Statistical NLP
Spring 2007

Lecture 16: PCFGs
Dan Klein – UC Berkeley

(Speech) Lattices
- There was nothing magical about words spanning exactly one position.
- When working with speech, we generally don’t know how many words there are, or where they break.
- We can represent the possibilities as a lattice and parse these just as easily.

A Simple Chart Parser
- Chart parsers are sparse dynamic programs
- Ingredients:
  - Nodes: positions between words
  - Edges: spans of words with labels, represent the set of trees over those words rooted at x
  - A chart: records which edges we’ve built
  - An agenda: a holding pen for edges (a queue)
- We’re going to figure out:
  - What edges can we build?
  - All the ways we built them.

Word Edges
- An edge found for the first time is called discovered. Edges go into the agenda on discovery.
- To initialize, we discover all word edges.

Unary Projection
- When we pop an word edge off the agenda, we check the lexicon to see what tag edges we can build from it

The “Fundamental Rule”
- When we pop edges off of the agenda:
  - Check for unary projections (NNS → critics, NP → NNS)
    \[ Y[i,j] \text{ with } X \rightarrow Y \text{ forms } X[i,j] \]
  - Combine with edges already in our chart (this is sometimes called the fundamental rule)
    \[ Y[i,j] \text{ and } Z[j,k] \text{ with } X \rightarrow Y Z \text{ form } X[i,k] \]
  - Enqueue resulting edges (if newly discovered)
  - Record backtraces (called traversals)
  - Stick the popped edge in the chart
- Queries a chart must support:
  - Is edge X[i,j] in the chart?
  - What edges with label Y end at position j?
  - What edges with label Z start at position i?

---

A Simple Chart Parser

<table>
<thead>
<tr>
<th></th>
<th>critics</th>
<th>write</th>
<th>reviews</th>
<th>with</th>
<th>computers</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Word Edges

<table>
<thead>
<tr>
<th>Critics</th>
<th>Write</th>
<th>Reviews</th>
<th>With</th>
<th>Computers</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Unary Projection

<table>
<thead>
<tr>
<th>Critics</th>
<th>Write</th>
<th>Reviews</th>
<th>With</th>
<th>Computers</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The “Fundamental Rule”

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
An Example

Order Independence
- A nice property:
  - It doesn’t matter what policy we use to order the agenda (FIFO, LIFO, random).
  - Why? Invariant: before popping an edge:
    - Any edge \(X[i,j]\) that can be directly built from chart edges and a single grammar rule is either in the chart or in the agenda.
    - Convince yourselves this invariant holds!
    - This will not be true once we get weighted parsers.

Exploiting Substructure
- Each edge records all the ways it was built (locally)
- Can recursively extract trees
- A chart may represent too many parses to enumerate (how many?)

Empty Elements
- Sometimes we want to posit nodes in a parse tree that don’t contain any pronounced words:
  - These are easy to add to our chart parser!
  - For each position \(i\), add the “word” edge \(\epsilon[i,i]\)
  - Add rules like \(NP \rightarrow \epsilon\) to the grammar
  - That’s it!

Treebank Sentences

( (S (NP-SBJ) (VBD) the move) (VP followed (NP (NP a round) (PP OF (NP (NP similar increases) (PP by (NP other lenders)) (PP against (NP Arizona real estate loans)))))) (S-ADV (NP-SBJ) (VBD reflecting (NP (NP a continuing decline) (PP-LOC in (NP that market)))))) )

Treebank Parsing in 20 sec
- Need a PCFG for broad coverage parsing.
- Can take a grammar right off the trees (doesn’t work well):
- Better results by enriching the grammar (e.g., lexicalization).
- Can also get reasonable parsers without lexicalization.
N-Ary Rules, Grammar States

- Often we want to write grammar rules like
  \[ VP \rightarrow \text{VBD} \text{ NP} \text{ PP} \text{ PP} \]
  which are not binary.
- We can work with these rules by introducing new intermediate symbols (states) into our grammar:

```
[VP → VBD NP PP PP]
```

```
VBD NP PP
```

Treebank Grammar Scale

- Treebank grammars can be enormous!
  - As a set of FSTs, the raw grammar has \( \sim 10K \) states (why?).
  - Better parsers usually make the grammars larger, not smaller.

