ACS Syntax and Semantics of Natural Language

Lecture 7: CCG Supertagging

Stephen Clark
Natural Language and Information Processing (NLIP) Group
sc609@cam.ac.uk
The first stage in the CCG parsing pipeline is to assign CCG lexical categories to the words in the sentence.

This process is known as *supertagging*, since the labels being assigned are detailed syntactic structures.

Srinivas and Joshi (1998) also called this process *almost parsing*, since the detailed labels mean that, once the supertagging has been performed, there is less work for the parser to do.

- Srinivas and Joshi used Lexicalised Tree Adjoining Grammar (LTAG), in which the ‘labels’ are LTAG elementary trees, but the principle is the same.
• We can apply any sequence labelling method to the supertagging task, e.g. HMMs, CRFs, maximum entropy models,...

• But the task is much harder than PTB pos tagging, since the label set is typically an order of magnitude larger.

• We will use a maximum entropy tagger (Ratnaparkhi, 1996), because of its ability to incorporate a large amount of contextual information for disambiguation.
Why Supertagging is Hard

- Categories in blue are all for prepositions
- Need to distinguish between complements and adjuncts, as well as make attachment decisions
  - PP attachment is known to be one of the hardest parsing sub-problems
How Hard is CCG Supertagging?

- Over 500 labels in the grammar used by the C&C parser, compared with around 50 PTB pos tags

- Useful baseline: for each word in the test set, assign the label most frequently seen with that word in the training data (and for unknown words assign N, say)

- For PTB pos tagging, this baseline is around 90%

- For CCG supertagging, this baseline is around 72%
The Need for CCG Multi-tagging

- Assigning a single category to each word (using Viterbi) results in around 92% per-word accuracy (using a standard MaxEnt tagger with a 5-word window, with words and pos tags as features)
  - around 2-3 mistakes per sentence!
- 92% is not accurate enough for reliable parsing
- We need to allow the supertagger to assign more than one category when the context cannot reliably disambiguate
\[ P(\text{category}|\text{sentence}) = \sum_{S} P(\text{category}, S|\text{sentence}) \]

where \( S \) ranges over all lexical category sequences for the sentence

- The *Forward-Backward Algorithm* is a DP algorithm for efficiently performing this sum
- Assign all categories to a word whose probability is greater than some dynamic threshold:

  assign category \( C \) if \( P(C|\text{sentence}) > \beta \cdot P(C_{\text{max}}|\text{sentence}) \)

where \( C_{\text{max}} \) is the category with the highest probability for that word
The number of categories assigned to each word on average is a crucial factor in the speed of the parser.

The following “adaptive” strategy has been found to work very well:

- Start with a high $\beta$ value/low ambiguity
- If the parser fails to find an analysis, decrease $\beta$
- Repeat until spanning analysis is found (success) or parsing is taking too long (failure)

See Section 10.3 of Clark and Curran (2007) for experimental details of the adaptive strategy.


• Stephen Clark and James R. Curran. The Importance of Supertagging for Wide-Coverage CCG Parsing. Proceedings of the 20th International Conference on Computational Linguistics (COLING-04), Geneva, Switzerland, 2004