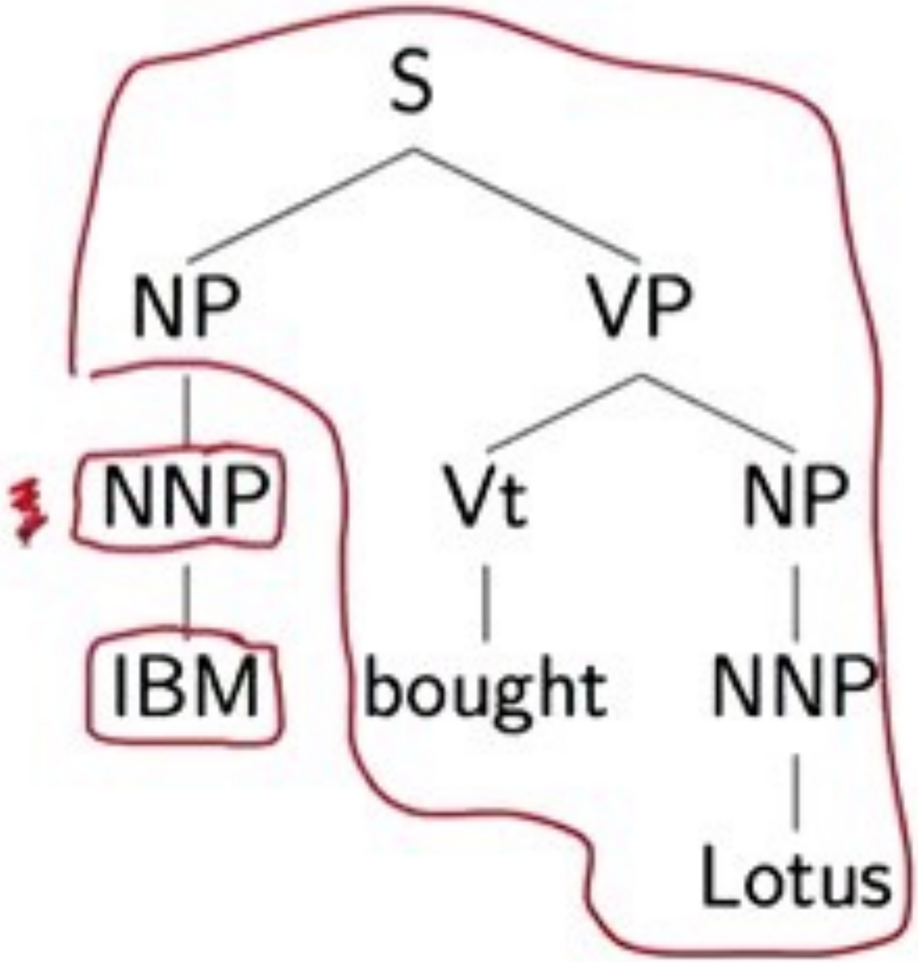
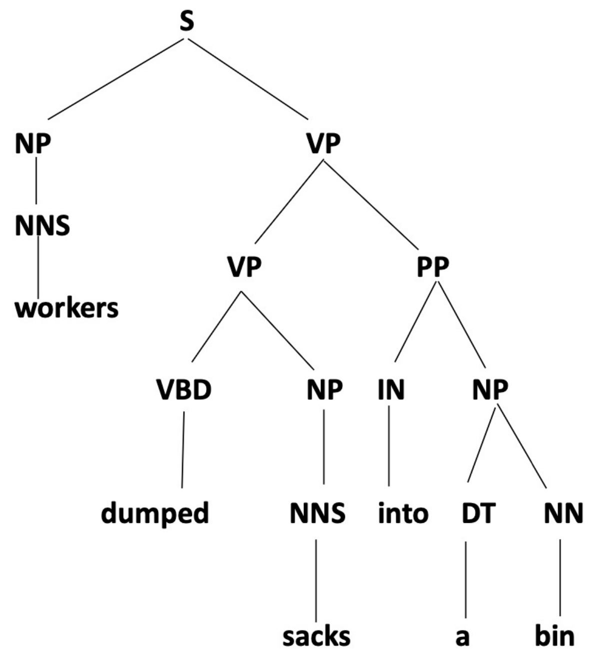
Lexicalized Probabilistic Context-Free Grammars

