Natural Language Processing

Lecture 15:
Treebanks and Probabilistic CFGs
Treebanks: a (Re)introduction
Grammars Can Be Encoded Explicitly and Implicitly

Explicit

\[ S \rightarrow \text{NP} \text{ VP} \]
\[ \text{NP} \rightarrow \text{Det} \text{ NP} \]
\[ \text{VP} \rightarrow \text{V} \text{ NP} \]
\[ \text{Det} \rightarrow \text{a, the} \]
\[ \text{N} \rightarrow \text{professor, students} \]
\[ \text{V} \rightarrow \text{annoyed, delighted} \]

Implicit

\[ (S) \]
\[ (NP) \]
\[ (VP) \]
\[ (Det \text{ the}) \]
\[ (N \text{ professor}) \]
\[ (V \text{ annoyed}) \]
\[ (NP) \]
\[ (Det \text{ the}) \]
\[ (N \text{ students}) \]
The First Big Treebank Was the PTB

**Penn Treebank**

- **Annotation of:**
  - Brown corpus
  - ATIS (Air Travel Information Service corpus)
  - Switchboard Corpus
  - Wall Street Journal corpus

- **Size:** about 1 million words

- **Rules:**
  - 17,500 types
  - “Flat”
  - Many types with only one token
There Are Many Other Treebanks

- Dependency
- Constituency/phrase structure
- Other kinds of linguistic structure
Universal Dependency Treebanks (UD) Are For Cross-Lingual Training

- Train a dependency parser in 111 languages
  - One at a time
  - Cross-lingually
Definition of Context-Free Grammars

- Vocabulary of terminal symbols, $\Sigma$
- Set of nonterminal symbols, $N$
- Special start symbol $S \in N$
- Production rules of the form $X \rightarrow \alpha$

where
- $X \in N$
- $\alpha \in (N \cup \Sigma)^*$  (in CNF: $\alpha \in N^2 \cup \Sigma$)
( (S
   (NP-SBJ
     (NP (NNP Pierre) (NNP Vinken) )
     (, ,)
     (ADJP
       (NP (CD 61) (NNS years) )
       (JJ old) )
     (, ,) )
   (VP (MD will)
     (VP (VB join)
       (NP (DT the) (NN board) )
       (PP-CLR (IN as)
         (NP (DT a) (JJ nonexecutive) (NN director) )
         (NP-TMP (NNP Nov.) (CD 29) )
       )
     )
   )
) )

Proper Ambivalence toward Treebanks
Proper Ambivalence

Why you should have great respect for treebanks.

Why you should be cautious around treebanks.
The Making of a Treebank

• Develop initial coding manual (hundreds of pages long)
  • Linguists define categories and tests
  • Try to foresee as many complications as possible
• Develop annotation tools (annotation UI, pre-parser)
• Collect data (corpora)
  • Composition depends on the purpose of the corpus
  • Must also be pre-processed
• Automatically parse the corpus/corpora
• Train annotators (“coders”)
• Manually correct the automatic annotations (“code”)
  • Generally done by non-experts under the direction of linguists
  • When cases are encountered that are not in the coding manual...
    • Revise the coding manual to include them
    • Check that already-annotated sections of the corpus are consistent with the new standard
This is expensive and time-consuming!
You Should Respect Treebanks

They require great skill

- Expert linguists make thousands of decisions
- Many annotators must all remember all of the decisions and use them consistently, including knowing which decision to use
- The “coding manual” containing all of the decisions is hundreds of pages long

They take many years to make

- Writing the coding manual, training coders, building user-interface tools, …
- and the coding itself with quality management

They are expensive

- Somebody had to secure the funding for these projects
You Should be Cautious Around Treebanks

- They are too big to fail
- They are produced under pressure of time and funding
- Although most of the decisions are made by experts, most of the coding is done by non-experts
To make a good model, you should understand what you’re modeling.
There are two sources of improvement in machine learning

