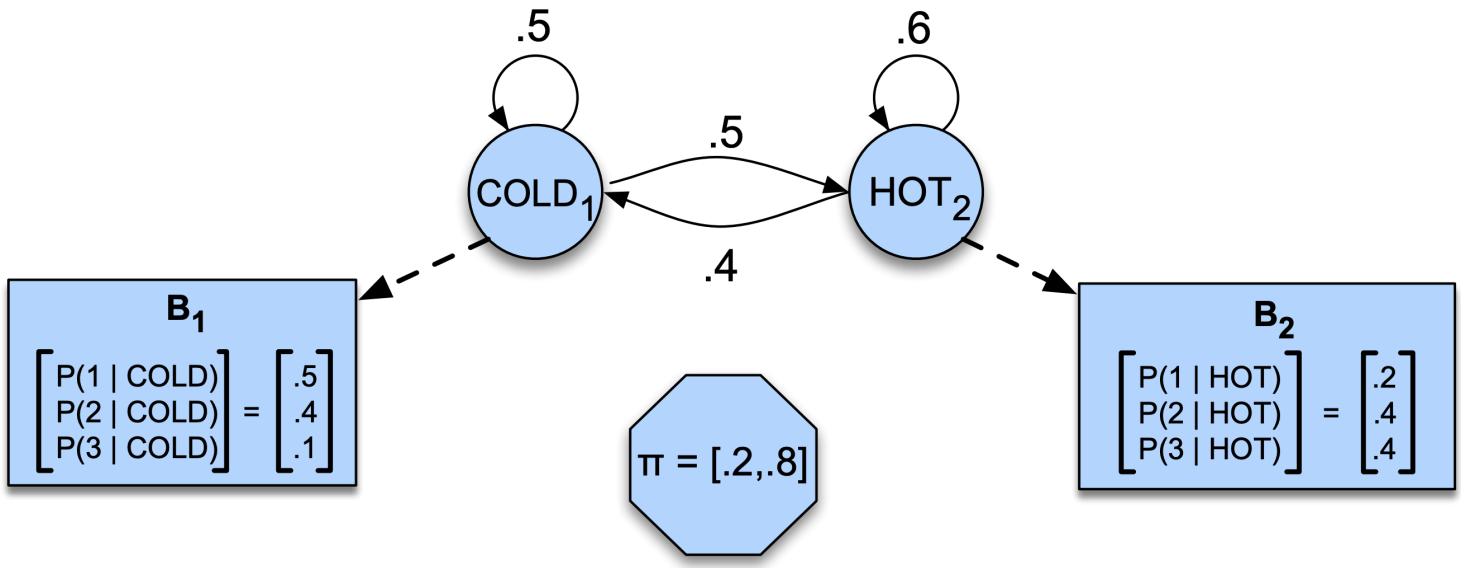
Computing the likelihood $P(o_1, o_2, \ldots, o_T)$



$$
\begin{aligned}
& P(o_1, o_2, \ldots, o_T) \\
&= \sum_{i=1}^{n} P(o_1, o_2, \ldots, o_T, s_T = i) && \text{marginalization} \\
&= \sum_{i=1}^{n} \alpha_{iT} && \text{sum of last column}
\end{aligned}
$$

Example[1]

- Two hidden states (Weather): $\{H, C\}$
- Observations (Ice creams): $\{1, 2, 3\}$

