CMPT-825
Natural Language Processing

Anoop Sarkar
http://www.cs.sfu.ca/~anoop

September 19, 2012
Probabilistic CFG (PCFG)

\[
\begin{align*}
S & \rightarrow \ NP \ VP & 1 \\
VP & \rightarrow \ V \ NP & 0.9 \\
VP & \rightarrow \ VP \ PP & 0.1 \\
PP & \rightarrow \ P \ NP & 1 \\
NP & \rightarrow \ NP \ PP & 0.25 \\
NP & \rightarrow \ Calvin & 0.25 \\
NP & \rightarrow \ monsters & 0.25 \\
NP & \rightarrow \ school & 0.25 \\
V & \rightarrow \ imagined & 1 \\
P & \rightarrow \ in & 1 \\
\end{align*}
\]

\[
P(input) = \sum_{\text{tree}} P(\text{tree} | \ input)
\]

\[
P(\text{Calvin imagined monsters in school}) = ?
\]

Notice that \( P(VP \rightarrow V \ NP) + P(VP \rightarrow VP \ PP) = 1.0 \)
Probabilistic CFG (PCFG)

\[ P(\text{Calvin imagined monsters in school}) = ? \]

(S (NP Calvin)
  (VP (V imagined)
    (NP (NP monsters)
      (PP (P in)
        (NP school)))))

(S (NP Calvin)
  (VP (VP (V imagined)
    (NP monsters))
    (PP (P in)
      (NP school))))
Probabilistic CFG (PCFG)

\[(S (NP Calvin)\]
\[(VP (V imagined)\]
\[(NP (NP monsters)\]
\[(PP (P in)\]
\[(NP school))\])\)

\[
P(tree_1) = P(S \rightarrow NP VP) \times P(NP \rightarrow Calvin) \times P(VP \rightarrow V NP) \times P(V \rightarrow imagined) \times P(NP \rightarrow NP PP) \times P(NP \rightarrow monsters) \times P(PP \rightarrow P NP) \times P(P \rightarrow in) \times P(NP \rightarrow school)\]
\[
= 1 \times 0.25 \times 0.9 \times 1 \times 0.25 \times 0.25 \times 1 \times 1 \times 0.25 = 0.003515625\]
Probabilistic CFG (PCFG)

(S (NP Calvin)
  (VP (VP (V imagined)
    (NP monsters))
  (PP (P in)
    (NP school))))

\[ P(\text{tree}_2) = P(S \rightarrow NP \ VP) \times P(NP \rightarrow Calvin) \times P(VP \rightarrow VP \ PP) \times P(VP \rightarrow V \ NP) \times P(V \rightarrow imagined) \times P(NP \rightarrow monsters) \times P(PP \rightarrow P \ NP) \times P(P \rightarrow in) \times P(NP \rightarrow school) \]
\[ = 1 \times 0.25 \times 0.1 \times 0.9 \times 1 \times 0.25 \times 1 \times 1 \times 0.25 = .00140625 \]
Probabilistic CFG (PCFG)

\[ P(\text{Calvin imagined monsters in school}) = P(\text{tree}_1) + P(\text{tree}_2) = 0.003515625 + 0.00140625 = 0.004921875 \]

Most likely tree is \( \text{tree}_1 = \arg \max_{\text{tree}} P(\text{tree} | \text{input}) \)

\[
\begin{align*}
(S \ (NP \ Calvin)) \\
(VP \ (V \ imagined)) \\
(NP \ (NP \ monsters)) \\
(PP \ (P \ in)) \\
(NP \ school)))))
\end{align*}
\]

\[
\begin{align*}
(S \ (NP \ Calvin)) \\
(VP \ (VP \ (V \ imagined)) \\
(NP \ monsters)) \\
(PP \ (P \ in)) \\
(NP \ school)))))
\end{align*}
\]
Central condition: $\sum_\alpha P(A \rightarrow \alpha) = 1$

Called a proper PCFG if this condition holds

Note that this means $P(A \rightarrow \alpha) = P(\alpha \mid A) = \frac{f(A,\alpha)}{f(A)}$

$P(T \mid S) = \frac{P(T,S)}{P(S)} = P(T,S) = \prod_i P(RHS_i \mid LHS_i)$
What is the PCFG that can be extracted from this single tree:

(S (NP (Det the) (NP man))
  (VP (VP (V played)
    (NP (Det a) (NP game)))
  (PP (P with)
    (NP (Det the) (NP dog))))

How many different rhs $\alpha$ exist for $A \rightarrow \alpha$ where $A$ can be $S$, $NP$, $VP$, $PP$, $Det$, $N$, $V$, $P$
We can do this with multiple trees. Simply count occurrences of CFG rules over all the trees.

