Lecture 14: PCFGs

Dan Klein – UC Berkeley

Treebank PCFGs [Charniak 96]

- Use PCFGs for broad coverage parsing
- Can take a grammar right off the trees (doesn’t work well):

```
ROOT → S 1
S → NP VP . 1
NP → PRP 1
VP → VBD ADJP 1
    .....  
```

<table>
<thead>
<tr>
<th>Model</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>72.0</td>
</tr>
</tbody>
</table>
Conditional Independence?

- Not every NP expansion can fill every NP slot
  - A grammar with symbols like “NP” won’t be context-free
  - Statistically, conditional independence too strong

Non-Independence

- 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!
Grammar Refinement

- Example: PP attachment

```
They
raised
a point of order
```

Grammar Refinement

```
S

NP - the

PRP VBD
She heard

VBD

DT NN
the noise
```

- Structure Annotation [Johnson '98, Klein&Manning '03]
- Lexicalization [Collins '99, Charniak '00]
- Latent Variables [Matsuzaki et al. 05, Petrov et al. '06]
The Game of Designing a Grammar

- Annotation refines base treebank symbols to improve statistical fit of the grammar
  - Structural annotation

Typical Experimental Setup

- Corpus: Penn Treebank, WSJ

<table>
<thead>
<tr>
<th>Training:</th>
<th>Development:</th>
<th>Test:</th>
</tr>
</thead>
<tbody>
<tr>
<td>sections</td>
<td>section</td>
<td>section</td>
</tr>
<tr>
<td>02-21</td>
<td>22 (here, first 20 files)</td>
<td>23</td>
</tr>
</tbody>
</table>

- 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 → NP CC
Vertical Markovization

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

Order 1

Order 2

Horizontal Markovization

Order 1

Order $\infty$
Unary Splits

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

```
Problem: Treebank tags are too coarse.

Example: Sentential, PP, and other prepositions are all marked IN.

Partial Solution:
- Subdivide the IN tag.
```
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^U^VP)
- SPLIT-AUX: mark auxiliary verbs with –AUX [cf. Charniak 97]
- SPLIT-CC: separate “but” and “&” from other conjunctions
- SPLIT-%: “%” gets its own tag.

<table>
<thead>
<tr>
<th></th>
<th>F1</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNARY-DT</td>
<td>80.4</td>
<td>8.1K</td>
</tr>
<tr>
<td>UNARY-RB</td>
<td>80.5</td>
<td>8.1K</td>
</tr>
<tr>
<td>TAG-PA</td>
<td>81.2</td>
<td>8.5K</td>
</tr>
<tr>
<td>SPLIT-AUX</td>
<td>81.6</td>
<td>9.0K</td>
</tr>
<tr>
<td>SPLIT-CC</td>
<td>81.7</td>
<td>9.1K</td>
</tr>
<tr>
<td>SPLIT-%</td>
<td>81.8</td>
<td>9.3K</td>
</tr>
</tbody>
</table>

A Fully Annotated (Unlex) Tree

```
ROOT

S^ROOT-v

"S NP S-B VP S-VBF-v ."S ."S

DT-U NP VBZ BE VP NP VP-B

This is NN NP NN NP

panic buying
```
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>Charniaky 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>88.7</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
  - Structural annotation [Johnson ’98, Klein and Manning 03]
  - Head lexicalization [Collins ’99, Charniak ’00]
Problems with PCFGs

- What's different between basic PCFG scores here?
- What (lexical) correlations need to be scored?

Lexicalized Trees

- Add “headwords” to each phrasal node
  - Syntactic vs. semantic heads
  - Headship not in (most) treebanks
  - Usually use head rules, e.g.:
    - NP:
      - Take leftmost NP
      - Take rightmost N
      - Take rightmost JJ
      - Take right child
    - VP:
      - Take leftmost VB
      - Take leftmost VP
      - Take left child
Lexicalized PCFGs?

- Problem: we now have to estimate probabilities like
  \[ VP(saw) \rightarrow VBD(saw)\ NP-C(\text{her})\ NP(\text{today}) \]

- Never going to get these atomically off of a treebank

- Solution: break up derivation into smaller steps

Lexical Derivation Steps

- A derivation of a local tree [Collins 99]

  - Choose a head tag and word
  - Choose a complement bag
  - Generate children (incl. adjuncts)
  - Recursively derive children
Lexicalized CKY

\[
\text{bestScore}(X, i, j, h) \\
\text{if } (j = i+1) \\
\quad \text{return } \text{tagScore}(X, s[i]) \\
\text{else} \\
\quad \text{return} \\
\quad \max \max \text{ score}(X[h] \rightarrow Y[h] \rightarrow Z[h']) \times \\
\quad \text{bestScore}(Y, i, k, h) \times \\
\quad \text{bestScore}(Z, k, j, h') \\
\quad \max \text{ score}(X[h] \rightarrow Y[h'] \rightarrow Z[h]) \times \\
\quad \text{bestScore}(Y, i, k, h') \times \\
\quad \text{bestScore}(Z, k, j, h)
\]

Pruning with Beams

- The Collins parser prunes with per-cell beams [Collins 99]
  - Essentially, run the O(n^3) CKY
  - Remember only a few hypotheses for each span <i,j>.
  - If we keep K hypotheses at each span, then we do at most O(nK^2) work per span (why?)
  - Keeps things more or less cubic

- Also: certain spans are forbidden entirely on the basis of punctuation (crucial for speed)
Pruning with a PCFG

- The Charniak parser prunes using a two-pass approach [Charniak 97+]
  - First, parse with the base grammar
  - For each X[i,j] calculate P(X[i,j,s])
    - This isn't trivial, and there are clever speed ups
  - Second, do the full $O(n^3)$ CKY
    - Skip any X[i,j] which had low (say, < 0.0001) posterior
    - Avoids almost all work in the second phase!