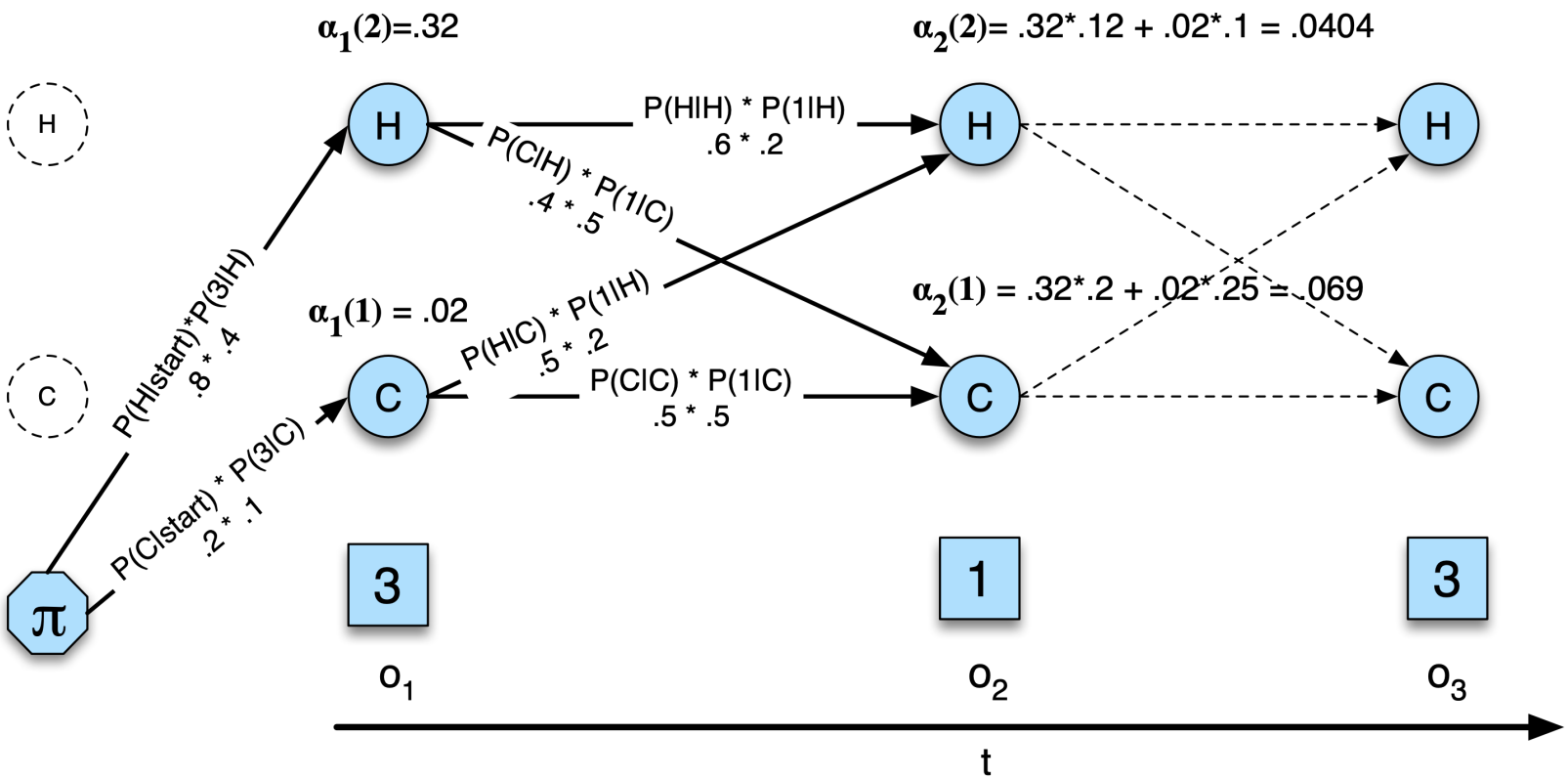


[1] Eisner, J. 2002. An interactive spreadsheet for teaching the forward-backward algorithm.

