Natural Language Processing: Assignment 2

Due: Friday 20 April 2018 at 16:00 | Submission via PointCarré or e-mail

Summary

In this assignment, you will use the Dependency Grammar framework to describe and formalize the syntactic structure of sentences. You will familiarize yourself with one of the standard approaches to automatic dependency parsing, viz. Transition-based parsing. Based on a set of toy sentences, you will implement the shift-reduce algorithm and construct an oracle to guide the algorithm in assigning the correct dependency relations. Finally, you will evaluate the results of your toy dependency parser against a gold standard using some standard evaluation measures for dependency parsing.

As background material to this assignment, please read "Chapter 14, Dependency Parsing" in Jurrafsky & Martin 3rd Edition (draft on https://web.stanford.edu/~jurafsky/slp3/)

Part 1: Annotation of toy sentences with dependency relations

Annotate the following sentences with universal part-of-speech tags and universal dependency relations (see the relevant pages on http://universaldependencies.org/ for descriptions and examples of POS-tags and dependency relations):

1. I gave an apple to the teacher
2. Mary missed her train to work
3. John gave the teacher a very heavy book
4. The sun shines

5. This is the dog that chased the cat
6. I saw the doctor this morning who is treating me
7. This is the cat that the dog chased
8. John is eager to please

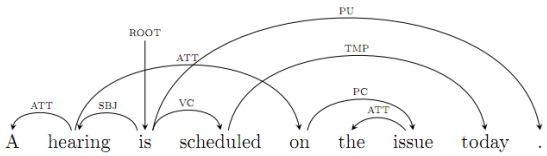
FORMAT

For all sentences: Provide your annotations in the CoNNL-U-format and add it as a separate txt file (filename <**annotation.txt**> in utf8 encoding) to the zip archive that you submit.

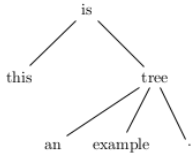
- You only need to provide **1 legitimate parse** per sentence
- In the annotated text file, you only need to specify values for the following fields of the CoNNL-U format: 1.ID, 2.FORM, 3.LEMMA, 4.UPOSTAG, 7.HEAD, 8.DEPREL.
- Other fields can be left unspecified, (indicated with an underscore in the CoNNL-U format)

For the first sentence only (*I gave an apple to the teacher*), also include in your report:

- A dependency diagram (Use the drawing functionality of your office suite. Latex users, see here):



- A dependency tree (Use the drawing functionality of your office suite. Latex users, see here):



This is an example tree.

In your report, also **discuss the following issues**:

- Are there ambiguous sentences with multiple possible parses? If yes, what form does the ambiguity take?
- Which sentence contains non-projective dependency relations? Which relations exactly and why are they non-projective?

Part 2: Implementation of a transition-based dependency parser

Write a transition-based dependency parser in Python (3.x) or Common Lisp using the following guidelines:

- Use the shift-reduce algorithm (see slides class 8 and J&M 3rd Ed. Ch.14)
- Only consider **unlabelled dependencies** and hence, only 3 possible actions (leftArc, rightArc, shift) for the algorithm
- Your parser should take as **INPUT**:
    1. a text file <input.txt> with sentences in the CoNNL-U format for which only the following fields are specified: *1.ID, 2.FORM, 3.LEMMA,* and *4.UPOSTAG*
    2. a file <feattemp.txt> with feature templates to be used by the parser to generate features for the oracle (see below)
- Your parser should produce the following **OUTPUT**:
    1. A **text file in CoNNL-U format** that adds values to the input file for the fields *7.HEAD,* and *8.DEPREL*. Since the parser only assigns unlabelled dependency relations, the dependency relations in *8.DEPREL* can be uniformly labelled as "DEP"
    2. A text file that contains **for each sentence** a **trace of the configuration states** that the shift-reduce parser went through while parsing the sentence. Use the trace table format from J&M 3rd Ed. Ch.14, fig.14.7. Write out the table in a simple tab separated value format. Put each table in an XML-like element with information about the input sentence as attributes in the start tag:

Step	Stack	Word List	Action	Relation Added
0	[root]	[book, me, the, morning, flight]	SHIFT	
1	[root, book]	[me, the, morning, flight]	SHIFT	
2	[root, book, me]	[the, morning, flight]	RIGHTARC	(book → me)
3	[root, book]	[the, morning, flight]	SHIFT	
4	[root, book, the]	[morning, flight]	SHIFT	
5	[root, book, the, morning]	[flight]	SHIFT	
6	[root, book, the, morning, flight]	[]	LEFTARC	(morning ← flight)
7	[root, book, the, flight]	[]	LEFTARC	(the ← flight)
8	[root, book, flight]	[]	RIGHTARC	(book → flight)
9	[root, book]	[]	RIGHTARC	(root → book)
10	[root]	[]	Done	