PCFGs and Independence

- Symbols in a PCFG define independence assumptions:

```
S → NP VP
NP → DT NN
```

- At any node, the material inside that node is independent of the material outside that node, given the label of that node.
- Any information that statistically connects behavior inside and outside a node must flow through that node.

Non-Independence I

- Independence assumptions are often too strong.
  - Example: the expansion of an NP is highly dependent on the parent of the NP (i.e., subjects vs. objects).
  - Also: the subject and object expansions are correlated!

Non-Independence II

- Who cares?
  - NB, HMMs, all make false assumptions!
  - For generation, consequences would be obvious.
  - For parsing, does it impact accuracy?

- Symptoms of overly strong assumptions:
  - Rewrites get used where they don’t belong.
  - Rewrites get used too often or too rarely.

Breaking Up the Symbols

- We can relax independence assumptions by encoding dependencies into the PCFG symbols:

  Parent annotation
  (Johnson 98)

  Marking possessive NPs

- What are the most useful "features" to encode?
Annotations

- Annotations split the grammar categories into sub-categories (in the original sense).
- Conditioning on history vs. annotating
  - \( P(NP^S \rightarrow PRP) \) is a lot like \( P(NP \rightarrow PRP \mid S) \)
  - \( P(NP\text{-POS} \rightarrow NNP \text{ POS}) \) isn’t history conditioning.
- Feature / unification grammars vs. annotation
  - Can think of a symbol like \( NP^NP\text{-POS} \) as \( NP \) [parent:NP, +POS]
- After parsing with an annotated grammar, the annotations are then stripped for evaluation.

Lexicalization

- Lexical heads important for certain classes of ambiguities (e.g., PP attachment):
- Lexicalizing grammar creates a much larger grammar. (cf. next week)
  - Sophisticated smoothing needed
  - Smarter parsing algorithms
  - More data needed
- How necessary is lexicalization?
  - Bilexical vs. monolexical selection
  - Closed vs. open class lexicalization

Unlexicalized PCFGs

- What is meant by an “unlexicalized” PCFG?
  - Grammar not systematically specified to the level of lexical items
    - \( NP \) [stocks] is not allowed
    - \( NP^S\text{-CC} \) is fine
  - Closed vs. open class words (\( NP^S \) [the])
    - Long tradition in linguistics of using function words as features or markers for selection
    - Contrary to the bilexical idea of semantic heads
    - Open-class selection really a proxy for semantics
  - It’s kind of a gradual transition from unlexicalized to lexicalized (but heavily smoothed) grammars.

Typical Experimental Setup

- Corpus: Penn Treebank, WSJ
  - Training: sections 02-21
  - Development: section 22 (here, first 20 files)
  - Test: section 23
- Accuracy – F1: harmonic mean of per-node labeled precision and recall.
- Here: also size – number of symbols in grammar.
  - Passive / complete symbols: \( NP, NP^S \)
  - Active / incomplete symbols: \( NP \rightarrow NP \text{ CC} \)

Multiple Annotations

- Each annotation done in succession
  - Order does matter
  - Too much annotation and we’ll have sparsity issues

Horizontal Markovization
Vertical Markovization

- Vertical Markov order: rewrites depend on past $k$ ancestor nodes. (cf. parent annotation)

![Vertical Markovization Diagram]

Vertical and Horizontal

- Examples:
  - Raw treebank: $v=1$, $h=\infty$
  - Johnson 98: $v=2$, $h=\infty$
  - Collins 99: $v=2$, $h=2$
  - Best F1: $v=3$, $h=2v$

![Vertical and Horizontal Diagram]

Unary Splits

- Problem: unary rewrites used to transmute categories so a high probability rule can be used.
- Solution: Mark unary rewrite sites with -U

![Unary Splits Diagram]

Tag Splits

- Problem: Treebank tags are too coarse.
- Example: Sentential, PP, and other prepositions are all marked IN.
- Partial Solution: Subdivide the IN tag.

![Tag Splits Diagram]

Other Tag Splits

- UNARY-DT: mark demonstratives as DT$^U$ ("the X" vs. "those")
- UNARY-RB: mark phrasal adverbs as RB$^U$ ("quickly" vs. "very")
- TAG-PA: mark tags with non-canonical parents ("not" is an RB$^VP$)
- SPLIT-AUX: mark auxiliary verbs with –AUX (cf. Charniak 97)
- SPLIT-CC: separate "but" and "&" from other conjunctions
- SPLIT-%: "%" gets its own tag.

