Recap: HMM

• Elements of HMM:
  – Set of states (tags)
  – Output alphabet (word types)
  – Start state (beginning of sentence)
  – State transition probabilities
  – Output probabilities from each state
More general notation

• Previous lecture:
  – Sequence of tags $T = t_1...t_n$
  – Sequence of words $S = w_1...w_n$

• This lecture:
  – Sequence of states $Q = q_1 ... q_T$
  – Sequence of outputs $O = o_1 ... o_T$
  – So $t$ is now a time step, not a tag! And $T$ is the sequence length.
Recap: HMM

• Given a sentence \( O = o_1 \ldots o_T \) with tags \( Q = q_1 \ldots q_T \), compute \( P(O,Q) \) as:

\[
P(O, Q) = \prod_{t=1}^{T} P(o_t|q_t)P(q_t|q_{t-1})
\]

• But we want to find \( \text{argmax}_Q P(Q|O) \) without enumerating all possible \( Q \)
  – Use Viterbi algorithm to store partial computations.
Tagging example

Words:

Possible tags: (ordered by frequency for each word)

<table>
<thead>
<tr>
<th></th>
<th>one</th>
<th>dog</th>
<th>bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;s&gt;</td>
<td>CD</td>
<td>NN</td>
<td>NN</td>
</tr>
<tr>
<td></td>
<td>NN</td>
<td>VB</td>
<td>VBD</td>
</tr>
<tr>
<td></td>
<td>PRP</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Tagging example

Words:
<\s> one dog bit </\s>

Possible tags:
(ordered by frequency for each word)

<table>
<thead>
<tr>
<th></th>
<th>one</th>
<th>dog</th>
<th>bit</th>
<th>&lt;/s&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;\s&gt;</td>
<td>CD</td>
<td>NN</td>
<td>NN</td>
<td>&lt;/\s&gt;</td>
</tr>
<tr>
<td></td>
<td>NN</td>
<td>VB</td>
<td>VBD</td>
<td></td>
</tr>
<tr>
<td>PRP</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Choosing the best tag for each word independently gives the wrong answer (<\s> CD NN NN </\s>).
- $P(\text{VB}|\text{bit}) < P(\text{NN}|\text{bit})$, but may yield a better *sequence* (<\s> CD NN VB </\s>)
  - because $P(\text{VBD}|\text{NN})$ and $P(</\s>|\text{VBD})$ are high.
Viterbi: intuition

Words:
<s> one dog bit </s>

Possible tags: (ordered by frequency for each word)

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;s&gt;</td>
<td>CD</td>
<td>NN</td>
<td>NN</td>
</tr>
<tr>
<td>NN</td>
<td>VB</td>
<td>VBD</td>
<td></td>
</tr>
<tr>
<td>PRP</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Possible tags:

Suppose we have already computed
  a) The best tag sequence for <s> … bit that ends in NN.
  b) The best tag sequence for <s> … bit that ends in VBD.

Then, the best full sequence would be either
  – sequence (a) extended to include </s>, or
  – sequence (b) extended to include </s>.
But similarly, to get
  a) The best tag sequence for <s> … bit that ends in NN.

We could extend one of:
  – The best tag sequence for <s> … dog that ends in NN.
  – The best tag sequence for <s> … dog that ends in VB.

And so on…
Viterbi: high-level picture

- Intuition: the best path of length \( t \) ending in state \( q \) must include the best path of length \( t-1 \) to the previous state. (\( t \) now a \textit{time step}, not a \textit{tag}).
Viterbi: high-level picture

• Intuition: the best path of length $t$ ending in state $q$ must include the best path of length $t-1$ to the previous state. ($t$ now a *time step*, not a *tag*). So,
  
  – Find the best path of length $t-1$ to each state.
  – Consider extending each of those by 1 step, to state $q$.
  – Take the best of those options as the best path to state $q$. 