Better models

Better data
Where do production rules come from?
( (S
  (NP-SBJ-1
    (NP (NNP Rudolph) (NNP Agnew) )
    (, , )
  )
  (UCP
    (ADJP
      (NP (CD 55) (NNS years) )
      (JJ old) )
    (CC and)
  )
  (NP
    (NP (JJ former) (NN chairman) )
    (PP (IN of)
      (NP (NNP Consolidated) (NNP Gold) (NNP Fields) (NNP PLC) ) ) )
    (, , ) )
  )
  (VP (VBD was)
    (VP (VBN named)
      (S
        (NP-SBJ (-NONE- *-1) )
        (NP-PRD
          (NP (DT a) (JJ nonexecutive) (NN director) )
          (PP (IN of)
            (NP (DT this) (JJ British) (JJ industrial) (NN conglomerate) ) ) ) )
        (, . ) )
      )
    )
  )
)
Some PTB Rules by Frequency

40717 PP → IN NP
33803 S → NP-SBJ VP
22513 NP-SBJ → -NONE-
21877 NP → NP PP
20740 NP → DT NN
14153 S → NP-SBJ VP .
12922 VP → TO VP
11881 PP-LOC → IN NP
11467 NP-SBJ → PRP
11378 NP → -NONE-
11291 NP → NN
...
989 VP → VBG S
985 NP-SBJ → NN
983 PP-MNR → IN NP
983 NP-SBJ → DT
969 VP → VBN VP
...
100 VP → VBD PP-PRD
100 PRN → : NP :
100 NP → DT JJS
100 NP-CLR → NN
99 NP-SBJ-1 → DT NNP
98 VP → VBN NP PP-DIR
98 VP → VBD PP-TMP
98 PP-TMP → VBG NP
97 VP → VBD ADV-P-TMP VP
...
10 WHNP-1 → WRB JJ
10 VP → VP CC VP PP-TMP
10 VP → VP CC VP ADV-MNR
10 VP → VBZ S , SBAR-ADV
10 VP → VBZ S ADV-P-TMP
Rules in the Penn Treebank

rules in the training section:
32,728
(+ 52,257 lexicon)

rules in the dev section:
3,128
(<78%)

4,021
The Most Frequent Rules Are Very Frequent. The Rest Are Infrequent.
Evaluation of Parsing
Evaluation for Parsing: Parseval

constituents in gold standard trees

constituents in parser output trees
Defining Parseval

**labeled recall:** \( = \frac{\text{\# of correct constituents in candidate parse of } s}{\text{\# of correct constituents in treebank parse of } s} \)

**labeled precision:** \( = \frac{\text{\# of correct constituents in candidate parse of } s}{\text{\# of total constituents in candidate parse of } s} \)

**cross-brackets:** the number of crossed brackets (e.g. the number of constituents for which the treebank has a bracketing such as \(((A \ B) \ C)\) but the candidate parse has a bracketing such as \((A \ (B \ C))\)).
The F-Measure and $F_1$

$$F_\beta = \frac{\left(\beta^2 + 1\right) PR}{\beta^2 P + R}$$

$$F_1 = \frac{2PR}{P + R}$$
Probabilistic context-free grammars
Two Related Problems

• **Input:** sentence $w = (w_1, \ldots, w_n)$ and CFG $G$

• **Output (recognition):** true iff $w \in \text{Language}(G)$

• **Output (parsing):** one or more derivations for $w$, under $G$
Probabilistic Context-Free Grammars

Vocabulary of terminal symbols, $\Sigma$

Set of nonterminal symbols, $N$

Special start symbol $S \in N$

Production rules of the form $X \to \alpha$, each with a positive weight $p(X \to \alpha)$,

where

$X \in N$

$\alpha \in (N \cup \Sigma)^*$  (in CNF: $\alpha \in N^2 \cup \Sigma$)

