SI485i : NLP

Set 9
Advanced PCFGs

Some slides from Chris Manning
Evaluating CKY

• How do we know if our parser works?

• Count the number of correct labels in your tree...the label and the span it dominates must both be correct.
  • [ label, start, finish ]

• Precision, Recall, F1 Score
Evaluation Metrics

- $C =$ number of correct non-terminals
- $M =$ total number of non-terminals produced
- $N =$ total number of non-terminals in the gold tree

- **Precision** $= \frac{C}{M}$
- **Recall** $= \frac{C}{N}$
- **F1 Score** (harmonic mean) $= \frac{2 \times P \times R}{P + R}$
Are PCFGs any good?

- Always produces *some* tree.
- Trees are reasonably good, giving a decent idea as to the correct structure.
- However, trees are rarely totally correct. Contain lots of errors.

- WSJ parsing accuracy = 73% F1
What’s missing in PCFGs?

This choice of VP->VP PP has nothing to do with the actual words in the sentence.

$t_2$:

```
S_1.0
  NP_0.1
   astronomers
  VP_0.3
    VP_0.7
     V_1.0
      saw
     NP_0.18
     stars
    PP_1.0
     P_1.0
      with
     NP_0.18
      ears
```
Words barely affect structure.

Correct!!!

Incorrect
PCFGs and their words

• The words in a PCFG only link to their POS tags.
• The head word of a phrase contains a ton of information that the grammar does not use.

• Attachment ambiguity
  • “The astronomer saw the moon with the telescope.”
• Coordination
  • “The dogs in the house and the cats.”
• Subcategorization
  • “give” versus “jump”
PCFGs and their words

• The words are ignored due to our current independence assumptions in the PCFG.

• The words under the NP do not affect the VP.

• Any information that statistically connects above and below a node must flow through that node, so regions are independent given that central node.
PCFGs and independence

• Independence assumptions are too strong.

• The NPs under an S are typically what syntactic category? What about under a VP?
Relax the Independence

- **Thought question:** how could you change your grammar to encode these probabilities?
Vertical Markovization

- Expand the grammar
  - NP^S → DT NN
  - NP^VP → DT NN
  - NP^NP → DT NN
  - etc.

Diagram:
- Order 1
  - S
    - NP
      - PRP
      - VBD
      - ADJP
    - VP
      - He
      - was
      - right
- Order 2
  - S^ROOT
    - NP^S
      - PRP
      - VBD
      - ADVP^VP
    - VP^S
      - He
      - was
      - right
Vertical Markovization

- Markovization can use k ancestors, not just k=1.
  - NP^VP^S -> DT NN
- The best distance in early experiments was k=3.

- **WARNING**: doesn’t this explode the size of the grammar? Yes. But the algorithm is O(n^3), so a bigger grammar (not n) can hurt but not excessively. The gain in performance can be worth it.
Horizontal Markovization

- Similar to vertical.
- Don’t label with the parents, but now label with the left siblings in your immediate tree.
- This takes into context where you are in your local tree structure.
## Markovization Results

<table>
<thead>
<tr>
<th>Vertical Order</th>
<th>Horizontal Markov Order</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$h = 0$</td>
</tr>
<tr>
<td>$v = 1$ No annotation</td>
<td>71.27</td>
</tr>
<tr>
<td></td>
<td>(854)</td>
</tr>
<tr>
<td>$v \leq 2$ Sel. Parents</td>
<td>74.75</td>
</tr>
<tr>
<td></td>
<td>(2285)</td>
</tr>
<tr>
<td>$v = 2$ All Parents</td>
<td>74.68</td>
</tr>
<tr>
<td></td>
<td>(2984)</td>
</tr>
<tr>
<td>$v \leq 3$ Sel. GParents</td>
<td>76.50</td>
</tr>
<tr>
<td></td>
<td>(4943)</td>
</tr>
<tr>
<td>$v = 3$ All GParents</td>
<td>76.74</td>
</tr>
<tr>
<td></td>
<td>(7797)</td>
</tr>
</tbody>
</table>

Figure 2: Markovizations: $F_1$ and grammar size.

Figure from Klein & Manning (2003)
More Context in the Grammar

• Markovization is just the beginning. You can label non-terminals with all kinds of other useful information
  • Label nodes dominating verbs
  • Label NP as NP-POSS that has a possessive child (his dog)
  • Split IN tags into 6 categories!
  • Label CONJ tags if they are but or and
  • Give % its own tag
  • Etc.
### Annotated Grammar Results

