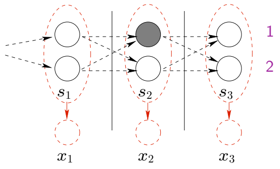
Machine learning: lecture 19

Tommi S. Jaakkola
MIT CSAIL
tommi@csail.mit.edu

Topics

- Hidden Markov models (HMMs)
  - the EM-algorithm
  - example
  - dynamic programming

Forward-backward probabilities: review



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

$$\alpha_t(i) \quad = \quad P(\mathbf{x}_1, \ldots, \mathbf{x}_t, s_t = i)$$

- Backward (diagnostic) 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)

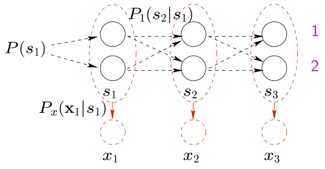
Uses of forward/backward probabilities

- The complementary forward/backward probabilities

$$\begin{aligned} \alpha_t(i) &= P(\mathbf{x}_1, \ldots, \mathbf{x}_t, s_t = i) \\ \beta_t(i) &= P(\mathbf{x}_{t+1}, \ldots, \mathbf{x}_n | s_t = i) \end{aligned}$$

permit us to evaluate various probabilities:

1. $P(\mathbf{x}_1, \ldots, \mathbf{x}_n)$
2. $\gamma_t(i) = P(s_t = i | \mathbf{x}_1, \ldots, \mathbf{x}_n)$
3. $\xi_t(i,j) = P(s_t = i, s_{t+1} = j | \mathbf{x}_1, \ldots, \mathbf{x}_n)$



The EM algorithm for HMMs

Assume we have L observation sequences $\mathbf{x}_1^{(l)}, \ldots, \mathbf{x}_{n_l}^{(l)}$

E-step: compute the posterior probabilities

$$\begin{aligned} &\gamma_t^{(l)}(i) && \text{for all } l,\ i, \text{ and } t\ (t = 1, \ldots, n_l) \\ &\xi_t^{(l)}(i,j) && \text{for all } l,\ i, \text{ and } t\ (t = 1, \ldots, n_l - 1) \end{aligned}$$

M-step: First, the initial state distribution can be updated according to the expected fraction of times the sequences started from a specific state i

$$\hat{P}(i) \ \leftarrow \ \frac{1}{L} \sum_{l=1}^{L} \gamma_1^{(l)}(i)$$

M-step cont'd

Second, the transition probabilities can be updated on the basis of the posterior counts:

$$\hat{P}_1(j|i) \ \leftarrow \ \frac{\hat{n}(i,j)}{\sum_{j'} \hat{n}(i,j')}$$

where

$$\hat{n}(i,j) \quad = \quad \sum_{l=1}^{L} \sum_{t=1}^{n-1} \xi_t^{(l)}(i,j)$$

defines the expected number of transitions from i to j

M-step cont'd

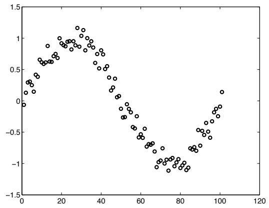
- Lastly, for the outputs we have to (in general) solve a weighted maximum likelihood estimation problem:

  Separately for each state i we maximize:

$$J(\theta_i) = \sum_{l=1}^{L} \sum_{t=1}^{n_l} \gamma_t^{(l)}(i) \, \log P(\mathbf{x}_t^{(l)}|\theta_i)$$

  with respect to the parameters θ_i (e.g, the mean and the covariance of a Gaussian).

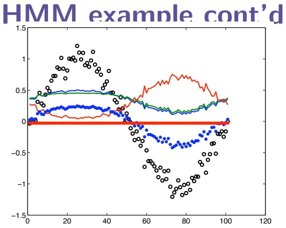
HMM example



Observed output as a function of time

- We will try to model this with a 3-state HMM with Gaussian outputs $p(x|s=i) = p(x|\mu_i, \sigma_i^2)$, $i = 1, 2, 3$.

HMM example cont'd



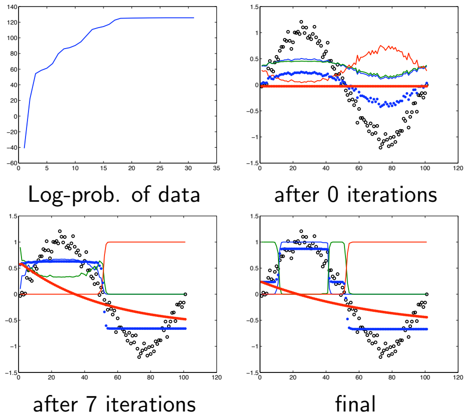
prior/posterior means and $\gamma_t(\cdot)$

$$\text{prior mean}(t) = \sum_i P_t(i)\hat{\mu}_i \quad (\text{'*'})$$

$$\text{posterior mean}(t) = \sum_i \gamma_t(i)\hat{\mu}_i \quad (\text{'*'})$$

where $P_t(i)$ is the probability of being in state i after t steps without observations; $\hat{\mu}_i$ is the mean output from the i^{th} state

