Agenda

- Homework
  - HW1 – graded (posting by email)
  - HW2 – graded by next Tuesday (maybe Thursday)
  - HW3 – due next Thursday 10/13
- Questions, comments, concerns?
  - Re-visit Viterbi & Forward Algorithms
  - Forward-Backward Algorithm

Viterbi Algorithm

- Use an \( N \times T \) trellis \([v_t^j]\)
  - Just like in forward algorithm
- \( v_t^j \) or \( v_t(j) \)
  \[ v_t^j \text{ or } v_t(j) = P(\text{in state } j \text{ after seeing } t \text{ observations and passing through the most likely state sequence so far}) \]
  \[ = P(q_0, q_1, q_2, \ldots, q_t, o_t) \]
  - Each cell = extension of most likely path from other cells
  \[ v_t^j = \max_i v_{t-1}^i \cdot a_{ij} \cdot b_j(o_t) \]

Viterbi Algorithm: Formal Definition

- Initialization
  - \( v_1^j = \pi_i(o_1); 1 \leq i \leq N \)
  - \( B_T^i = 0 \)
- Recursion
  - \( a_j = \max_i [v_{t-1}^i \cdot a_{ij} \cdot b_j(o_t)]; 1 \leq i \leq N, 2 \leq t \leq T \)
  - \( B_T^j = \argmax_i [v_{t-1}^i \cdot a_{ij}] \)
- Termination
  - \( \rho = \max_j v_T^j \)
  - \( \hat{q}_T = \argmax_j v_T^j \)

HMM Tagger – Initialization (v2)

- word sequence: \( W = w_1 \ldots w_n \), for time \( 1 \leq t \leq n \)
- total corpus size: \( N \)
- input (word) vocabulary: \( v_i \in V \) for \( 1 \leq i \leq k \)
- output (tag) vocabulary: \( \tau_j \in T \) for \( 1 \leq j \leq m \)
- Let \( b_j(v) = P(v_i | \tau_j) = c(t,v) \cdot c(t) \cdot c(t+m) \)
- Let \( a_j(t) = P(\tau_j | \tau_i) = c(t,j+1) \cdot c(t) \cdot c(t+m) \)
- Let \( \alpha_0(0) = 1 \) and \( \alpha_t(o) = \max_i [\alpha_{t-1}^i \cdot a_i] \cdot b(o) \)
- \( \xi(t) = \max_{t-1} [\alpha_{t-1}^i \cdot a_i] \)

Viterbi Algorithm (version 2)

- word sequence: \( W = w_1 \ldots w_n \), size of tagset \( |T| = m \)
- for \( t = 1 \) to \( n \)
  - for \( j = 1 \) to \( m \)
    - \( \xi(t) \leftarrow \argmax_{t-1} [\alpha(t-1) \cdot a_j] \)
    - \( a(t) \leftarrow \max_i [\alpha(t-1) \cdot a_i] \cdot b(o) \)
    - \( \xi(n+1) \leftarrow \argmax_{t-1} [\alpha(n)] \)
  - \( \rho(n+1) \leftarrow 0 \)
- for \( t = n \) to \( 1 \)
  - \( \rho(t) \leftarrow \xi(t+1) \cdot (t+1) \)
  - \( \tau(t) \leftarrow \xi(t+1) \cdot (t+1) \)
Viterbi Algorithm (version 3)
• pseudocode for the Viterbi algorithm is also given in the textbook
  • Just be sure to initialize as defined on slide 41 of lecture 9

Forward Algorithm
• Use an $N \times T$ trellis or chart $[a_{ij}]$
• Forward probabilities: $a_j(t)$
  • $= P(\text{being in state } j \text{ after seeing } t \text{ observations})$
  • $= P(q_t = j | o_1, o_2, ..., o_t)$
• Each cell $= \sum$ extensions of all paths from other cells
  $$a_j(t) = \sum_{i} a_i(t-1) a_{ij} b_j(o_t)$$
  • $a_i(t-1)$: forward path probability until $(t-1)$
  • $a_{ij}$: transition probability of going from state $i$ to $j$
  • $b_j(o_t)$: probability of emitting symbol $o_t$ in state $j$
  • $P(O|\lambda) = \sum_i a_i(T)$

Forward-Backward (Baum-Welsch) Algorithm
• What if, instead of wanting to know:
  • $P(\text{being in state } j \text{ after seeing } t \text{ observations})$
  (Forward Algorithm)
  • $P(\text{in state } j \text{ after seeing } t \text{ observations and passing through the most likely state sequence so far})$
  (Viterbi Algorithm)
• We want to know:
  • $P(\text{being in state } j \text{ at time } t \text{ given the entire observation sequence})$
  • $P(\text{being in state } j \text{ at time } t \text{ and being in state } k \text{ at time } t+1 \text{ given the entire observation sequence})$
• Our forward probability $a_j(t)$ is insufficient to calculate these conditional probabilities
  • Also need a backward probability