![Other Tag Splits Diagram]

Treebank Splits

- The treebank comes with some annotations (e.g., -LOC, -SUBJ, etc).
  - Whole set together hurt the baseline.
  - One in particular is very useful (NP-TMP) when pushed down to the head tag (why?).
  - Can mark gapped S nodes as well.

![Treebank Splits Diagram]
Yield Splits

- Problem: sometimes the behavior of a category depends on something inside its future yield.
- Examples:
  - Possessive NPs
  - Finite vs. infinite VPs
  - Lexical heads!
- Solution: annotate future elements into nodes.
  - Lexicalized grammars do this (in very careful ways – why?).

<table>
<thead>
<tr>
<th>Annotation</th>
<th>F1</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Previous</td>
<td>92.3</td>
<td>9.7K</td>
</tr>
<tr>
<td>POSS-NP</td>
<td>83.1</td>
<td>9.8K</td>
</tr>
<tr>
<td>SPLIT-VP</td>
<td>85.7</td>
<td>10.5K</td>
</tr>
</tbody>
</table>

Distance / Recursion Splits

- Problem: vanilla PCFGs cannot distinguish attachment heights.
- Solution: mark a property of higher or lower sites:
  - Contains a verb.
  - Is (non)-recursive.
    - Base NPs [cf. Collins 99]
    - Right-recursive NPs

<table>
<thead>
<tr>
<th>Annotation</th>
<th>F1</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Previous</td>
<td>85.7</td>
<td>10.5K</td>
</tr>
<tr>
<td>BASE-NP</td>
<td>86.0</td>
<td>11.7K</td>
</tr>
<tr>
<td>DOMINATES-V</td>
<td>86.9</td>
<td>14.1K</td>
</tr>
<tr>
<td>RIGHT-REC-NP</td>
<td>87.0</td>
<td>15.2K</td>
</tr>
</tbody>
</table>

A Fully Annotated (Unlex) Tree

Some Test Set Results

<table>
<thead>
<tr>
<th>Parser</th>
<th>LP</th>
<th>LR</th>
<th>F1</th>
<th>CB</th>
<th>0 CB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Magerman 95</td>
<td>84.9</td>
<td>84.6</td>
<td>84.7</td>
<td>1.26</td>
<td>56.6</td>
</tr>
<tr>
<td>Collins 96</td>
<td>86.3</td>
<td>85.8</td>
<td>86.0</td>
<td>1.14</td>
<td>59.9</td>
</tr>
<tr>
<td>Unlexicalized</td>
<td>86.9</td>
<td>85.7</td>
<td>86.3</td>
<td>1.10</td>
<td>60.3</td>
</tr>
<tr>
<td>Charniak 97</td>
<td>87.4</td>
<td>87.5</td>
<td>87.4</td>
<td>1.00</td>
<td>62.1</td>
</tr>
<tr>
<td>Collins 99</td>
<td>86.7</td>
<td>88.6</td>
<td>88.6</td>
<td>0.90</td>
<td>67.1</td>
</tr>
</tbody>
</table>

- Beats “first generation” lexicalized parsers.
- Lots of room to improve – more complex models next.

The Game of Designing a Grammar

- Annotation refines base treebank symbols to improve statistical fit of the grammar
- Parent annotation [Johnson ‘98]

The Game of Designing a Grammar

- Annotation refines base treebank symbols to improve statistical fit of the grammar
- Parent annotation [Johnson ‘98]
- Head lexicalization [Collins ‘99, Charniak ‘00]
The Game of Designing a Grammar

- Annotation refines base treebank symbols to improve statistical fit of the grammar
  - Parent annotation [Johnson '98]
  - Head lexicalization [Collins '99, Charniak '00]
  - Automatic clustering?

Manual Annotation

- Manually split categories
  - NP: subject vs object
  - DT: determiners vs demonstratives
  - IN: sentential vs prepositional

- Advantages:
  - Fairly compact grammar
  - Linguistic motivations

- Disadvantages:
  - Performance leveled out
  - Manually annotated

<table>
<thead>
<tr>
<th>Model</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Naïve Treebank Grammar</td>
<td>72.6</td>
</tr>
<tr>
<td>Klein &amp; Manning '03</td>
<td>86.3</td>
</tr>
</tbody>
</table>

Automatic Annotation Induction

- Advantages:
  - Automatically learned:
    - Label all nodes with latent variables.
    - Same number $k$ of subcategories for all categories.

