

# Hw6: beam search

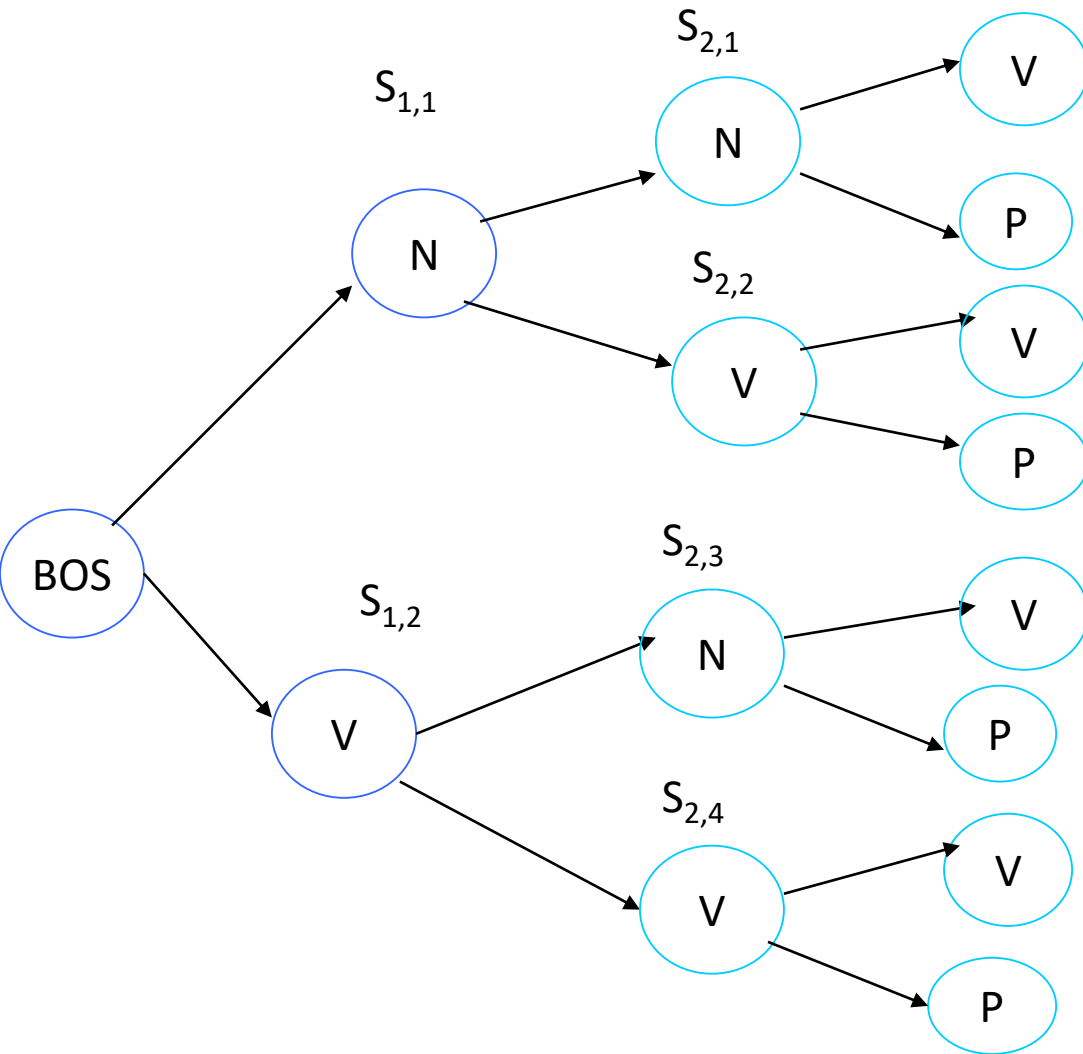
# Q1: Beam search

- 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  $\text{prevT}=\text{tag}$  and  $\text{prevTwoTags}=\text{tag}_{-2}+\text{tag}_{-1}$  before calculating  $P(y \mid x)$ .
- Features:
  - $\text{tag}_{-2}$  is the tag of  $w_{-2}$  and  $\text{tag}_{-1}$  is the tag of  $w_{-1}$
  - For the list of tags, see  $\text{prevT}=\text{tag}$  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  $lg(prob_{i,j}) + \text{beam\_size} \geq lg(max\_prob)$

Note: `sys_output` includes  $P(\text{tag} | \text{word})$ , so at each node you should also store  $P(t_k | h_k)$ .