Visualization as an aid to Text Processing

Honours Project 2015

Nikola Nikolov

s1245947

**Food for thought**

Bookeh + Ipython vs Flask + D3  
  
Which one is a better way to create a visual representation of dependency parsing.

How to extract data from each step of the algorithm to be able to visualize it step by step?

How to evaluate which implementation is better?

Is bokeh more ‘static’ than d3 in terms of animation?

Presentation, executable plan (timetable).

Added Werkzeug python library to check if the file uploaded by the user (the ConLL file) has a secure name. (Pass it a filename and it will return a secure version of it. This filename can then safely be stored on a regular file system and passed to [**os.path.join()**](http://docs.python.org/dev/library/os.path.html#os.path.join). The filename returned is an ASCII only string for maximum portability.

On windows systems the function also makes sure that the file is not named after one of the special device files.

>>> secure\_filename**(**"My cool movie.mov"**)**

'My\_cool\_movie.mov'

)

**What is Dependency Parsing?**  
Dependency Parsing is basically the process of parsing natural language to a Dependency Tree. There are two ways of achieving this goal – by using constituency grammar or dependency grammar. The main difference between the two is that in dependency grammar for every word or morph in the sentence there is exactly one single node corresponding to it, where as in constituency grammar we can have one or more nodes corresponding to the same word or morph.

**Different ways of visualizing dependency trees**

Conventions can vary! There are many different ways we can use Dependency Parsing to visualize dependencies. In Dependency Grammars the root is always the verb and we can see this in the pictures. Pictures [ 1] through [7] showcase some of the different approaches that can be used. Some of them are in the form of the trees, other look more like a graph than a tree, or as shown in pictures [7] and [4] we can focus on the hierarchical order of the words, even with the connections omitted like in the case of [7]. Even though picture [4] looks like a tree, dependency trees reflect the actual word order of the sentence. Pictures [1] through [3] are dependency trees, however they differ in the amount of information represented. Picture [1] displays not only the words, but also the part of speech tokens for each word. The technique in in picture [6] is rarely used, however it is another way of giving the words a hierarchical order (children nodes are enclosed in more brackets than their parent nodes). The representation we will be using is shown in picture [5].

**Our choice of visualization**

**/SUMMERIZE and use link in bookmakrks to showcase the different ways of connecting words**  
Whenever two words are connected by a dependency relation, we say that one of them is the head and the other is the dependent, and that there is a link connecting them. In general, the dependent is the modifier, object, or complement; the head plays the larger role in determining the behavior of the pair. The dependent presupposes the presence of the head; the head may require the presence of the dependent. Figure 2 shows the dependency structure of a sentence. Essentially, a dependency link is an arrow pointing from head to dependent. The dependency structure is a tree (directed acyclic graph) with the main verb as its root (head). Figure 3 shows a way to display the word order and the tree structure at once. To get from a word to its dependents in this kind of diagram, go downhill. In what follows, a dependent that precedes its head is called a predependent; one that follows its head, a postdependent. Figure 3: This representation of a dependency tree preserves the word order while depicting the tree structure plainly. To get from a head to its dependents, go downhill. I shall say that a word is independent (headless) if it is not a dependent of any other word. Note that in the dependency tree, constituents (phrases) still exist. Any word and all its dependents, their dependents, etc., form a phrase. I shall say that dependents, dependents of dependents, etc., are subordinate to the original word, which in turn dominates (is superior to) them. A word comprises itself and all the words that it dominates. That is, the head of a phrase comprises the whole phrase