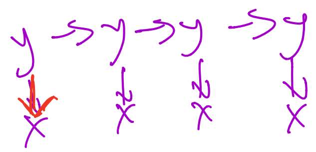
UMass CS 685, Spring 2021
HMM example from the Jurafsky and Martin textbook (Jason Eisner's ice cream example)



Model

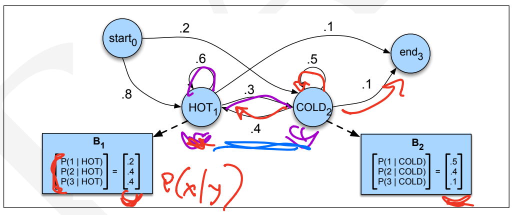


Figure 7.3 A hidden Markov model for relating numbers of ice creams eaten by Jason (the observations) to the weather (H or C, the hidden variables).

Forward algorithm $= P(x)$

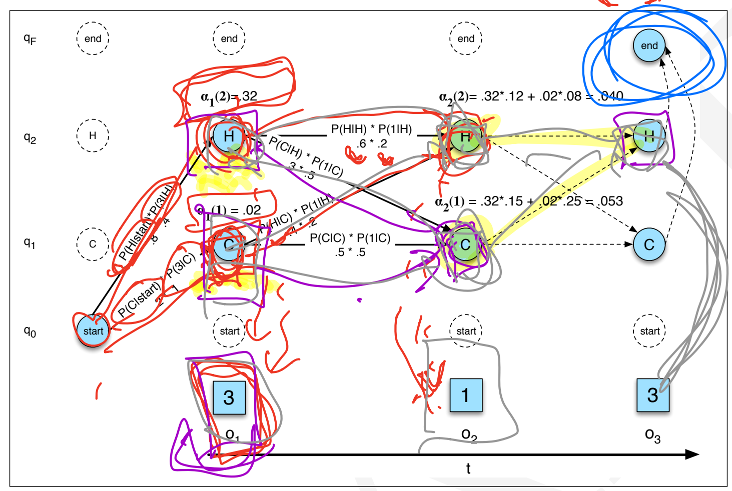


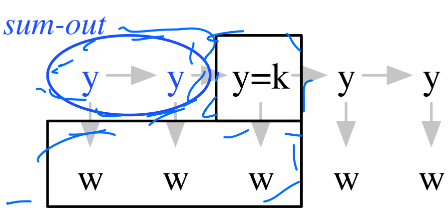
Figure 7.7 The forward trellis for computing the total observation likelihood for the ice-cream events *3 1 3*. Hidden states are in circles, observations in squares. White (unfilled) circles indicate illegal transitions. The figure shows the computation of $\alpha_t(j)$ for two states at two time steps. The computation in each cell follows Eq. 7.14: $\alpha_t(j) = \sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij} b_j(o_t)$. The resulting probability expressed in each cell is Eq. 7.13: $\alpha_t(j) = P(o_1, o_2 \ldots o_t, q_t = j|\lambda)$.

Forward-Backward

Declaratively:

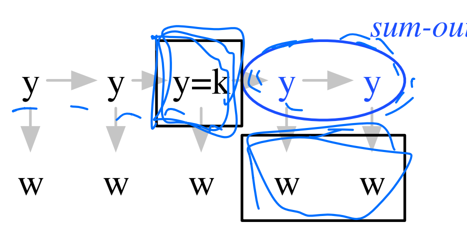
Forward probs

$$\alpha_t[k] = \sum_{y_1 \ldots y_{t-1}} P(y_t = k, w_1..w_t, y_1..y_{t-1})$$



Backward probs

$$\beta_t[k] = \sum_{y_{t+1} \ldots y_n} P(y_t = k, w_{t+1}..w_n, y_{t+1}..y_n)$$



Forward Algo.: for each *t=1..N*, for each *k*,

$$\alpha_t[k] := \sum_{j=1..K} \left(\alpha_{t-1}[j] \; P_{trans}(k \mid j) \; P_{emit}(w_t \mid k) \right)$$

Backward Algo.: for each *t=N..1*, for each *j*,

$$\beta_t[j] := \sum_{k=1..K} \left(\beta_{t+1}[j] \; P_{trans}(k \mid j) \; P_{emit}(w_{t+1} \mid k) \right)$$

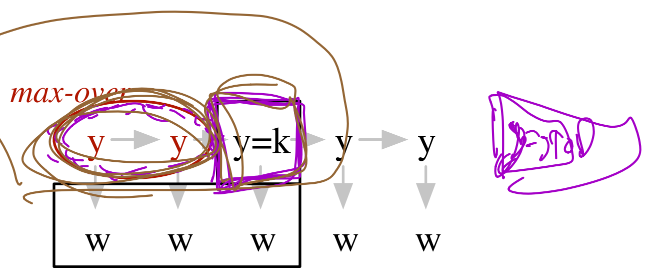
$\alpha_{N+1}[end] = P(\vec{w})$

Tag Marginals:

$$P(y_t = k \mid w_1..w_n) \propto \alpha_t[k] \; \beta_t[k]$$

$$P(y_{t-1} = j, y_t = k \mid w_1..w_n) \propto \alpha_t[j] \; P_{trans}(k \mid j) \; \beta_t[k]$$

TODO: Data?



