Natural Language Processing

Lecture 15:
Treebanks and Probabilistic CFGs

Treebanks: a (Re)introduction

Grammars Can Be Encoded Explicitly and Implicitly

Explicit

S → NP VP

NP → Det NP

VP → V NP

Det → a, the

N → professor, students

V → annoyed, delighted

Implicit

```
(S
  (NP
    (Det the)
    (N professor))
  (VP
    (V annoyed)
    (NP
      (Det the)
      (N students)))
```

The First Big Treebank Was the PTB

Penn Treebank

- **Annotation of:**
  - Brown corpus
  - ATIS (Air Travel Information Service corpus)
  - Switchboard Corpus
  - Wall Street Journal corpus
- **Size:** about 1 million words
- **Rules**:
  - 17,500 types
  - “Flat”
  - Many types with only one token

There Are Many Other Treebanks

- Dependency
- Constituency/phrase structure
- Other kinds of linguistic structure

Universal Dependency Treebanks (UD) Are For Cross-Lingual Training

- Train a dependency parser in 111 languages
  - One at a time
  - Cross-lingually

Definition of Context-Free Grammars

Vocabulary of terminal symbols, Σ

Set of nonterminal symbols, N

Special start symbol $S \in N$

Production rules of the form $X \rightarrow \alpha$

where

$X \in N$

$\alpha \in (N \cup \Sigma)^*$ (in CNF: $\alpha \in N^2 \cup \Sigma$)

Treebank Tree Example

```
( (S
    (NP-SBJ
      (NP (NNP Pierre) (NNP Vinken) )
      (, ,)
      (ADJP
        (NP (CD 61) (NNS years) )
        (JJ old) )
      (, ,) )
    (VP (MD will)
      (VP (VB join)
        (NP (DT the) (NN board) )
        (PP-CLR (IN as)
          (NP (DT a) (JJ nonexecutive) (NN
director) ) )
        (NP-TMP (NNP Nov.) (CD 29) ) ) )
    (. .) ) )
```

Proper Ambivalence toward Treebanks

Proper Ambivalence

Why you should have great respect for treebanks.

Why you should be cautious around treebanks.

The Making of a Treebank

- Develop initial coding manual (hundreds of pages long)
  - Linguists define categories and tests
  - Try to foresee as many complications as possible
- Develop annotation tools (annotation UI, pre-parser)
- Collect data (corpora)
  - Composition depends on the purpose of the corpus
  - Must also be pre-processed
- Automatically parse the corpus/corpora
- Train annotators ("coders")
- Manually correct the automatic annotations ("code")
  - Generally done by non-experts under the direction of linguists
  - When cases are encountered that are not in the coding manual…
    - Revise the coding manual to include them
    - Check that already-annotated sections of the corpus are consistent with the new standard

This is expensive and time-consuming!

You Should Respect Treebanks

They require great skill

- Expert linguists make thousands of decisions
- Many annotators must all remember all of the decisions and use them consistently, including knowing which decision to use
- The “coding manual” containing all of the decisions is hundreds of pages long

They take many years to make

- Writing the coding manual, training coders, building user-interface tools, ...
- and the coding itself with quality management

They are expensive

- Somebody had to secure the funding for these projects

You Should be Cautious Around Treebanks

They are too big to fail

They are produced under pressure of time and funding

Although most of the decisions are made by experts, most of the coding is done by non-experts

To make a good model, you should understand what you're modeling.

There are two sources of improvement in machine learning

Better models

Better data

Where do production rules come from?

```
( (S
    (NP-SBJ-1
      (NP (NNP Rudolph) (NNP Agnew) )
      (, ,)
      (UCP
        (ADJP
          (NP (CD 55) (NNS years) )
          (JJ old) )
        (CC and)
        (NP
          (NP (JJ former) (NN chairman) )
          (PP (IN of)
            (NP (NNP Consolidated) (NNP Gold) (NNP Fields) (NNP PLC) ) ) ) )
      (, ,) )
    (VP (VBD was)
      (VP (VBN named)
        (S
          (NP-SBJ (-NONE- *-1) )
          (NP-PRD
            (NP (DT a) (JJ nonexecutive) (NN director) )
            (PP (IN of)
              (NP (DT this) (JJ British) (JJ industrial) (NN conglomerate) ) ) ) ) ) )
    (. .) ) )
```