Michael Collins, Columbia University

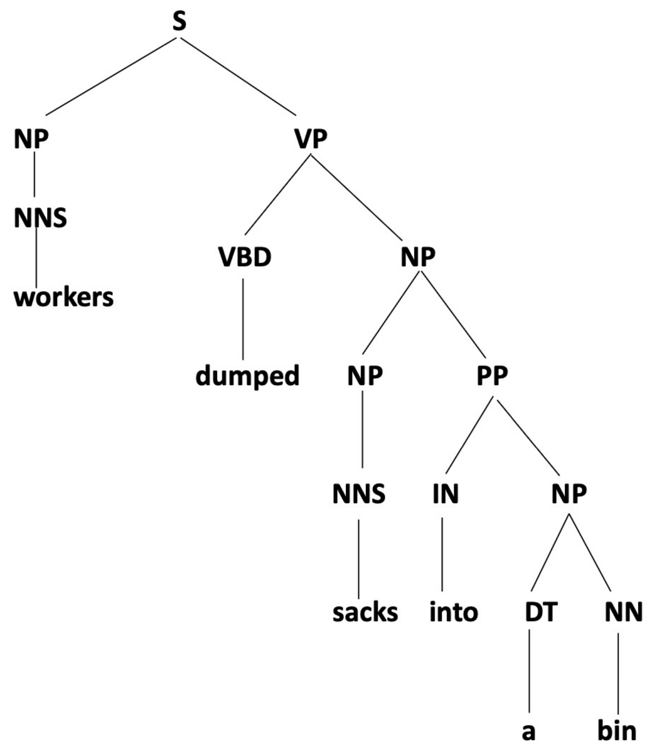
PCFGs的精确度只能达到72%，而现代的方法能达到92%，它的一个明显的缺点就是对词汇不敏感


S
NP
VP
NNP
Vt
NP
IBM
bought
NNP
Lotus



Rules
S→NP VP
NP→NNS
VP→VP PP
VP→VBD NP
NP→NNS
PP→IN NP
NP→DT NN
NNS→workers
VBD→dumped
NNS→sacks
IN→into
DT→a
NN→bin

Rules
S→NP VP
NP→NNS
NP→NP PP
VP→VBD NP
NP→NNS
PP→IN NP
NP→DT NN
NNS→workers
VBD→dumped
NNS→sacks
IN→into
DT→a
NN→bin



Overview

- Lexicalization of a treebank
- Lexicalized probabilistic context-free grammars
- Parameter estimation in lexicalized probabilistic context-free grammars
- Accuracy of lexicalized probabilistic context-free grammars

Heads in Context-Free Rules

Add annotations specifying the "head" of each rule:

S	$\Rightarrow$	NP	VP
VP	$\Rightarrow$	Vi	
VP	$\Rightarrow$	Vt	NP
VP	$\Rightarrow$	VP	PP
NP	$\Rightarrow$	DT	NN
NP	$\Rightarrow$	NP	PP
PP	$\Rightarrow$	IN	NP

Vi	$\Rightarrow$	sleeps
Vt	$\Rightarrow$	saw
NN	$\Rightarrow$	man
NN	$\Rightarrow$	woman
NN	$\Rightarrow$	telescope
DT	$\Rightarrow$	the
IN	$\Rightarrow$	with
IN	$\Rightarrow$	in

More about Heads

- Each context-free rule has one "special" child that is the head of the rule. e.g.,

S	⇒	NP	VP		(VP is the head)
VP	⇒	Vt	NP		(Vt is the head)
NP	⇒	DT	NN	NN	(NN is the head)

- A core idea in syntax
(e.g., see X-bar Theory, Head-Driven Phrase Structure Grammar)

- Some intuitions:
  - The central sub-constituent of each rule.
  - The semantic predicate in each rule.