Example - Forward Algorithm

$$\alpha_{i1} = \pi_i \, b_i(o_1)$$

$$\alpha_{j,t+1} = \sum_{i=1}^{n} \alpha_{it} \, a_{ij} \, b_j(o_{t+1})$$



Viterbi algorithm

The most likely sequence of hidden states

$$\{s_1^*, s_2^*, \ldots, s_T^*\}$$

$$= \text{argmax}_{s_1, s_2, \ldots, s_T} P(s_1, s_2, \ldots, s_T | o_1, o_2, \ldots, o_T)$$

$$= \text{argmax}_{s_1, s_2, \ldots, s_T} \left[\frac{P(s_1, s_2, \ldots, s_T, o_1, o_2, \ldots, o_T)}{P(o_1, o_2, \ldots, o_T)} \right]$$

product rule

$$= \text{argmax}_{s_1, s_2, \ldots, s_T} P(s_1, s_2, \ldots, s_T, o_1, o_2, \ldots, o_T)$$

no states in denominator

$$= \text{argmax}_{s_1, s_2, \ldots, s_T} \log P(s_1, s_2, \ldots, s_T, o_1, o_2, \ldots, o_T)$$

log is monotonic

The matrix ℓ^*

- **Definition**

  For a particular sequence of observations $\{o_1, o_2, \ldots, o_T\}$, we define the following matrix:

$$\ell^*_{it} = \max_{s_1, s_2, \ldots, s_{t-1}} \log P(s_1, s_2, \ldots, s_{t-1}, S_t = i, o_1, o_2, \ldots, o_t)$$

$$n \text{ rows} \left\{ \begin{bmatrix} \ell^*_{11} & \ell^*_{12} & \cdots & \ell^*_{1,T-1} & \ell^*_{1T} \\ \ell^*_{21} & \ell^*_{22} & \cdots & \ell^*_{2,T-1} & \ell^*_{2T} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \ell^*_{n1} & \ell^*_{n2} & \cdots & \ell^*_{n,T-1} & \ell^*_{nT} \end{bmatrix} \right.$$

- **Intuition**

ℓ^*_{it}	log-probability of the *t-step path* of hidden states $s_1, s_2, \ldots, s_t$ that best explains the observations $o_1, o_2, \ldots, o_t$ and ends at state $S_t = i$ at time t

Computing the matrix ℓ^*

$$\ell^*_{it} = \max_{s_1, s_2, \ldots, s_{t-1}} \log P(s_1, s_2, \ldots, s_{t-1}, S_t = i, o_1, o_2, \ldots, o_t)$$

$$\overset{n \text{ rows}}{\left\{ \left[\begin{array}{|c|cccc} \ell^*_{11} & \ell^*_{12} & \cdots & \ell^*_{1,T-1} & \ell^*_{1T} \\ \ell^*_{21} & \ell^*_{22} & \cdots & \ell^*_{2,T-1} & \ell^*_{2T} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \ell^*_{n1} & \ell^*_{n2} & \cdots & \ell^*_{n,T-1} & \ell^*_{nT} \end{array} \right] \right.}$$

- First column ($t = 1$)

$$\begin{aligned} \ell^*_{i1} &= \log P(S_1 = i, o_1) \\ &= \log \Big[P(S_1 = i)\, P(o_1 | S_1 = i) \Big] && \text{product rule} \\ &= \log \pi_i + \log b_i(o_1) && \text{CPTs} \end{aligned}$$