Forward and Backward Probabilities

<table>
<thead>
<tr>
<th>$w_t$</th>
<th>$w_{t+1}$</th>
<th>$w_{t+2}$</th>
<th>$w_{n}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a_j(t)$</td>
<td>$a_{ij}$</td>
<td>$a_{ij}$</td>
<td>$a_{ij}$</td>
</tr>
<tr>
<td>$\beta_j(t)$</td>
<td>$\sum_{i=1}^{m} a_i(t-1) a_{ij} b_j(o_t)$</td>
<td>$\sum_{i=1}^{m} a_i(t-1) a_{ij} b_j(o_{t+1})$</td>
<td>$\sum_{i=1}^{m} a_i(t-1) a_{ij} b_j(o_{n})$</td>
</tr>
<tr>
<td>$P(w_1 \ldots w_n) = \sum_{i=1}^{m} a_i(t-1) a_{ij} b_j(o_{n})$</td>
<td>$\sum_{i=1}^{m} a_i(t-1) a_{ij} b_j(o_{n})$</td>
<td>$\sum_{i=1}^{m} a_i(t-1) a_{ij} b_j(o_{n})$</td>
<td>$\sum_{i=1}^{m} a_i(t-1) a_{ij} b_j(o_{n})$</td>
</tr>
</tbody>
</table>

New Parameters for Forward-Backward

Probability of having tag $i$ at time $t$ given $w_1 \ldots w_n$

$$\gamma_i(t) = \frac{a_i(t) \beta_i(t)}{\sum_{j=1}^{m} \alpha_j(t) \beta_j(t)}$$

Probability of having tag $i$ at time $t$ and tag $j$ at time $t+1$, given $w_1 \ldots w_n$

$$\xi_{ij}(t) = \frac{\gamma_i(t) a_{ij} b_j(w_{t+1}) \beta_j(t+1)}{\beta_i(t)}$$
Forward-Backward Algorithm

word sequence: \( W = w_1, \ldots, w_T \), size of tagset \( T = m \) \( \alpha_0(t) = 1 \)

for \( j = 1 \) to \( m \)

\( \alpha_j(t) \leftarrow \sum_{a_{j+1}} \alpha_{j+1}(t)a_j h_j(n) \)

for \( i = 1 \) to \( m \)

\( \beta_i(t) \leftarrow \gamma_i(t) \sum_{a_{j+1}} \beta_{j+1}(t)a_j h_j(n) \)

for \( t = 1 \) to \( T \)

\( \gamma_i(t) \leftarrow \frac{\alpha_i(t) \beta_i(t)n}{\sum_{a_{j+1}} \alpha_j(t)a_j h_j(n)} \)

for \( j = 1 \) to \( m \)

\( \lambda\text{stock} \)

\( \pi_1 = 0.5 \)

\( \pi_2 = 0.2 \)

\( \pi_3 = 0.3 \)

\[ \sum \alpha_{i+1}(Bull) \times a_{BullBull} \times b_{Bull} \times \gamma(t) \]

\[ \sum \beta_{i+1}(Bull) \times a_{BullBull} \times b_{Bull} \times \gamma(t) \]

\[ \sum \gamma_i(t) \]

\[ \sum \alpha_{i+1}(Bear) \times a_{BearBear} \times b_{Bear} \times \gamma(t) \]

\[ \sum \beta_{i+1}(Bear) \times a_{BearBear} \times b_{Bear} \times \gamma(t) \]

\[ \sum \gamma_i(t) \]

\[ \sum \alpha_{i+1}(Static) \times a_{StaticStatic} \times b_{Static} \times \gamma(t) \]

\[ \sum \beta_{i+1}(Static) \times a_{StaticStatic} \times b_{Static} \times \gamma(t) \]

\[ \sum \gamma_i(t) \]
Forward-Backward Algorithm, E-step

word sequence: $W = w_1, \ldots, w_n$, size of tagset $|T| = m$, $a_0(0) = 1$

for $i = 1$ to $m$
  for $j = 1$ to $m$
    $a(i) = \sum_{k=1}^{m} \alpha(i-1) a(k) b(j)(k)$
  
for $i = 1$ to $m$
  $B(i) = \sum_{j=1}^{m} \alpha(i-1) a(j) b(j)(i)$

for $i = 1$ to $m$
  $\gamma(i) = \frac{\alpha(i) B(i)}{\sum_{m} \alpha(i) B(i)}$

for $i = n$ to $1$
  for $j = 1$ to $m$
    $\hat{\beta}(i) = \frac{\alpha(i) b(j)(i) \gamma(i)}{\sum_{m} \alpha(i) b(j)(i) \gamma(i)}$
  
for $i = 1$ to $m$
  $\hat{\gamma}(i) = \frac{\alpha(i) \hat{\beta}(i)}{\sum_{m} \alpha(i) \hat{\beta}(i)}$

Agenda: Summary

- Review Viterbi, Forward Algorithms
- Forward-Backward (Baum-Welsh) Algorithm
- Midterm

Forward-Backward, M-step

corpus of $N$ sentences, $W = w_1^1, \ldots, w_n^N$, size of tagset $|T| = m$

initialize $a_{ij}, a_{j0}, a_{0j}$, and $b_j(v_k)$ to 0 for all $i, j, k$

for $i = 1$ to $m$
  $e_{ij} = \sum_{k=1}^{N} w_{i+1}(i, j)$
  $a_{0i} = \frac{1}{N} \sum_{k=1}^{N} \gamma_{ji}^k(1)$
  $a_{00} = \frac{1}{N} \sum_{k=1}^{N} \gamma_{ji}^k([W_0])$

for $j = 1$ to $m$
  $a_{ij} = \frac{1}{N} \sum_{k=1}^{N} \gamma_{ji}^k([W_i])$

for $k = 1$ to $|V|$
  $b_k(v_k) = \frac{1}{N} \sum_{j=1}^{N} \gamma_{ji}^k(\delta_{jk}(i))$