Lecture 17:

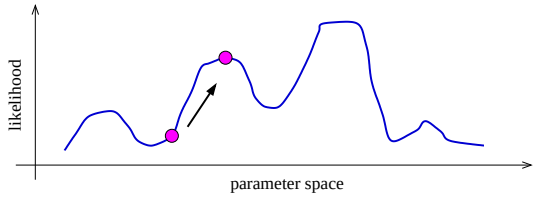
Hidden Markov Model Learning

Sam Roweis

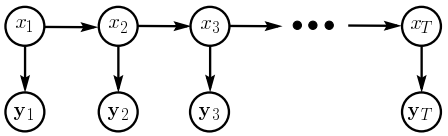
March 10, 2003

Baum-Welch Algorithm: EM Training

1. Intuition: if only we *knew* the true state path then ML parameter estimation would be trivial.
2. But: can *estimate* state path using another trick.
3. *Baum-Welch algorithm* (special case of EM): estimate the states, then compute params, then re-estimate states, and so on ...
4. This works and we can *prove* that it always improves likelihood.
5. However: finding the ML parameters is NP complete, so initial conditions matter a lot and convergence is hard to tell.



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 $\gamma_i(t) = \mathsf{P}(x_t = i|\{\mathbf{y}\})$.

Parameter Estimation using EM

- S_{ij} are transition probs; state j has output distribution $A_j(\mathbf{y})$

$$\mathsf{P}(x_{t+1} = j|x_t = i) = S_{ij} \qquad \mathsf{P}(x_1 = j) = \pi_j$$
$$\mathsf{P}(\mathbf{y}_t = y|x_t = j) = A_j(y)$$

- Complete log likelihood:

$$\log p(x,y) = \log\{\pi_{x_1} \prod_{t=1}^{T-1} S_{x_t,x_{t+1}} \prod_{t=1}^{T} A_{x_t}(\mathbf{y}_t)\}$$
$$= \log\{\prod_i \pi_i^{[x_1^i]} \prod_{t=1}^{T-1} \prod_{ij} S_{ij}^{[x_t^i,x_{t+1}^j]} \prod_{t=1}^{T} \prod_k A_k(\mathbf{y}_t)^{[x_t^k]}\}$$
$$= \sum_i [x_1^i] \log \pi_i + \sum_{t=1}^{T-1} \sum_{ij} [x_t^i, x_{t+1}^j] \log S_{ij} + \sum_{t=1}^{T} \sum_k [x_t^k] \log A_k(\mathbf{y}_t)$$

  where the indicator $[x_t^i] = 1$ if $x_t = i$ and 0 otherwise

- For EM, we need to compute the *expected complete log likelihood*.

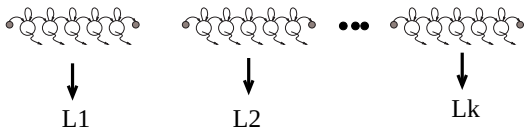
State expectations required from the E-Step

- The expected complete log likelihood requires
  $\gamma_i(t) =< [x_t^i] >$ *and* $\xi_{ij}(t) =< [x_t^i, x_{t+1}^j] >$
- So in the E-step we need to compute both
  $\gamma_i(t) = p(x_t = i|\{\mathbf{y}\})$ and $\xi_{ij}(t) = p(x_t = i, x_{t+1} = j|\{\mathbf{y}\})$.
- We already know how to compute $\gamma_i(t)$ using α and β recursions. We can compute $\xi_{ij}(t)$ the same way:

$$\begin{aligned}
\xi_{ij}(t) &= p(x_{it}, x_{jt+}|\{\mathbf{y}\}) = p(x_{it}|\{\mathbf{y}\})p(x_{jt+}|x_{it}, \{\mathbf{y}\}) \\
&= p(x_{it}, y_1^t|y_{t+1}^T)p(x_{jt+}|x_{it}, y_{t+1}^T)/p(y_1^t|y_{t+1}^T) \\
&= \frac{p(x_{it}, y_1^t)p(y_{t+1}^T|x_{it}, y_1^t)}{p(y_1^t|y_{t+1}^T)p(y_{t+1}^T)} \frac{p(y_{t+1}^T|x_{jt+}, x_{it})p(x_{jt+}|x_{it})}{p(y_{t+1}^T|x_i = t)} \\
&= \frac{p(x_{it}, y_1^t)p(y_{t+1}^T|x_{it})}{p(y_1^T)} \frac{p(y_{t+1}|x_{jt+})p(y_{t+2}^T|x_{jt+})p(x_{jt+}|x_{it})}{p(y_{t+1}^T|x_i = t)} \\
&= \alpha_i(t)A_j(y_{t+1})S_{ij}\beta_j(t+1)/L
\end{aligned}$$