Notation

- Sequence of observations over time $o_1, o_2, \ldots, o_T$
  - here, words in sentence
- Vocabulary size $V$ of possible observations
- Set of possible states $q^1, q^2, \ldots, q^N$ (see note next slide)
  - here, tags
- $A$, an $N \times N$ matrix of transition probabilities
  - $a_{ij}$: the prob of transitioning from state $i$ to $j$. (Fig 8.5, slide 19 of prev. lect.)
- $B$, an $N \times V$ matrix of output probabilities
  - $b_i(o_t)$: the prob of emitting $o_t$ from state $i$. (Fig 8.6, slide 20 of prev. lect.)
Note on notation

- J&M use $q_1, q_2, \ldots, q_N$ for set of states, but *also* use $q_1, q_2, \ldots, q_T$ for state sequence over time.
  - So, just seeing $q_1$ is ambiguous (though usually disambiguated from context).
  - I’ll instead use $q_i$ for state names, and $q_t$ for state at time $t$.
  - So we could have $q_t = q_i$, meaning: the state we’re in at time $t$ is $q_i$. 
HMM example w/ new notation

- States \{q^1, q^2\} (or \{<s>, q^1, q^2\})
- Output alphabet \{x, y, z\}

Adapted from Manning & Schuetze, Fig 9.2
Transition and Output Probabilities

• Transition matrix $A$:
  
  $$a_{ij} = P(q^j | q^i)$$
  
<table>
<thead>
<tr>
<th></th>
<th>$q^1$</th>
<th>$q^2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$&lt;s&gt;$</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$q^1$</td>
<td>.7</td>
<td>.3</td>
</tr>
<tr>
<td>$q^2$</td>
<td>.5</td>
<td>.5</td>
</tr>
</tbody>
</table>

• Output matrix $B$:

  $$b_i(o) = P(o | q^i)$$

  for output $o$

<table>
<thead>
<tr>
<th></th>
<th>$x$</th>
<th>$y$</th>
<th>$z$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q^1$</td>
<td>.6</td>
<td>.1</td>
<td>.3</td>
</tr>
<tr>
<td>$q^2$</td>
<td>.1</td>
<td>.7</td>
<td>.2</td>
</tr>
</tbody>
</table>
Joint probability of (states, outputs)

• Let $\lambda = (A, B)$ be the parameters of our HMM.

• Using our new notation, given state sequence $Q = (q_1 ... q_T)$ and output sequence $O = (o_1 ... o_T)$, we have:

$$P(O, Q | \lambda) = \prod_{t=1}^{T} P(o_t|q_t)P(q_t|q_{t-1})$$
Joint probability of (states, outputs)

• Let \( \lambda = (A, B) \) be the parameters of our HMM.

• Using our new notation, given state sequence \( Q = (q_1 \ldots q_T) \) and output sequence \( O = (o_1 \ldots o_T) \), we have:

\[
P(O, Q|\lambda) = \prod_{t=1}^{T} P(o_t|q_t)P(q_t|q_{t-1})
\]

• Or:

\[
P(O, Q|\lambda) = \prod_{t=1}^{T} b_{q_t}(o_t) a_{q_{t-1}q_t}
\]
Joint probability of (states, outputs)

- Let $\lambda = (A, B)$ be the parameters of our HMM.

- Using our new notation, given state sequence $Q = (q_1 \ldots q_T)$ and output sequence $O = (o_1 \ldots o_T)$, we have:

  $$P(O, Q|\lambda) = \prod_{t=1}^{T} P(o_t|q_t)P(q_t|q_{t-1})$$

- Or:

  $$P(O, Q|\lambda) = \prod_{t=1}^{T} b_{q_t}(o_t) a_{q_{t-1}q_t}$$

- Example:

  $$P(O = (y, z), Q = (q^1, q^1)|\lambda) = b_1(y) \cdot b_1(z) \cdot a_{s>1} \cdot a_{11}$$

  $$= (.1)(.3)(1)(.7)$$
Viterbi: high-level picture

- Want to find $\arg\max_Q P(Q|O)$

- Intuition: the best path of length $t$ ending in state $q$ must include the best path of length $t-1$ to the previous state. So,
  - Find the best path of length $t-1$ to each state.
  - Consider extending each of those by 1 step, to state $q$.
  - Take the best of those options as the best path to state $q$. 
Viterbi algorithm

• Use a **chart** to store partial results as we go
  – **NxT** table, where $v(j,t)$ is the probability* of the best state sequence for $o_1...o_t$ that ends in state $j$.

*Specifically, $v(j,t)$ stores the max of the joint probability $P(o_1...o_t,q_1...q_{t-1},q_t=j|\lambda)$
Viterbi algorithm

• Use a **chart** to store partial results as we go
  – **NxT** table, where $v(j,t)$ is the probability* of the best state sequence for $o_1...o_t$ that ends in state $j$.

