Recap: Adding Head Words/Tags to Trees

- We now have **lexicalized** context-free rules, e.g.,

\[
S(\text{questioned}, \text{Vt}) \Rightarrow \text{NP(lawyer, NN)} \quad \text{VP(questioned, Vt)}
\]

Recap: Lexicalized PCFGs

- We now need to estimate rule probabilities such as

\[
Prob(S(\text{questioned}, \text{Vt}) \Rightarrow \text{NP(lawyer, NN)} \quad \text{VP(questioned, Vt)} \mid S(\text{questioned}, \text{Vt}))
\]

- Sparse data is a problem. We have a **huge** number of non-terminals, and a **huge** number of possible rules. We have to work hard to estimate these rule probabilities...

- Once we have estimated these rule probabilities, we can find the highest scoring parse tree under the lexicalized PCFG using dynamic programming methods (see Problem set 1).

Recap: Charniak’s Model

- The general form of a lexicalized rule is as follows:

\[
X(h, t) \Rightarrow L_n(lw_n, lt_n) \cdots L_1(lw_1, lt_1) \ H(h, t) \ R_1(rw_1, rt_1) \cdots R_m(rw_m, rt_m)
\]

- Charniak’s model decomposes the probability of each rule as:

\[
Prob(X(h, t) \Rightarrow L_n(lw_n, lt_n) \cdots L_1(lw_1, lt_1) \ H(h, t) \ R_1(rw_1, rt_1) \cdots R_m(rw_m, rt_m) \mid X(h, t))
\]

\[
\times \prod_{i=1}^{n} \ Prob(lw_i \mid X(h, t), H, L_i(lt_i)) \times \prod_{i=1}^{m} \ Prob(rw_i \mid X(h, t), H, R_i(rt_i))
\]

- For example,

\[
Prob(S(\text{questioned}, \text{Vt}) \Rightarrow \text{NP(lawyer, NN)} \quad \text{VP(questioned, Vt)} \mid S(\text{questioned}, \text{Vt}))
\]

\[
= Prob(S(\text{questioned}, \text{Vt}) \Rightarrow \text{NP(NN)} \quad \text{VP(Vt)} \mid S(\text{questioned}, \text{Vt}))
\]

\[
\times Prob(\text{lawyer} \mid S(\text{questioned}, \text{Vt}), \text{VP}, \text{NP(NN)})
\]
Motivation for Breaking Down Rules

- First step of decomposition of (Charniak 1997):
  \[ S(\text{questioned}, \text{Vt}) \]

  \[ \downarrow \]

  \[ P(\text{NP(NN) VP} \mid S(\text{questioned}, \text{Vt})) \]

  \[ \text{NP(\text{NN}) VP(\text{questioned}, \text{Vt})} \]

- Relies on counts of entire rules
- These counts are sparse:
  - 40,000 sentences from Penn treebank have 12,409 rules.
  - 15% of all test data sentences contain a rule never seen in training

The General Form of Model 1

- The general form of a lexicalized rule is as follows:
  \[ X(h,t) \Rightarrow L_n(lw_n,l_t_n) \ldots L_1(lw_1,l_t_1) H(h,t) R_1(rw_1,rt_1) \ldots R_m(rw_m,rt_m) \]

- Collins model 1 decomposes the probability of each rule as:
  \[
  P_h(H \mid X, h, t) \times \\
  \prod_{i=1}^{n} P_d(L_i(lw_i,l_t_i) \mid X, H, h, t, \text{LEFT}) \times \\
  P_d(\text{STOP} \mid X, H, h, t, \text{LEFT}) \times \\
  \prod_{i=1}^{m} P_d(R_i(rw_i,rt_i) \mid X, H, h, t, \text{RIGHT}) \times \\
  P_d(\text{STOP} \mid X, H, h, t, \text{RIGHT})
  \]

Modeling Rule Productions as Markov Processes

- Collins (1997), Model 1

  \[ S(\text{told}, \text{V}) \]

  \[ \text{STOP NP(yesterday,NN) NP(Hillary,NNP) VP(\text{told}, \text{V}) STOP} \]

  We first generate the head label of the rule
  Then generate the left modifiers
  Then generate the right modifiers

  \[
  P_h(\text{VP} \mid S, \text{told}, \text{V}) \times \\
  P_d(\text{NP(Hillary,NNP)} \mid S, \text{VP,told, V,LEFT}) \times \\
  P_d(\text{NP(yesterday,NN)} \mid S, \text{VP,told, V,LEFT}) \times \\
  P_d(\text{STOP} \mid S, \text{VP,told, V,LEFT}) \times \\
  P_d(\text{STOP} \mid S, \text{VP,told, V,RIGHT})
  \]
Overview of Today’s Lecture

- Refinements to Model 1
- Evaluating parsing models
- Extensions to the parsing models

A Refinement: Adding a Distance Variable

- \( \Delta = 1 \) if position is adjacent to the head.