Computing the matrix ℓ^*

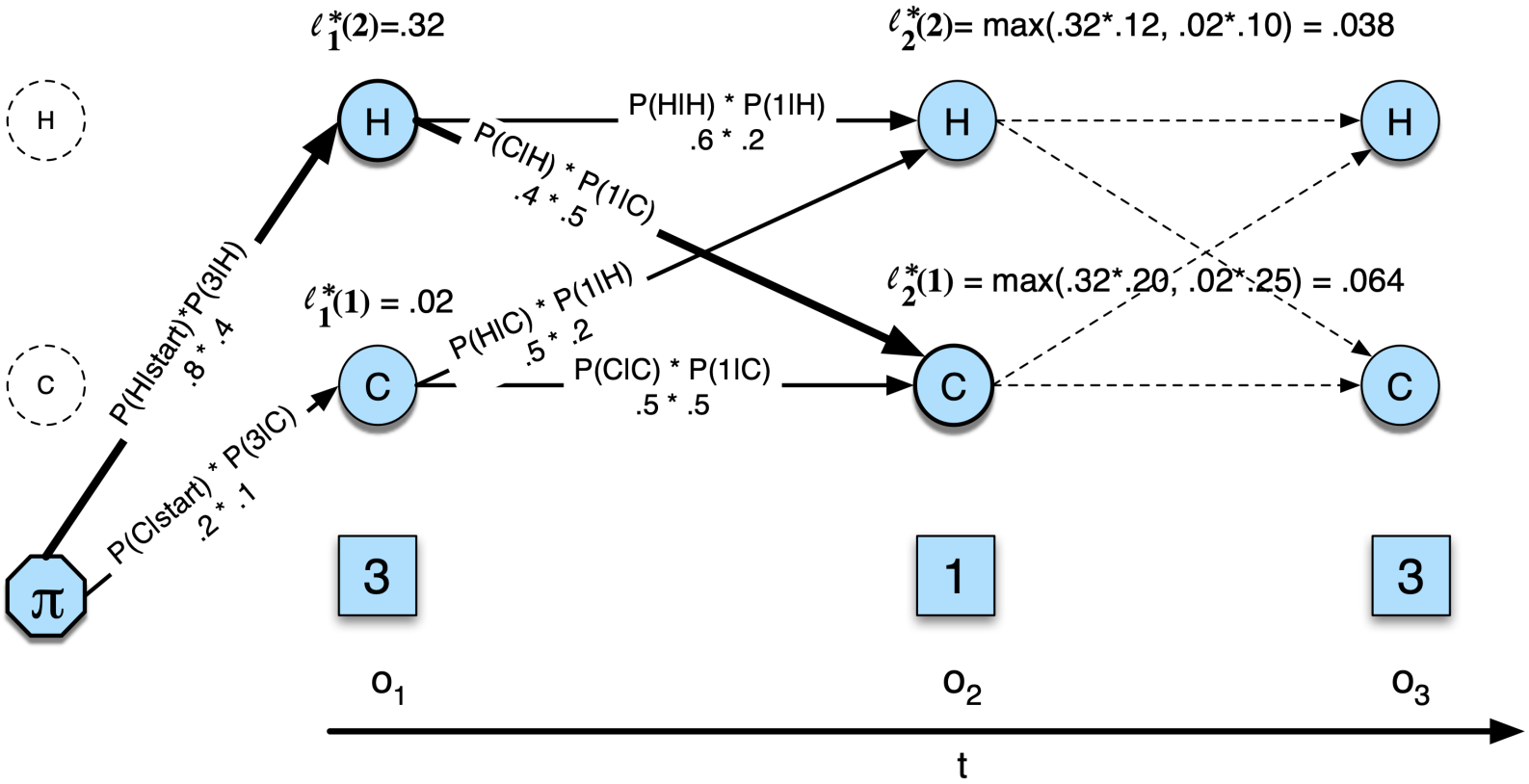
- Next columns ($t > 1$)

$$
\begin{aligned}
\ell^*_{j,t+1} &= \max_{s_1,\ldots,s_t} \log P(s_1,\ldots,s_t, S_{t+1}=j, o_1,\ldots,o_{t+1}) \\
&= \max_i \max_{s_1,\ldots,s_{t-1}} \log P(s_1,\ldots,s_{t-1}, S_t=i, S_{t+1}=j, o_1,\ldots,o_{t+1}) \\
&= \max_i \max_{s_1,\ldots,s_{t-1}} \log \Big[P(s_1,\ldots,s_{t-1}, S_t=i, o_1,\ldots,o_t) \cdot \quad \boxed{\text{product rule}} \\
&\qquad\qquad P(S_{t+1}=j | s_1,\ldots,s_{t-1}, S_t=i, o_1,\ldots,o_t) \cdot \\
&\qquad\qquad P(o_{t+1} | s_1,\ldots,s_{t-1}, S_t=i, S_{t+1}=j, o_1,\ldots,o_t) \Big] \\
&= \max_i \max_{s_1,\ldots,s_{t-1}} \log \Big[P(s_1,\ldots,s_{t-1}, S_t=i, o_1,\ldots,o_t) \cdot \\
&\qquad\qquad P(S_{t+1}=j | S_t=i) \cdot P(o_{t+1} | S_{t+1}=j) \Big] \quad \boxed{\text{CI}} \\
&= \max_i \max_{s_1,\ldots,s_{t-1}} \Big[\log P(s_1,\ldots,s_{t-1}, S_t=i, o_1,\ldots,o_t) + \log a_{ij} + \log b_j(o_{t+1}) \Big] \\
&= \max_i \Big[\ell^*_{it} + \log a_{ij} \Big] + \log b_j(o_{t+1})
\end{aligned}
$$