$\forall X \in N, \ \Sigma_{\alpha} p(X \to \alpha) = 1$
A Sample PCFG

<table>
<thead>
<tr>
<th>Grammar Rule</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S \rightarrow NP \ VP$</td>
<td>.80</td>
</tr>
<tr>
<td>$S \rightarrow Aux \ NP \ VP$</td>
<td>.15</td>
</tr>
<tr>
<td>$S \rightarrow VP$</td>
<td>.05</td>
</tr>
<tr>
<td>$NP \rightarrow Det \ Nom$</td>
<td>.20</td>
</tr>
<tr>
<td>$NP \rightarrow Proper-Noun$</td>
<td>.35</td>
</tr>
<tr>
<td>$NP \rightarrow Nom$</td>
<td>.05</td>
</tr>
<tr>
<td>$NP \rightarrow Pronoun$</td>
<td>.40</td>
</tr>
<tr>
<td>$Nom \rightarrow Noun$</td>
<td>.75</td>
</tr>
<tr>
<td>$Nom \rightarrow Noun \ Nom$</td>
<td>.20</td>
</tr>
<tr>
<td>$Nom \rightarrow Proper-Noun \ Nom$</td>
<td>.05</td>
</tr>
<tr>
<td>$VP \rightarrow Verb$</td>
<td>.55</td>
</tr>
<tr>
<td>$VP \rightarrow Verb \ NP$</td>
<td>.40</td>
</tr>
<tr>
<td>$VP \rightarrow Verb \ NP \ NP$</td>
<td>.05</td>
</tr>
<tr>
<td>$Det \rightarrow that [.05]$</td>
<td>$\text{the} [.80]$</td>
</tr>
<tr>
<td>$Noun \rightarrow book$</td>
<td>[.10]</td>
</tr>
<tr>
<td>$Noun \rightarrow flights$</td>
<td>[.50]</td>
</tr>
<tr>
<td>$Noun \rightarrow meal$</td>
<td>[.40]</td>
</tr>
<tr>
<td>$Verb \rightarrow book$</td>
<td>[.30]</td>
</tr>
<tr>
<td>$Verb \rightarrow include$</td>
<td>[.30]</td>
</tr>
<tr>
<td>$Verb \rightarrow want$</td>
<td>[.40]</td>
</tr>
<tr>
<td>$Aux \rightarrow can$</td>
<td>[.40]</td>
</tr>
<tr>
<td>$Aux \rightarrow does$</td>
<td>[.30]</td>
</tr>
<tr>
<td>$Aux \rightarrow do$</td>
<td>[.30]</td>
</tr>
<tr>
<td>$Proper-Noun \rightarrow TWA$</td>
<td>[.40]</td>
</tr>
<tr>
<td>$Proper-Noun \rightarrow Denver$</td>
<td>[.40]</td>
</tr>
<tr>
<td>$Pronoun \rightarrow you [.40]$</td>
<td>$I [.60]$</td>
</tr>
</tbody>
</table>

**Figure 12.1** A PCFG; a probabilistic augmentation of the miniature English grammar and lexicon in Figure 10.2. These probabilities are not based on a corpus; they were made up merely for expository purposes.
The joint probability of a particular parse $T$ and sentence $S$, is defined as the product of the probabilities of all the rules $r$ used to expand each node $n$ in the parse tree:

$$P(T, S) = \prod_{n \in T} p(r(n))$$

**The Probability of a Parse Tree**

The joint probability of a particular parse $T$ and sentence $S$, is defined as the product of the probabilities of all the rules $r$ used to expand each node $n$ in the parse tree:
An Example—Disambiguation
### An Example—Disambiguation

- Consider the productions for each parse:

<table>
<thead>
<tr>
<th>Rules</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>S → Aux NP VP</td>
<td>.15</td>
</tr>
<tr>
<td>NP → Pro</td>
<td>.40</td>
</tr>
<tr>
<td>VP → V NP NP</td>
<td>.05</td>
</tr>
<tr>
<td>NP → Nom</td>
<td>.05</td>
</tr>
<tr>
<td>NP → PNoun</td>
<td>.35</td>
</tr>
<tr>
<td>Nom → Noun</td>
<td>.75</td>
</tr>
<tr>
<td>Aux → Can</td>
<td>.40</td>
</tr>
<tr>
<td>NP → Pro</td>
<td>.40</td>
</tr>
<tr>
<td>Pro → you</td>
<td>.40</td>
</tr>
<tr>
<td>Verb → book</td>
<td>.30</td>
</tr>
<tr>
<td>PNoun → TWA</td>
<td>.40</td>
</tr>
<tr>
<td>Noun → flights</td>
<td>.50</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Rules</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>S → Aux NP VP</td>
<td>.15</td>
</tr>
<tr>
<td>NP → Pro</td>
<td>.40</td>
</tr>
<tr>
<td>VP → V NP</td>
<td>.40</td>
</tr>
<tr>
<td>NP → Nom</td>
<td>.05</td>
</tr>
<tr>
<td>Nom → PNoun Nom</td>
<td>.05</td>
</tr>
<tr>
<td>Nom → Noun</td>
<td>.75</td>
</tr>
<tr>
<td>Aux → Can</td>
<td>.40</td>
</tr>
<tr>
<td>NP → Pro</td>
<td>.40</td>
</tr>
<tr>
<td>Pro → you</td>
<td>.40</td>
</tr>
<tr>
<td>Verb → book</td>
<td>.30</td>
</tr>
<tr>
<td>PNoun → TWA</td>
<td>.40</td>
</tr>
<tr>
<td>Noun → flights</td>
<td>.50</td>
</tr>
</tbody>
</table>
Probabilities of the Two Parses