Viterbi algorithm (for HMMs)

Declaratively:

$$V_t[k] = \max_{y_1 \dots y_{t-1}} P(y_t = k, \quad y_1..y_{t-1}, w_1..w_t)$$

for $\max\limits_{\vec{y}} P(\vec{w}, \vec{y})$

$\max\limits_{\vec{y}} \prod_t P(y_{t+1}|y_t)\, P(w_t|y_t)$

Algorithm, for each *t=1..N,*

$$V_t[k] := \max_{j=1..K} \Big(V_{t-1}[j]\ P_{trans}(k \mid j)\ P_{emit}(w_t \mid k) \Big)$$

$$B_t[k] := \arg\max_{j=1..K} \Big(\dots \Big)$$

For solution: choose best tag at last position.
Trace backpointers to find best tag at second-to-last, e tc.

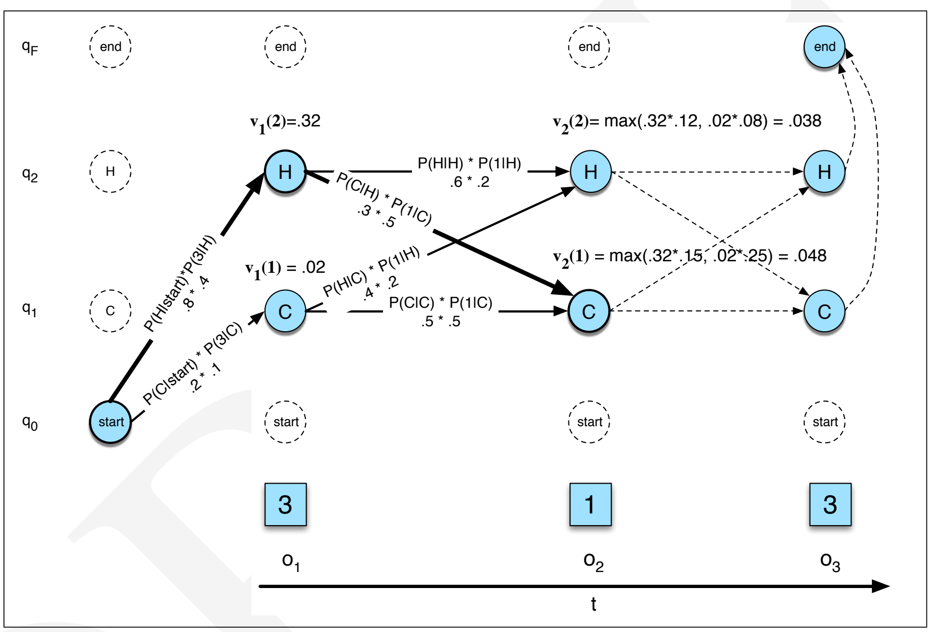
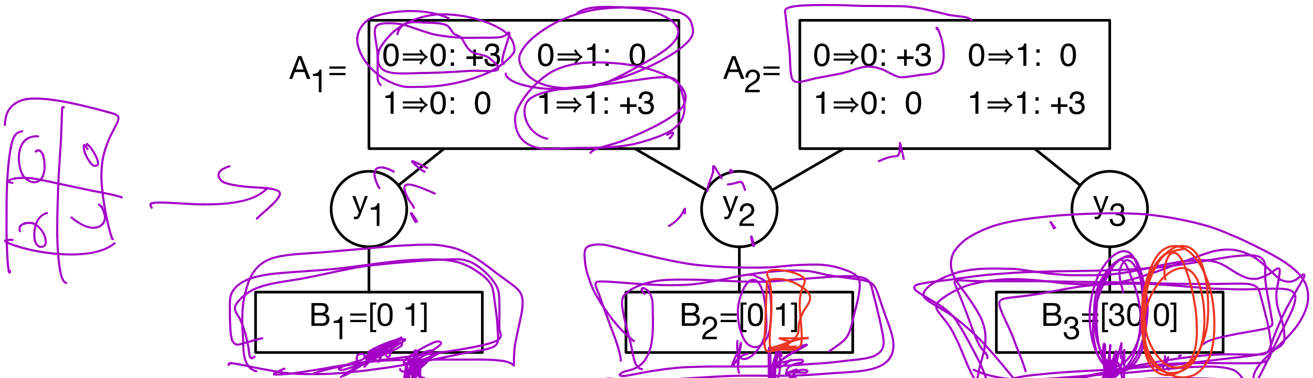


