Homework 5

Statistical NLP

Due: April 20

1 Lexical Preferences

One way of determining whether verbs have similar meaning is to determine whether the verbs take the same kinds of arguments. Your task is to implement a simple variant of Resnik's lexical selection system by extracting from the parsed Penn Treebank corpus the verbs and their common-noun subjects (leave out names) (anything in this kind of configuration:
`(S  (NP ... (N..  noun)) (VP ... (V.. verb) ...) ...)` should be fine). Calculate probability distribution for each verb of p(noun|verb). Calculate the selectional preference strength of the verb by computing the total KL divergence between the noun's unigram distribution p(noun) (as subject) and p(noun|verb) for all nouns. (equation 8.28). List the verbs in order of their decreasing selectional preference.

2 Word Sense Disambiguation

There is a file `partyNNannotations.cvs` shared with you. This contains 175 sentences containing the ambiguous target word *party*, which has three meanings: political, festivity, and entity (*the ruling party*, *a birthday party*, *an interested party*). WordNet lists five senses (subdividing these a bit more). But let's keep it simple, and number these senses 1 (political) 2 (festivity) and 3 (entity) senses. Your first task is to disambiguate the word *party) in these sentences, adding your annotations to this file (I've left two columns open for annotators to put annotations in). I've divided up the sentences so each sentence is annotated by two of you. Once the annotations are done, you should compute the Cohen kappa-score for the sentences that you annotated and for the entire annotation as a whole. This is to be turned in.*

We now have double annotations for all sentences in our training corpus. Retain for training only those sentences that have the same double annotation (where there is agreement). From each of the sets of 25 sentences that you annotated with someone chose two sentence that had double agreement **at random** *(really: in python use `int(random.random()*24` if you have 24 sentences with double agreement). You and your partner have to make the choice jointly (don't duplicate). Copy these into the file `test.cvs` for use in testing.*

Build a Naive Bayesian model, making use of the following features: word before, word after, word two words before, word two after. (For each sentence this involves extracting these four features and the annotation - you can do this by hand or build a program to do it. If the target word party *appears near the end of the word, use the symbols* **BEG** *and* **END** *in place of the words that aren't there; you may have to use two* **BEG** *or two* **END** *symbols.) Your model should have a value for the prior probability of each of the senses $P(s_k)$ and for each feature and sense a value for the probability of the feature given the sense $P(feat \mid s_k)$.*

Finally, use your model to disambiguate each of the 14 test sentences. Compute precision and recall values for each sense on the basis of this test set. [1]

[1] A time saving trick, which is reasonable to use (and often is used) but only fair to use if you truly chose your test sentences randomly is only to compute values for the parameters that you will be tested on. I will say no more about this hint.