The Final Probabilities

\[
P_h(VP \mid S, told, V) \times 
P_d(NP(Hillary, NNP) \mid S, VP, told, V, \text{LEFT}) \times 
P_d(NP(yesterday, NN) \mid S, VP, told, V, \text{LEFT}, \Delta = 0) \times 
P_d(STOP \mid S, VP, told, V, \text{RIGHT}, \Delta = 1)
\]
Adding the Complement/Adjunct Distinction

- **Hillary** is the subject
- *yesterday* is a temporal modifier
- **But nothing to distinguish them.**

Complements vs. Adjuncts

- Complements are closely related to the head they modify, adjuncts are more indirectly related
- Complements are usually arguments of the thing they modify yesterday Hillary told ... ⇒ **Hillary is doing the telling**
- Adjuncts add modifying information: time, place, manner etc. yesterday Hillary told ... ⇒ **yesterday is a temporal modifier**
- Complements are usually required, adjuncts are optional

yesterday Hillary told ... (grammatical) vs. Hillary told ... (grammatical) vs. yesterday told ... (ungrammatical)

Adding Tags Making the Complement/Adjunct Distinction

- **Bill** is the object
- *yesterday* is a temporal modifier
- **But nothing to distinguish them.**
Adding Tags Making the Complement/Adjunct Distinction

Adding Subcategorization Probabilities

- Step 1: generate category of head child

- Step 2: choose left subcategorization frame

- Step 3: generate left modifiers in a Markov chain
Summary

- Identify heads of rules ⇒ dependency representations
- Presented two variants of PCFG methods applied to *lexicalized grammars.*
  - Break generation of rule down into small (markov process) steps
  - Build dependencies back up (distance, subcategorization)

Overview of Today’s Lecture

- Refinements to Model 1
- Evaluating parsing models
- Extensions to the parsing models

Evaluation: Representing Trees as Constituents

Precision and Recall

<table>
<thead>
<tr>
<th>Label</th>
<th>Start Point</th>
<th>End Point</th>
</tr>
</thead>
<tbody>
<tr>
<td>NP</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>NP</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>NP</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>PP</td>
<td>6</td>
<td>8</td>
</tr>
<tr>
<td>NP</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>VP</td>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>S</td>
<td>1</td>
<td>8</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Label</th>
<th>Start Point</th>
<th>End Point</th>
</tr>
</thead>
<tbody>
<tr>
<td>NP</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>NP</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>PP</td>
<td>6</td>
<td>8</td>
</tr>
<tr>
<td>NP</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>VP</td>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>S</td>
<td>1</td>
<td>8</td>
</tr>
</tbody>
</table>

- \( G = \text{number of constituents in gold standard} = 7 \)
- \( P = \text{number in parse output} = 6 \)
- \( C = \text{number correct} = 6 \)

Recall = \(100\% \times \frac{C}{G} = 100\% \times \frac{6}{7} \)  
Precision = \(100\% \times \frac{C}{P} = 100\% \times \frac{6}{6} \)
### Results

<table>
<thead>
<tr>
<th>Method</th>
<th>Recall</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCFGs (Charniak 97)</td>
<td>70.6%</td>
<td>74.8%</td>
</tr>
<tr>
<td>Conditional Models – Decision Trees (Magerman 95)</td>
<td>84.0%</td>
<td>84.3%</td>
</tr>
<tr>
<td>Generative Lexicalized Model (Charniak 97)</td>
<td>86.7%</td>
<td>86.6%</td>
</tr>
<tr>
<td>Model 1 (no subcategorization)</td>
<td>87.5%</td>
<td>87.7%</td>
</tr>
<tr>
<td>Model 2 (subcategorization)</td>
<td>88.1%</td>
<td>88.3%</td>
</tr>
</tbody>
</table>

