Estimation for Lexicalised PCFGs

ACS Introduction to NLP
Stephen Clark
Estimation for PCFGs

- Easy!

\[ \hat{P}(RHS|LHS) = \frac{f(LHS \rightarrow RHS)}{f(LHS)} \]

where \( f(LHS \rightarrow RHS) \) is the number of times \( LHS \) rewrites as the \( RHS \) in a treebank, and \( f(LHS) \) is the total number of times \( LHS \) is rewritten as anything

- These relative frequency estimates can be justified as maximum likelihood estimates:

\[ \hat{P} = \arg \max_P \prod_{i=1}^n \prod_{j=1}^m P(RHS^i_j|LHS^i_j) \]

where \( LHS^i_j \rightarrow RHS^i_j \) is the \( j \)th rule application in the \( i \)th training example (Collins has a proof of this)
Smoothing for Lexicalised PCFGs

- The grammar Collins uses is (roughly speaking) a lexicalised PCFG (I say roughly speaking because of the Markov process generating the subcat frames)

- Lexicalised PCFGs can be thought of as PCFGs with much larger sets of non-terminal symbols (the standard non-terminals embellished with lexical items)

- So relative frequency estimation isn't going to work (many combinations of LHS's and RHS's won’t appear in the data)
Backoff and Interpolation

• Backoff levels for $p_h(H|P,w,t)$ where $H$ is the head category, $P$ is the parent, $w$ is the head word associated with the head category, and $t$ is the pos tag of the head word

  $- p_h(H|P,w,t)$
  $- p_h(H|P,t)$
  $- p_h(H|P)$

• Use a linear combination of these (linear interpolation):

  $\tilde{p}_h(H|P,w,t) = \lambda_1\hat{p}_h(H|P,w,t) + \lambda_2\hat{p}_h(H|P,t) + \lambda_3\hat{p}_h(H|P)$

  $\lambda_i \geq 0, \sum_i \lambda_i = 1$
Setting the Lambdas

- A neat way to set the values of the $\lambda$s based on the *diversity*:

$$\lambda_i = \frac{f_i}{f_i + 5u_i}$$

where $f_i$ is the number of times we’ve seen the denominator from the relative frequency estimate and $u_i$ is the number of unique outcomes in the distribution (see p.185 of Collins’ thesis); and 5 is set empirically
More Backoff and Interpolation

- \( p_L(L_i(lw_i, lt_i)|P, H, w, t, LC) \) where \( L_i(lw_i, lt_i) \) is a left complement consisting of non-terminal \( L_i \), word \( lw_i \), and pos tag \( lt_i \); \( P \) is the parent category; \( H \) is the category of the head; \( w \) is the head word; \( t \) is the pos tag of the head word, and \( LC \) is the left subcat frame

\[
p_L(L_i(lw_i, lt_i)|P, H, w, t, LC) = p_L(L_i(lt_i)|P, H, w, t, LC) \times p_L(lw_i|L_i, lt_i, P, H, w, t, LC)
\]
More Backoff and Interpolation

- $p_L(L_i(l_i)|P, H, w, t, LC')$ where $L_i(l_i)$ is a left complement, $P$ is the parent category, $H$ is the category of the head, $w$ is the head word, $t$ is the pos tag of the head word, and $LC'$ is the left subcat frame

  - $p_L(L_i(l_i)|P, H, t, LC')$
  - $p_L(L_i(l_i)|P, H, t, LC')$
  - $p_L(L_i(l_i)|P, H, LC')$

$$p_L(L_i(l_i)|P, H, t, LC') = \lambda_1 p_L(L_i(l_i)|P, H, t, LC') + \lambda_2 p_L(L_i(l_i)|P, H, LC') + \lambda_3 p_L(L_i(l_i)|P, H, LC')$$
Dealing with Unknown Words

• All words occurring less than 5 times in the training data, and all words in test data never seen in training, are replaced with an “UNKNOWN” token

• Question: why does this work?
  – we’re replacing a rare word with “UNKNOWN” (which is now quite common!)
  – so the joint model isn’t very accurate at generating rare words? (over-estimates their probabilities)
  – why isn’t this a problem?
Distance

- All Collins’ models have “distance” parameters which improve the results
- I’ve ignored them only because they clutter the equations further and adding them as extra parameters is not complicated
Results

- Model 1 achieves 87.5/87.7 LP/LR on WSJ section 23 according to the Parseval measures

- Model 2 achieves 88.1/88.3 LP/LR

- Current best scores on this task are around 91 (eg Charniak and Johnson (2005), Coarse-to-fine n-best parsing and MaxEnt discriminative reranking)