Vanilla PCFG Parsing

\[ T^* = \arg \max_T Pr(T|S) \]

\[ Pr(T|S) = \prod_{\alpha \rightarrow \beta \in T} Pr(\alpha \rightarrow \beta | \alpha) \]

<table>
<thead>
<tr>
<th>DERIVATION</th>
<th>RULES USED</th>
<th>PROBABILITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>S \rightarrow NP VP</td>
<td>1.0</td>
</tr>
<tr>
<td>NP VP</td>
<td>NP \rightarrow DT N</td>
<td>0.3</td>
</tr>
<tr>
<td>DT N VP</td>
<td>DT \rightarrow the</td>
<td>1.0</td>
</tr>
<tr>
<td>the N VP</td>
<td>N \rightarrow dog</td>
<td>0.1</td>
</tr>
<tr>
<td>the dog VP</td>
<td>VP \rightarrow VB</td>
<td>0.4</td>
</tr>
<tr>
<td>the dog VB</td>
<td>VB \rightarrowlaughs</td>
<td>0.5</td>
</tr>
<tr>
<td>the dog laughs</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**TOTAL PROBABILITY** = 1.0 \times 0.3 \times 1.0 \times 0.1 \times 0.4 \times 0.5
A rule application is not necessarily independent of its context:

- Parents, children, or sibling categories influence choice of rule.
  
  e.g., NPs under S different from NPs under VP

- Head word of the current child constituent and the head word of the phrase influence choice of rule.
  
  e.g., Prepositional phrases attachment depend on the head verb.
Lexicalized Charniak Parser

- Key idea is to identify heads of constituents and use them to condition probabilities.
  - There are a handful of rules that specify how to identify heads.

- Probability of lexicalized parse tree is computed using these two quantities.

\[
P(\text{cur}_\text{head} = \text{profits} \mid \text{cur}_\text{category} = \text{NP}, \text{parent}_\text{head} = \text{rose}, \text{parent}_\text{category} = \text{S})
\]

\[
P(\text{rule} = r_i \mid \text{cur}_\text{head} = \text{profits}, \text{cur}_\text{category} = \text{NP}, \text{parent}_\text{category} = \text{S})
\]
• Why bother?
  – Lexicalization increases grammar size
  – Estimation headaches.
  – Domain adaptation is an issue.
  – Asymptotic complexity jumps to $O(N^5)$.
    • Requires clever algorithms to get it down to $O(N^3)$.
      [Eisner and Satta, 1996]

• What is the main idea?
  – Improve vanilla PCFG by adding different forms of context annotations.
There is a horizontal context and a vertical context.

Vanilla PCFG
-- Conditions on the immediate vertical context i.e., parent.
-- Uses all of the horizontal context i.e., expands to all children categories simultaneously.
Horizontal Markovization

**Intuition:**

Estimating over entire RHS can lead to poor estimates when RHS has many non-terminals.

Break it down by assuming that expansions from **head** are independent given immediate neighbors.

**Example:**

\[
\Pr(VP \rightarrow \text{VBZ NP PP PP} | VP) = \Pr(<VP:[\text{VBZ}]> \rightarrow \text{VBZ} | <VP:[\text{VBZ}]>) \times \\
\Pr(<VP:[\text{VBZ}]…NP> \rightarrow <VP:[\text{VBZ}]>NP | <VP:[\text{VBZ}]…NP>) \times \\
\Pr(<VP:[\text{VBZ}]…PP> \rightarrow <VP:[\text{VBZ}]>NP PP | <VP:[\text{VBZ}]…PP>)
\]
**Intuition:**

Expanding conditioning on the one parent assumes too little vertical context.

Use all ancestors as vertical context. Explore various length vertical histories.

**Example:**

\[ \text{Pr}(T|S) = ??? \]
### Markovization Results

**Summary:**

Adding vertical context helps.

Restricting horizontal context to depend on fewer ancestors helps.

Trade-off between grammar size and utility of history.

<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 (854)</td>
</tr>
<tr>
<td>$v \leq 2$ Sel. Parents</td>
<td>74.75 (2285)</td>
</tr>
<tr>
<td>$v = 2$ All Parents</td>
<td>74.68 (2984)</td>
</tr>
<tr>
<td>$v \leq 3$ Sel. GParents</td>
<td>76.50 (4943)</td>
</tr>
<tr>
<td>$v = 3$ All GParents</td>
<td>76.74 (7797)</td>
</tr>
</tbody>
</table>
Unary production $S^\text{VP} \rightarrow \text{VP}^S$ is incorrectly used here. The content of the expanded VP precludes the use of this rule. Internal S nodes are often complements of communication verbs. e.g., She thought that John was killed in the accident.

Mark the non-terminal in unary productions to say it has only one child.
Mark nodes that have no siblings.

Useful in the case of pre-terminals where internal annotation is meaningless
  -- All pre-terminal to terminal expansions are unary!

PTB conflates demonstratives (those, that) from true determiners (a, the).
  -- Adding UNARY-DT distinguishes between these two cases.

  e.g., The apples that were good vs. Those apples were good.
Many POS tags in PTB conflate distinct categories with different attachment preferences.

E.g. Preposition tag IN can be sub-ordinating conjunctions or regular prepositions.

Sub-ordinating conjunctions typically associate with a sentence category (S) to form SBARs.

1) Mark pre-terminals with parent information when they occur in non-canonical categories.

Tags associating with non-canonical categories have a specific distribution.

2) SPLIT-IN to mark prep categories.
Other Modifications

- SPLIT-CC, SPLIT-VP etc.
- Mark S nodes with empty subjects (GAPPED-S).
  - She was planning [ ____ to apply for the position.]
- TMP (temporal tags) are helpful.
  - TMP tags on NPs are propagated down from heads.
Present tense verbs do not usually take infinitival verb complements
-- “is” doesn’t take the infinitive “panic”, rather panic is a modifier on buying.
-- Marking non-infinitival verbs as VBFs fixes the problem.

Similarly marking POSS-NP constructions also helps.
-- In PTB annotations, NP → NP * is used only for possessives.
  e.g., John’s pizza as opposed to Long Island beaches
How far can an unlexicalized grammar go?

Pretty far! Nearly a 10% absolute improvement in F1.

<table>
<thead>
<tr>
<th>Annotation</th>
<th>Cumulative</th>
<th>Indiv.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline ($v \leq 2, h \leq 2$)</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>
So what is being done here?

- Linguistic insights i.e., knowledge about language constructions are being used.

- What is problematic about this approach?
  - Well, we need people knowledgeable about language (aka linguists).
  - Adapting to new domains requires manual intervention.

- Not sure if we’ve captured all relevant insights!
Summary

• One can think of many of these annotations as introducing sub-categories.

• By sub-categorizing are we effectively doing what lexicalization does?
  – The sub-categorization or state-splitting can never quite get close to the sparsity issues of open-class words.
  – Care taken to not back-off to uncategorized rules.

• Is there a way to automatically induce these sub-categories rather than manually specifying them?
  – Latent variable models w/ EM [Petrov and Klein, 2007]

• Essentially there is a trade-off in grammar size vs. effectiveness.
  – What are the implications for how much data we need?
  – Can un-lexicalized grammars be learnt sooner?