• Fill in columns from left to right, with

$$v(j, t) = \max_{i=1}^N v(i, t - 1) \cdot a_{ij} \cdot b_j(o_t)$$

*Specifically, $v(j,t)$ stores the max of the joint probability $P(o_1...o_t,q_1...q_{t-1},q_t=j|\lambda)$
Viterbi algorithm

• Use a **chart** to store partial results as we go
  – NxT table, where $v(j,t)$ is the probability* of the best state sequence for $o_1 \ldots o_t$ that ends in state $j$.

• Fill in columns from left to right, with

  \[ v(j, t) = \max_{i=1}^{N} v(i, t - 1) \cdot a_{ij} \cdot b_j(o_t) \]

• Store a **backtrace** to show, for each cell, which state at $t-1$ we came from.

*Specifically, $v(j,t)$ stores the max of the joint probability $P(o_1 \ldots o_t, q_1 \ldots q_{t-1}, q_t=j | \lambda)$
Example

- Suppose $0=xzy$. Our initially empty table:

<table>
<thead>
<tr>
<th></th>
<th>$o_1=x$</th>
<th>$o_2=z$</th>
<th>$o_3=y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q^1$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$q^2$</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Filling the first column

<table>
<thead>
<tr>
<th></th>
<th>$o_1 = x$</th>
<th>$o_2 = z$</th>
<th>$o_3 = y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q^1$</td>
<td>.6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$q^2$</td>
<td>0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

$v(1,1) = a_{<s>_1} \cdot b_1(x) = (1)(.6)$
$v(2,1) = a_{<s>_2} \cdot b_2(x) = (0)(.1)$
Starting the second column

\[
\begin{array}{|c|c|c|}
\hline
& o_1=x & o_2=z & o_3=y \\
\hline
q^1 & .6 & & \\
\hline
q^2 & 0 & & \\
\hline
\end{array}
\]

\[
v(1,2) = \max_{i=1}^{N} v(i, 1) \cdot a_{i1} \cdot b_1(z)
\]

\[
= \max \left\{ v(1,1) \cdot a_{11} \cdot b_1(z) = (.6)(.7)(.3) \\
v(2,1) \cdot a_{21} \cdot b_1(z) = (0)(.5)(.3) \right\}
\]
Starting the second column

<table>
<thead>
<tr>
<th></th>
<th>$o_1 = x$</th>
<th>$o_2 = z$</th>
<th>$o_3 = y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q^1$</td>
<td>.6</td>
<td>.126</td>
<td></td>
</tr>
<tr>
<td>$q^2$</td>
<td>0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[ v(1,2) = \max_{i=1}^N v(i, 1) \cdot a_{i1} \cdot b_1(z) \]

\[ = \max \left\{ v(1,1) \cdot a_{11} \cdot b_1(z) = (.6)(.7)(.3), \quad v(2,1) \cdot a_{21} \cdot b_1(z) = (0)(.5)(.3) \right\} \]
Finishing the second column

<table>
<thead>
<tr>
<th></th>
<th>$o_1 = x$</th>
<th>$o_2 = z$</th>
<th>$o_3 = y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q^1$</td>
<td>.6</td>
<td>.126</td>
<td></td>
</tr>
<tr>
<td>$q^2$</td>
<td>0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

$v(2, 2) = \max_{i=1}^{N} v(i, 1) \cdot a_{i2} \cdot b_2(z)$

$$= \max \begin{cases} v(1, 1) \cdot a_{12} \cdot b_2(z) = (.6)(.3)(.2) \\ v(2, 1) \cdot a_{22} \cdot b_2(z) = (0)(.5)(.2) \end{cases}$$
Finishing the second column

<table>
<thead>
<tr>
<th></th>
<th>$o_1=x$</th>
<th>$o_2=z$</th>
<th>$o_3=y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q^1$</td>
<td>.6</td>
<td>.126</td>
<td></td>
</tr>
<tr>
<td>$q^2$</td>
<td>0</td>
<td>.036</td>
<td></td>
</tr>
</tbody>
</table>

$$v(2,2) = \max_{i=1}^N v(i, 1) \cdot a_{i2} \cdot b_2(z)$$

$$= \max \left\{ v(1,1) \cdot a_{12} \cdot b_2(z) = (.6)(.3)(.2) \\
v(2,1) \cdot a_{22} \cdot b_2(z) = (0)(.5)(.2) \right\}$$
• Exercise: make sure you get the same results!

