Natural Language Processing

Lab 8: Dependency Annotation

1 Introduction

In this lab, we will perform grammatical annotation of English sentences, following the guidelines of the Universal Dependencies (UD) project.[1] The input to annotation will be text that has been tokenized, tagged and lemmatized, so the task will be limited to annotating syntactic structures in the form of dependencies. We will use the UD Annotatrix tool for annotation and the MaltEval tool for comparing annotations. We will also need to consult the UD guidelines for syntactic annotation.[2]

2 Data

The data to be annotated consists of 10 English sentences, selected from one of the English UD treebanks, stored in the file `en10-anno.conll` (available in `/local/kurs/nlp/syntax/`), and represented in the CoNLL-X format shown below.

```
1    This          this          DET      DET      _
2    department    department    NOUN     NOUN     _
3    now           now           ADV      ADV      _
4    faces         face          VERB     VERB     _
5    new           new           ADJ      ADJ      _
6    challenges    challenge     NOUN     NOUN     _
7    .             .             PUNCT    PUNCT    _
```

In this format, each word is represented by one line, consisting of tab-separated columns with the following interpretation:[3]

1. ID: Word index, integer starting at 1 for each new sentence.
2. FORM: Word form or punctuation symbol.
3. LEMMA: Lemma or stem of word form.
4. CPOSTAG: Coarse-grained part-of-speech tag.
5. POSTAG: Fine-grained part-of-speech tag (same as CPOSTAG if not available).
6. FEATS: List of morphological features (underscore if not available).

The syntactic dependency annotation will consist in adding two more columns:

1. HEAD: Head of the current token, which is either a value of ID or zero (0).
2. DEPREL: Dependency relation to the HEAD (root iff HEAD = 0).

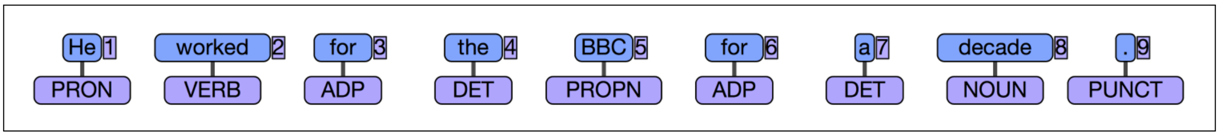
However, instead of manipulating the CoNLL-X file directly, which would be theoretically possible but practically difficult and error-prone, we will use the Brat tool that provides a graphical user interface for annotation.

3 Start UD Annotatrix

UD Annotatrix is a web-based annotation tool available at:

`https://maryszmary.github.io/ud-annotatrix/standalone/annotator.html`

Load the file `en10-anno.conll` by clicking on the "import corpus" button and selecting the file. You should now see the first sentence in CoNLL-U format in the text window at the top of the screen. If you scroll down, you should also see a graphical display of the sentence, with words and tags but no dependencies, as shown below.



You can now annotate dependency trees as follows:

1. To add a dependency arc, click first on the head word and then on the dependent word.
2. To add a label to an arc, click on the arc and type the label (and hit the return key).
3. To remove a dependency arc, right-click to select and then hit the delete or backspace key.

[1] `http://universaldependencies.org`

[2] `http://universaldependencies.org/u/overview/syntax.html`

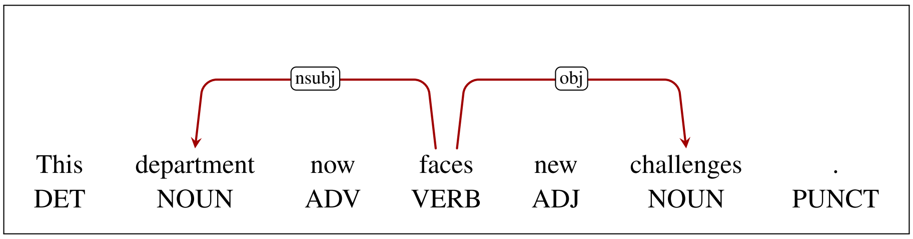
[3] For more information about the CoNLL-X format, see `http://ilk.uvt.nl/conll/#dataformat`. UD actually uses an evolved version of CoNLL-X called CoNLL-U, but the differences are irrelevant for this lab.