- Charniak et al 06: can use more passes
- Petrov et al 07: can use many more passes

Pruning with A*

- You can also speed up the search without sacrificing optimality
- For agenda-based parsers:
  - Can select which items to process first
  - Can do with any “figure of merit” [Charniak 98]
  - If your figure-of-merit is a valid A* heuristic, no loss of optimiality [Klein and Manning 03]
Projection-Based $A^*$

$\pi_{\text{SYNTACTIC}}$

$\pi_{\text{SEMANTIC}}$

A* Speedup

- Total time dominated by calculation of $A^*$ tables in each projection… $O(n^3)$
**Results**

- **Some results**
  - Collins 99 – 88.6 F1 (generative lexical)
  - Charniak and Johnson 05 – 89.7 / 91.3 F1 (generative lexical / reranked)
  - Petrov et al 06 – 90.7 F1 (generative unlexical)
  - McClosky et al 06 – 92.1 F1 (gen + rerank + self-train)

- **However**
  - Bilexical counts rarely make a difference (why?)
  - Gildea 01 – Removing bilexical counts costs < 0.5 F1

---

**The Game of Designing a Grammar**

- Annotation refines base treebank symbols to improve statistical fit of the grammar
  - Structural annotation
  - Head lexicalization
  - Automatic clustering?
Latent Variable Grammars

Parse Tree
Sentence $w$
Derivations $t : T$
Parameters $\Theta$

Learning Latent Annotations

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

Just like Forward-Backward for HMMs.
Refinement of the DT tag

Hierarchical refinement
Hierarchical Estimation Results

<table>
<thead>
<tr>
<th>Total Number of grammar symbols</th>
<th>Parsing accuracy (F1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>74</td>
</tr>
<tr>
<td>300</td>
<td>78</td>
</tr>
<tr>
<td>500</td>
<td>82</td>
</tr>
<tr>
<td>700</td>
<td>86</td>
</tr>
<tr>
<td>900</td>
<td>90</td>
</tr>
<tr>
<td>1100</td>
<td>88</td>
</tr>
<tr>
<td>1300</td>
<td>88</td>
</tr>
<tr>
<td>1500</td>
<td>86</td>
</tr>
<tr>
<td>1700</td>
<td>88</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Model</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flat Training</td>
<td>87.3</td>
</tr>
<tr>
<td>Hierarchical Training</td>
<td>88.4</td>
</tr>
</tbody>
</table>

Refinement of the , tag

- Splitting all categories equally is wasteful:
Adaptive Splitting

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

Adaptive Splitting Results

<table>
<thead>
<tr>
<th>Model</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Previous</td>
<td>88.4</td>
</tr>
<tr>
<td>With 50% Merging</td>
<td>89.5</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>John</td>
<td>Robert</td>
<td>James</td>
<td>J.</td>
<td>E.</td>
<td>L.</td>
</tr>
<tr>
<td>Robinson</td>
<td>Noriega</td>
<td>Peters</td>
<td>New</td>
<td>San</td>
<td>Wall</td>
</tr>
<tr>
<td>York</td>
<td>Francisco</td>
<td>Street</td>
<td></td>
<td></td>
<td></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>it</td>
<td>he</td>
<td>they</td>
</tr>
<tr>
<td>it</td>
<td>them</td>
<td>him</td>
</tr>
</tbody>
</table>

#### Relative adverbs (RBR):

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

#### Cardinal Numbers (CD):

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

- Example: PP attachment

For each chart item $X[i,j]$, compute posterior probability:

$$\frac{P_{\text{IN}}(X, i, j) \cdot P_{\text{OUT}}(X, i, j)}{P_{\text{IN}}(\text{root, 0, n})} < \text{threshold}$$

E.g. consider the span 5 to 12:
Bracket Posteriors

Hierarchical Pruning

course:

split in two:

split in four:

split in eight:
## Final Results (Accuracy)

<table>
<thead>
<tr>
<th></th>
<th>ENG</th>
<th></th>
<th>GER</th>
<th></th>
<th>CHN</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>≤ 40 words F1</td>
<td>all F1</td>
<td></td>
<td>≤ 40 words F1</td>
<td>all F1</td>
<td></td>
</tr>
<tr>
<td>Charniak&amp;Johnson ‘05 (generative)</td>
<td>90.1</td>
<td>89.6</td>
<td></td>
<td>Split / Merge</td>
<td>90.6</td>
<td>90.1</td>
</tr>
</tbody>
</table>

Still higher numbers from reranking / self-training methods