A repository of such trees labelled by a human is called a TreeBank.
Ambiguity

- Part of Speech ambiguity
  saw → noun
  saw → verb

- Structural ambiguity: Prepositional Phrases
  I saw (the man) with the telescope
  I saw (the man with the telescope)

- Structural ambiguity: Coordination
  a program to promote safety in ((trucks) and (minivans))
  a program to promote ((safety in trucks) and (minivans))
  ((a program to promote safety in trucks) and (minivans))
Ambiguity ← attachment choice in alternative parses

```
(a program to promote safety in trucks and minivans)
```

```
(a program to promote safety in trucks and minivans)
```
Parsing as a machine learning problem

- $S = \text{a sentence}$
- $T = \text{a parse tree}$

A statistical parsing model defines $P(T | S)$
Parsing as a machine learning problem

- $S =$ a sentence
  $T =$ a parse tree
  A statistical parsing model defines $P(T \mid S)$
- Find best parse: $\arg\max_T P(T \mid S)$

  e.g. for PCFGs: $P(T, S) = \prod_{i=1}^{n} P(RHS_i \mid LHS_i)$
Parsing as a machine learning problem

- $S =$ a sentence
  - $T =$ a parse tree
    - A statistical parsing model defines $P(T \mid S)$
- Find best parse: $\arg \max_T P(T \mid S)$
- $P(T \mid S) = \frac{P(T,S)}{P(S)} = P(T, S)$
Parsings as a machine learning problem

- $S = \text{a sentence}$
  - $T = \text{a parse tree}$
  - A statistical parsing model defines $P(T \mid S)$
- Find best parse: $\arg \max_T P(T \mid S)$
- $P(T \mid S) = \frac{P(T,S)}{P(S)} = P(T, S)$
- Best parse: $\arg \max_T P(T, S)$
Parsing as a machine learning problem

- $S =$ a sentence
- $T =$ a parse tree
- A statistical parsing model defines $P(T \mid S)$
- Find best parse: $\text{arg max}_T P(T \mid S)$
- $P(T \mid S) = \frac{P(T, S)}{P(S)} = P(T, S)$
- Best parse: $\text{arg max}_T P(T, S)$
- e.g. for PCFGs: $P(T, S) = \prod_{i=1}^{n} P(\text{RHS}_i \mid \text{LHS}_i)$
Adding Lexical Information to PCFG

S
  ..
      VP{indicated}
        VB{indicated}  NP{difference}  PP{in}
          indicated  difference  P  NP
                      in  ..
Adding Lexical Information to PCFG (Collins 99, Charniak 00)

\[ P_h(VB \mid VP, \text{indicated}) \times P_l(\text{STOP} \mid VP, VB, \text{indicated}) \times P_r(\text{NP(difference)} \mid VP, VB, \text{indicated}) \times P_r(\text{PP(in)} \mid VP, VB, \text{indicated}) \times P_r(\text{STOP} \mid VP, VB, \text{indicated}) \]
Evaluation of Parsing

- Consider a candidate parse to be evaluated against the truth (or gold-standard parse):

  candidate: (S (A (P this) (Q is)) (A (R a) (T test)))
  gold: (S (A (P this)) (B (Q is) (A (R a) (T test))))

- In order to evaluate this, we list all the constituents:

<table>
<thead>
<tr>
<th>Candidate</th>
<th>Gold</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0,4,S)</td>
<td>(0,4,S)</td>
</tr>
<tr>
<td>(0,2,A)</td>
<td>(0,1,A)</td>
</tr>
<tr>
<td>(2,4,A)</td>
<td>(1,4,B)</td>
</tr>
<tr>
<td></td>
<td>(2,4,A)</td>
</tr>
</tbody>
</table>

- Skip spans of length 1 which would be equivalent to part of speech tagging accuracy.

- Precision is defined as $\frac{\#_{\text{correct}}}{\#_{\text{proposed}}} = \frac{2}{3}$ and recall as $\frac{\#_{\text{correct}}}{\#_{\text{in gold}}} = \frac{2}{4}$.