Some PTB Rules by Frequency

40717 PP → IN NP

33803 S → NP-SBJ VP

22513 NP-SBJ → -NONE-

21877 NP → NP PP

20740 NP → DT NN

14153 S → NP-SBJ VP .

12922 VP → TO VP

11881 PP-LOC → IN NP

11467 NP-SBJ → PRP

11378 NP → -NONE-

11291 NP → NN

...

989 VP → VBG S

985 NP-SBJ → NN

983 PP-MNR → IN NP

983 NP-SBJ → DT

969 VP → VBN VP

...

100 VP → VBD PP-PRD

100 PRN → : NP :

100 NP → DT JJS

100 NP-CLR → NN

99 NP-SBJ-1 → DT NNP

98 VP → VBN NP PP-DIR

98 VP → VBD PP-TMP

98 PP-TMP → VBG NP

97 VP → VBD ADVP-TMP VP

...

10 WHNP-1 → WRB JJ

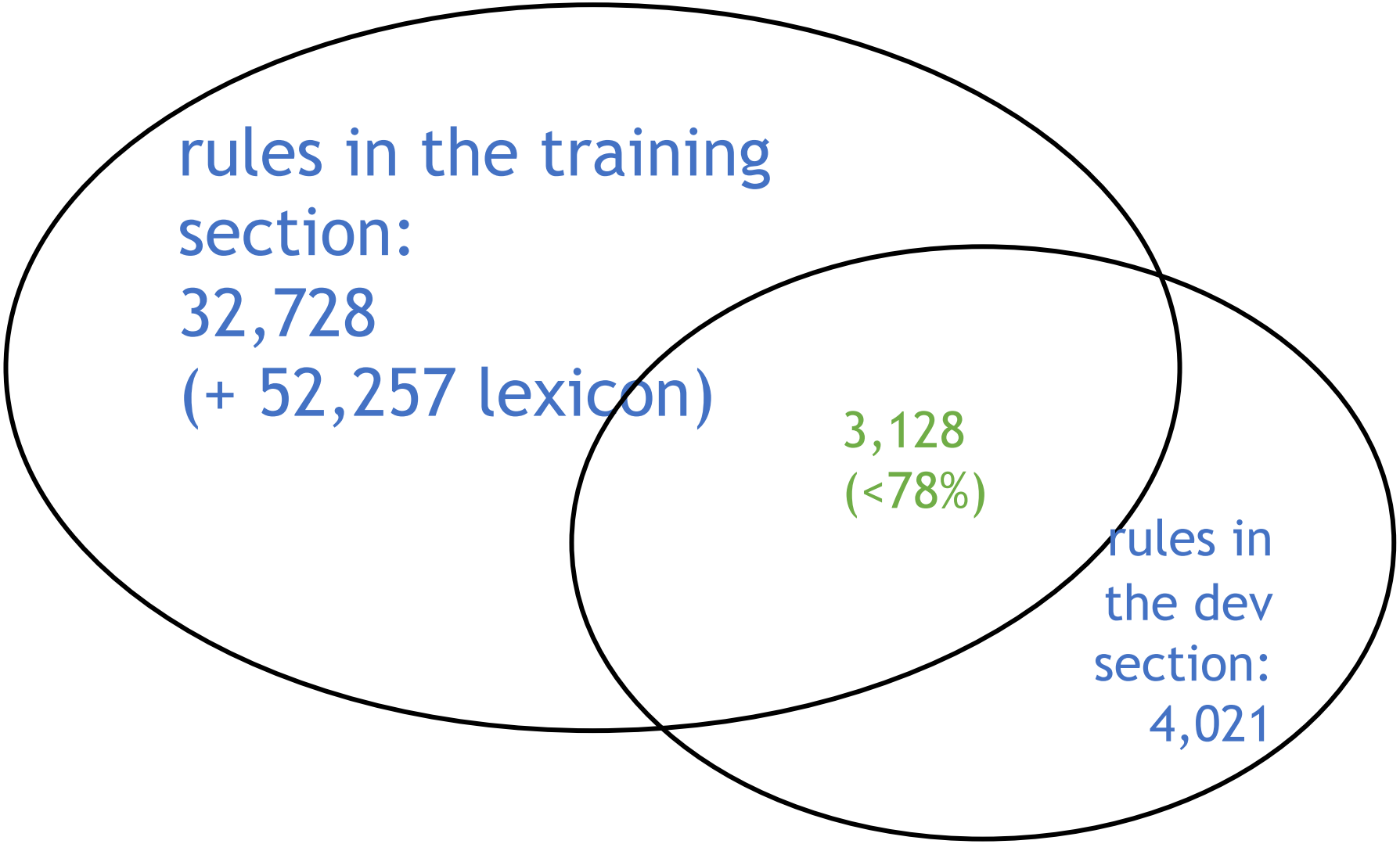
10 VP → VP CC VP PP-TMP