- Disadvantages:
  - Grammar gets too large
  - Most categories are oversplit while others are undersplit.

Learning Latent Annotations

EM algorithm:
- Brackets are known
- Base categories are known
- Only induce subcategories

Refinement of the DT tag

Hierarchical refinement
Adaptive Splitting

- Want to split complex categories more
- Idea: split everything, roll back splits which were least useful

Evaluate loss in likelihood from removing each split =
\[
\frac{\text{Data likelihood with split reversed}}{\text{Data likelihood with split}}
\]

No loss in accuracy when 50% of the splits are reversed.

Adaptive Splitting Results

<table>
<thead>
<tr>
<th>Model</th>
<th>Previous F1</th>
<th>With 50% Merging F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>50% Merging</td>
<td>88.4</td>
<td>89.5</td>
</tr>
<tr>
<td>Hierarchical Training</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Flat Training</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Number of Phrasal Subcategories

Final Results

<table>
<thead>
<tr>
<th>Parser</th>
<th>( F_1 \leq 40 ) words</th>
<th>( F_1 ) all words</th>
</tr>
</thead>
<tbody>
<tr>
<td>Klein &amp; Manning '03</td>
<td>86.3</td>
<td>85.7</td>
</tr>
<tr>
<td>Matsuzaki et al. '05</td>
<td>86.7</td>
<td>86.1</td>
</tr>
<tr>
<td>Collins '99</td>
<td>88.6</td>
<td>88.2</td>
</tr>
<tr>
<td>Charniak &amp; Johnson '05</td>
<td>90.1</td>
<td>89.6</td>
</tr>
<tr>
<td>Petrov et. al. 06</td>
<td>90.2</td>
<td>89.7</td>
</tr>
</tbody>
</table>
### Learned Splits

**Proper Nouns (NNP):**

<table>
<thead>
<tr>
<th>NNP-14</th>
<th>NNP-12</th>
<th>NNP-2</th>
<th>NNP-1</th>
<th>NNP-15</th>
<th>NNP-3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>J.</td>
<td>E.</td>
<td>L.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Bush</td>
<td>Noriega</td>
<td>Peters</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>New</td>
<td>San</td>
<td>Wall</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>York</td>
<td>Francisco</td>
<td>Street</td>
</tr>
</tbody>
</table>

**Personal pronouns (PRP):**

<table>
<thead>
<tr>
<th>PRP-0</th>
<th>PRP-1</th>
<th>PRP-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>It</td>
<td>He</td>
<td>I</td>
</tr>
<tr>
<td></td>
<td>it</td>
<td>he</td>
</tr>
<tr>
<td></td>
<td>them</td>
<td>him</td>
</tr>
</tbody>
</table>

### Learned Splits

**Relative adverbs (RBR):**

<table>
<thead>
<tr>
<th>RBR-0</th>
<th>further</th>
<th>lower</th>
<th>higher</th>
</tr>
</thead>
<tbody>
<tr>
<td>RBR-1</td>
<td>more</td>
<td>less</td>
<td>More</td>
</tr>
<tr>
<td>RBR-2</td>
<td>earlier</td>
<td>Earlier</td>
<td>later</td>
</tr>
</tbody>
</table>

**Cardinal Numbers (CD):**

<table>
<thead>
<tr>
<th>CD-7</th>
<th>one</th>
<th>two</th>
<th>Three</th>
</tr>
</thead>
<tbody>
<tr>
<td>CD-4</td>
<td>1989</td>
<td>1990</td>
<td>1988</td>
</tr>
<tr>
<td>CD-11</td>
<td>million</td>
<td>billion</td>
<td>trillion</td>
</tr>
<tr>
<td>CD-0</td>
<td>1</td>
<td>50</td>
<td>100</td>
</tr>
<tr>
<td>CD-3</td>
<td>1</td>
<td>30</td>
<td>31</td>
</tr>
<tr>
<td>CD-9</td>
<td>78</td>
<td>58</td>
<td>34</td>
</tr>
</tbody>
</table>