<table>
<thead>
<tr>
<th></th>
<th>$o_1=x$</th>
<th>$o_2=z$</th>
<th>$o_3=y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q^1$</td>
<td>.6</td>
<td>.126</td>
<td>.00882</td>
</tr>
<tr>
<td>$q^2$</td>
<td>0</td>
<td>.036</td>
<td>.02646</td>
</tr>
</tbody>
</table>
Best Path

<table>
<thead>
<tr>
<th></th>
<th>0₁=x</th>
<th>0₂=z</th>
<th>0₃=y</th>
</tr>
</thead>
<tbody>
<tr>
<td>q¹</td>
<td>.6</td>
<td>.126</td>
<td>.00882</td>
</tr>
<tr>
<td>q²</td>
<td>0</td>
<td>.036</td>
<td>.02646</td>
</tr>
</tbody>
</table>

- Choose best final state: \( \max_{i=1}^{N} v(i,T) \)
- Follow backtraces to find best full sequence: \( q¹q¹q² \)
HMMs: what else?

• Using Viterbi, we can find the best tags for a sentence (decoding), and get $P(O, Q|\lambda)$.

• We might also want to
  – Compute the likelihood $P(O|\lambda)$, i.e., the probability of a sentence regardless of tags (a language model!)
  – learn the best set of parameters $\lambda = (A, B)$ given only an unannotated corpus of sentences.
Computing the likelihood

• From probability theory, we know that

\[ P(O|\lambda) = \sum_{Q} P(O,Q|\lambda) \]

• There are an exponential number of Qs.

• Again, by computing and storing partial results, we can solve efficiently.

• (Next slides show the algorithm but I’ll likely skip them)
Forward algorithm

• Use a table with cells $\alpha(j,t)$: the probability of being in state $j$ after seeing $o_1...o_t$ (forward probability).

$$\alpha(j, t) = P(o_1, o_2, ... o_t, q_t = j | \lambda)$$

• Fill in columns from left to right, with

$$\alpha(j, t) = \sum_{i=1}^{N} \alpha(i, t - 1) \cdot a_{ij} \cdot b_j(o_t)$$

  – Same as Viterbi, but sum instead of max (and no backtrace).
**Example**

- Suppose $O=xzy$. Our initially empty table:

<table>
<thead>
<tr>
<th></th>
<th>$o_1=x$</th>
<th>$o_2=z$</th>
<th>$o_3=y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q^1$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$q^2$</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Filling the first column

$$
\begin{array}{c|c|c|c|}
 & o_1=x & o_2=z & o_3=y \\
\hline
q^1 & .6 & & \\
\hline
q^2 & 0 & & \\
\end{array}
$$

\[
\alpha(1,1) = a_{<s>1} \cdot b_1(x) = (1)(.6) \\
\alpha(2,1) = a_{<s>2} \cdot b_2(x) = (0)(.1)
\]
Starting the second column

<table>
<thead>
<tr>
<th></th>
<th>$o_1=x$</th>
<th>$o_2=z$</th>
<th>$o_3=y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q^1$</td>
<td>0.6</td>
<td>0.126</td>
<td></td>
</tr>
<tr>
<td>$q^2$</td>
<td>0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

$$\alpha(1,2) = \sum_{i=1}^{N} \alpha(i, 1) \cdot a_{i1} \cdot b_1(z)$$

$$= \alpha(1,1) \cdot a_{11} \cdot b_1(z) + \alpha(2,1) \cdot a_{21} \cdot b_1(z)$$

$$= (0.6)(0.7)(0.3) + (0)(0.5)(0.3)$$

$$= 0.126$$
Finishing the second column

<table>
<thead>
<tr>
<th>q^1</th>
<th>o_1=x</th>
<th>o_2=z</th>
<th>o_3=y</th>
</tr>
</thead>
<tbody>
<tr>
<td>.6</td>
<td>.126</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td>.036</td>
<td></td>
</tr>
</tbody>
</table>

\[ \alpha(2,2) = \sum_{i=1}^{N} \alpha(i, 1) \cdot a_{i2} \cdot b_{2(z)} \]

\[ = \alpha(1,1) \cdot a_{12} \cdot b_{2(z)} + \alpha(2,1) \cdot a_{22} \cdot b_{2(z)} \]

\[ = (.6)(.3)(.2) + (0)(.5)(.2) \]

\[ = .036 \]
Third column and finish

<table>
<thead>
<tr>
<th></th>
<th>o₁=x</th>
<th>o₂=z</th>
<th>o₃=y</th>
</tr>
</thead>
<tbody>
<tr>
<td>q¹</td>
<td>.6</td>
<td>.126</td>
<td>.01062</td>
</tr>
<tr>
<td>q²</td>
<td>0</td>
<td>.036</td>
<td>.03906</td>
</tr>
</tbody>
</table>

- Add up all probabilities in last column to get the probability of the entire sequence:

\[ P(O|\lambda) = \sum_{i=1}^{N} \alpha(i, T) \]
Learning

• Given *only* the output sequence, learn the best set of parameters $\lambda = (A, B)$.