<table>
<thead>
<tr>
<th>Annotation</th>
<th>Cumulative</th>
<th>Indiv.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Size</td>
<td>$F_1$</td>
</tr>
<tr>
<td><strong>Baseline</strong> ($v \leq 2, h \leq 2$)</td>
<td>7619</td>
<td>77.77</td>
</tr>
<tr>
<td>UNARY-INTERNAL</td>
<td>8065</td>
<td>78.32</td>
</tr>
<tr>
<td>UNARY-DT</td>
<td>8066</td>
<td>78.48</td>
</tr>
<tr>
<td>UNARY-RB</td>
<td>8069</td>
<td>78.86</td>
</tr>
<tr>
<td>TAG-PA</td>
<td>8520</td>
<td>80.62</td>
</tr>
<tr>
<td>SPLIT-IN</td>
<td>8541</td>
<td>81.19</td>
</tr>
<tr>
<td>SPLIT-AUX</td>
<td>9034</td>
<td>81.66</td>
</tr>
<tr>
<td>SPLIT-CC</td>
<td>9190</td>
<td>81.69</td>
</tr>
<tr>
<td>SPLIT-%</td>
<td>9255</td>
<td>81.81</td>
</tr>
<tr>
<td>TMP-NP</td>
<td>9594</td>
<td>82.25</td>
</tr>
<tr>
<td>GAPPED-S</td>
<td>9741</td>
<td>82.28</td>
</tr>
<tr>
<td>POSS-NP</td>
<td>9820</td>
<td>83.06</td>
</tr>
<tr>
<td>SPLIT-VP</td>
<td>10499</td>
<td>85.72</td>
</tr>
<tr>
<td>BASE-NP</td>
<td>11660</td>
<td>86.04</td>
</tr>
<tr>
<td>DOMINATES-V</td>
<td>14097</td>
<td>86.91</td>
</tr>
<tr>
<td>RIGHT-REC-NP</td>
<td>15276</td>
<td>87.04</td>
</tr>
</tbody>
</table>

Figure from Klein & Manning (2003)
Lexicalization

• Markovization and these grammar additions relax the independence assumptions between “neighbor” nodes.

• We still haven’t used the words yet.

• **Lexicalization** is the process of adding the main word of the subtree to its non-terminal parent.
Lexicalization

- The *head word* of a phrase is the main content-bearing word.
- Use the *head word* to label non-terminals.
Lexicalization Benefits

- PP-attachment problems are better modeled
  - “announced rates in january”
  - “announced in january rates”
  - The VP-announce will prefer having “in MONTH” as its child

- Subcategorization frames are now used!
  - VP-give expects two NP children
  - VP-sit expects no NP children, maybe one PP

- And many others…
Lexicalization and Frames

- Different probabilities of each VP rule if lexicalized with each of these four verbs:

<table>
<thead>
<tr>
<th>Local Tree</th>
<th>come</th>
<th>take</th>
<th>think</th>
<th>want</th>
</tr>
</thead>
<tbody>
<tr>
<td>VP → V</td>
<td>9.5%</td>
<td>2.6%</td>
<td>4.6%</td>
<td>5.7%</td>
</tr>
<tr>
<td>VP → V NP</td>
<td>1.1%</td>
<td>32.1%</td>
<td>0.2%</td>
<td>13.9%</td>
</tr>
<tr>
<td>VP → V PP</td>
<td>34.5%</td>
<td>3.1%</td>
<td>7.1%</td>
<td>0.3%</td>
</tr>
<tr>
<td>VP → V SBAR</td>
<td>6.6%</td>
<td>0.3%</td>
<td>73.0%</td>
<td>0.2%</td>
</tr>
<tr>
<td>VP → V S</td>
<td>2.2%</td>
<td>1.3%</td>
<td>4.8%</td>
<td>70.8%</td>
</tr>
<tr>
<td>VP → V NP S</td>
<td>0.1%</td>
<td>5.7%</td>
<td>0.0%</td>
<td>0.3%</td>
</tr>
<tr>
<td>VP → V PRT NP</td>
<td>0.3%</td>
<td>5.8%</td>
<td>0.0%</td>
<td>0.0%</td>
</tr>
<tr>
<td>VP → V PRT PP</td>
<td>6.1%</td>
<td>1.5%</td>
<td>0.2%</td>
<td>0.0%</td>
</tr>
</tbody>
</table>
Lexicalization

Independence Assumptions

- PCFGs

```
S
  /\NP  VP
  /\   /\  /
  DT N  V  DT N
  the lawyer questioned the witness
```

- Lexicalized PCFGs

```
S(questioned)
  /\NP lawyer VP questioned
  /\   /\   /
  DT N  V  DT N
  the lawyer questioned the witness
```
Exercise!

The plane flew heavy cargo with its big engines.

1. Draw the parse tree. Binary rules not required.
2. Add lexicalization to the grammar rules.
3. Add 2\textsuperscript{nd} order vertical markovization.
Putting it all together

• Lexicalized rules give you a massive gain. This was a big breakthrough in the 90’s.
• You can combine lexicalized rules with markovization and all other features.

• Grammars explode.
• Lexicalization … there are lots of details and backoff models that are required to make this work in reasonable time (not covered in this class).
State of the Art

- Parsing doesn’t have to use these PCFG models.
- **Discriminative Learning** has been used to get the best gains. Instead of computing probabilities from MLE counts, it **weights** each rule through optimization techniques that we do not cover in this class.

- The best parsers output multiple trees, and then use a different algorithm to **rank** those possibilities.
- Best F1 performance: low-mid 90’s.
Key Ideas

1. Parsing evaluation: precision/recall/F1
2. Independence assumptions of non-terminals
3. Markovization of grammar rules
4. Adding misc. features to rules
5. Lexicalization of grammar rules