Figure 14.7 Trace of a transition-based parse.

```
<sentence file="J&MCh14" id="14.7" text="book me the morning flight">
Step›Stack→Word List›Action→RelationAdded
0›[root]→[book,me,the,morning,flight]›SHIFT
1›[root,book]→[me,the,morning,flight]›SHIFT
2›[root,book,me]›[the,morning,flight]→RIGHTARC›(book->me)
3›[root,book]→[the,morning,flight]→SHIFT
4›[root,book,the]›[morning,flight]›SHIFT
5›[root,book,the,morning]›[flight]›SHIFT
6›[root,book,the,morning,flight]›[]›LEFTARC→(morning<-flight)
7›[root,book,the,flight]›[]›LEFTARC→(the<-flight)
8›[root,book,flight]→[]›RIGHTARC›(book->flight)
9›[root,book]→[]›RIGHTARC›(root->book)
10›[root]→[]›Done
</sentence>
```

- Manually construct a **rule-based oracle** using the following guidelines:
    - Use **sentences 1 to 4** that you manually annotated in Part 1, as a starting point to choose and define **feature templates** which your parser will use to generate features from the configurations while executing the shift-reduce algorithm. Make sure that you choose your feature templates in such a way that they will generate features that are informative for the oracle to select the correct action for the configurations in the parsing process of the 4 sentences. Discuss the motivations behind your choice of features in the report
    - Feature templates should be defined in a generic way so that they are not only applicable to 1 specific sentence or configuration: Therefore, only use the Part-Of-Speech information of the words located on the stack, the input buffer, and/or dependency relation list.
    - You can use both single-word and double-word feature templates (see J&M 3rd Ed. Ch.14, fig.14.8
    - The feature templates should be specified in a format based on J&M 3rd Ed. Ch.14, fig.14.8 and listed in a separate file <feattemp.txt> that can be read by your parser.
    - The oracle itself should be a function that takes as argument a set of generated features and that uses rules (in the form of (nested) IF/ELSE statements) to return the appropriate action (*leftArc*, *rightArc* or *shift*). The rules should be based on your motivation for your selection of feature templates, i.e. the rules operationalize your analysis of why specific features are informative.

Part 3: Evaluation

For didactic purposes and in the toy-world setting of this assignment, you will evaluate your parser against the same four sentences used in the development stage. (In a real-life setting and with an oracle based on a trained statistical classifier, you would of course use a test-set of sentences that was not part of training/development set). Your manually annotated dependency relations from Part 1 will function as the "gold standard" for the evaluation. Since your parser only assigns unlabelled dependency relations, the specific dependency labels in your manual annotation do not have to be taken into consideration.

- Run your parser on sentences 1 to 4 listed in part 1 of the assignment
- Manually check the dependency relations in your parser's output against your manually annotated dependency relations for each sentence
- In your report, give the **unlabelled attachment accuracy** across all 4 sentences and discuss the errors (if any) and why your think the parser made them.
- Also report the precision and recall for the head-dependency relations across all 4 sentences.

Part 4: Report

Your report should be a running text that begins with a section that briefly introduces the context of the assignment and the problem at stake. Discuss parts 1, 2 and 3 of the assignment in separate sections: Describe annotation issues (part 1), design choices and oracle construction (part 2), and evaluation (part 3).

Handing in your assignment:

Submit your assignment **by Friday 20 April 2018 (16h00)** via PointCarré or via e-mail (katrien@ai.vub.ac.be) as a **zip archive** that includes the following 8 files:

1. **report.pdf** : your report as a pdf-file
2. **annotation.txt**: your manually annotated sentences 1-8 in CoNNL-U format
3. **parser.py:** your parser code as 1 python script, that is easily runnable, takes input.txt as input and produces the output files output.txt and conftable.txt
4. **input.txt**: the input file for your parser with sentences 1-4 in CoNNL-U format (with unspecified head and dependency fields)
5. **feattemp.txt** with the feature templates you defined and selected
6. **output.txt**: parsed output in CoNNL-U format (with head and dependency information)
7. **conftable.txt**: a text file with the trace tables of the configuration states for sentences 1-4
8. **README**: a text file that explains how to run the code

If you have any questions about the assignment, please post them on the forum on PointCarré.

Good luck!

Katrien Beuls