11-712: NLP Lab

Noah A. Smith

Spring 2014
version 1 (1/8/14)

The focus of this lab is natural language dependency parsing. To successfully complete the laboratory exercise, you will design, implement, evaluate, document, and openly release a dependency parser for a language of your choice.

1 Background and Goals

Syntactic analysis is a long-studied and widely used part of NLP. Despite annotation efforts and algorithmic research, syntactic analyzers are available for relatively few languages and genres.

The goal of this lab is to give you hands-on experience developing and documenting an open-source NLP tool. It is our hope that you will build a tool with good enough performance to be useful to others, and that you will design it in such a way that it can be extended and improved by others. You are encouraged to use existing resources, with one exception: you may not use any existing annotated dataset. You must develop your own annotated *test* set (details below), and you may develop your own annotated *training* and *development* sets. You may use any existing software tools for annotation, training, and parsing, as long as you acknowledge and cite them appropriately.

Before you start, you need to understand the theory of dependency syntax. There are different defnitions of "head," and different approaches to categorizing dependency relations. These different theories are not all mutually consistent! This means that you will have to define your own heuristics and conventions, justify them, and *document* them.

Some useful readings:

- Linguistic theory of dependencies: Zwicky (1985); chapters 2–3 of Valin (2001)—we will give you access to a scanned copy of these chapters
- Survey of dependency parsing algorithms: Kübler et al. (2009)
- Slides on dependency parsing from Algorithms for NLP (2013): `http://demo.clab.cs.cmu.edu/fa2013-11711/images/8/86/Deparsing.pdf`
- Survey of linguistic fundamentals (morphology and syntax) targeting computer scientist readers: Bender (2013)
- CoNLL dependency parsing shared task overviews: Buchholz and Marsi (2006); Hajič et al. (2009); Nivre et al. (2007); Surdeanu et al. (2008)

Some useful tools:

- GitHub (`http://github.com`): version control and software publishing platform.
- Fragmentary unlabeled dependency grammar tools (`http://www.ark.cs.cmu.edu/FUDG`).
- TurboParser (`http://www.ark.cs.cmu.edu/TurboParser`); there are many other openly available statistical dependency parsing packages, this one just happens to have been developed in the instructor's research group.

This exercise is primarily an individual exercise. You are, however, encouraged to work with at least one informant, and you are welcome to enlist the assistance of additional annotators. You are also encouraged to discuss your project with other students in the class, and other researchers more generally. You must acknowledge everyone whose assistance helped you finish the exercise.

2 Schedule

The lab is tightly structured across 15 weeks. You will be writing documentation from the very beginning. There are deadlines every one or two weeks. The amount of work each week is relatively small, but you must complete each milestone on schedule.

1. 1/15 (optional): Attend Lori Levin's lecture on syntactic dependencies and headedness to make sure you have a good foundation for working with dependency syntax.
   January 15, 12:00–1:00 pm (GHC 7501).
2. By 1/17: Select a language and genre.[1] You must choose a language with a writing system in which tokenization into words will be easy. If possible, you should identify a native speaker to whom you can turn for advice. Ideally this person will have enough linguistics training to have conversations with you about the syntax of your language. If you cannot find a native speaker, try to find someone who knows the language. You may serve as your own informant. Also, create an open-source project on GitHub where you will openly develop your project and write your report. Write part 1 of the report.
3. **There will be a meeting January 21, 4:30–5:30 pm (GHC 4405).**
4. By 1/24: Identify and review past work on this language's syntax. This includes linguistics papers, computational modeling papers, and system descriptions. To the greatest extent possible, you will want to make use of previous work (with the exception of annotated datasets). Write part 2 of the report.
5. By 1/31: Identify existing resources that might be useful in building your analyzer. Open-source lexicons, unannotated corpora, and reference grammars are worth looking for. You will need two test corpora of at least 1,000 words apiece. Call these A and B. You must select data that you can redistribute without violating any copyright. Write part 3 of your report.
6. By 2/7: Catalog the attested phenomena in the language, based on your literature search, the resources you've identified, and a meeting with your informant. Write part 4 of your report.
7. By 2/14: Annotate your test datasets A and B, with assistance from your informant if necessary. Decide on your strategy for implementation. We suggest four possibilities:

[1] Because we want you to produce a tool that is *useful*, it is not recommended that you choose a language for which sophisticated tools are already available. But this is not a hard constraint.