HMM example cont'd



Log-prob. of data

after 0 iterations

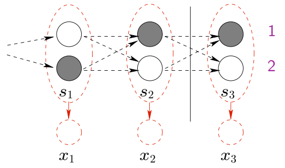
after 7 iterations

final

HMM problems

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

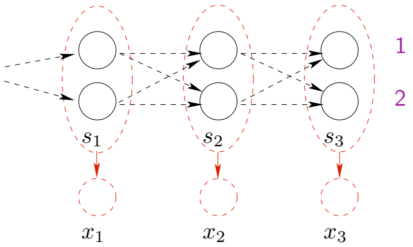
Max-probabilities



- We can recover the most likely hidden state sequence corresponding to a sequence of observations by evaluating the following max-probabilities:

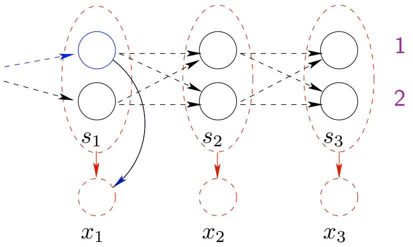
$$\delta_t(i) = \max_{s_1,\ldots,s_{t-1}} P(\mathbf{x}_1, \ldots, \mathbf{x}_t, s_1, \ldots, s_{t-1}, s_t = i)$$

Viterbi algorithm



$\delta_1(1) = P(x_1, s_1 = 1)$
$\delta_1(2) = P(x_1, s_1 = 2)$

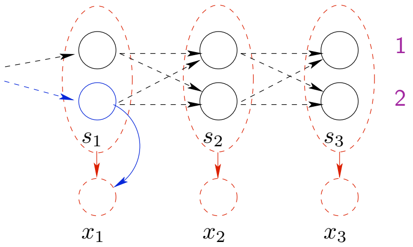
Viterbi algorithm



$\delta_1(1) = P(x_1, s_1 = 1)$
$\delta_1(2) = P(x_1, s_1 = 2)$

$$\delta_1(1) \;=\; P(1)P_x(\mathbf{x}_1|1)$$

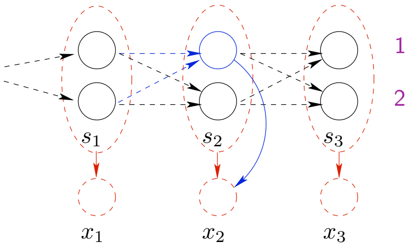
Viterbi algorithm



$\delta_1(1) = P(x_1, s_1 = 1)$
$\delta_1(2) = P(x_1, s_1 = 2)$

$$\begin{aligned}\delta_1(1) &= P(1)P_x(\mathbf{x}_1|1)\\ \delta_1(2) &= P(2)P_x(\mathbf{x}_1|2)\end{aligned}$$

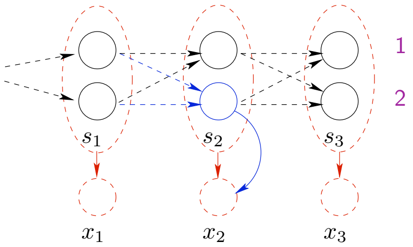
Viterbi algorithm



$\delta_2(1) = \max_{s_1} P(x_1, x_2, s_1, s_2 = 1)$
$\delta_2(2) = \max_{s_1} P(x_1, x_2, s_1, s_2 = 2)$

$$\delta_2(1) \;=\; \max\left\{\delta_1(1)P_1(1|1), \delta_1(2)P_1(1|2)\right\}P_x(\mathbf{x}_2|1)$$

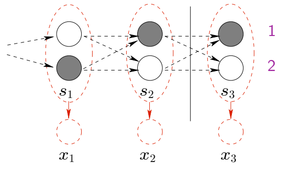
Viterbi algorithm



$\delta_2(1) = \max_{s_1} P(x_1, x_2, s_1, s_2 = 1)$
$\delta_2(2) = \max_{s_1} P(x_1, x_2, s_1, s_2 = 2)$

$$\begin{aligned}\delta_2(1) &= \max\left\{\delta_1(1)P_1(1|1), \delta_1(2)P_1(1|2)\right\}P_x(\mathbf{x}_2|1)\\ \delta_2(2) &= \max\left\{\delta_1(1)P_1(2|1), \delta_1(2)P_1(2|2)\right\}P_x(\mathbf{x}_2|2)\end{aligned}$$

