Last Class: Part-of-Speech Tagging

Today: Parsing

1. Grammars and Parsing
2. Top-down and bottom-up parsing

Key Ideas:
- Sentence analysis: from the Greek σύνθεσις, meaning "setting out together or arrangement." Refers to the way words are arranged together.

Why worry about syntax?
- The boy whom the frog was eaten by died.
- The frog was eaten by the boy.
- The boy ate the frog.
- The frog died.

Why worry about syntax?
- The frog was eaten by the boy.
- The boy ate the frog.
- The frog died.
- The boy was eaten by the frog.

Why worry about syntax?
- The frog was eaten by the boy.
- The boy ate the frog.
- The frog died.
- The boy was eaten by the frog.

Why worry about syntax?
- The frog was eaten by the boy.
- The boy ate the frog.
- The frog died.
- The boy was eaten by the frog.

Syntax

Context-free grammars:
All can be modeled by various kinds of grammars that are based on
- The boy ate the frog.
- The frog was eaten by the boy.
- The frog died.
- The boy was eaten by the frog.

Sentence

Need a grammar: a formal specification of the structures allowable in
- The boy ate the frog.
- The frog was eaten by the boy.
- The frog died.
- The boy was eaten by the frog.

Grammar and Parsing

Sentence

Need a parser: an algorithm for assigning syntactic structure to an input
- The boy ate the frog.
- The frog was eaten by the boy.
- The frog died.
- The boy was eaten by the frog.

Sentence

Need a grammar: a formal specification of the structures allowable in
- The boy ate the frog.
- The frog was eaten by the boy.
- The frog died.
- The boy was eaten by the frog.

Sentence

Need a parser: an algorithm for assigning syntactic structure to an input
- The boy ate the frog.
- The frog was eaten by the boy.
- The frog died.
- The boy was eaten by the frog.

Sentence

Need a grammar: a formal specification of the structures allowable in
- The boy ate the frog.
- The frog was eaten by the boy.
- The frog died.
- The boy was eaten by the frog.

Sentence

Need a parser: an algorithm for assigning syntactic structure to an input
- The boy ate the frog.
- The frog was eaten by the boy.
- The frog died.
- The boy was eaten by the frog.

Sentence

Need a grammar: a formal specification of the structures allowable in
- The boy ate the frog.
- The frog was eaten by the boy.
- The frog died.
- The boy was eaten by the frog.

Sentence

Need a parser: an algorithm for assigning syntactic structure to an input
- The boy ate the frog.
- The frog was eaten by the boy.
- The frog died.
- The boy was eaten by the frog.

Sentence

Need a grammar: a formal specification of the structures allowable in
- The boy ate the frog.
- The frog was eaten by the boy.
- The frog died.
- The boy was eaten by the frog.

Sentence

Need a parser: an algorithm for assigning syntactic structure to an input
- The boy ate the frog.
- The frog was eaten by the boy.
- The frog died.
- The boy was eaten by the frog.

Sentence

Need a grammar: a formal specification of the structures allowable in
- The boy ate the frog.
- The frog was eaten by the boy.
- The frog died.
- The boy was eaten by the frog.

Sentence

Need a parser: an algorithm for assigning syntactic structure to an input
- The boy ate the frog.
- The frog was eaten by the boy.
- The frog died.
- The boy was eaten by the frog.

Sentence

Need a grammar: a formal specification of the structures allowable in
- The boy ate the frog.
- The frog was eaten by the boy.
- The frog died.
- The boy was eaten by the frog.

Sentence

Need a parser: an algorithm for assigning syntactic structure to an input
- The boy ate the frog.
- The frog was eaten by the boy.
- The frog died.
- The boy was eaten by the frog.

Sentence

Need a grammar: a formal specification of the structures allowable in
- The boy ate the frog.
- The frog was eaten by the boy.
- The frog died.
- The boy was eaten by the frog.

Sentence

Need a parser: an algorithm for assigning syntactic structure to an input
- The boy ate the frog.
- The frog was eaten by the boy.
- The frog died.
- The boy was eaten by the frog.

Sentence

Need a grammar: a formal specification of the structures allowable in
- The boy ate the frog.
- The frog was eaten by the boy.
- The frog died.
- The boy was eaten by the frog.

Sentence

Need a parser: an algorithm for assigning syntactic structure to an input
- The boy ate the frog.
- The frog was eaten by the boy.
- The frog died.
- The boy was eaten by the frog.

Sentence

Need a grammar: a formal specification of the structures allowable in
- The boy ate the frog.
- The frog was eaten by the boy.
- The frog died.
- The boy was eaten by the frog.

Sentence

Need a parser: an algorithm for assigning syntactic structure to an input
- The boy ate the frog.
- The frog was eaten by the boy.
- The frog died.
- The boy was eaten by the frog.

Sentence

Need a grammar: a formal specification of the structures allowable in
- The boy ate the frog.
- The frog was eaten by the boy.
- The frog died.
- The boy was eaten by the frog.

Sentence

Need a parser: an algorithm for assigning syntactic structure to an input
- The boy ate the frog.
- The frog was eaten by the boy.
- The frog died.
- The boy was eaten by the frog.

Sentence

Need a grammar: a formal specification of the structures allowable in
- The boy ate the frog.
- The frog was eaten by the boy.
- The frog died.
- The boy was eaten by the frog.

