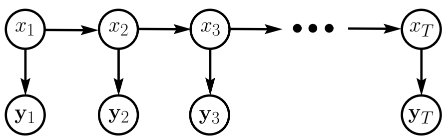
Lecture 17:

Inference for Profile HMMs

March 15, 2006

Reminder: HMM Graphical Model



- Hidden states $\{x_t\}$, outputs $\{\mathbf{y}_t\}$
  Joint probability factorizes:

$$\mathsf{P}(\{x\},\{\mathbf{y}\}) = \prod_{t=1}^{T} \mathsf{P}(x_t|x_{t-1})\mathsf{P}(\mathbf{y}_t|x_t)$$
$$= \pi_{x_1} \prod_{t=1}^{T-1} S_{x_t,x_{t+1}} \prod_{t=1}^{T} A_{x_t}(\mathbf{y}_t)$$

- We saw efficient recursions for computing
  $L = \mathsf{P}(\{\mathbf{y}\}) = \sum_{\{x\}} \mathsf{P}(\{x\},\{\mathbf{y}\})$ and $\gamma_i(t) = \mathsf{P}(x_t = i|\{\mathbf{y}\})$.

Forward-Backward ($\alpha\beta$) Inference Recursions

- Estimate the marginal over a single hidden state:

$$\gamma(x_t) = p(x_t|\{\mathbf{y}\} = \frac{\alpha(x_t)\beta(x_t)}{p(\mathbf{y}_1^T)}$$

$$\text{where} \quad \alpha_j(t) = p(\,\mathbf{y}_1^t\,,\ x_t = j\,)$$
$$\beta_j(t) = p(\mathbf{y}_{t+1}^T \mid x_t = j\,)$$
$$\gamma_i(t) = p(x_t = i \mid \mathbf{y}_1^T)$$

- There are simple recursions for $\alpha_j(t)$ and $\beta_j(t)$:

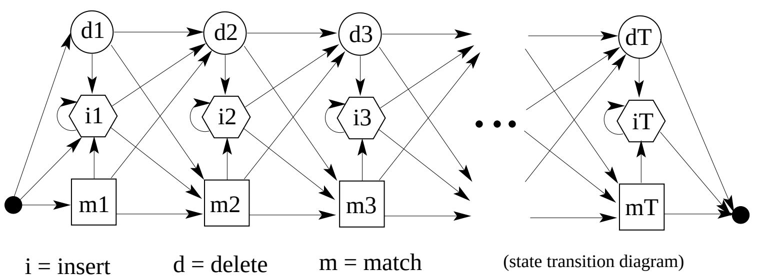
$$\alpha_k(t+1) = \{\sum_j \alpha_j(t) S_{jk}\} A_k(\mathbf{y}_{t+1}); \qquad \alpha_j(1) = \pi_j A_j(\mathbf{y}_1)$$
$$\beta_j(t) = \sum_i S_{ji}\beta_i(t+1)A_i(\mathbf{y}_{t+1}); \qquad \beta_j(T) = 1$$

- $\alpha_i(t)$ gives total *inflow* of prob. to node (t,i)
  $\beta_i(t)$ gives total *outflow* of prob.

Viterbi Decoding

- The numbers $\gamma_j(t)$ above gave the probability distribution over all states at any time.
- By choosing the state $\gamma_*(t)$ with the largest probability at each time, we can make a "best" state path. This is the path with the *maximum expected number of correct states*.
- But it *is not* the single path with the highest likelihood of generating the data. In fact it may be a path of prob. zero!
- To find the single best path, we do *Viterbi decoding* which is just Bellman's dynamic programming algorithm applied to this problem.
- The recursions look the same, except with $\max$ instead of $\sum$.
- Bugs once more: same trick except at each step kill all bugs but the one with the highest value at the node.

Profile (String-Edit) HMMs



- A "profile HMM" or "string-edit" HMM is used for probabilistically matching an observed input string to a stored template pattern with possible insertions and deletions.
- Three kinds of states: match, insert, delete.
  m_n – use position n in the template to match an observed symbol
  i_n – insert extra symbol(s) observations after template position n
  d_n – delete (skip) template position n

Forward-Backward for Profile HMMs

- The equations for the delete states in profile HMMs need to be modified slightly, since they don't emit any symbols.
- For delete states k, the forward equations become:

$$\alpha_k(t) = \sum_j \alpha_j(t) S_{jk}$$

  which should be evaluated after the insert and match state updates.
- For all states, the backward equations become:

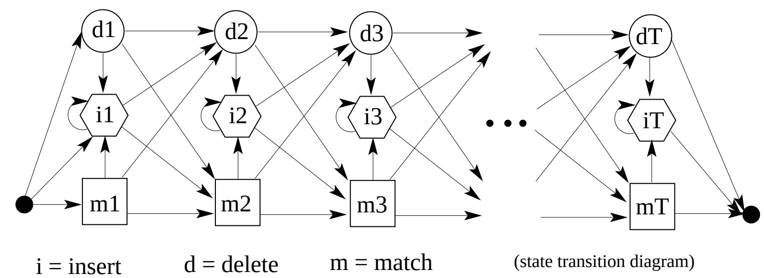
$$\beta_k(t) = \sum_{i \in \text{match,ins}} S_{ki}\beta_i(t+1)A_i(\mathbf{y}_{t+1}) + \sum_{j \in \text{del}} S_{kj}\beta_j(t)$$

  which should be evaluated first for delete states k; then for the rest.
- The gamma equations remain the same:

$$\gamma_i(t) = p(x_t = i \mid \mathbf{y}_1^T) = \alpha_i(t)\beta_i(t)/L$$

- Notice that each summation above contains only three terms, regardless of the total number of states!

