L&S II: Assignment 3

Prof. Noah Smith

Due: Thursday, October 26 (part hardcopy, in class, part electronic)

1 An Empirical Problem

The goal of this assignment is to get more probabilistic modeling practice and write a decoder for a simple problem. As in earlier assignments, there is a dataset for training and a dataset for testing. You are encouraged to reserve some development data for use in tuning your models to generalize well.

The modeling task is to predict the types of grammatical relations among words in dependency trees. The language we'll be working with is Portuguese.[1] At training time, you will get a set of $\approx$ 9,000 dependency trees, with each dependency relationship (one per word) labeled. At test time, you will get a set of $\approx$ 300 unlabeled dependency trees, which you will use your model to label.

Here are some questions to consider before and during the construction of your model. Will you predict each of the grammatical relations separately, or will your decisions for different words influence each other? Either way, be very clear about the independence assumptions in your model (state them). Given those independence assumptions, what method of combinatorial optimization is required? Describe your search method clearly, stating what runtime/space/optimality guarantees it offers (or doesn't); if it's a dynamic programming algorithm, give the equations.

Data format The data is in UTF-8. To make it work with Perl, `use bytes`. Sentences are separated by a blank line. Each token is on its own line and is described by seven fields:

1. position of the word (1-indexed)
2. word form or punctuation mark
3. lemma
4. part-of-speech tag
5. morphological features, separated by a `|` symbol

[1] Portuguese is spoken by more than 200 million people as a native language, ranking fifth or sixth among all human languages. It is spoken in Portugal, Brazil, Angola, and Mozambique.

6. the position of the word's parent; if 0, means that the word attaches to the virtual "wall" symbol (it's a root)

7. dependency relation between the word and its parent

The last field is what you need to predict, for every word in the test data.

Warning The training and test data for this exercise are a publicly available corpus distributed by the *Floresta Sintá(c)tica* project that you can find here: `http://acdc.linguateca.pt/treebank/info_floresta_English.html`. You are welcome and encouraged to explore that page to find out more about the annotation style, the meanings of the grammatical relations and tags used in the data, etc. **You are not, however, allowed to look at or in any way make use of corpus data that I haven't given you.** Doing so will be considered **cheating**. To safeguard yourself, be prepared to turn in all of your source code (training *and* testing) so that I can replicate your results and see that they do not depend on the test data.

Deliverables Turn in, electronically, a file that contains the missing seventh column for the test data. The file should not contain the other columns. If your file doesn't "line up" with the test data, your grade will suffer, so be sure it's right. The hardcopy you turn in should describe your model, how you trained it, and explain any other approaches you tried. The thought questions above should be answered in your description. You should also answer this question: how would your approach have changed if, at test time, you had the output of an unlabeled dependency parser, rather than gold-standard unlabeled trees? Discuss some solutions to that problem.

2 A Formal Problem

Show how an HMM can be represented as a PCFG. Show how using the dynamic programming PCFG-parsing algorithm of your choice (e.g., Earley's algorithm, CKY) reduces to the Viterbi algorithm when the PCFG implements an HMM. To do this, you should write the algorithm as a set of recursive equations and show how the equations can be simplified under the assumption that the PCFG implements an HMM. Your goal is to show that the equations are essentially the same as the equations for the Viterbi algorithm. Would it make sense to use probabilistic CFG parsing code to run the Viterbi algorithm (why/why not)?