Rules which Recover Heads: An Example for NPs

If the rule contains NN, NNS, or NNP:
    Choose the rightmost NN, NNS, or NNP

Else If the rule contains an NP: Choose the leftmost NP

Else If the rule contains a JJ: Choose the rightmost JJ

Else If the rule contains a CD: Choose the rightmost CD

Else Choose the rightmost child

e.g.,

NP	⇒	DT	NNP	NN
NP	⇒	DT	NN	NNP
NP	⇒	NP	PP	
NP	⇒	DT	JJ	
NP	⇒	DT		

Rules which Recover Heads: An Example for VPs

If the rule contains Vi or Vt: Choose the leftmost Vi or Vt

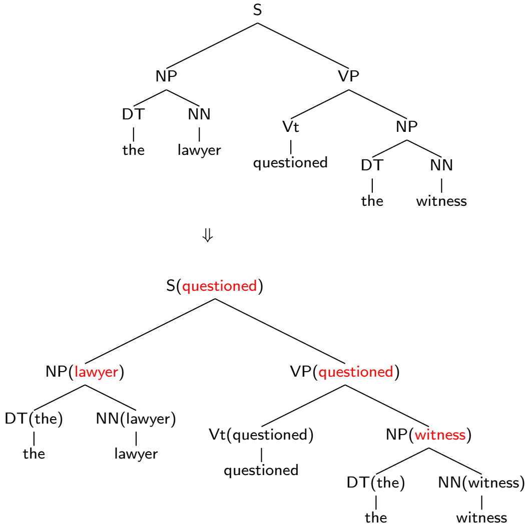
Else If the rule contains an VP: Choose the leftmost VP

Else Choose the leftmost child

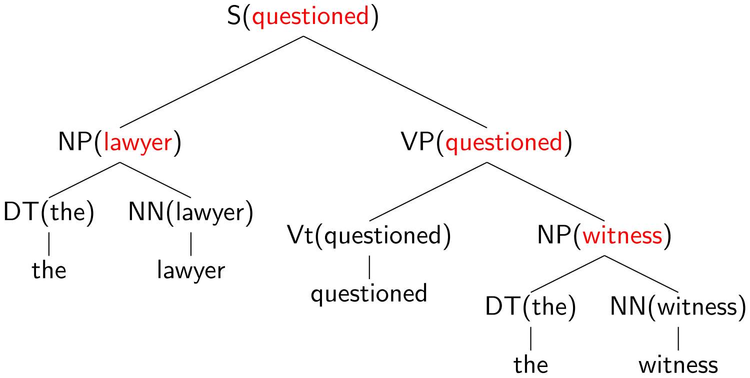
e.g.,

$$\begin{array}{llll} \text{VP} & \Rightarrow & \textcolor{red}{\text{Vt}} & \text{NP} \\ \text{VP} & \Rightarrow & \textcolor{red}{\text{VP}} & \text{PP} \end{array}$$

Adding Headwords to Trees



Adding Headwords to Trees (Continued)



- A constituent receives its headword from its **head child**.

S	$\Rightarrow$	NP	VP		(S receives headword from VP)
VP	$\Rightarrow$	Vt	NP		(VP receives headword from Vt)
NP	$\Rightarrow$	DT		NN	(NP receives headword from NN)

Overview

- Lexicalization of a treebank
- Lexicalized probabilistic context-free grammars
- Parameter estimation in lexicalized probabilistic context-free grammars
- Accuracy of lexicalized probabilistic context-free grammars

Chomsky Normal Form

A context free grammar $G = (N, \Sigma, R, S)$ in Chomsky Normal Form is as follows

- N is a set of non-terminal symbols
- Σ is a set of terminal symbols
- R is a set of rules which take one of two forms:
  - $X \rightarrow Y_1 Y_2$ for $X \in N$, and $Y_1, Y_2 \in N$
  - $X \rightarrow Y$ for $X \in N$, and $Y \in \Sigma$
- $S \in N$ is a distinguished start symbol

We can find the highest scoring parse under a PCFG in this form, in $O(n^3|N|^3)$ time where n is the length of the string being parsed.

Lexicalized Context-Free Grammars in Chomsky Normal Form

- N is a set of non-terminal symbols
- Σ is a set of terminal symbols
- R is a set of rules which take one of three forms:
  - $X(h) \rightarrow_1 Y_1(h)\ Y_2(w)$ for $X \in N$, and $Y_1, Y_2 \in N$, and $h, w \in \Sigma$
  - $X(h) \rightarrow_2 Y_1(w)\ Y_2(h)$ for $X \in N$, and $Y_1, Y_2 \in N$, and $h, w \in \Sigma$
  - $X(h) \rightarrow h$ for $X \in N$, and $h \in \Sigma$
- $S \in N$ is a distinguished start symbol

