SFU NatLangLab

CMPT-413: Computational Linguistics

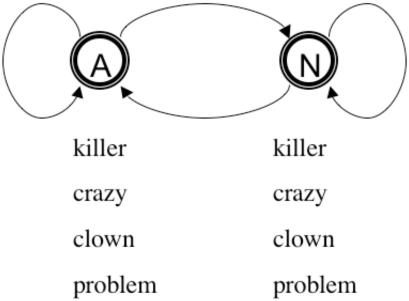
HMM5: Language Models and Hidden Markov Models

Anoop Sarkar
http://www.cs.sfu.ca/∼anoop

February 28, 2013

Hidden Markov Model

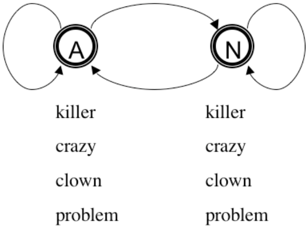
$$\text{Model } \theta = \begin{cases} \pi_i & \text{probability of starting at state } i \\ a_{i,j} & \text{probability of transition from state } i \text{ to state } j \\ b_i(o) & \text{probability of output } o \text{ at state } i \end{cases}$$



Hidden Markov Model Algorithms

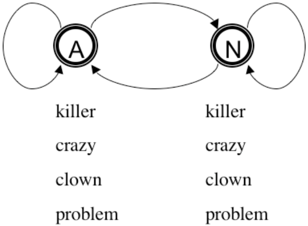
- HMM as parser: compute the best sequence of states for a given observation sequence.
- HMM as language model: compute probability of given observation sequence.
- HMM as learner: given a corpus of observation sequences, learn its distribution, i.e. learn the parameters of the HMM from the corpus.
  - Learning from a set of observations with the sequence of states provided (states are not hidden) [Supervised Learning]
  - Learning from a set of observations without any state information. [Unsupervised Learning]

HMM as Language Model



- Find $P(\textit{killer clown}) = \sum_y P(y, \textit{killer clown})$
- $P(\textit{killer clown}) = P(AA, \textit{killer clown}) + P(AN, \textit{killer clown}) + P(NN, \textit{killer clown}) + P(NA, \textit{killer clown})$

HMM as Language Model



- Consider the input *killer crazy clown problem*
- So the task is to find the sum over all sequences of states:

$$\sum_{s_1, s_2, s_3, s_4} P(\textit{killer crazy clown problem}, s_1, s_2, s_3, s_4)$$

- A sub-problem is to find the most likely sequence of states for *killer crazy clown*:

$$\sum_{s_1, s_2, s_3} P(\textit{killer crazy clown}, s_1, s_2, s_3)$$

HMM as Language Model

- In our example there are two possible values for s_4:

$$\sum_{s_1,\ldots,s_4} P(\textit{killer crazy clown problem}, s_1, s_2, s_3, s_4) =$$
$$\sum_{s_1,s_2,s_3} P(\textit{killer crazy clown problem}, s_1, s_2, s_3, N) +$$
$$\sum_{s_1,s_2,s_3} P(\textit{killer crazy clown problem}, s_1, s_2, s_3, A)$$

- Very similar to the Viterbi algorithm. Sum instead of max, and that's the only difference!

HMM as Language Model

- Provide an index for each input symbol:
  1:killer 2:crazy 3:clown 4:problem

$$V[N,3] = \sum_{s_1,s_2} P(\textit{killer crazy clown}, s_1, s_2, N)$$

$$V[N,4] = \sum_{s_1,s_2,s_3} P(\textit{killer crazy clown problem}, s_1, s_2, s_3, N)$$

- Putting them together:

$$\begin{aligned} V[N,4] &= V[N,3] \cdot a_{N,N} \cdot b_N(\textit{problem}) + \\ &\quad V[A,3] \cdot a_{A,N} \cdot b_N(\textit{problem}) \end{aligned}$$

$$\begin{aligned} V[A,4] &= V[N,3] \cdot a_{N,A} \cdot b_A(\textit{problem}) + \\ &\quad V[A,3] \cdot a_{A,A} \cdot b_A(\textit{problem}) \end{aligned}$$

- The best score for the input is given by: $V[N,4] + V[A,4]$

HMM as Language Model

- For input of length T: $o_1, \ldots, o_T$, we want to find $P(o_1, \ldots, o_T) = \sum_{y_1, \ldots, y_T} P(y_1, \ldots, y_T, o_1, \ldots, o_T)$
- Each y_t in this sequence is one of the states in the HMM.
- For each state q we initialize our table: $V[q, 1] = \pi_q \cdot b_q(o_1)$
- Then compute recursively for $t = 1 \ldots T-1$ for each state q:

$$V[q, t+1] = \sum_{q'} \left\{ V[q', t] \cdot a_{q',q} \cdot b_q(o_{t+1}) \right\}$$

- After the loop terminates, the best score is $\sum_q V[q, T]$
- So: Viterbi with sum instead of max gives us an algorithm for HMM as a language model.
- This algorithm is sometimes called the *forward algorithm*.