Summary

We have shown how to efficiently compute the matrix ℓ^* one column at a time, from left to right:

Example - Viterbi (Fill ℓ^*)



Computing $\{s_1^*, s_2^*, \ldots, s_T^*\}$

- Form one more matrix:

$$\Phi_{t+1}(j) \;=\; \arg\max_i \left[\ell_{it}^* + \log a_{ij}\right]$$

Intuitively, given the observations $o_1, o_2, \ldots, o_{t+1}$, record the most likely state at time t given that $S_{t+1}=j$.

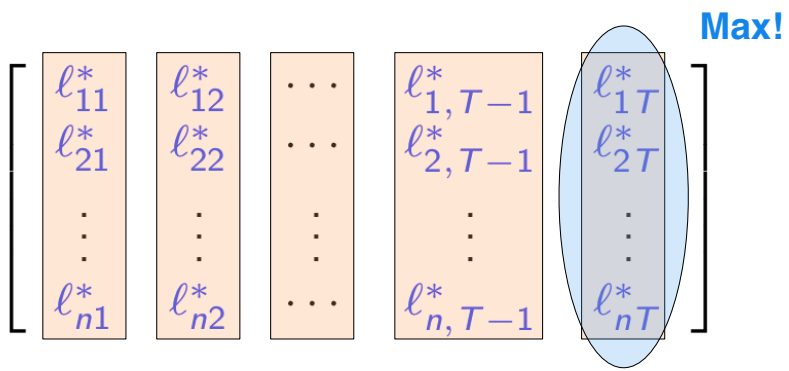
- Compute the most likely states by backtracking:

