Further details of the Baum-Welch algorithm

Martin Emms

November 30, 2017

real Baum-Welch: summing the 'clock-tick' probs

- brute-force EM would for each \( o^d \) calculate 'responsibility' \( \gamma^d(s) = p(s|o^d) \) for all \( s \) and from these calculate various expectations (eg. \( E^d(i), E^d(ij) \))

- Baum-Welch instead first runs \( \alpha \) and \( \beta \) for \( o^d \). By various terms involving \( \alpha \) and \( \beta \) can derive various per \( t \) 'clock tick' mini responsibilities and then summations over these give various expectations (\( E^d(i), E^d(ij) \) etc).

These are

'transition' \( \xi_t(i, j) = \) the cond. prob. of transition \( ij \) at \( t \) given \( o^d \)

\[
\xi_t(i, j) = \frac{[\alpha_t(i) \times a_{ij} \times b_j (o_{t+1}) \times \beta_{t+1}(j) \times P(o^d)]}{P(o^d)}
\]
re-estimation of transition probs $A$

the re-estimation for the transition probs $a_{ij}$ involves getting the expected count of transition $ij$ and comparing to the expected count of $i$

$$a_{ij} = \frac{\sum_d \sum_{t=1}^{T-1} \xi_d(i,j)}{\sum_d \sum_{t=1}^{T-1} \gamma_d(t)}$$

Note the limit $T-1$: at the last time tick there is no defined $ij$ transition, nor should any expected state value at $T$ be relevant.

re-estimating the observation probs $B$

the re-estimation for the obs probs $b_j(k)$ involves getting the expected count being in state $j$ and producing observation symbol $k$ and comparing this to the expected count of being in state $j$

$$b_j(k) = \frac{\sum_d \sum_{t=1}^{T} |o_t = k \gamma_t(j)|}{\sum_d \sum_{t=1}^{T} \gamma_t(j)}$$

in the numerator just the time ticks where $o_t = k$ are taken, and in the denominator every time tick is taken.

Further details of the Baum-Welch algorithm

Further details of the Baum-Welch algorithm
Further details of the Baum-Welch algorithm

re-estimating the start probs $\pi$

the re-estimation for start prob $\pi[i]$ involves getting the expected count of
being in state $i$ at $t = 1$ and comparing to number of observations $D$

$$\hat{\pi}[i] = \frac{\sum_d \gamma_d(i)}{D}$$

The backward algorithm

'forward probability' $\alpha_t(i) = P(o_1 \ldots o_t, s_t = i)$

Recursion for $\alpha$

base

$$\alpha_1(i) = \pi(i)b(o_1)$$

recursive

$$\alpha_t(j) = \sum_{i=1}^N \alpha_{t-1}(i)a_{ij}b(o_t), \text{ for } t = 2, \ldots, T$$

corresponding for $\beta$

'backward probability' $\beta_t(i) = P(o_{t+1} \ldots o_T|s_t = i)$

Recursion for $\beta$

base

$$\beta_T(i) = 1$$

recursive

$$\beta_t(i) = \sum_{j=1}^N a_{ij}b(o_{t+1})\beta_{t+1}(j), \text{ for } t = T - 1, \ldots, 1$$

deriving $\beta$

for $\beta_t(i)$ need $P(o_{t+1} \ldots o_T|s_t = i)$. let $j$ be some arbitrary state at $t + 1$. If
had $P(s_{t+1} = j, o_{t+1} \ldots o_T|s_t = i)$, could sum over the $j$ to get desired quantity.

$$P(s_{t+1} = j, o_{t+1} \ldots o_T|s_t = i) = \frac{P(s_t = i, s_{t+1} = j, o_{t+1} \ldots o_T)}{P(s_t = i)}$$

$$= \frac{P(s_t = 1, s_{t+1} = j, o_{t+1})\beta_{t+1}(j)}{P(s_t = i)}$$

$$= \frac{P(s_t = 1, s_{t+1} = j)b(o_{t+1})\beta_{t+1}(j)}{P(s_t = i)}$$

$$= a_{ij}b(o_{t+1})\beta_{t+1}(j)$$

hence

$$\beta_t(i) = \sum_{j=1}^N a_{ij}b(o_{t+1})\beta_{t+1}(j)$$