Using HMMs for Recognition

- Use many HMMs for recognition by:
  1. training one HMM for each class (requires *labelled* training data)
  2. evaluating probability of an unknown sequence under each HMM
  3. classifying unknown sequence: HMM with highest likelihood



- This requires the solution of two problems:
  1. Given model, evaluate prob. of a sequence.
     (We can do this exactly & efficiently.)
  2. Give some training sequences, estimate model parameters.
     (We can find the local maximum of parameter space nearest our starting point using Baum-Welch (EM).)

M-step: New Parameters are just Ratios of Frequency Counts

- Initial state distribution: expected #times in state i at time 1:

$$\hat{\pi}_i = \gamma_i(1)$$

- Expected #transitions from state i to j which begin at time t:

$$\xi_{ij}(t) = \alpha_i(t)S_{ij}A_j(\mathbf{y}_{t+1})\beta_j(t+1)/L$$

  so the estimated transition probabilities are:

$$\hat{S}_{ij} = \sum_{t=1}^{T-1} \xi_{ij}(t) \Big/ \sum_{t=1}^{T-1} \gamma_i(t)$$

- The output distributions are the expected number of times we observe a particular symbol in a particular state:

$$\hat{A}_j(y_0) = \sum_{t|\mathbf{y}_t = y_0} \gamma_j(t) \Big/ \sum_{t=1}^{T} \gamma_j(t)$$

Applications of HMMs

- Speech recognition.
- Language modeling.
- Information retrieval.
- Motion video analysis/tracking.
- Protein sequence and genetic sequence alignment and analysis.
- Financial time series prediction.
- ...

HMM Practicalities

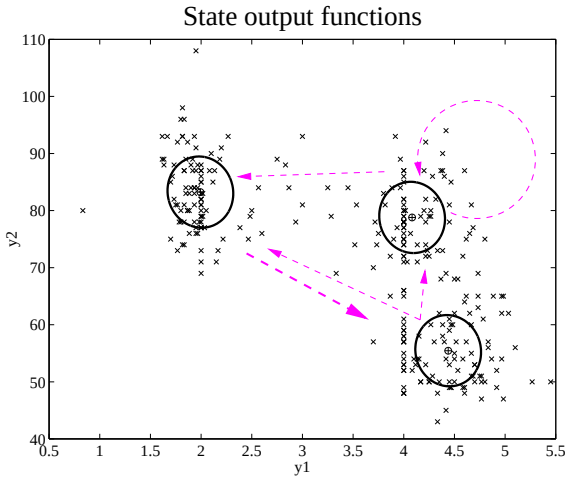
- Multiple observation sequences: can be dealt with by averaging numerators and averaging denominators in the ratios given above.
- Initialization: mixtures of Naive Bayes or mixtures of Gaussians
- Numerical scaling: the probability values that the bugs carry get tiny for big times and so can easily underflow. Good rescaling trick:

$$\rho_t = \mathsf{P}(\mathbf{y}_t|\mathbf{y}_1^{t-1}) \qquad \alpha(t) = \tilde{\alpha}(t)\prod_{t'=1}^{t}\rho_{t'}$$

or represent all probabilities as logs and use `logsum`

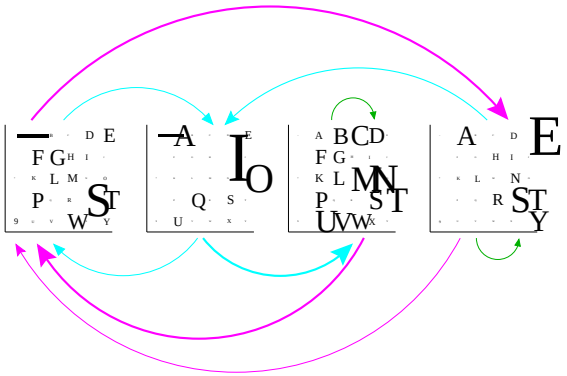
Mixture HMM Example

- Geyser data (continuous outputs)