- Another measure: crossing brackets,

  candidate: [ an [incredibly expensive] coat ] (1 CB)
  gold: [ an [incredibly [expensive coat]]]
Evaluation of Parsing

Bracketing recall \( R \) = \( \frac{\text{num of correct constituents}}{\text{num of constituents in the goldfile}} \)

Bracketing precision \( P \) = \( \frac{\text{num of correct constituents}}{\text{num of constituents in the parsed file}} \)

Complete match = % of sents where recall & precision are both 100%

Average crossing = \( \frac{\text{num of constituents crossing a goldfile constituent}}{\text{num of sents}} \)

No crossing = % of sents which have 0 crossing brackets

2 or less crossing = % of sents which have \( \leq 2 \) crossing brackets
### Statistical Parsing Results

\[
F1\text{-}score = 2 \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}
\]

<table>
<thead>
<tr>
<th>System</th>
<th>(\leq 100\text{wds} F1\text{-}score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shift-Reduce (Magerman, 1995)</td>
<td>84.14</td>
</tr>
<tr>
<td>PCFG with Lexical Features (Collins, 1999)</td>
<td>88.19</td>
</tr>
<tr>
<td>PCFG with Lexical Features (Charniak, 1999)</td>
<td>89.54</td>
</tr>
<tr>
<td>(n)-best Re-ranking (Collins, 2000)</td>
<td>89.74</td>
</tr>
<tr>
<td>Unlexicalized Berkeley parser (Petrov et al, 2007)</td>
<td>90.10</td>
</tr>
<tr>
<td>(n)-best Re-ranking (Charniak and Johnson, 2005)</td>
<td>91.02</td>
</tr>
<tr>
<td>Tree-insertion grammars (Carreras, Collins, Koo, 2008)</td>
<td>91.10</td>
</tr>
<tr>
<td>Ensemble (n)-best Re-ranking (Johnson and Ural, 2010)</td>
<td>91.49</td>
</tr>
<tr>
<td>Forest Re-ranking (Huang, 2010)</td>
<td>91.70</td>
</tr>
<tr>
<td>Unlabeled Data with Self-Training (McCloskey et al, 2006)</td>
<td>92.10</td>
</tr>
</tbody>
</table>
Probability of nonterminal $X$ spanning $j \ldots k$: $N[X, j, k]$

Beam Thresholding compares $N[X, j, k]$ with every other $Y$ where $N[Y, j, k]$

But what should be compared?

Just the inside probability: $P(X \Rightarrow t_j \ldots t_k)$? written as $\beta(X, j, k)$

Perhaps $\beta(\text{FRAG}, 0, 3) > \beta(\text{NP}, 0, 3)$, but NPs are much more likely than FRAGs in general
Practical Issues: Beam Thresholding and Priors

- The correct estimate is the *outside probability*:

\[ P(S \Rightarrow t_1 \ldots t_{j-1} X t_{k+1} \ldots t_n) \]

written as \( \alpha(X, j, k) \)

- Unfortunately, you can only compute \( \alpha(X, j, k) \) efficiently after you finish parsing and reach \((S, 0, n)\)
Practical Issues: Beam Thresholding and Priors

- To make things easier we multiply the prior probability $P(X)$ with the inside probability.
- In beam Thresholding we compare every new insertion of $X$ for span $j, k$ as follows:
  
  Compare $P(X) \cdot \beta(X, j, k)$ with the most probable $Y$
  
  $P(Y) \cdot \beta(Y, j, k)$

- Assume $Y$ is the most probable entry in $j, k$, then we compare

  $\text{beam} \cdot P(Y) \cdot \beta(Y, j, k)$  \hspace{1cm} (1)

  $P(X) \cdot \beta(X, j, k)$ \hspace{1cm} (2)

- If (2) < (1) then we prune $X$ for this span $j, k$
- beam is set to a small value, say 0.001 or even 0.01.
- As the beam value increases, the parser speed increases (since more entries are pruned).
- A simpler (but not as effective) alternative to using the beam is to keep only the top $K$ entries for each span $j, k$. 

Experiments with Beam Thresholding
Experiments with Beam Thresholding

![Graph showing the relationship between sentence length and time with different beam thresholds. The graph includes lines for 'No pruning, w/ prior', 'Beam = 10^-5, w/ prior', and 'Beam = 10^-4, w/ prior'.]