• Assume ‘best’ = maximum-likelihood.

• Other definitions are possible, won’t discuss here.
Unsupervised learning

• Training an HMM from an annotated corpus is simple.
  – **Supervised** learning: we have examples labelled with the right ‘answers’ (here, tags): no hidden variables in training.

• Training from unannotated corpus is trickier.
  – **Unsupervised** learning: we have no examples labelled with the right ‘answers’: all we see are outputs, state sequence is hidden.
Circularity

• If we know the state sequence, we can find the best $\lambda$.
  – E.g., use MLE: $P(q^j|q^i) = \frac{C(q_i \rightarrow q_j)}{C(q^i)}$

• If we know $\lambda$, we can find the best state sequence.
  – use Viterbi

• But we don't know either!
Expectation-maximization (EM)

Essentially, a bootstrapping algorithm.

- Initialize parameters $\lambda^{(0)}$

- At each iteration $k$,
  - E-step: Compute **expected counts** using $\lambda^{(k-1)}$
  - M-step: Set $\lambda^{(k)}$ using MLE on the expected counts

- Repeat until $\lambda$ doesn't change (or other stopping criterion).
Expected counts??

Counting transitions from $q^i \rightarrow q^j$:

- **Real counts:**
  - count 1 each time we see $q^i \rightarrow q^j$ in true tag sequence.

- **Expected counts:**
  - With current $\lambda$, compute probs of all possible tag sequences.
  - If sequence $Q$ has probability $p$, count $p$ for each $q^i \rightarrow q^j$ in $Q$.
  - Add up these fractional counts across all possible sequences.
Example

- Notionally, we compute expected counts as follows:

<table>
<thead>
<tr>
<th>Possible sequence</th>
<th>Probability of sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q₁ = q₁ q₁ q₁</td>
<td>p₁</td>
</tr>
<tr>
<td>Q₂ = q₁ q₂ q₁</td>
<td>p₂</td>
</tr>
<tr>
<td>Q₃ = q₁ q₁ q₂</td>
<td>p₃</td>
</tr>
<tr>
<td>Q₄ = q₁ q₂ q₂</td>
<td>p₄</td>
</tr>
</tbody>
</table>

Observs: x z y
Example

- Notionally, we compute expected counts as follows:

<table>
<thead>
<tr>
<th>Possible sequence</th>
<th>Probability of sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Q_1 = q^1 q^1 q^1$</td>
<td>$p_1$</td>
</tr>
<tr>
<td>$Q_2 = q^1 q^2 q^1$</td>
<td>$p_2$</td>
</tr>
<tr>
<td>$Q_3 = q^1 q^1 q^2$</td>
<td>$p_3$</td>
</tr>
<tr>
<td>$Q_4 = q^1 q^2 q^2$</td>
<td>$p_4$</td>
</tr>
</tbody>
</table>

Observs: $x$, $z$, $y$

$$\hat{C}(q^1 \rightarrow q^1) = 2p_1 + p_3$$
Forward-Backward algorithm

• As usual, avoid enumerating all possible sequences.

• **Forward-Backward** (Baum-Welch) algorithm computes expected counts using forward probabilities and **backward probabilities**:

$$\beta(j, t) = P(q_t = j, o_{t+1}, o_{t+2}, \ldots o_T | \lambda)$$

  – Details, see J&M 6.5

• EM idea is much more general: can use for many latent variable models.
Guarantees

- EM is guaranteed to find a **local** maximum of the likelihood.
Guarantees

- EM is guaranteed to find a local maximum of the likelihood.

- Not guaranteed to find global maximum.

- Practical issues: initialization, random restarts, early stopping.
Summary

• HMM: a generative model of sentences using hidden state sequence

• Dynamic programming algorithms to compute
  – Best tag sequence given words (Viterbi algorithm)
  – Likelihood (forward algorithm)
  – Best parameters from unannotated corpus (forward-backward algorithm, an instance of EM)
Announcements

• Assignment 1 is now out.

• If you are not yet registered for the class, make sure you either
  – Have already added your ID# to partner signup
  – Email Henry NOW with your name and ID#.
  – Otherwise he won’t know to assign you a partner!