An Example

$$
\begin{array}{llll}
\text{S(saw)} & \rightarrow_2 & \text{NP(man)} & \text{VP(saw)} \\
\text{VP(saw)} & \rightarrow_1 & \text{Vt(saw)} & \text{NP(dog)} \\
\text{NP(man)} & \rightarrow_2 & \text{DT(the)} & \text{NN(man)} \\
\text{NP(dog)} & \rightarrow_2 & \text{DT(the)} & \text{NN(dog)} \\
\text{Vt(saw)} & \rightarrow & \text{saw} & \\
\text{DT(the)} & \rightarrow & \text{the} & \\
\text{NN(man)} & \rightarrow & \text{man} & \\
\text{NN(dog)} & \rightarrow & \text{dog} &
\end{array}
$$

Parameters in a Lexicalized PCFG

- An example parameter in a PCFG:

$$q(\mathsf{S} \rightarrow \mathsf{NP\ VP})$$

- An example parameter in a Lexicalized PCFG:

$$q(\mathsf{S(saw)} \rightarrow_2 \mathsf{NP(man)\ VP(saw)})$$

Parsing with Lexicalized CFGs

- The new form of grammar looks just like a Chomsky normal form CFG, but with potentially $O(|\Sigma|^2 \times |N|^3)$ possible rules.

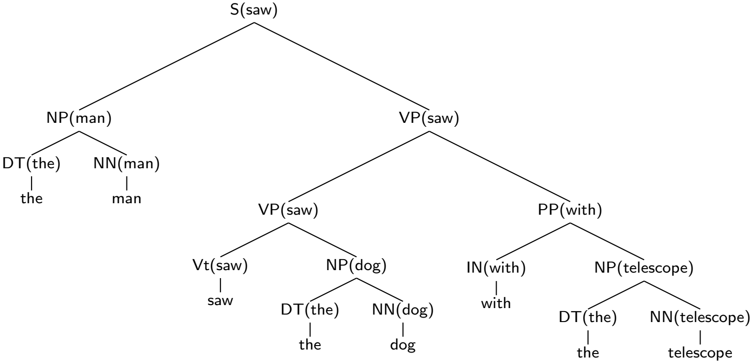
- Naively, parsing an n word sentence using the dynamic programming algorithm will take $O(n^3|\Sigma|^2|N|^3)$ time. But $|\Sigma|$ can be huge!!

- Crucial observation: at most $O(n^2 \times |N|^3)$ rules can be applicable to a given sentence $w_1, w_2, \ldots w_n$ of length n. This is because any rules which contain a lexical item that is not one of $w_1 \ldots w_n$, can be safely discarded.

- The result: we can parse in $O(n^5|N|^3)$ time.

Overview

- Lexicalization of a treebank
- Lexicalized probabilistic context-free grammars
- Parameter estimation in lexicalized probabilistic context-free grammars
- Accuracy of lexicalized probabilistic context-free grammars



$$
\begin{aligned}
\mathsf{p(t)} = {} & q(\mathsf{S(saw)} \rightarrow_2 \mathsf{NP(man)\ VP(saw)}) \\
& \times q(\mathsf{NP(man)} \rightarrow_2 \mathsf{DT(the)\ NN(man)}) \\
& \times q(\mathsf{VP(saw)} \rightarrow_1 \mathsf{VP(saw)\ PP(with)}) \\
& \times q(\mathsf{VP(saw)} \rightarrow_1 \mathsf{Vt(saw)\ NP(dog)}) \\
& \times q(\mathsf{PP(with)} \rightarrow_1 \mathsf{IN(with)\ NP(telescope)}) \\
& \times \ldots
\end{aligned}
$$

A Model from Charniak (1997)

- An example parameter in a Lexicalized PCFG:

$$q(\text{S(saw)} \rightarrow_2 \text{NP(man) VP(saw)})$$

- First step: decompose this parameter into a product of two parameters

$$\begin{aligned}
& q(\text{S(saw)} \rightarrow_2 \text{NP(man) VP(saw)}) \\
= \;& q(\text{S} \rightarrow_2 \text{NP VP}|\text{S, saw}) \times q(\text{man}|\text{S} \rightarrow_2 \text{NP VP, saw})
\end{aligned}$$

A Model from Charniak (1997) (Continued)

$$
\begin{aligned}
& q(\text{S(saw)} \rightarrow_2 \text{NP(man) VP(saw)}) \\
= \; & q(\text{S} \rightarrow_2 \text{NP VP}|\text{S, saw}) \times q(\text{man}|\text{S} \rightarrow_2 \text{NP VP, saw})
\end{aligned}
$$