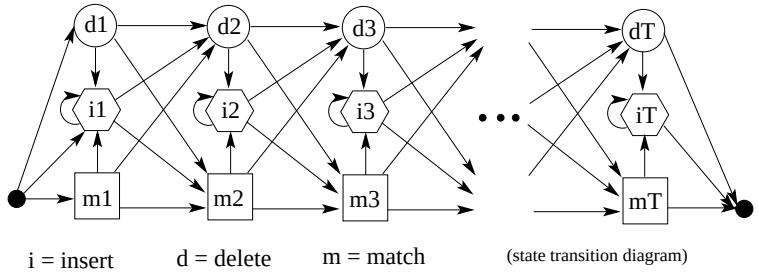


Symbol HMM Example

- Character sequences (discrete outputs)

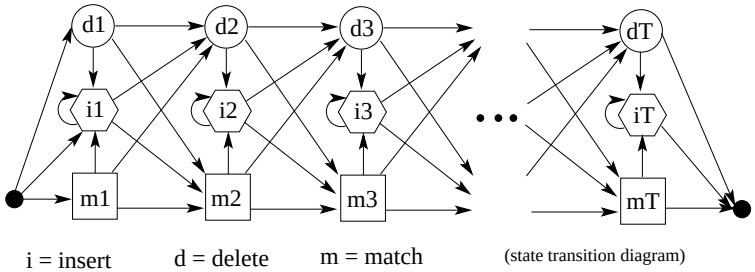


Profile (String-Edit) HMMs



- 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_j – use position j in the template to match an observed symbol
  i_j – insert extra symbol(s) observations after template position j
  d_j – delete (skip) template position j

Profile HMMs have Linear Costs



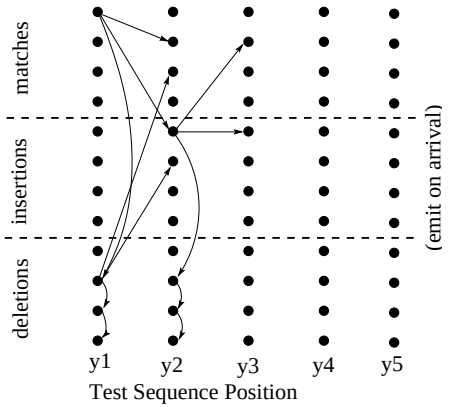
- number of states = 3(length_template)
- Only insert and match states can generate output symbols.
- Once you visit or skip a match state you can never return to it.
- At most 3 destination states from any state, so S_{ij} very sparse.
- Storage/Time cost *linear* in #states, not quadratic.
- State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

String-Edit HMM Grid Costs

$C_{x_t \to x_{t+1}} = -\log T_{x_t, x_{t+1}} - \log A_{x_{t+1}}(\mathbf{y}_t)$ if x_{t+1} is match or insert

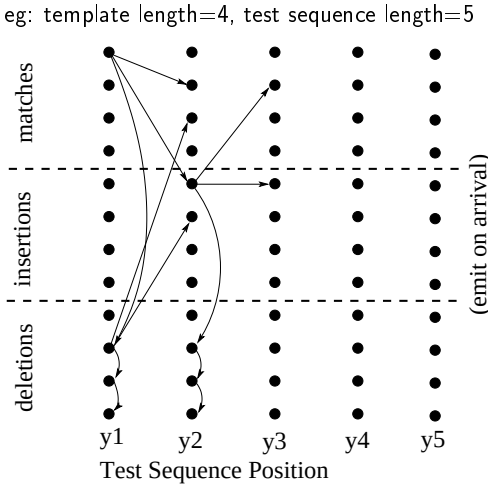
$C_{x_t \to x_{t+1}} = -\log T_{x_t, x_{t+1}}$ if x_{t+1} is a delete state

State $x_t \in \{m_j, i_j, d_j\}$ has nonzero transition probabilities only to states $x_{t+1} \in \{m_{j+1}, i_j, d_{j+1}\}$.



DP for Profile HMMs

- How do we fill in the costs for a DP grid using a string-edit HMM?
- Almost the same as normal except:
  - Now the grid is 3 times its normal height.
  - It is possible to move down without moving right if you move into a deletion state.



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.