10 VP → VP CC VP ADVP-MNR

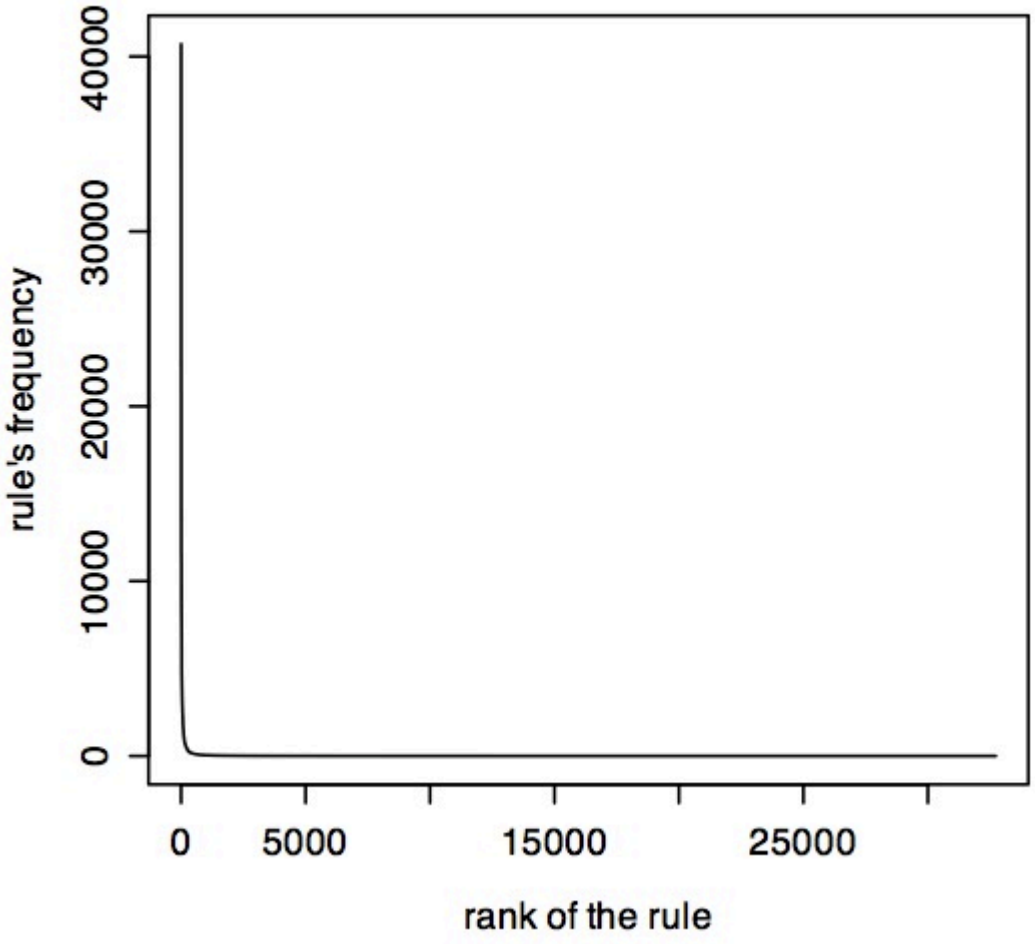
10 VP → VBZ S , SBAR-ADV

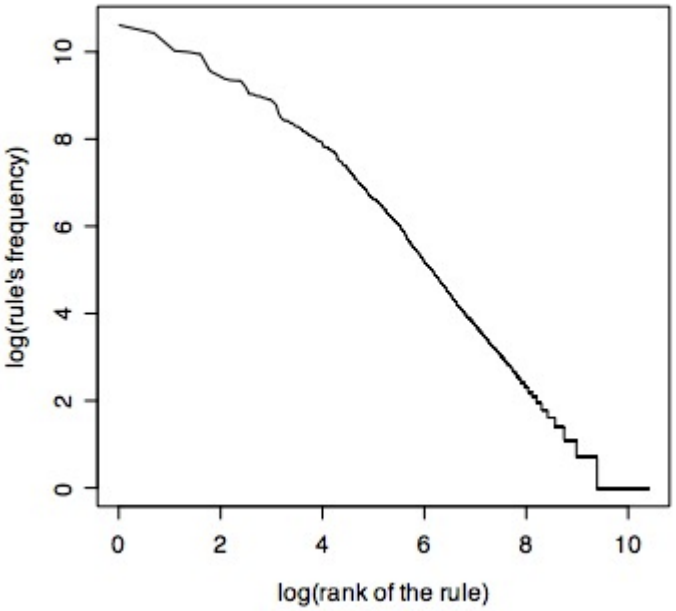
10 VP → VBZ S ADVP-TMP

Rules in the Penn Treebank



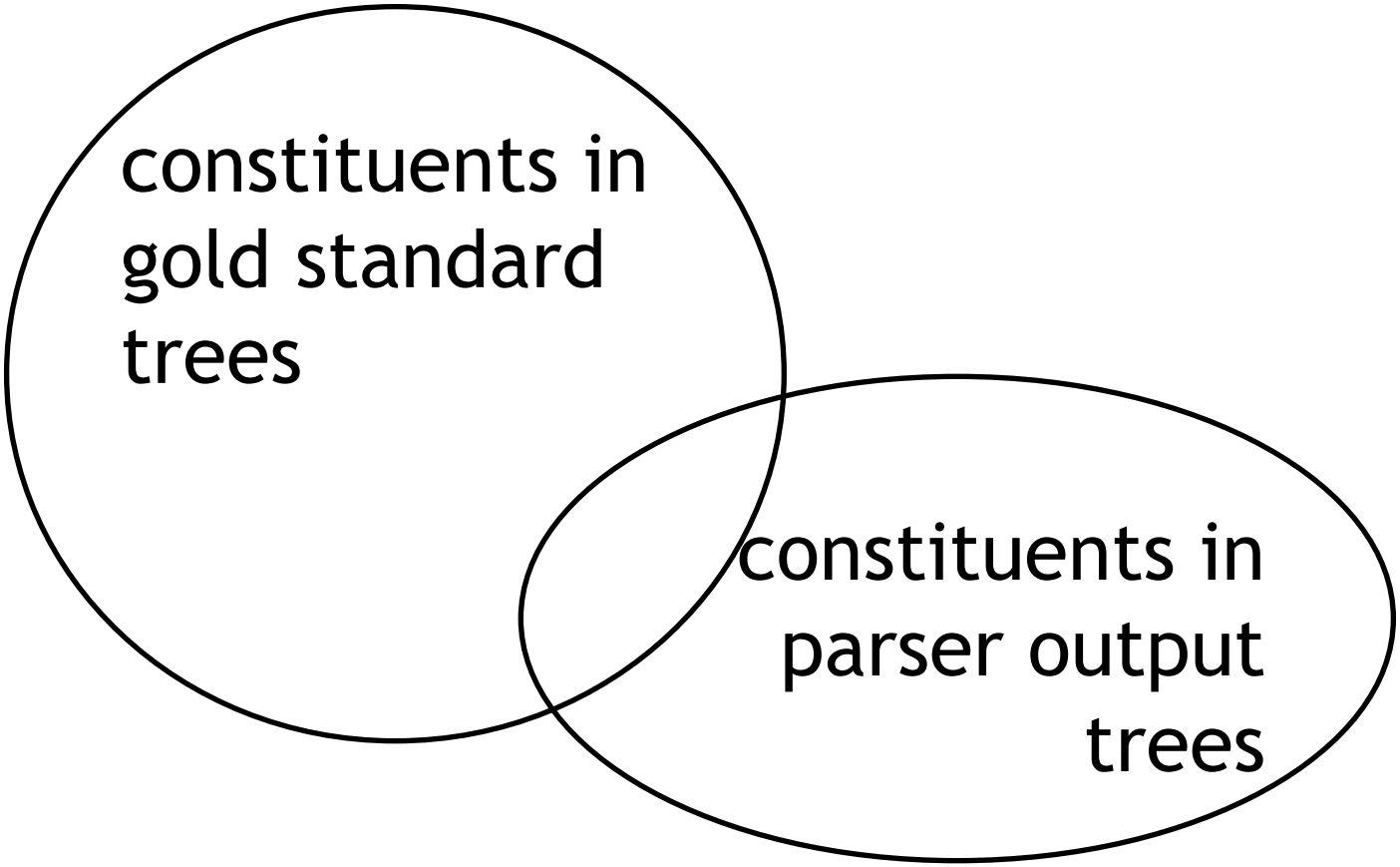
The Most Frequent Rules Are **Very** Frequent. The Rest Are Infrequent.