(a) Writing rules. You must choose a grammar formalism, an algorithm, and develop the set of rules to be used to parse text in your language/genre.

(b) Supervised learning. You must annotate additional data for training, select an algorithmic framework, and train your parser.

(c) Semisupervised learning. You must annotate additional data for training, devise a learning algorithm, and train your parser.

(d) Unsupervised learning. You must devise a learning algorithm and train your parser.

Regardless of which path you choose, your final parser must generate one unlabeled dependency parse for any input string. Note that when we say "you must annotate the data," it is acceptable to enlist others (e.g., your informant). As with test sets A and B, any data you use must be data you have permission to redistribute without violating copyright; any data you use in the development of your parser must be openly released along with your parser. Write part 5 of your report.

8. 2/15–2/28: Based on your prioritized list, first round of development.
9. **There will be a meeting March 6, 10–11 am (GHC 4405).**
10. By 3/7: First round of evaluation. For every sentence in corpus A, run it through your parser. Organize the output in a way that will make it easier for your informant to give feedback. Evaluate the unlabeled attachment score of your parser. Write part 6 of your report.
11. By 3/14: Write up the lessons you learned and any revisions to the design of your parser and prioritized list (part 7 of your report) .
12. 3/15–3/28: Continue development, based on the revised plan.
13. By 4/4: Repeat the evaluation, this time with corpus B. Write part 8 of your report.
14. By 4/5–4/24: Continued development, based on the outcomes of the second evaluation. Write up part 9 of your report.
15. By 4/25: Upload your corpora and your system's parses on test sets A and B, so that potential users of your tool, and researchers who might build on it, may inspect. Write up the future work (part 10 of your report). Turn in your report and release version 1.0 of your parser.
16. **There will be a meeting May 1, 1–3 pm (GHC 2109).** You will be expected to give a presentation about your project.

3 Report

Your report should be written in LaTeX, using the template provided at `http://www.cs.cmu.edu/~nasmith/NLPLab/report-template-2014.tex`. The outline for your report is fixed:

1. Basic information about the language
2. Past work on syntax for this language
3. Available resources, including your corpora
4. Survey of phenomena

5. Initial design
6. System analysis on corpus A
7. Lessons learned and revised design
8. System analysis on corpus B
9. Final revisions
10. Future work

You must write each portion of the report on schedule and maintain the report in your GitHub repository.

References

Emily M. Bender. *Linguistic Fundamentals for Natural Language Processing: 100 Essentials from Morphology and Syntax*. Morgan and Claypool, 2013.

Sabine Buchholz and Erwin Marsi. CoNLL-X shared task on multilingual dependency parsing. In *Proceedings of the Tenth Conference on Computational Natural Language Learning*, pages 149–164, New York City, USA, June 2006.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–18, Boulder, Colorado, USA, June 2009.

Sandra Kübler, Ryan McDonald, and Joakim Nivre. *Dependency Parsing*. Morgan and Claypool, 2009.

Joakim Nivre, Johan Hall, Sandra Kübler, Ryan McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. The CoNLL 2007 shared task on dependency parsing. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 915–932, Prague, Czech Republic, June 2007.

Mihai Surdeanu, Richard Johansson, Adam Meyers, Lluís Màrquez, and Joakim Nivre. The CoNLL 2008 shared task on joint parsing of syntactic and semantic dependencies. In *Proceedings of the 12th Conference on Computational Natural Language Learning*, pages 159–177, Manchester, UK, August 2008.

Robert D. Van Valin. *An Introduction to Syntax*. Cambridge University Press, 2001.

Arnold M. Zwicky. Heads. *Journal of Linguistics*, 21(1):1–29, 1985.