Viterbi algorithm

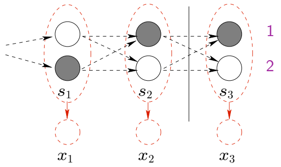


$$\delta_t(i) = \max_{s_1,\ldots,s_{t-1}} P(\mathbf{x}_1, \ldots, \mathbf{x}_t, s_1, \ldots, s_{t-1}, s_t = i)$$

- We get the following recursive equation for calculating the max probabilities:

$$\begin{aligned}\delta_1(i) &= P(i)P_x(\mathbf{x}_1|i)\\ \delta_t(i) &= \max_j \left\{\delta_{t-1}(j)P_1(i|j)\right\}P_x(\mathbf{x}_t|i)\end{aligned}$$

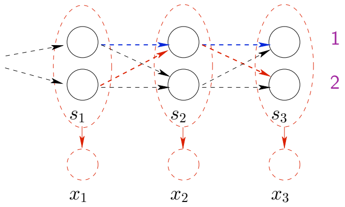
Viterbi algorithm: back-tracking



- We can recover the most likely state sequence by working backwards:

$$\begin{aligned} s_n^* &= \arg\max_i \delta_n(i) \\ s_t^* &= \arg\max_j \left\{ \delta_t(j) P_1(s_{t+1}^*|j) \right\} \end{aligned}$$

Viterbi algorithm: properties

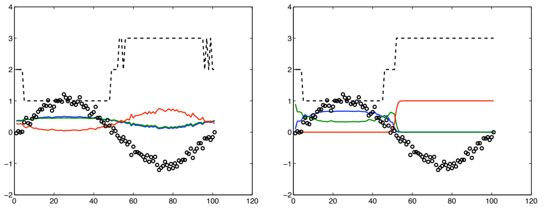


- The most likely path has the property that any partial path is also optimal:

  If $s_t^* = i$ then $\{s_1^*, \ldots, s_t^*\}$ is also the most likely state sequence forced to end up in $s_t = i$ at time t given only $\mathbf{x}_1, \ldots, \mathbf{x}_t$.

Viterbi algorithm: example

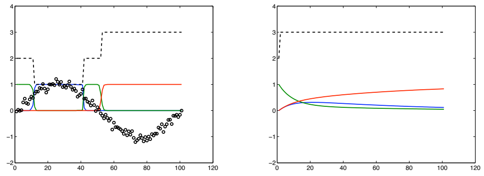
- Same example as in the EM case (3 states, Gaussian outputs)



after 0 iterations after 7 iterations

- The most likely hidden state sequence $\{s_0^*, \ldots, s_n^*\}$ need not agree with the most likely states derived from the posterior marginals $\gamma_t(i)$

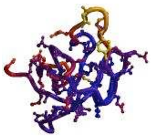
Example cont'd



final ML model, no observations

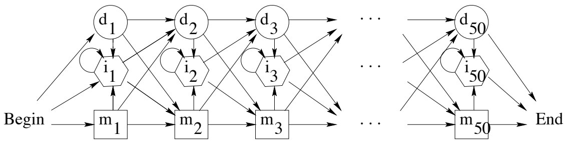
Linear HMMs, alignment

- Proteins



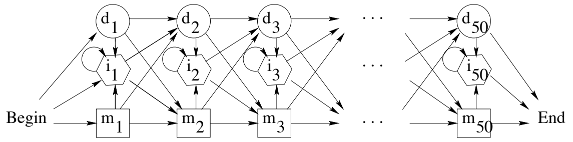
```
GVAALGAKVLAQIGVAVSHLGDEGKMVAQMKAVGVRHKGYGNKHIKAQYFEPLGASLLSAMEHRIG
```

- A Linear HMM model for protein sequences



```
GVAALGAKVLAQIGVAVSHLGDegkMVAQMKAVGVRHKgygNK-HIKAQYFEPLGASLLSAMEHRIG
```

Linear HMMs, multiple alignment



```
-VKGHGKKVADALTNAVAHVDD.....MPNALSALSDLHA...HKLRVDPV.NFKLLSHCLLVTLAAHLP
KVKAHGKKVLGAFSDGLAHLDN.....LKGTFATLSELHC...DKLHVDPE.NFRLLGNVLVCVLAHHFG
DLKKHGVTVLTALGAILKKKGH.....HEAELKPLAQSHA...TK-HKIPIkYLEFISEAIIHVLHSRHP
PFETHANRIVGFFSKIIGELPN.....IEADVNTFVASHK...PR-GVTHD.QLNNFRAGFVSYMKAH--
DVRWHAERIINAVNDAVASMDDtek..MSMKLRDLSGKHA...KSFQVDPQ.YFKVLAAVIADTVAA---
ELQAHAGKVFKLVYEAAIQLQVtgvvvTDATLKNLGSVHV...SK-GVADA.HFPVVKEAILKTIKEVVG
GVAALGAKVLAQIGVAVSHLGDegk..MVAQMKAVGVRHKgygNK-HIKAQ.YFEPLGASLLSAMEHRIG
```