Profile HMMs have Linear Costs



- number of states = 3(length_template)
- Only insert and match states can generate output symbols.
- Once you visit or skip a match state you can never return to it.
- At most 3 destination states from any state, so S_{ij} very sparse.
- Storage/Time cost *linear* in #states, not quadratic.
- State variables and observations no longer in sync.
  (e.g. y1:m1 ; d2 ; y2:i2 ; y3:i2 ; y4:m3 ; ...)

Initializing Forward-Backward for Profile HMMs

- The initialization equations for Profile HMMs also need to be fixed up, to reflect the fact that the model can only begin in states m_1, i_1, d_1 and can only finish in states m_N, i_N, d_N.
- In particular, $\pi_j = 0$ if j is not one of m_1, i_1, d_1.
- When initializing $\alpha_k(1)$, delete states k have zeros, and all other states have the product of the transition probabilities through only delete states up to them, plus the final emission probability.
- When initializing $\beta_k(T)$, similar adjustments must be made.
- To enforce the condition that the model finishes in states m_N, i_N, d_N, we create a special END state, accessible only from m_N, i_N, d_N, and append a special "END" symbol in the final position of each sequence. We then define $A(END, k)$ to be zero unless k is the END state, in which case $A(END, k)$ is one. [$A(z, END)$ is also zero for any z other than the END symbol.]

M-step for Profile HMMs

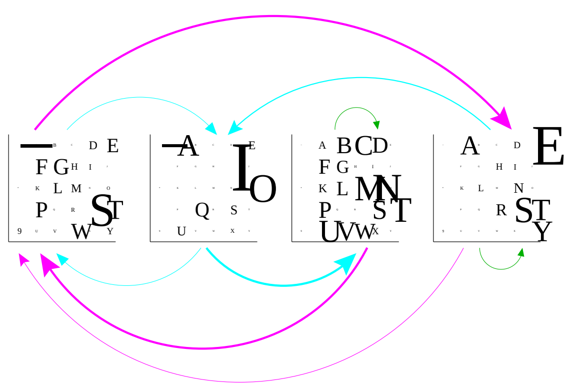
- The emission probabilities $A_j()$ for match and insert states and the initial state distribution π (for m_1, i_1, d_1) are updated exactly as in the regular M-step.
- The expected #transitions from state i to j which begin at time t are different when j is a delete state:

$$\xi_{ij}(t) = \alpha_i(t) S_{ij} \beta_j(t) / L$$

- Given this change, the updates to the transition parameters is the same as in the normal M-step.

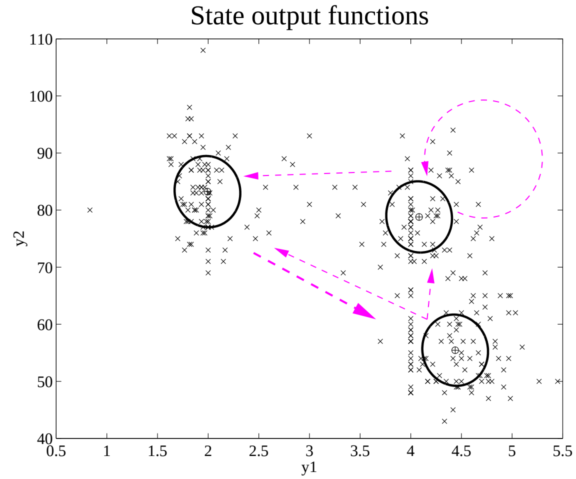
Symbol HMM Example

- Character sequences (discrete outputs)



Mixture HMM Example

- Geyser data (continuous outputs)



Some HMM History

- Markov ('13) and later Shannon ('48,'51) studied *Markov chains*.
- Baum et. al (BP'66, BE'67, BS'68, BPSW'70, B'72) developed much of the theory of "probabilistic functions of Markov chains".
- Viterbi ('67) (now Qualcomm) came up with an efficient optimal decoder for state inference.
- Applications to speech were pioneered independently by:
  - Baker ('75) at CMU (now Dragon)
  - Jelinek's group ('75) at IBM (now Hopkins)
  - communications research division of IDA (Ferguson '74 unpublished)
- Dempster, Laird & Rubin ('77) recognized a general form of the Baum-Welch algorithm and called it the *EM* algorithm.
- A landmark open symposium in Princeton ('80) hosted by IDA reviewed work till then.