More info on the Help button.
Note that the CoNLL-U representation at the top gets modified as well. In fact, you can also annotate by directly editing the CoNLL-U file. In general, we do not recommend doing this, because it is easier to make mistakes, but you have to use this method to add the root depependency. In other words, after you have added all dependencies between words, you should change the HEAD value to 0 and the DEPREL value to “root” for the word that is the root of the dependency tree.

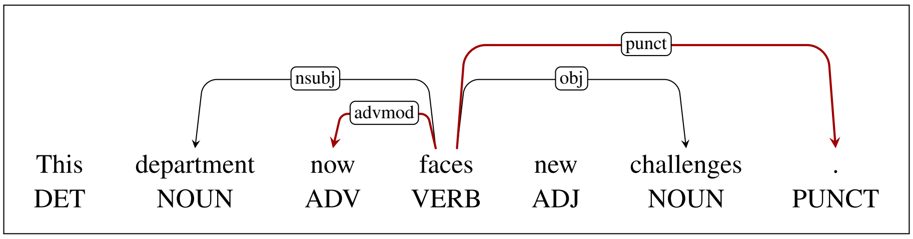
Important: When you have finished the annotation, you must use the ”Download corpus” button to download it to your computer. This puts it in the downloads folder, from which you may need to copy it to your working directory for the final comparison.

4 Annotate sentences

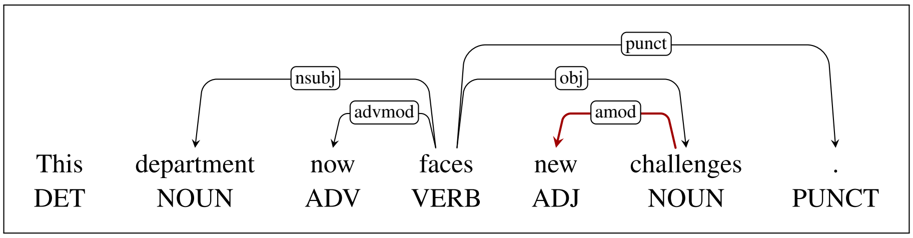
When annotating a sentence, start by identifying the main predicate and annotate its core arguments.


nsubj
obj
This department now faces new challenges .
DET NOUN ADV VERB ADJ NOUN PUNCT

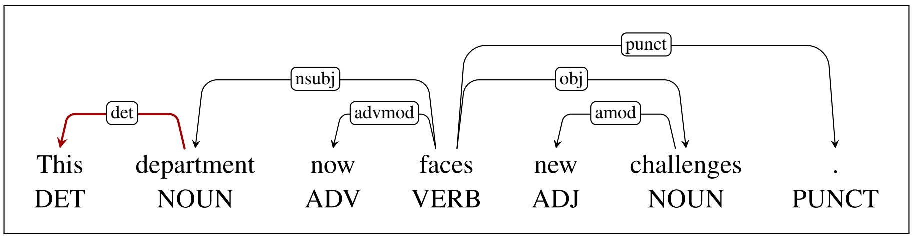
Go on by annotating additional modifiers of the main predicate (including punctuation).


punct
nsubj
obj
advmod
This department now faces new challenges .
DET NOUN ADV VERB ADJ NOUN PUNCT

Then connect modifiers to the dependents of the main predicate (possibly in several steps).


punct
nsubj
obj
advmod
amod
This department now faces new challenges .
DET NOUN ADV VERB ADJ NOUN PUNCT

Finally, attach function words to the content word they specify.


punct
nsubj
obj
det
advmod
amod
This department now faces new challenges .
DET NOUN ADV VERB ADJ NOUN PUNCT

Repeat the process for all 10 sentences, consulting the UD guidelines as needed.

5 Compare annotations

When you have annotated all 10 sentences, you can either compare your annotations to those of another group or ask the lab instructor for the official UD annotation (contained in the file `en10-gold.conll`). What differences do you find? Did you make a mistake or do you want to argue for your analysis? Is it possible that more than one analysis could be acceptable? You can visualize the differences using the MaltEval tool, which you can get by copying the file `MaltEval.jar` from `/local/kurs/nlp/syntax/`. If you want to compare two annotations that are stored in `FILE1` and `FILE2`, you can display both annotations with differences marked in red by running:

```
java -jar -Xmx2g MaltEval.jar -g FILE1 -s FILE2 -v 1
```