- Second step: use smoothed estimation for the two parameter estimates

$$
\begin{aligned}
& q(\text{S} \rightarrow_2 \text{NP VP}|\text{S, saw}) \\
= \; & \lambda_1 \times q_{ML}(\text{S} \rightarrow_2 \text{NP VP}|\text{S, saw}) + \lambda_2 \times q_{ML}(\text{S} \rightarrow_2 \text{NP VP}|\text{S})
\end{aligned}
$$

A Model from Charniak (1997) (Continued)

$$
\begin{aligned}
& q(\text{S(saw)} \rightarrow_2 \text{NP(man) VP(saw)}) \\
= \; & q(\text{S} \rightarrow_2 \text{NP VP} | \text{S, saw}) \times q(\text{man} | \text{S} \rightarrow_2 \text{NP VP, saw})
\end{aligned}
$$

- Second step: use smoothed estimation for the two parameter estimates

$$
\begin{aligned}
& q(\text{S} \rightarrow_2 \text{NP VP} | \text{S, saw}) \\
= \; & \lambda_1 \times q_{ML}(\text{S} \rightarrow_2 \text{NP VP} | \text{S, saw}) + \lambda_2 \times q_{ML}(\text{S} \rightarrow_2 \text{NP VP} | \text{S})
\end{aligned}
$$

$$
\begin{aligned}
& q(\text{man} | \text{S} \rightarrow_2 \text{NP VP, saw}) \\
= \; & \lambda_3 \times q_{ML}(\text{man} | \text{S} \rightarrow_2 \text{NP VP, saw}) + \lambda_4 \times q_{ML}(\text{man} | \text{S} \rightarrow_2 \text{NP VP}) \\
& + \lambda_5 \times q_{ML}(\text{man} | \text{NP})
\end{aligned}
$$

Other Important Details

- Need to deal with rules with more than two children, e.g.,

$$\text{VP(told)} \rightarrow \text{V(told) NP(him) PP(on) SBAR(that)}$$

Other Important Details

- Need to deal with rules with more than two children, e.g.,

$$\text{VP(told)} \rightarrow \text{V(told) NP(him) PP(on) SBAR(that)}$$

- Need to incorporate parts of speech (useful in smoothing)

$$\text{VP-V(told)} \rightarrow \text{V(told) NP-PRP(him) PP-IN(on) SBAR-COMP(that)}$$

Other Important Details

- Need to deal with rules with more than two children, e.g.,

  $$\text{VP(told)} \rightarrow \text{V(told) NP(him) PP(on) SBAR(that)}$$

- Need to incorporate parts of speech (useful in smoothing)

  $$\text{VP-V(told)} \rightarrow \text{V(told) NP-PRP(him) PP-IN(on) SBAR-COMP(that)}$$

- Need to encode preferences for close attachment
  John was believed to have been shot by Bill

Other Important Details

- Need to deal with rules with more than two children, e.g.,

  $$\text{VP(told)} \rightarrow \text{V(told) NP(him) PP(on) SBAR(that)}$$

- Need to incorporate parts of speech (useful in smoothing)

  $$\text{VP-V(told)} \rightarrow \text{V(told) NP-PRP(him) PP-IN(on) SBAR-COMP(that)}$$

- Need to encode preferences for close attachment
  John was believed to have been shot by Bill

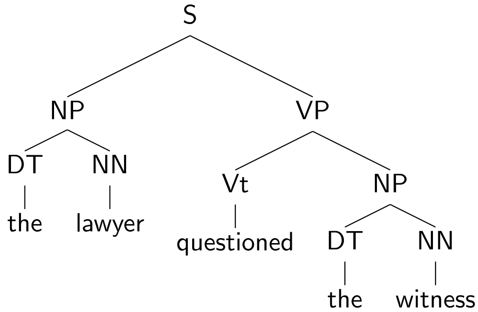
- Further reading:

  Michael Collins. 2003. Head-Driven Statistical Models for Natural Language Parsing. *In Computational Linguistics.*

Overview

- Lexicalization of a treebank
- Lexicalized probabilistic context-free grammars
- Parameter estimation in lexicalized probabilistic context-free grammars
- Accuracy of lexicalized probabilistic context-free grammars

