Hidden Markov Models

Chris Williams

School of Informatics, University of Edinburgh

November 2010
Overview

- Definitions
- Inference Problems
- Recursion formulae
- Viterbi alignment
- Training a HMM
- Reading: Bishop §13.1, 13.2 (but not 13.2.3, 13.2.4, 13.2.5), Rabiner paper
Dynamical models used in many areas for modelling sequences, including

- Speech recognition
- Molecular biology sequences
- Linguistic sequences (e.g. part-of-speech tagging)
- Multi-electrode spike-train analysis
- Tracking objects through time
Markov Chain

\[ p(a, b, c, d) = p(a)p(b|a)p(c|b)p(d|c) \]

Hidden Markov Model
A HMM is defined by

- $K$ the number of states
- $A$ the state transition matrix
- $\rho(x_n|z_n)$ the output probability distribution (independent of $n$)
- $\pi$ the initial transition probabilities
- $z_n$ is a multinomial variable with components $z_{ni}$, if $z_n$ is in state $i$ then $z_{ni} = 1$ and $z_{nj} = 0$ for $j \neq i$
• Let $\pi_{z_1} = \prod_{i=1}^{K} [\pi_i]^{z_{1i}}$ and
  
  $$a_{z_nz_{n+1}} = \prod_{i,j} [a_{ij}]^{z_{ni}z_{n+1,j}}$$

• Let $X = x_1, \ldots, x_N$ and $Z = z_1, \ldots, z_N$

• For any state sequence $Z$

  $$p(X, Z) = \pi_{z_1} p(x_1|z_1) a_{z_1z_2} p(x_2|z_2) \cdots a_{z_{N-1}z_N} p(x_N|z_N)$$
Independence relationships

Conditioning on $z_n$ renders $z_{n-1}$ and $z_{n+1}$ independent, i.e.

$I(z_{n-1}, z_{n+1}| z_n)$

$I(z_s, z_u| z_n)$ for all $s < n$, $u > n$
\[ l(x_s, x_u | z_n) \text{ for all } s \leq n, \ u > n \]

- the future is independent of the past given the present
- Note that conditioning on \( x_n \) does not yield any conditional independences
- HMM as a dynamical mixture model; choice of state is not independent at each time frame, but depends on the past
- HMM as a finite state automaton

```plaintext
P(x|s_3)  
\[s_3\]  
P(x|s_1)  
\[s_1\]  
P(x|s_2)  
\[s_2\]  
\[a_{21}\]  
\[a_{32}\]  
\[a_{22}\]  
```
Inference Problems

- $p(z_1 \ldots z_N | X)$ inferring hidden state given $X$
- $p(z_n | X)$ marginal of above
- $p(z_n | x_1, \ldots, x_n)$ filtering
- $p(z_n | x_1, \ldots, x_s)$ $n > s$, prediction
- $p(z_n | x_1, \ldots, x_u)$ $n < u$, smoothing
- $p(x_1, \ldots, x_N)$ likelihood calculation
- Find sequence $z_1^* \ldots z_N^*$ that maximizes $p(Z | X)$ [Viterbi alignment]
filtering

smoothing

prediction

denotes the extent of data available
- Naive approach $O(K^N)$
- Efficient $O(K^2N)$ algorithm is available by pushing sums through products
Computing $p(z_n|X)$

$$p(z_n|X) = \frac{p(X|z_n)p(z_n)}{p(X)}$$

$$= \frac{p(x_1, \ldots, x_n|z_n)p(x_{n+1} \ldots x_N|z_n)p(z_n)}{p(X)}$$

$$= \frac{p(x_1, \ldots, x_n, z_n)p(x_{n+1} \ldots x_N|z_n)}{p(X)}$$

$$\equiv \frac{\alpha(z_n)\beta(z_n)}{p(X)}$$

The alphas and betas can be calculated recursively.
\[ \sum_{z_n} p(z_n|X) = 1 = \frac{\sum_{z_n} \alpha(z_n) \beta(z_n)}{p(X)} \]

implies

\[ p(X) = \sum_{z_n} \alpha(z_n) \beta(z_n) \]