$$s_T^* \;=\; \arg\max_i \left[\ell_{iT}^*\right]$$

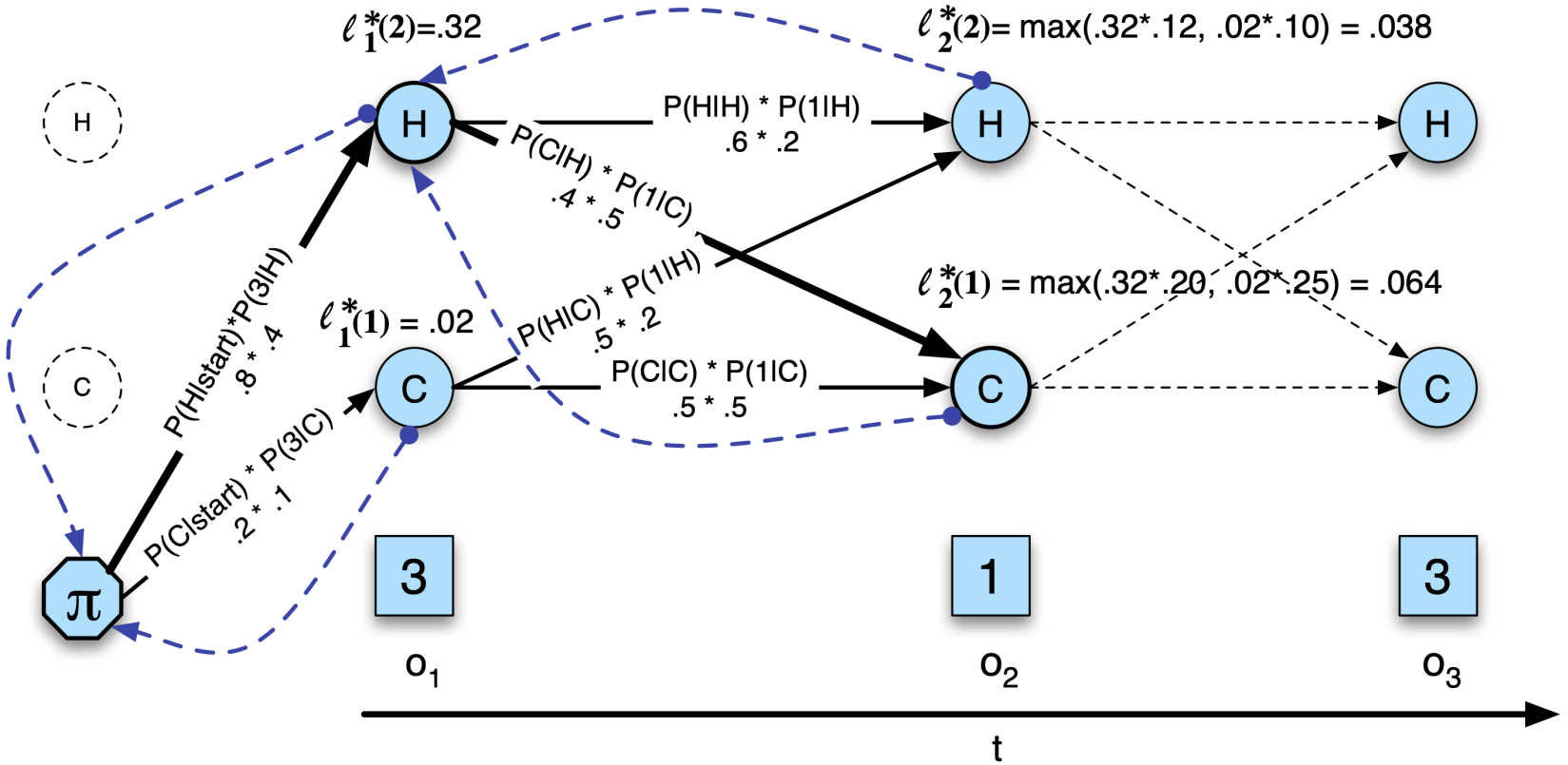
for $t = T-1$ to 1

$$s_t^* \;=\; \Phi_{t+1}(s_{t+1}^*)$$

end



Example - Viterbi (Backtrack through ℓ^*)



Summary of Viterbi algorithm

- Fill ℓ^* matrix from left to right:

$$\boxed{t = 1} \qquad \ell^*_{i1} = \log \pi_i + \log b_i(o_1)$$

$$\boxed{t > 1} \qquad \ell^*_{j,t+1} = \max_i \left[\ell^*_{it} + \log a_{ij} \right] + \log b_j(o_{t+1})$$

- Backtrack through ℓ^* from right to left:

$$\boxed{t = T} \qquad s^*_T = \operatorname{argmax}_i \left[\ell^*_{iT} \right]$$

$$\boxed{t < T} \qquad s^*_t = \operatorname{argmax}_i \left[\ell^*_{it} + \log a_{is^*_{t+1}} \right]$$

- Where you've seen this before:

  This algorithm is an instance of dynamic programming. Sometimes $\{s^*_1, s^*_2, \ldots, s^*_T\}$ is called the Viterbi path.

Check in

Which of the following statements is true(in terms of asymptotic time complexity)?

A. The Viterbi algorithm generally runs faster than the Forward algorithm because it finds a single most likely sequence.