Figure 7.10 The Viterbi trellis for computing the best path through the hidden state space for the ice-cream eating events *3 1 3*. Hidden states are in circles, observations in squares. White (unfilled) circles indicate illegal transitions. The figure shows the computation of $v_t(j)$ for two states at two time steps. The computation in each cell follows Eq. 7.19: $v_t(j) = \max_{1\le i\le N-1} v_{t-1}(i)\, a_{ij}\, b_j(o_t)$. The resulting probability expressed in each cell is Eq. 7.18: $v_t(j) = P(q_0, q_1, \dots, q_{t-1}, o_1, o_2, \dots, o_t, q_t = j|\lambda)$.

CS690N Viterbi exercise



Sticky-favoring model over hidden state vocab {0,1}. Factor scores are "goodness points" are in log-scale additive form. (They're positive, though for an HMM they would all be negative.)

log P(y | w) = (constant) + G(y1, y2, y3)

G(y1, y2, y3) = A(y1,y2) + A(y2,y3) + B1(y1) + B2(y2) + B3(y3)

inference goal: argmax$_y$ G(y)

Additive Viterbi

For t=1..T,

For k in {0,1},

$$V_t[k] := \max_j \left(V_{t-1}[j] + A(j,k) + B_t(k)\right)$$

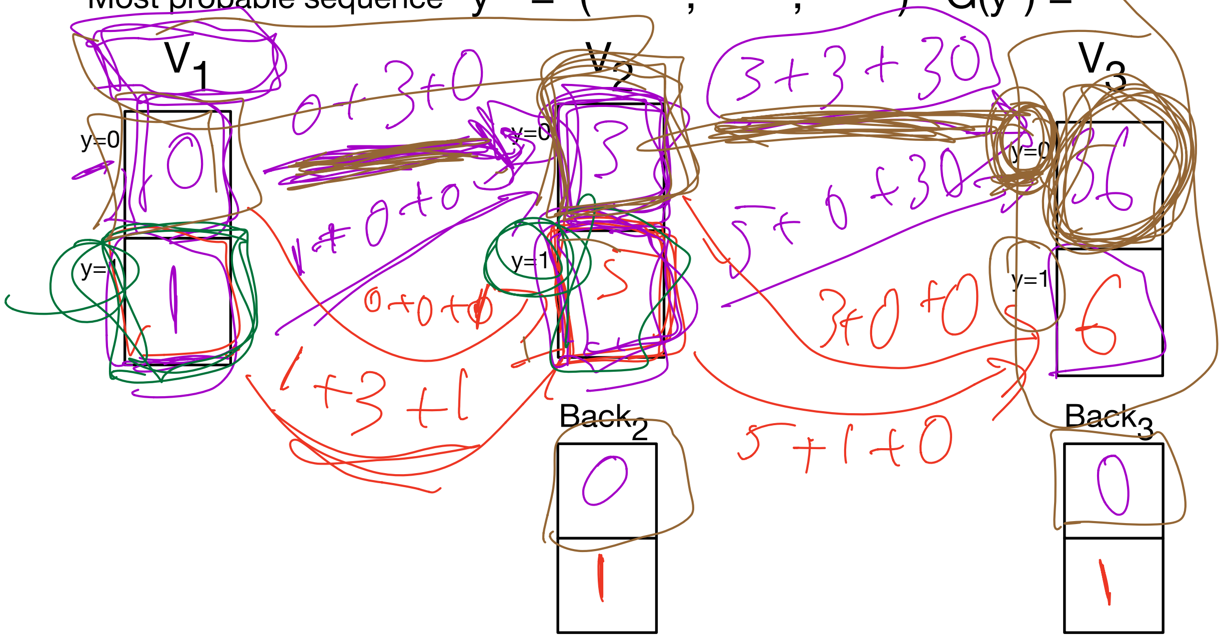
$$B[k] := \arg\max_j \left(\ldots\right)$$

For t=1, set A_0(anything)=0 and V_0[anything]=0

Final backtrace step: take best-scoring from last V_T, follow the backpointers all the way back

Run Viterbi and fill out the trellis with arcs like in the textbook's HMM example.

Most probable sequence $y^* = ($ 0 , 0 , 0 $)$ $G(y^*) =$



$$B\left[k\right] := \underset{j}{\arg\max}\left(\ldots\right)$$