### Effect of the Different Features

<table>
<thead>
<tr>
<th>MODEL</th>
<th>A</th>
<th>V</th>
<th>R</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model 1</td>
<td>NO</td>
<td>NO</td>
<td>75.0%</td>
<td>76.5%</td>
</tr>
<tr>
<td>Model 1</td>
<td>YES</td>
<td>NO</td>
<td>86.6%</td>
<td>86.7%</td>
</tr>
<tr>
<td>Model 1</td>
<td>YES</td>
<td>YES</td>
<td>87.8%</td>
<td>88.2%</td>
</tr>
<tr>
<td>Model 2</td>
<td>NO</td>
<td>NO</td>
<td>85.1%</td>
<td>86.8%</td>
</tr>
<tr>
<td>Model 2</td>
<td>YES</td>
<td>NO</td>
<td>87.7%</td>
<td>87.8%</td>
</tr>
<tr>
<td>Model 2</td>
<td>YES</td>
<td>YES</td>
<td>88.7%</td>
<td>89.0%</td>
</tr>
</tbody>
</table>

Results on Section 0 of the WSJ Treebank. Model 1 has no subcategorization, Model 2 has subcategorization. A = YES, V = YES mean that the adjacency/verb conditions respectively were used in the distance measure. R/P = recall/precision.

### Weaknesses of Precision and Recall

<table>
<thead>
<tr>
<th>Label</th>
<th>Start Point</th>
<th>End Point</th>
</tr>
</thead>
<tbody>
<tr>
<td>NP</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>NP</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>NP</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>PP</td>
<td>6</td>
<td>8</td>
</tr>
<tr>
<td>NP</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>VP</td>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>S</td>
<td>1</td>
<td>8</td>
</tr>
</tbody>
</table>

**NP attachment:**

(S (NP The men) (VP dumped (NP (NP large sacks) (PP of (NP the substance)))))

**VP attachment:**

(S (NP The men) (VP dumped (NP large sacks) (PP of (NP the substance))))
Dependency Accuracies

- All parses for a sentence with \( n \) words have \( n \) dependencies. Report a single figure, dependency accuracy.

- Model 2 with all features scores 88.3\% dependency accuracy (91\% if you ignore non-terminal labels on dependencies).

- Can calculate precision/recall on particular dependency types e.g., look at all subject/verb dependencies ⇒ all dependencies with label (S, VP, NP-C, LEFT)

\[
\text{Recall} = \frac{\text{number of subject/verb dependencies correct}}{\text{number of subject/verb dependencies in gold standard}} \\
\text{Precision} = \frac{\text{number of subject/verb dependencies correct}}{\text{number of subject/verb dependencies in parser’s output}}
\]

