Tutorial 12: Statistical language models

Some more on grammars

1. Give three examples of each of the following word classes:
   (a) count noun
   (b) mass noun
   (c) proper noun
   (d) determiner
   (e) preposition
   (f) intransitive verb
   (g) transitive verb
   (h) ditransitive verb
2. Quiz: Give three examples of each of the following phrase types:
   (a) noun phrase
   (b) verb phrase
   (c) prepositional phrase

Linguistic ambiguity

3. What kind of ambiguity do you think is present in each of the following sentences?
   (a) *Rugby team's coach set on fire.*
   (b) *The robber tickled the man under the chin.*
   (c) *Toni Morisson won a pullet surprise in 1989.*
   (d) *All that glitters is not gold.*

NLTK: practice with a grammar using variables

4. The file `feature_grammar.fcfg` holds the grammar we discussed in Lecture 22, that uses variables for syntactic agreement. Download this file into a directory `[dir]`. Write down some sentences which you think are covered by this grammar.

5. The file `feature_grammar_parser.py` holds python code for parsing sentences using the feature grammar. Download that file into the same directory. Then change the `load_parser` function so that it points to the feature grammar file.

   (a) Try parsing the sentences you thought the grammar could cover. Were you right? If not, try again to find some sentences that can be parsed using the grammar. (You might find it useful to turn on trace in the parse function; see the arguments of the `load_parser` function.)

   (b) Find two ways in which the grammar can generate an infinitely long sentence.

   (c) **Quiz (part 1):** Add a rule to the grammar that allows all nouns (N) to be modified by PPs. The rule should allow nouns like `cat on the table`. (And thus NPs like `the cat on the table`.)

   (d) Try parsing a sentence that includes a noun modified by a PP. You should find the sentence is *syntactically ambiguous*. Explain how the ambiguity arises.

   (e) **Quiz (part 2):** The grammar leaks in a couple of places. One problem is that it admits sentences like *I love she*. In English, there should be two kinds of pronoun: those that can occur in *subject* position (e.g. *I*, *she*) and those that can occur in *object* position (e.g. *me*, *her*). Update the grammar to add this distinction, so that it admits *I love her*, but not *her loves I*.

Linear probabilistic language models

6. n-gram models

   (a) Explain how an n-gram model can be trained to predict the next word in a corpus of text.

   (b) What problems arise if we build an n-gram model with a small n? (E.g. $n = 1$ or 2?)

   (c) What problem arises if we build an n-gram model with a large n?

   (d) How can an n-gram model be used to resolve word sense ambiguities?

7. n-grams with NLTK
   Load the file `pride_and_prejudice.txt` into a directory `[dir]`. Open the file `ngrams_demo.py` in Canopy. This file contains some functions that create a simple n-gram model of a text. (You will need to set `Absolute_Path_Name` to `[dir]` before you can run it.)

   (a) Run `ngrams_demo.py`, and get a feel for how n-grams are used to build a conditional probability distribution for the next word given a sequence of words.

   (b) Write a function `generate_text(cpd, n)` that takes a conditional probability distribution `cpd` and generates a sequence of `n` words sampled from this distribution.

8. Linear models of language with neural networks

   (a) What is an **Elman network**? How does it differ from a regular back-prop network? Sketch it, and explain how it is trained.

   (b) How is the language model learned by an Elman network similar to an n-gram model? How does it differ from an n-gram model?