Evaluation: Representing Trees as Constituents



Label	Start Point	End Point
NP	1	2
NP	4	5
VP	3	5
S	1	5

Precision and Recall

Label	Start Point	End Point
NP	1	2
NP	4	5
NP	4	8
PP	6	8
NP	7	8
VP	3	8
S	1	8

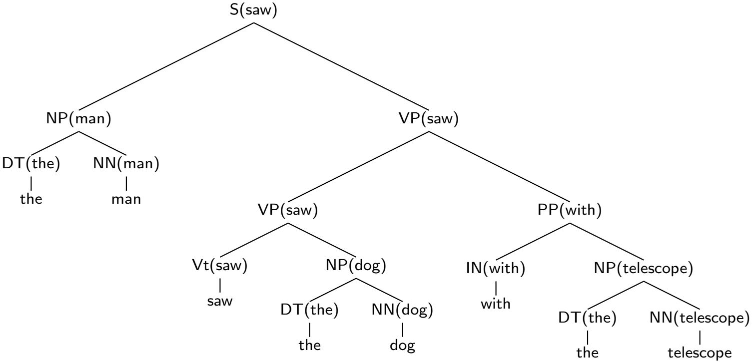
Label	Start Point	End Point
NP	1	2
NP	4	5
PP	6	8
NP	7	8
VP	3	8
S	1	8

- G = number of constituents in gold standard = 7
- P = number in parse output = 6
- C = number correct = 6

$$\text{Recall} = 100\% \times \frac{C}{G} = 100\% \times \frac{6}{7} \qquad \text{Precision} = 100\% \times \frac{C}{P} = 100\% \times \frac{6}{6}$$

Results

- Training data: 40,000 sentences from the Penn Wall Street Journal treebank. Testing: around 2,400 sentences from the Penn Wall Street Journal treebank.
- Results for a PCFG: 70.6% Recall, 74.8% Precision
- Magerman (1994): 84.0% Recall, 84.3% Precision
- Results for a lexicalized PCFG: 88.1% recall, 88.3% precision (from Collins (1997, 2003))
- More recent results: 90.7% Recall/91.4% Precision (Carreras et al., 2008); 91.7% Recall, 92.0% Precision (Petrov 2010); 91.2% Recall, 91.8% Precision (Charniak and Johnson, 2005)



$\langle$ ROOT_0,	saw_3,	ROOT $\rangle$
$\langle$ saw_3,	man_2,	$\mathsf{S} \rightarrow_2 \mathsf{NP\ VP}$ $\rangle$
$\langle$ man_2,	the_1,	$\mathsf{NP} \rightarrow_2 \mathsf{DT\ NN}$ $\rangle$
$\langle$ saw_3,	with_6,	$\mathsf{VP} \rightarrow_1 \mathsf{VP\ PP}$ $\rangle$
$\langle$ saw_3,	dog_5,	$\mathsf{VP} \rightarrow_1 \mathsf{Vt\ NP}$ $\rangle$
$\langle$ dog_5,	the_4,	$\mathsf{NP} \rightarrow_2 \mathsf{DT\ NN}$ $\rangle$
$\langle$ with_6,	$\mathsf{telescope}_8$,	$\mathsf{PP} \rightarrow_1 \mathsf{IN\ NP}$ $\rangle$
$\langle$ $\mathsf{telescope}_8$,	the_7,	$\mathsf{NP} \rightarrow_2 \mathsf{DT\ NN}$ $\rangle$

Dependency Accuracies

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

- Results from Collins, 2003: 88.3% dependency accuracy

- Can calculate precision/recall on particular dependency types
  e.g., look at all subject/verb dependencies $\Rightarrow$
  all dependencies with label S $\rightarrow_2$ NP VP

  $$\text{Recall} = \frac{\text{number of subject/verb dependencies correct}}{\text{number of subject/verb dependencies in gold standard}}$$

  $$\text{Precision} = \frac{\text{number of subject/verb dependencies correct}}{\text{number of subject/verb dependencies in parser's output}}$$

Strengths and Weaknesses of Modern Parsers

(Numbers taken from Collins (2003))

- Subject-verb pairs: over 95% recall and precision
- Object-verb pairs: over 92% recall and precision
- Other arguments to verbs: $\approx$ 93% recall and precision
- Non-recursive NP boundaries: $\approx$ 93% recall and precision
- PP attachments: $\approx$ 82% recall and precision
- Coordination ambiguities: $\approx$ 61% recall and precision

Summary

- Key weakness of PCFGs: lack of sensitivity to lexical information
- Lexicalized PCFGs:
  - Lexicalize a treebank using head rules
  - Estimate the parameters of a lexicalized PCFG using smoothed estimation
- Accuracy of lexicalized PCFGs: around 88% in recovering constituents or depenencies