<table>
<thead>
<tr>
<th>R</th>
<th>CP</th>
<th>P</th>
<th>Count</th>
<th>Relation</th>
<th>Rec</th>
<th>Prec</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>29.65</td>
<td>29.65</td>
<td>11786</td>
<td>NPB TAG TAG L</td>
<td>94.60</td>
<td>93.46</td>
</tr>
<tr>
<td>2</td>
<td>40.55</td>
<td>10.90</td>
<td>4335</td>
<td>PP TAG NP-C R</td>
<td>94.72</td>
<td>94.04</td>
</tr>
<tr>
<td>3</td>
<td>48.72</td>
<td>8.17</td>
<td>3248</td>
<td>S VP NP-C L</td>
<td>95.75</td>
<td>95.11</td>
</tr>
<tr>
<td>4</td>
<td>54.03</td>
<td>5.31</td>
<td>2112</td>
<td>NP NPB PP R</td>
<td>84.99</td>
<td>84.35</td>
</tr>
<tr>
<td>5</td>
<td>59.30</td>
<td>5.27</td>
<td>2095</td>
<td>VP TAG NP-C R</td>
<td>92.41</td>
<td>92.15</td>
</tr>
<tr>
<td>6</td>
<td>64.18</td>
<td>4.88</td>
<td>1941</td>
<td>VP TAG VP-C R</td>
<td>97.42</td>
<td>97.98</td>
</tr>
<tr>
<td>7</td>
<td>68.71</td>
<td>4.53</td>
<td>1801</td>
<td>VP TAG PP R</td>
<td>83.62</td>
<td>81.14</td>
</tr>
<tr>
<td>8</td>
<td>73.13</td>
<td>4.42</td>
<td>1757</td>
<td>TOP TOP S R</td>
<td>96.36</td>
<td>96.85</td>
</tr>
<tr>
<td>9</td>
<td>74.53</td>
<td>1.40</td>
<td>558</td>
<td>VP TAG SBAR-C R</td>
<td>94.27</td>
<td>93.93</td>
</tr>
<tr>
<td>10</td>
<td>75.83</td>
<td>1.30</td>
<td>518</td>
<td>QP TAG TAG R</td>
<td>86.49</td>
<td>86.65</td>
</tr>
<tr>
<td>11</td>
<td>77.08</td>
<td>1.25</td>
<td>495</td>
<td>NP NPB NP R</td>
<td>74.34</td>
<td>75.72</td>
</tr>
<tr>
<td>12</td>
<td>78.28</td>
<td>1.20</td>
<td>477</td>
<td>SBAR TAG S-C R</td>
<td>94.55</td>
<td>92.04</td>
</tr>
<tr>
<td>13</td>
<td>79.48</td>
<td>1.20</td>
<td>476</td>
<td>NP NPB SBAR R</td>
<td>79.20</td>
<td>79.54</td>
</tr>
<tr>
<td>14</td>
<td>80.40</td>
<td>0.92</td>
<td>367</td>
<td>VP TAG ADV P R</td>
<td>74.93</td>
<td>78.57</td>
</tr>
<tr>
<td>15</td>
<td>81.30</td>
<td>0.90</td>
<td>358</td>
<td>NPB TAG NPB L</td>
<td>97.49</td>
<td>92.82</td>
</tr>
<tr>
<td>16</td>
<td>82.18</td>
<td>0.88</td>
<td>349</td>
<td>VP TAG TAG R</td>
<td>90.54</td>
<td>93.49</td>
</tr>
<tr>
<td>17</td>
<td>82.97</td>
<td>0.79</td>
<td>316</td>
<td>VP TAG SG-C R</td>
<td>92.41</td>
<td>88.22</td>
</tr>
</tbody>
</table>

Accuracy of the 17 most frequent dependency types in section 0 of the treebank, as recovered by model 2. \( R = \) rank; \( CP = \) cumulative percentage; \( P = \) percentage; \( \text{Rec} = \) Recall; \( \text{Prec} = \) precision.

<table>
<thead>
<tr>
<th>Type</th>
<th>Sub-type</th>
<th>Description</th>
<th>Count</th>
<th>Recall</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>Complement to a verb</td>
<td>S VP NP-C L</td>
<td>Subject</td>
<td>3248</td>
<td>95.75</td>
<td>95.11</td>
</tr>
<tr>
<td></td>
<td>VP TAG NP-C R</td>
<td>Object</td>
<td>2095</td>
<td>92.41</td>
<td>92.15</td>
</tr>
<tr>
<td></td>
<td>VP TAG SBAR-C R</td>
<td></td>
<td>558</td>
<td>94.27</td>
<td>93.93</td>
</tr>
<tr>
<td></td>
<td>VP TAG SG-C R</td>
<td></td>
<td>316</td>
<td>92.41</td>
<td>88.22</td>
</tr>
<tr>
<td></td>
<td>VP TAG S-C R</td>
<td></td>
<td>150</td>
<td>74.67</td>
<td>78.32</td>
</tr>
<tr>
<td></td>
<td>S VP S-C L</td>
<td></td>
<td>104</td>
<td>93.27</td>
<td>78.86</td>
</tr>
<tr>
<td></td>
<td>S VP SG-C L</td>
<td></td>
<td>14</td>
<td>78.57</td>
<td>67.85</td>
</tr>
<tr>
<td>TOTAL</td>
<td></td>
<td></td>
<td>6495</td>
<td>93.76</td>
<td>92.96</td>
</tr>
</tbody>
</table>

