Hidden Markov Models

Transition Matrix

\[
\begin{bmatrix}
0.5 & 0.2 \\
0.2 & 0.2 \\
0.2 & 0.3 \\
\end{bmatrix}
\]

Emission Probabilities

\[
\begin{array}{c|cc}
        & 1 & 2 \\
\hline
1 & 0.5 & 0.5 \\
2 & 0.3 & 0.3 \\
E & 0.2 & 0.2 \\
\end{array}
\]

\[
\begin{array}{c|c|c}
\text{observe } & A & B \\
\hline
1 & A & \frac{1}{3} \\
2 & A & \frac{2}{3} \\
E & A & \frac{2}{3} \\
\end{array}
\]

\[
\begin{array}{c|c|c}
\alpha & \frac{1}{2} & \frac{1}{2} \\
\hline
\beta & \frac{1}{2} & \frac{1}{2} \\
\end{array}
\]

\[
\begin{array}{c|c|c}
\gamma & \frac{1}{2} & \frac{1}{2} \\
\hline
\kappa & \frac{1}{2} & \frac{1}{2} \\
\end{array}
\]

\[
\alpha(i, t) = \text{total mass of all paths that get to state } i \text{ at time } t \text{ and emit symbol } w_t, \text{ up to and including the emission}
\]

\[
\beta(i, t) = \text{total mass of all paths that leave state } i \text{ at time } t, \text{ not including the emission at time } t
\]

\[
\alpha(1, 2) = \left(\frac{3}{8} \frac{1}{2} + \frac{1}{8} \frac{1}{4}\right) \frac{3}{4} = \frac{21}{24}
\]

\[
\beta(1, 1) = \frac{1}{2} \frac{3}{4} \frac{1}{4} + \frac{1}{4} \frac{1}{4} \frac{1}{4} = \frac{7}{24}
\]

\[
\alpha(2, 2) = \left(\frac{3}{8} \frac{1}{4} + \frac{1}{8} \frac{1}{2}\right) \frac{1}{4} = \frac{5}{24}
\]

\[
\beta(2, 1) = \frac{1}{4} \frac{3}{4} \frac{1}{4} + \frac{1}{2} \frac{1}{4} \frac{1}{4} = \frac{5}{24}
\]
\[ \alpha(E, 0) = 1 \]
\[ \alpha(i, t+1) = \left( \sum_j \beta(j, t) \cdot P(i \rightarrow j) \right) \cdot P(i \Rightarrow \omega_{t+1}) \]
\[ \beta(E, t+1) = 1 \]
\[ \beta(i, t-1) = \sum_j P(i \rightarrow j) \cdot P(j \Rightarrow \omega_t) \cdot \beta(j, t) \]

**NOTE**

\[ \alpha(E, t+1) = \beta(E, 0) = Z \text{ total sequence probability} \]
Also for any \( t \), \( \sum \alpha_{itf} = Z \), b/c all paths must go through exactly one state at time \( t \) (great for debugging EM!)

**Updates**

\[ E \text{ count of } 1 \rightarrow 1 \text{ at time } t = \frac{\alpha(1, 1) \cdot P(1 \rightarrow 1) \cdot P(1 \Rightarrow \omega_t) \cdot \beta(1, 2)}{Z} \]
\[ E \text{ count of } 2 \rightarrow H \text{ at time } t = \frac{\alpha(2, 1) \cdot \beta(2, 1)}{Z} \]

For EM, just sum all these \( E \) counts over all times in all sequences, normalize, and that's it.

**NOTE**

\[ E(1 \rightarrow 1) = \frac{9}{28} > \frac{9}{13} = \frac{9}{13} > \frac{1}{2} \text{ so the transition } 1 \rightarrow 1 \text{ is "encouraged" to increase} \]
Decoding

Viterbi:

\[ V(i, t) = \text{best path through } i \text{ at time } t \]

\[ = \text{best path from } \arg \max_j V(j, t-1) \cdot \text{prob } p(j \rightarrow i) \cdot \text{prob } p(i \rightarrow w) \]

Max Prob Tag

for time \( t \), pick \( \arg \max_i \text{fit} \)

Can lead to different results from Viterbi; often better fit to real world needs (e.g. # of correct tags)
A Rainbow of HMMs

Continuous Outputs
- Speech Recognition → FFT snippets
- OCR → like MNIST, but w/ a Language Model
- Stocks → use generative/predictive power

Hidden Semi-Markov Models
- HMM assumes itself transition R.V is geometrically distributed
- HSMM uses arbitrary different distribution, but can't use normal FB algorithm

Sticky HMM's encourage self transitions through bias in the prior - important for HDP-HMM's, which have potentially ∞ # of states

Used in Unsupervised POS tagging

Consider

Tok "That's what she said" she said.
POS "DT VBZ WP PRP UBD PP PRP UBD ."
also, we get 2 classes of "the", because of different noun types.

Tricks to improve performance:

Condition on more nodes (Tri/i, Bi/i, Uni/i smoothing)

Use Morphology to figure out rare words
i.e., if "running", "swimming", etc. are all in a class, then spell-checking should be too, even without much context info.

Posterior Regularization — how do you constrain one word to be generated by a small # of states?