Sentence

Need a parser: an algorithm for assigning syntactic structure to an input
- The boy ate the frog.
- The frog was eaten by the boy.
- The frog died.
- The boy was eaten by the frog.

Sentence

Need a grammar: a formal specification of the structures allowable in
- The boy ate the frog.
- The frog was eaten by the boy.
- The frog died.
- The boy was eaten by the frog.

Sentence

Need a parser: an algorithm for assigning syntactic structure to an input
- The boy ate the frog.
- The frog was eaten by the boy.
- The frog died.
- The boy was eaten by the frog.

Sentence

Need a grammar: a formal specification of the structures allowable in
- The boy ate the frog.
- The frog was eaten by the boy.
- The frog died.
- The boy was eaten by the frog.

Sentence

Need a parser: an algorithm for assigning syntactic structure to an input
- The boy ate the frog.
- The frog was eaten by the boy.
- The frog died.
- The boy was eaten by the frog.

Sentence

Need a grammar: a formal specification of the structures allowable in
- The boy ate the frog.
- The frog was eaten by the boy.
- The frog died.
- The boy was eaten by the frog.

Sentence

Need a parser: an algorithm for assigning syntactic structure to an input
- The boy ate the frog.
- The frog was eaten by the boy.
- The frog died.
- The boy was eaten by the frog.
CFG example

CFG's are also called phrase-structure grammars. Equivalent to Backus–Naur Form (BNF).

1. S → NP VP
2. VP → VN P
3. NP → NAME
4. NP → ART N
5. NAME → Beavis
6. V → ate
7. ART → the
8. N → can
9. NP → ART N

• CFG's are powerful enough to describe most of the structure in natural languages.
• CFG's are restricted enough so that efficient parsers can be built.

Definitions

Derivations

Let α, γ, and β be strings in the set (Σ ∪ N)∗ such that

Then we say that αγβ directly derives αγβ.

Then we say the αγβ directly derives αγβ.

If the rule A → αβγ, then we say that αAγ directly derives αβγ.

Let α1, α2, ..., αm be strings in (Σ ∪ N)∗, m > 1, such that

Then we say that α1Aα2Aα3A...Aαm−1Aαm directly derives α1Aα2Aα3A...Aαm.

The set of strings composed of terminal symbols that can be derived from the designated start symbol S is called the language L(G) generated by the grammar G, which is the set of strings

L(G) = {w ∈ Σ∗ | S∗ ⇒ w}

Parse: the problem of mapping from a string of words to its parse tree, which is a diagram that shows the syntactic structure of the sentence.

A context-free grammar consists of:

1. a set of non-terminal symbols N
2. a set of terminal symbols Σ (disjoint from N)
3. a set of productions, P, each of the form A → α, where A ∈ N and α ∈ (Σ ∪ N)∗
4. a designated start symbol S
General Parsing Strategies

Grammar

Top-Down

1. \( S \rightarrow NP \ VP \)

2. \( VP \rightarrow VN \ P \)

3. \( NP \rightarrow NAME \)

4. \( NP \rightarrow ART \ N \)

5. \( NAME \rightarrow Beav \)

6. \( V \rightarrow ate \)

7. \( ART \rightarrow the \)

8. \( N \rightarrow cat \)


Bottom-Up

1. \( S \rightarrow NAME \ ate \ cat \)

2. \( VP \rightarrow NAME \ V \ ART \ cat \)

3. \( NP \rightarrow NAME \ V \ ART \ N \)

4. \( NP \rightarrow NAME \ V \ ART \ cat \)

5. \( NAME \rightarrow Beav \ ate \)

6. \( V \rightarrow Beav \ ate \)

7. \( ART \rightarrow Beav \ ate \)

8. \( N \rightarrow Beav \ ate \)


A Top-Down Parser

Input: CFG grammar, lexicon, sentence to parse

Output: yes/no

State of the parse: (symbol list, position)

Algorithm for a Top-Down Parser

1. The old 3 man 4 card is
2. Select the current state, if \( \text{PSL} \) is empty, return NO.
3. Check for failure. If \( \text{PSL} \) is empty, return YES.
4. Otherwise, generate the next possible states.
   (a) If \( s_1 \) is a lexical symbol and next word can be in that class, create a new state by removing \( s_1 \), updating the word position, and adding it to \( \text{PSL} \).
   (b) If \( s_1 \) is a non-terminal, generate a new state for each rule in the grammar that can rewrite \( s_1 \). Add all to \( \text{PSL} \).
Problems with the Top-Down Parser

1. Only judges grammaticality.
2. Stops when it finds a single derivation.
3. No semantic knowledge employed.
4. No way to rank the derivations.
5. Problems with left-recursive rules.
6. Problems with ungrammatical sentences.
8. Only judges grammaticality.

Example

1. ((S) 1)
2. ((NP VP) 1)
3. ((art n VP) 1)
4. ((art adj n VP) 1)
5. ((VP) 2)
6. ((art adj n VP) 1)
7. (() 1)
8. ((v NP) 3) ((art adj n VP) 1) Backtrack
9. ((v NP) 3)
10. ((art adj n VP) 1)
11. ((adj n VP) 2)
12. ((n VP) 3)
13. ((VP) 4)
14. ((v) 4) ((v NP) 4)
15. (() 5)
16. ((v NP) 4)

DONE!

Efficient Parsing

The top-down parser is terribly inefficient.

DONE!