Other complements | PP TAG NP-C R | | 4335 | 94.72 | 94.04 |
| | VP TAG VP-C R | | 1941 | 97.42 | 97.98 |
| | SBAR TAG S-C R | | 477 | 94.55 | 92.04 |
| | SBAR WHNP SG-C R | | 286 | 90.56 | 90.56 |
| | PP TAG SG-C R | | 125 | 94.40 | 89.39 |
| | SBAR WHADVP S-C R | | 83 | 97.59 | 98.78 |
| | PP TAG PP-C R | | 51 | 84.31 | 70.49 |
| | SBAR WHNP S-C R | | 42 | 66.67 | 84.85 |
| | SBAR TAG SG-C R | | 23 | 69.57 | 69.57 |
| | PP TAG S-C R | | 18 | 38.89 | 63.64 |
| | SBAR WHNP S-C R | | 16 | 100.00 | 100.00 |
| | S ADJP NP-C L | | 15 | 46.67 | 46.67 |
| | PP TAG SBAR-C R | | 15 | 100.00 | 88.24 |
| TOTAL | | | 7473 | 94.47 | 94.12 |
### Type: PP modification

<table>
<thead>
<tr>
<th>Sub-type</th>
<th>Description</th>
<th>Count</th>
<th>Recall</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>NPNPB PP R</td>
<td>VP TAG PP R</td>
<td>2112</td>
<td>84.99</td>
<td>84.35</td>
</tr>
<tr>
<td>S VP PP L</td>
<td>ADJP TAG PP R</td>
<td>1801</td>
<td>83.62</td>
<td>81.14</td>
</tr>
<tr>
<td>ADVP TAG PP R</td>
<td>NP NP PP R</td>
<td>287</td>
<td>90.24</td>
<td>81.96</td>
</tr>
<tr>
<td>PP PP P L</td>
<td>NAC TAG PP R</td>
<td>90</td>
<td>75.56</td>
<td>78.16</td>
</tr>
<tr>
<td>NP NP PP R</td>
<td>PP PP P L</td>
<td>23</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>PP PP P L</td>
<td>NAC TAG PP R</td>
<td>19</td>
<td>21.05</td>
<td>26.67</td>
</tr>
<tr>
<td>NAC TAG PP R</td>
<td>TOTAL</td>
<td>12</td>
<td>50.00</td>
<td>100.00</td>
</tr>
</tbody>
</table>

#### Coordination

<table>
<thead>
<tr>
<th>Sub-type</th>
<th>Description</th>
<th>Count</th>
<th>Recall</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>NPNPNNPR</td>
<td>VP TAG NP S R</td>
<td>4473</td>
<td>82.29</td>
<td>81.51</td>
</tr>
</tbody>
</table>

#### Count

- 4473 = 11.2% of all cases
- 763 = 1.9% of all cases

#### Some Conclusions about Errors in Parsing

- “Core” sentential structure (complements, NP chunks) recovered with over 90% accuracy.
- Attachment ambiguities involving adjuncts are resolved with much lower accuracy (≈ 80% for PP attachment, ≈ 50 – 60% for coordination).
Overview of Today’s Lecture

- Refinements to Model 1
- Evaluating parsing models
- Extensions to the parsing models

Trigram Language Models (from Lecture 2)

Step 1: The chain rule (note that \( w_{n+1} = \text{STOP} \))

\[
P(w_1, w_2, \ldots, w_n) = \prod_{i=1}^{n+1} P(w_i | w_1 \ldots w_{i-1})
\]

Step 2: Make Markov independence assumptions:

\[
P(w_1, w_2, \ldots, w_n) = \prod_{i=1}^{n+1} P(w_i | w_{i-2}, w_{i-1})
\]

For Example

\[
P(\text{the, dog, laughs}) = P(\text{the} | \text{START}) \times P(\text{dog} | \text{START, the}) \times P(\text{laughs} | \text{the, dog}) \times P(\text{STOP} | \text{dog, laughs})
\]

Parsing Models as Language Models

- Generative models assign a probability \( P(T, S) \) to each tree/sentence pair
- Say sentence is \( S \), set of parses for \( S \) is \( T(S) \), then

\[
P(S) = \sum_{T \in T(S)} P(T, S)
\]

- Can calculate perplexity for parsing models

A Quick Reminder of Perplexity

- We have some test data, \( n \) sentences

\[
S_1, S_2, S_3, \ldots, S_n
\]