• Define \( p(z_n|X) = \gamma(z_n) \)
Recursion formulae

- **Alpha**

\[ \alpha(z_{n+1}) = \sum_{z_n} \alpha(z_n) a_{z_n z_{n+1}} p(x_{n+1}|z_{n+1}) \]

Initialization

\[ \alpha(z_1) = p(x_1, z_1) = p(x_1|z_1)p(z_1) = p(x_1|z_1)\pi_{z_1} \]

- **Beta**

\[ \beta(z_n) = \sum_{z_{n+1}} \beta(z_{n+1}) a_{z_n z_{n+1}} p(x_{n+1}|z_{n+1}) \]

Initialization: \( \beta(z_N) \) is the vector of ones as

\[ \sum_i \alpha(z_{Ni}) \beta(z_{Ni}) = \sum_i \alpha(z_{Ni}) = \sum_i p(x_1, \ldots, x_N, z_N = i) = p(X) \]

Each step is \( O(K^2) \)
Viterbi alignment

Find the best state sequence \( z_1^*z_2^* \ldots z_N^* \) that maximizes \( p(Z|X) \)

Define

\[
\delta_n(i) = \max_{z_1, \ldots, z_{n-1}} p(z_1, \ldots, z_n = i, x_1 \ldots x_n)
\]

i.e. \( \delta_n(i) \) is the best score along a single path up to time \( n \) which account for the first \( n \) observations and ends in state \( S_i \).

- There is a recursive formula for the \( \delta \)s similar to the \( \alpha \)-recursion, except that a max rather than sum operation is used.
- For further details see, for example, L. Rabiner, Proc. IEEE 77(2) 1989 pp 257-285
Calculating $p(z_n, z_{n+1}|X)$

$$\xi(z_n, z_{n+1}) \equiv p(z_n, z_{n+1}|X)$$

$$= \frac{p(z_n, z_{n+1}, X)}{p(X)}$$

$$= \frac{p(X|z_n, z_{n+1})p(z_{n+1}|z_n)p(z_n)}{p(X)}$$

$$= p(x_1, \ldots, x_n|z_n)p(x_{n+1}|z_{n+1}) \times$$

$$\frac{p(x_{n+2} \ldots x_N|z_{n+1})p(z_{n+1}|z_n)p(z_n)}{p(X)}$$

$$= \frac{\alpha(z_n)p(x_{n+1}|z_{n+1})\beta(z_{n+1})a_{z_n,z_{n+1}}}{p(X)}$$
Training a HMM

- Use the EM algorithm to estimate $\pi$, $A$ and $\eta$, the parameters of $p(x_n|z_n)$. Let $\theta = (\pi, A, \eta)$.
- If we knew the “true” state sequence, parameter estimation would be easy. The trick is to use the probability distribution over state paths to weight these estimates.

\[
\hat{\pi}_i \leftarrow \gamma(Z_{1i})
\]
\[
\hat{a}_{ij} \leftarrow \frac{\sum_{n=1}^{N-1} \xi(Z_{ni}, Z_{n+1,j})}{\sum_{n=1}^{N-1} \gamma(Z_{ni})}
\]
If the output is a multinomial distribution with
\[ p(x_{nj} = 1 | z_{ni} = 1) = \eta_{ij} \] and \[ \sum_k x_{nk} = 1 \]

\[ \hat{\eta}_{ij} \leftarrow \frac{\sum_{n=1}^{N} \gamma(z_{ni}) x_{nj}}{\sum_{n=1}^{N} \gamma(z_{ni})} \]

For HMMs these are known as the Baum-Welch equations
Example: Harmonizing Chorales in the Style of J S Bach

- Moray Allan and Chris Williams (NIPS 2004)
  http://www.tardis.ed.ac.uk/~moray/harmony/,
  online demo at
  http://www.anc.inf.ed.ac.uk/demos/hmmbach/

- Visible states are the melody (quarter notes)
- Hidden states are the harmony (which chord)
- Trained using labelled melody/harmony data (no need for EM)
- Task: find Viterbi alignment for harmony given melody (or sample from $p(\text{harmony} | \text{melody})$.)
- Actually uses HMMs for three subtasks: harmonic skeleton, chord skeleton, ornamentation