Lab 2 – Natural Language Understanding in Prolog

CS427 / CS527

Fall 2016

1 Description

The purpose of this lab is to introduce you to natural language understanding in Prolog. In this lab, you will extend the Prolog probabilistic parser (section 8.4, Probabilistic Context-Free Parser) and context sensitive parser (section 8.5) as defined in Luger's AI Algorithms, Data Structures, and Idioms. You must provide Prolog code for a subset of English grammar rules. **Don't forget to make your solutions recursive**–they should take any number of compounds! You will then use this parser to parse sentences you compose yourself. These sentences may only use words defined in your lexicon (see Lexicon section below).

1.1 Probabilistic Context-Free Parser Example

Take for example the sentence, "i love it." This sentence might have the following probabilty (example only).

```
?- utterance(Prob, [i, love, it]).
Prob = 0.08765
```

1.2 Context Sensitive Parser Examples

Take for example the sentence, "trump joins the wriwenis cult." This sentence will not be recognized as a valid sentence.

```
?- utterance([trump, joins, the, wriwenis, cult]).
Yes.
```

Take for example the sentence, "trump join the wriwenis cult." This sentence will not be recognized as a valid sentence.

```
?- utterance([trump, join, the, wriwenis, cult]).
No.
```

Take for example the sentence, "trump likes the hollow earth." This sentence will be recognized as a valid sentence.

```
?- utterance([trump, likes, the, hollow, earth]).
Yes.
```

2 The Lexicon File

This project comes with a Lexicon file. This file contains the list of valid words you can use to construct sentences. You may add words to the lexicon, but be sure to document which words you add in your README file. Whether or not you add words to the lexicon, you are required to provide probabilities for each noun, verb, adjective, etc.

3 Requirements

- You may work individually or in a team of up to two members. (If you work in teams of two, each member has to work on rules).
- You must implement the probabilistic rules described in section 8.4.
- You must implement the context-sensitive rules desecribed in section 8.5
- You must add additional rules so the parser can parse more complicated sentences.
  - adverbs, (i.e., words that modify verbs; e.g., "The writer writes ELOQUENTLY.")

  - adjectives, (i.e., words that modify nouns; e.g., “The GRACEFUL ballerina dances on stage.”)
  - conjunctions, (i.e., words that join two or more words, phrases, or clauses; e.g., I like pizza AND cheese BUT I like neither pepperoni NOR sausage)

- prepositions and prepositional phrases (The hat is ON THE CAT; the dog UNDER THE PORCH; the robot IN THE BOX ON THE SHELF is expensive)

4 Submission

Please package your files in a single archive file (*.tar or *.zip file) and submit it to Learn. The archive file should contain:

- README, the README file that describes how to run your code
- teamwork.txt, a text file that describes how work was distributed
- the original and updated lexicon files
- probabilistic.pl, your Prolog source files for your probabilistic parser
- context_sensitive.pl, your context sensitive parser
- traces.txt, a text file containing traces of the following test cases

5 What to Trace, and How

- `utterance([alient, exist, under, the, flat, earth]).`
- `utterance([black, helicopters, are, terribly, real]).`
- `utterance([trump,wants,a,great,great,wall]).`
- `utterance([the,robot,scans,small,and,round,brains]).`

To generate a trace, type and enter `trace.` (with the period) into the Prolog interpreter; subsequent inputs to the interpreter will generate a trace. These traces can then be cut-and-pasted into your `traces.txt` file. To stop tracing, type and enter `notrace.` (with the period).

6 Notes

To generate a trace, type and enter `trace.` (with the period) into the Prolog interpreter; subsequent inputs to the interpreter will generate a trace. These traces can then be cut-and-pasted into your `traces.txt` file. To stop tracing, type and enter `notrace.` (with the period at the end).

7 References

- Luger, G.F., and Stubblefield, W.A. AI Algorithms, Data Structures, and Idioms in Prolog, Lisp, and Java. 2009. Pearson Addison-Wesley.
- Grammarly. https://www.grammarly.com/handbook