- We could look at the probability under our model \( \prod_{i=1}^{n} P(S_i) \).
  Or more conveniently, the log probability

\[
\log \prod_{i=1}^{n} P(S_i) = \sum_{i=1}^{n} \log P(S_i)
\]

- In fact the usual evaluation measure is perplexity

\[
\text{Perplexity} = 2^{-x} \quad \text{where} \quad x = \frac{1}{W} \sum_{i=1}^{n} \log P(S_i)
\]

and \( W \) is the total number of words in the test data.
Trigrams Can’t Capture Long-Distance Dependencies

**Actual Utterance:** He is a resident of the U.S. and of the U.K.

**Recognizer Output:** He is a resident of the U.S. and *that* the U.K.

- Bigram *and that* is around 15 times as frequent as *and of*
  ⇒ Bigram model gives over 10 times greater probability to incorrect string

- Parsing models assign 78 times higher probability to the correct string

---

Examples of Long-Distance Dependencies

**Subject/verb dependencies**

*Microsoft*, the world’s largest software company, **acquired** . . .

**Object/verb dependencies**

. . . **acquired** the New-York based software *company* . . .

**Appositives**

*Microsoft*, the world’s largest software *company*, acquired . . .

**Verb/Preposition Collocations**

I **put** the coffee mug **on the table**

The USA **elected** the son of George Bush Sr. as president

**Coordination**

She said *that* . . . and *that* . . .

---

Work on Parsers as Language Models

- “The Structured Language Model”. Ciprian Chelba and Fred Jelinek, see also recent work by Peng Xu, Ahmad Emami and Fred Jelinek.

- “Probabilistic Top-Down Parsing and Language Modeling”. Brian Roark.

- “Immediate Head-Parsing for Language Models”. Eugene Charniak.

---

Some Perplexity Figures from (Charniak, 2000)

<table>
<thead>
<tr>
<th>Model</th>
<th>Trigram</th>
<th>Grammar</th>
<th>Interpolation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chelba and Jelinek</td>
<td>167.14</td>
<td>158.28</td>
<td>148.90</td>
</tr>
<tr>
<td>Roark</td>
<td>167.02</td>
<td>152.26</td>
<td>137.26</td>
</tr>
<tr>
<td>Charniak</td>
<td>167.89</td>
<td>144.98</td>
<td>133.15</td>
</tr>
</tbody>
</table>

- **Interpolation** is a mixture of the trigram and grammatical models

- Chelba and Jelinek, Roark use trigram information in their grammatical models, Charniak doesn’t!

- **Note:** Charniak’s parser in these experiments is as described in (Charniak 2000), and makes use of Markov processes generating rules (a shift away from the Charniak 1997 model).
Extending Charniak’s Parsing Model

She said that the lawyer questioned him

⇒ bigram lexical probabilities

\[ P(\text{questioned} \mid \text{SBAR,COMP,S,Vt, that,COMP}) \]
\[ P(\text{lawyer} \mid \text{S,VP,NP,NN, questioned,Vt}) \]
\[ P(\text{him} \mid \text{VP,Vt,NP,PRP, questioned,Vt}) \ldots \]
Some Perplexity Figures from (Charniak, 2000)

<table>
<thead>
<tr>
<th>Model</th>
<th>Trigram</th>
<th>Grammar</th>
<th>Interpolation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chelba and Jelinek</td>
<td>167.14</td>
<td>158.28</td>
<td>148.90</td>
</tr>
<tr>
<td>Roark</td>
<td>167.02</td>
<td>152.26</td>
<td>137.26</td>
</tr>
<tr>
<td>Charniak (Bigram)</td>
<td>167.89</td>
<td>144.98</td>
<td>133.15</td>
</tr>
<tr>
<td>Charniak (Trigram)</td>
<td>167.89</td>
<td>130.20</td>
<td>126.07</td>
</tr>
</tbody>
</table>

Model 3: A Model of Wh-Movement

- Examples of Wh-movement:
  
  **Example 1** The person (SBAR who [S-CLPRP) bought the shoes)
  
  **Example 2** The shoes (SBAR that I bought [S-CLPRP) last week)
  
  **Example 3** The person (SBAR who I bought the shoes from [S-CLPRP)
  
  **Example 4** The person (SBAR who Jeff said I bought the shoes from [S-CLPRP)

- Key ungrammatical examples:
  
