1 Introduction

In this exercise, we are going to explore a simple kind of syntactic rules called D-rules, which describe the syntactic relations between head words and dependent words. In the next exercise, we are going to use these rules to construct a dependency parser.

2 Data

The data comes from the Swedish treebank included in the Universal Dependency Treebanks and is divided into a training set (sv-universal-train.conll), a development set (sv-universal-dev.conll), and a test set (sv-universal-test.conll). In this exercise, we will only use the training set. Let us look at the first sentence of the training set:

1 Individuell _ ADJ JJ POS|UTR|SIN|IND|NOM 2 amod _ _
2 beskattning _ NOUN NN UTR|SIN|IND|NOM 0 ROOT _ _
3 av _ ADP PP _ 2 adpmod _ _
4 arbetsinkomster _ NOUN NN UTR|PLU|IND|NOM 3 adpobj _ _

As usual, each token is represented on a single line (with blank lines as sentence separators) with tabs separating its different attributes. The only attributes we care about here are the word form in column 2, the (universal) part-of-speech tag in column 4, the syntactic head in column 7, and the dependency relation in column 8.1

In this sentence, the root word is the second word beskattning (taxation), which is preceded by the adjectival modifier (amod) Individuell (individual) and followed by the adpositional modifier (adpmod) av (of), which takes the final word arbetsinkomster (work income) as its adpositional object (adpobj). The four words are tagged as adjective (ADJ), noun (NOUN), adposition (ADP), and noun (NOUN), respectively. Please make sure that you understand the representation before you continue.

Note on visualization: Although not strictly needed for this exercise, it may be helpful to use a tool for visualizing the syntactic analyses as dependency trees. For this I recommend using whatswrong, which can be downloaded from https://code.google.com/p/whatswrong/downloads/. (Use the standalone jar version.)

3 D-rules

A D-rule is a quadruple \((T_h, T_d, D, L)\), which specifies that it is possible to have a dependency relation between a head word with tag \(T_h\) and a dependent word with tag \(T_d\) with direction \(D\) and dependency label \(L\). For example, based on the evidence from the first sentence in the training set, we may conclude that the following D-rules are valid for Swedish (where < and > specifies the that the dependent occurs to the left and to the right of the head, respectively):

- NOUN ADJ < amod
- NOUN ADP > adpmod
- ADP NOUN > adpobj

4 Extract D-rules manually

The first task is to manually extract D-rules from the first three sentences of the training set, that is, the sentence above and the two following ones. Make a list of all the D-rules instantiated in these sentences and record the frequency of each rule.

5 Extract D-rules automatically

Write a program to automatically extract D-rules from the training set and count their frequency. The output should be a list of all D-rules and their frequency in the following format:

- TAG-head TAG-dep DIR LABEL freq

How many rules did you find overall? How many rules occur only ones? Which are the ten most frequent rules? How common is it to find distinct rules where the first three elements \((T_h, T_d \text{ and } D)\) are the same?

---

1For more information about this format known as the CoNLL-X format, see http://nextens.uvt.nl/depparse-wiki/DataFormat.
Useful linux commands:  To sort the output file with respect to (descending) frequency, run:

```
sort -f 5 -n -r < output_file > sorted_output_file
```

To extract the D-rules with frequency 1:

```
grep '[^0-9]1$' < output_file > freq1_file
```

To count the number of D-rules:

```
w -l < output_file
```