B. The Forward algorithm generally runs faster than the Viterbi algorithm because it computes probabilities for all possible sequences.

C. Both algorithms have similar running times.

D. The Viterbi algorithm runs slower than the Forward algorithm due to its additional backtracking step.

Check in

What is the running time for Viterbi algorithm?

A. $O(n)$

B. $O(n^2)$

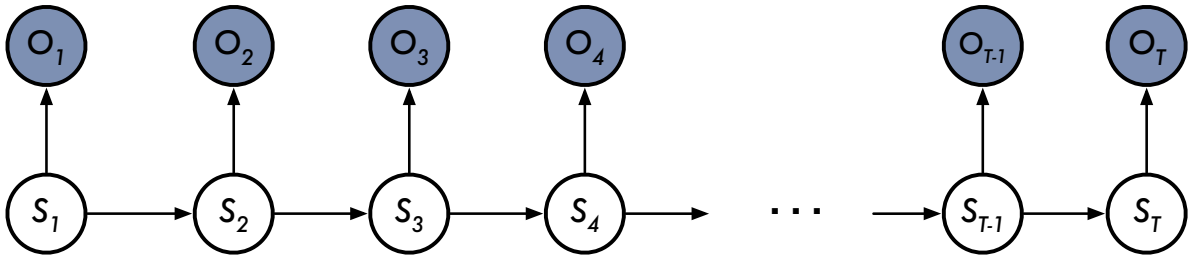
C. $O(Tn^2)$

D. $O(T^2n^4)$

E. $O(n^T)$

Learning in HMMs

Learning in HMMs



Given: one or more sequences of observations $\{o_1, o_2, \ldots, o_T\}$.
For simplicity, we'll assume just one.

Goal: estimate $\{\pi_i, a_{ij}, b_{ik}\}$ to maximize $P(o_1, o_2, \ldots, o_T)$,
the likelihood of the observed data.

Assume: the cardinality n of the hidden state space is fixed.
$s_t \in \{1, 2, \ldots, n\}$

How do we estimate $\{\pi_i, a_{ij}, b_{ik}\}$?

EM Algorithm!
How EM works in general:

To re-estimate $P(X_i = x | \mathrm{pa}_i = \pi)$ in the M-step,
we must compute $P(X_i = x, \mathrm{pa}_i = \pi | V)$ in the E-step.

EM algorithm for HMMs

- CPTs to re-estimate:

$$\begin{aligned} \pi_i &= P(S_1 = i) \\ a_{ij} &= P(S_{t+1} = j | S_t = i) \\ b_{ik} &= P(O_t = k | S_t = i) \end{aligned}$$

- E-step in HMMs must compute:

$$P(S_1 = i | o_1, o_2, \ldots, o_T)$$

$$P(S_{t+1} = j, S_t = i | o_1, o_2, \ldots, o_T)$$

$$P(O_t = k, S_t = i | o_1, o_2, \ldots, o_T) = I(o_t, k)\, P(S_t = i | o_1, o_2, \ldots, o_T)$$

$P(S_1 = i | o_1, o_2, \ldots, o_T)$: special case of below $(t = 1)$

How to efficiently compute these posteriors?

Computing $P(S_t = i | o_1, \ldots, o_T)$

$$P(S_t = i | o_1, \ldots, o_T) = \frac{P(S_t = i, o_1, o_2, \ldots, o_T)}{P(o_1, o_2, \ldots, o_T)}$$

product rule

- Numerator

$$\begin{aligned} & P(S_t = i, o_1, o_2, \ldots, o_T) \\ &= P(o_1, \ldots, o_t, S_t = i)\, P(o_{t+1}, \ldots, o_T | S_t = i, o_1, \ldots, o_t) \quad \text{product rule} \\ &= P(o_1, \ldots, o_t, S_t = i)\, P(o_{t+1}, \ldots, o_T | S_t = i) \quad \text{conditional independence} \\ &= \alpha_{it}\, P(o_{t+1}, \ldots, o_T | S_t = i) \end{aligned}$$