  **Example 1** The person (SBAR who Fran and [S-CLPRP) bought the shoes)
  (derived from Fran and Jeff bought the shoes)
  
  **Example 2**
  The store (SBAR that Jeff bought the shoes because Fran likes [S-CLPRP)
  (derived from Jeff bought the shoes because Fran likes the store)

The Parse Trees at this Stage

It’s difficult to recover “shoes” as the object of “bought”

Adding Gaps and Traces

It’s easy to recover “shoes” as the object of “bought”
Adding Gaps and Traces

- This information can be recovered from the treebank
- Doubles the number of non-terminals (with/without gaps)
- Similar to treatment of Wh-movement in GPSG (generalized phrase structure grammar)
- If our parser recovers this information, it’s easy to recover syntactic relations

New Rules: Rules that Generate Gaps

- Modeled in a very similar way to previous rules

New Rules: Rules that Pass Gaps down the Tree

- Passing a gap to a modifier
  \[ \text{SBAR}(that,WDT)(+gap) \]
  \[ \text{WHNP}(that,WDT) \quad \text{S-C}(bought,Vt)(+gap) \]
- Passing a gap to the head
  \[ \text{S-C}(bought,Vt)(+gap) \]
  \[ \text{NP-C}(I,PRP) \quad \text{VP}(bought,Vt)(+gap) \]

New Rules: Rules that Discharge Gaps as a Trace

- Discharging a gap as a TRACE
  \[ \text{VP}(bought,Vt)(+gap) \]
  \[ \text{Vt}(bought,Vt) \quad \text{TRACE} \quad \text{NP}(week,NN) \]
**Adding Gap Propagation (Example 1)**

- **Step 1:** generate category of head child

  \[
  \begin{align*}
  &\text{SBAR(that,WDT)(+gap)} \\
  &\quad \downarrow \\
  &\text{SBAR(that,WDT)(+gap)} \\
  &\quad \text{WHNP(that,WDT)} \\
  &P_h(\text{WHNP} \mid \text{SBAR, that, WDT})
  \end{align*}
  \]

- **Step 2:** choose to propagate the gap to the head, or to the left or right of the head

  \[
  \begin{align*}
  &\text{SBAR(that,WDT)(+gap)} \\
  &\quad \downarrow \\
  &\text{SBAR(that,WDT)(+gap)} \\
  &\quad \text{WHNP(that,WDT)} \\
  &P_h(\text{WHNP} \mid \text{SBAR, that, WDT}) 	imes P_y(\text{RIGHT} \mid \text{SBAR, that, WDT})
  \end{align*}
  \]

- **Step 3:** choose right subcategorization frame

  \[
  \begin{align*}
  &\text{SBAR(that,WDT)(+gap)} \\
  &\quad \downarrow \\
  &\text{WHNP(that,WDT)} \\
  &\quad \text{SBAR(that,WDT)(+gap)} \\
  &\quad \text{WHNP(that,WDT)} \\
  &\quad \{\text{S-C, +gap}\}
  \end{align*}
  \]

  \[
  P_h(\text{WHNP} \mid \text{SBAR, that, WDT}) \times P_y(\text{RIGHT} \mid \text{SBAR, that, WDT}) \times P_{rec}(\{\text{S-C}\} \mid \text{SBAR, WHNP, that, WDT})
  \]

- **Step 4:** Generate right modifiers

  \[
  \begin{align*}
  &\text{SBAR(that,WDT)(+gap)} \\
  &\quad \downarrow \\
  &\text{WHNP(that,WDT)} \\
  &\quad \{\text{S-C, +gap}\}
  \end{align*}
  \]

  \[
  \begin{align*}
  &\text{SBAR(that,WDT)(+gap)} \\
  &\quad \downarrow \\
  &\text{WHNP(that,WDT)} \\
  &\quad \{\text{S-C(bought, Vt), +gap}\}
  \end{align*}
  \]

  \[
  P_h(\text{WHNP} \mid \text{SBAR, that, WDT}) \times P_y(\text{RIGHT} \mid \text{SBAR, that, WDT}) \times P_{rec}(\{\text{S-C}\} \mid \text{SBAR, WHNP, that, WDT}) \times
  \]

  \[
  P_d(\text{S-C(bought, Vt), +gap} \mid \text{SBAR, WHNP, that, WDT, RIGHT, \{S-C, +gap\}})
  \]