\[ P(T_l) = \frac{.15 \times .40 \times .05 \times .05 \times .35 \times .75 \times .40 \times .40 \times .40 \times .30 \times .40 \times .50}{.15 \times .40 \times .40 \times .05 \times .05 \times .35 \times .75 \times .40 \times .40 \times .40 \times .30 \times .40 \times .50} = 1.5 \times 10^{-6} \]

\[ P(T_r) = \frac{.15 \times .40 \times .40 \times .05 \times .05 \times .35 \times .75 \times .40 \times .40 \times .40 \times .30 \times .40 \times .50}{.15 \times .40 \times .40 \times .05 \times .05 \times .35 \times .75 \times .40 \times .40 \times .40 \times .30 \times .40 \times .50} = 1.7 \times 10^{-6} \]

We favor the tree on the right in disambiguation because it has a higher probability.
What Can You Do With a PCFG?

• Just as with CFGs, PCFGs can be used for both parsing and generation, but they have advantages in both areas:
  • Parsing
    • CFGs are good for “precision” parsers
    • PCFGs are good for robust parsers
    • PCFGs can disambiguate
  • Generation
    • CFGs generate grammatical sentences
    • PCFGs can generate more natural sentences
Where Do the Probabilities in PCFGs Come From?

• From a **tree bank**

\[
P(\alpha \rightarrow \beta | \alpha) = \frac{\text{Count}(\alpha \rightarrow \beta)}{\sum_{\gamma} \text{Count}(\alpha \rightarrow \gamma)} = \frac{\text{Count}(\alpha \rightarrow \beta)}{\text{Count}(\alpha)}
\]

• From a **corpus**
  • Parse the corpus with your CFG
  • Count the rules for each parse
  • Normalize
  • **But wait, most sentences are ambiguous!**
    • “Keep a separate count for each parse of a sentence and weigh each partial count by the probability of the parse it appears in.”
PCFGs as a Noisy Channel

`delete all except the leaves`

PCFG

source

 derivation

`Y`

channel

decode

yield

`X`
Problems with PCFGs
Subject NPs are more likely to have *the*; Object NPs are more likely to have *a*.

An example of a structural dependency!
Vanilla PCFGs Cannot Capture Structural Dependencies

• (P)CFG rules apply independently
  • $S \rightarrow \textbf{NP} \ \textbf{VP}$
  • $\textbf{VP} \rightarrow V \ \textbf{NP}$
  • Same rules apply to all $\textbf{NPs}$, regardless of context
  • $\textit{the}$ more common in subjects than $a$
Vanilla PCFGs Cannot Capture Lexical Dependencies
Probabilistic Lexicalized
context free grammars
The Word that Defines/Constrains the Rest of a Constituent Is the Head

Noun phrases (NPs) are headed by nouns

Verb phrases (VPs) are headed by verbs

In lexicalized grammars/trees we augment the label with the head
A Example of a Lexicalized Tree

Figure 12.5  A lexicalized tree from Collins (1999).
Lexicalized Grammars

• Fancy lexicalized grammars exist (LFG, HPSG, etc.)
• We keep it simple
• A PCFG with more rules
Lexicalized Grammars

• Viewed in this way…
  • Lexicalized grammars are huge (perhaps impractically huge)
  • However, they can be used with the same algorithms we have already learned

• Lexicalized grammars require more training data than vanilla PCFGs, but they can capture probabilistic patterns that PCFGs could never capture

• In most contemporary applications, lexicalized grammars are chosen over vanilla PCFGs
Questions?