Hw6: Beam search

Q1: Beam search

- format: beamsearch_maxent.sh test_data boundary_file model_file syst_output beam_size topN topK
- test_data: the format is "instanceName goldClass f1 v1 f2 v2 ..."
  - This includes words from all the test sentences.
  - prevT=xx and prevTwoTags=xx+yy are NOT in the feature vector. You need to add them on the fly.
- boundary_file: length of each sentence so you know when to end a beam search
- model_file: MaxEnt model in text format
- sys_output: instanceName goldClass sysClass prob
  - instanceName and goldClass are copied from test_data
  - sysClass is the y for x according to the best sequence for the sentence
  - Prob is P(y | x): y is the tag, x is the word.
- beam_size, topN, and topK: see slide #5-6

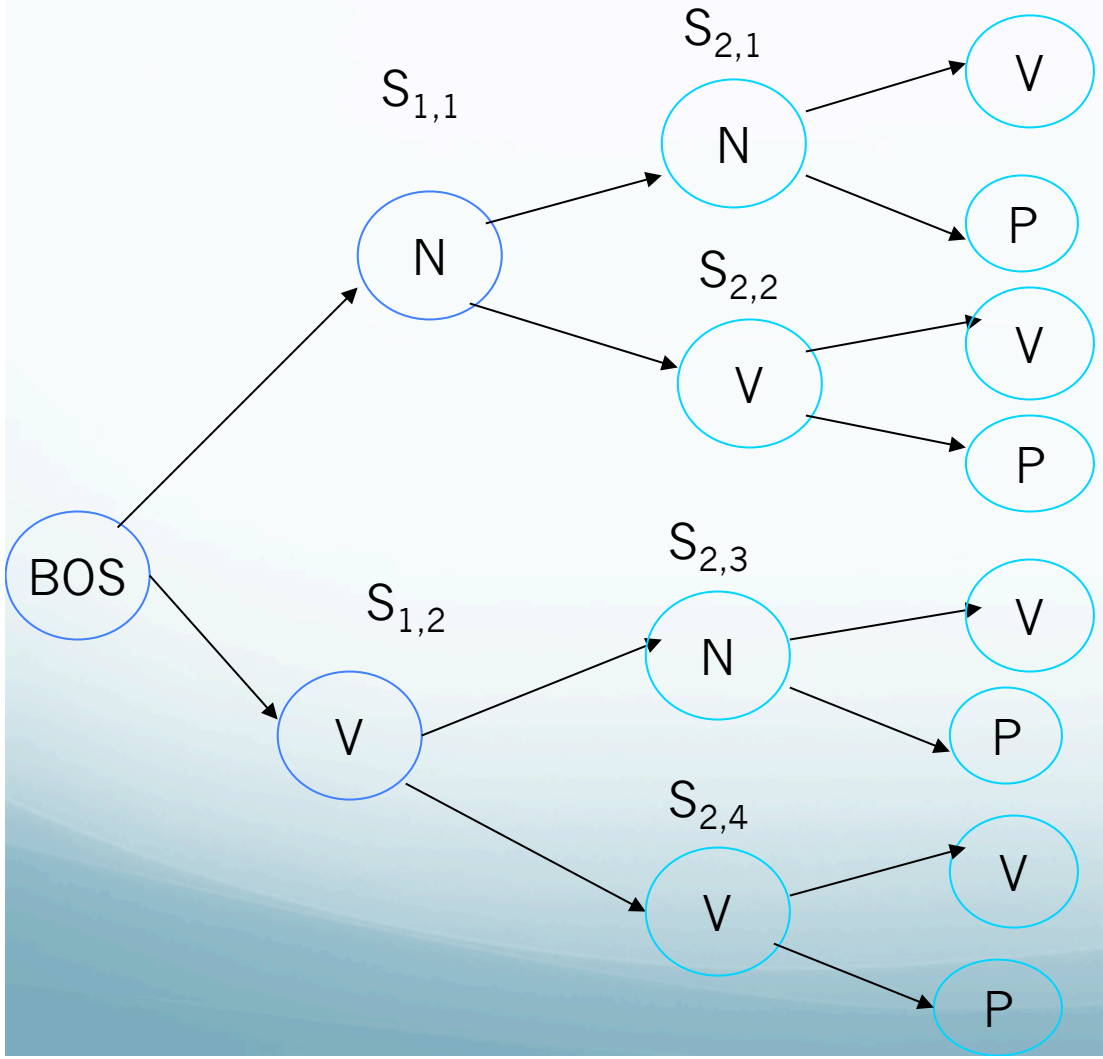
- Remember to add prevT=tag and prevTwoTags=tag_{-2}+tag_{-1} before calculating P(y | x).

- Features:
  - tag_{-2} is the tag of w_{-2} and tag_{-1} is the tag of w_{-1}
  - For the list of tags, see prevT=tag in the model file
  - If the model file does not have weights for those feature functions, that means the weights for them are zero.

- Test your code with small data files first.

Beam search

time flies like an arrow



Beam search

Parameters: topN, topK, beam_size

(1) Get topN tags for w_1 and form nodes $s_{1,j}$

(2) For i=2 to n (n is the sentence length)
For each surviving node $s_{i-1,j}$
form the vector for w_i
get topN tags for w_i and
form new nodes
Prune nodes at position i

(3) Pick the node at position n with highest prob

Pruning at Position i

Each node at Position i should store a tag for w_i and a prob, where the prob is $\prod_{k=1}^{i} P(t_k|h_k)$.

Let max_prob be the highest prob among the nodes at Position i

For each node $s_{i,j}$ at Position i
    Let $prob_{i,j}$ be the probability stored at the node
    keep the node iff $prob_{i,j}$ is among the topK of the nodes
        and $lg(prob_{i,j}) +$ beam_size $\geq lg(max\_prob)$

Note: sys_output includes P(tag|word), so at each node you should also store: $P(t_k|h_k)$