Backward Algorithm

We need one more matrix ...

$$\begin{aligned} \text{Analogous to } \alpha_{it} &= P(o_1, o_2, \ldots, o_t, S_t = i), \\ \text{define } \beta_{it} &= P(o_{t+1}, o_{t+2}, \ldots, o_T | S_t = i). \end{aligned}$$

$$n \text{ rows} \left\{ \begin{bmatrix} \beta_{11} & \beta_{12} & \cdots & \beta_{1,T-1} & \beta_{1T} \\ \beta_{21} & \beta_{22} & \cdots & \beta_{2,T-1} & \beta_{2T} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \beta_{n1} & \beta_{n2} & \cdots & \beta_{n,T-1} & \beta_{nT} \end{bmatrix} \right.$$

Understand the **differences** between these matrices:

- α_{it} predicts observations up to and including time t.
- β_{it} predicts observations from **time** $t+1$ to time T.

Computing $\beta_{it} = P(o_{t+1}, o_{t+2}, \ldots, o_T | S_t = i)$

- **Last column ($t = T$)**

$$\beta_{iT} = P(\underline{\quad} | S_T = i)$$ What does this mean?

Note: β_{it} computes the probability of the future given $S_t = i$.

But we don't see *any* observations beyond time T.
Put another way, the future after time T is unspecified.

What is the probability of some unspecified future occurring?
By definition, we set:

$$\boxed{\beta_{iT} = 1 \quad \textbf{for all} \quad i \in \{1, 2, \ldots, n\}}$$

Computing $\beta_{it} = P(o_{t+1}, o_{t+2}, \ldots, o_T | S_t = i)$

- Previous columns ($t < T$)

$$
\begin{aligned}
\beta_{it} &= P(o_{t+1}, o_{t+2}, \ldots, o_T | S_t = i) \\
&= \sum_{j=1}^{n} P(S_{t+1} = j, o_{t+1}, o_{t+2}, \ldots, o_T | S_t = i) && \text{marginalization} \\
&= \sum_{j=1}^{n} \Big[P(S_{t+1} = j | S_t = i) \cdot \\
&\qquad P(o_{t+1} | S_t = i, S_{t+1} = j) \cdot \\
&\qquad P(o_{t+2}, \ldots, o_T | S_t = i, S_{t+1} = j, o_{t+1}) \Big] && \text{product rule} \\
&= \sum_{j=1}^{n} \Big[P(S_{t+1} = j | S_t = i)\, P(o_{t+1} | S_{t+1} = j)\, P(o_{t+2}, \ldots, o_T | S_{t+1} = j) \Big] && \text{CI} \\
&= \sum_{j=1}^{n} a_{ij}\, b_j(o_{t+1})\, \beta_{j,t+1} && \text{CPTs}
\end{aligned}
$$

Backward algorithm

The backward algorithm fills in the matrix of β_{it} elements one column at a time:

Computing $P(S_t = i | o_1, \ldots, o_T)$

$$P(S_t = i | o_1, \ldots, o_T) = \frac{P(S_t = i, o_1, o_2, \ldots, o_T)}{P(o_1, o_2, \ldots, o_T)}$$ product rule

- Numerator

$$\begin{aligned}
&P(S_t = i, o_1, o_2, \ldots, o_T) \\
&\quad = P(o_1, \ldots, o_t, S_t = i)\, P(o_{t+1}, \ldots, o_T | S_t = i, o_1, \ldots, o_t) && \text{product rule} \\
&\quad = P(o_1, \ldots, o_t, S_t = i)\, P(o_{t+1}, \ldots, o_T | S_t = i) && \text{conditional independence} \\
&\quad = \alpha_{it}\, P(o_{t+1}, \ldots, o_T | S_t = i) \\
&\quad = \alpha_{it}\, \beta_{it}
\end{aligned}$$

Next lecture: EM with the forward-backward algorithm.

That’s all folks!