- In this case left modifiers are generated as before
Adding Gap Propagation (Example 2)

- Step 1: generate category of head child

\[
\begin{align*}
S-C(bought,Vt)(+gap) \\
\downarrow \\
S-C(bought,Vt)(+gap) \\
\mid \\
VP(bought,Vt)
\end{align*}
\]

\[
P_h(VP \mid S-C, bought, Vt)
\]

Adding Gap Propagation (Example 3)

- Step 1: generate category of head child

\[
\begin{align*}
VP(bought,Vt)(+gap) \\
\downarrow \\
VP(bought,Vt)(+gap) \\
\mid \\
Vt(bought,Vt)
\end{align*}
\]

\[
P_h(Vt \mid VP, bought, Vt)
\]

Adding Gap Propagation (Example 2)

- Step 2: choose to propagate the gap to the head, or to the left or right of the head

\[
\begin{align*}
S-C(bought,Vt)(+gap) \\
\downarrow \\
S-C(bought,Vt)(+gap) \\
\mid \\
VP(bought,Vt)
\end{align*}
\]

\[
P_h(VP \mid S-C, bought, Vt) \times P_g(\text{HEAD} \mid S-C, VP, bought, Vt)
\]

- In this case we’re done: rest of rule is generated as before

Adding Gap Propagation (Example 3)

- Step 2: choose to propagate the gap to the head, or to the left or right of the head

\[
\begin{align*}
VP(bought,Vt)(+gap) \\
\downarrow \\
VP(bought,Vt)(+gap) \\
\mid \\
Vt(bought,Vt)
\end{align*}
\]

\[
P_h(Vt \mid SBAR, that, WDT) \times P_g(\text{RIGHT} \mid VP, Vt, bought, Vt)
\]

- In this case left modifiers are generated as before
**Adding Gap Propagation (Example 3)**

- Step 3: choose right subcategorization frame

```
VP(bought,Vt)(+gap)  
Vt(bought,Vt)  

\[ P_h(Vt \mid SBAR, \text{that}, \text{WDT}) \times P_g(\text{RIGHT} \mid VP, Vt, bought, Vt) \times P_r_c(\{NP-C\} \mid VP, Vt, bought, Vt) \]
```

69

**Adding Gap Propagation (Example 3)**

- Step 4: generate right modifiers

```
VP(bought,Vt)(+gap)  
Vt(bought,Vt)  

\[ P_h(Vt \mid SBAR, \text{that}, \text{WDT}) \times P_g(\text{RIGHT} \mid VP, Vt, bought, Vt) \times P_r_c(\{NP-C\} \mid VP, Vt, bought, Vt) \times P_d(\text{TRACE} \mid VP, Vt, bought, Vt, RIGHT, \{NP-C,+gap\}) \times P_d(\text{NP}(\text{yesterday},\text{NN}) \mid VP, Vt, bought, Vt, RIGHT, \{\}) \]
```

71

**Adding Gap Propagation (Example 3)**

```
VP(bought,Vt)(+gap)  
Vt(bought,Vt)  

\[ P_h(Vt \mid SBAR, \text{that}, \text{WDT}) \times P_g(\text{RIGHT} \mid VP, Vt, bought, Vt) \times P_r_c(\{NP-C\} \mid VP, Vt, bought, Vt) \times P_d(\text{TRACE} \mid VP, Vt, bought, Vt, RIGHT, \{NP-C,+gap\}) \times P_d(\text{NP}(\text{yesterday},\text{NN}) \mid VP, Vt, bought, Vt, RIGHT, \{\}) \times P_d(\text{STOP} \mid VP, Vt, bought, Vt, RIGHT, \{\}) \]
```

72
Ungrammatical Cases Contain Low Probability Rules

**Example 1** The person (SBAR who Fran and TRACE bought the shoes)

```
  S-C(bought,Vt)(+gap)
   NP-C(Fran,NNP)(+gap) VP(bought,Vt)
   NP(Fran,NNP) CC TRACE
```

**Example 2** The store (SBAR that Jeff bought the shoes because Fran likes TRACE)

```
  VP(bought,Vt)(+gap)
   Vt(bought,Vt) NP-C(shoes,NNS) SBAR(because,COMP)(+gap)
```

73