Evaluation of Parsing

Evaluation for Parsing: **Parseval**



Defining Parseval

labeled recall: $= \dfrac{\text{\# of correct constituents in candidate parse of } s}{\text{\# of correct constituents in treebank parse of } s}$

labeled precision: $= \dfrac{\text{\# of correct constituents in candidate parse of } s}{\text{\# of total constituents in candidate parse of } s}$

cross-brackets: the number of crossed brackets (e.g. the number of constituents for which the treebank has a bracketing such as ((A B) C) but the candidate parse has a bracketing such as (A (B C))).

The F-Measure and F_1

$$F_\beta = \frac{(\beta^2 + 1)PR}{\beta^2 P + R}$$

$$F_1 = \frac{2PR}{P + R}$$

Probabilistic context-free grammars

Two Related Problems

- **Input:** sentence $\boldsymbol{w} = (w_1, \ldots, w_n)$ and CFG G
- **Output (recognition):** true iff $\boldsymbol{w} \in$ Language(G)
- **Output (parsing):** one or more derivations for $\boldsymbol{w}$, under G

Probabilistic Context-Free Grammars

Vocabulary of terminal symbols, Σ

Set of nonterminal symbols, N

Special start symbol $S \in N$

Production rules of the form $X \rightarrow \alpha$, each with a positive weight $p(X \rightarrow \alpha)$,

where

$X \in N$

$\alpha \in (N \cup \Sigma)^*$ (in CNF: $\alpha \in N^2 \cup \Sigma$)

$\forall X \in N, \ \sum_\alpha p(X \rightarrow \alpha) = 1$

A Sample PCFG

$S \rightarrow NP\ VP$	[.80]	$Det \rightarrow that$ [.05] | the [.80] | a	[.15]
$S \rightarrow Aux\ NP\ VP$	[.15]	$Noun \rightarrow book$	[.10]
$S \rightarrow VP$	[.05]	$Noun \rightarrow flights$	[.50]
$NP \rightarrow Det\ Nom$	[.20]	$Noun \rightarrow meal$	[.40]
$NP \rightarrow Proper\text{-}Noun$	[.35]	$Verb \rightarrow book$	[.30]
$NP \rightarrow Nom$	[.05]	$Verb \rightarrow include$	[.30]
$NP \rightarrow Pronoun$	[.40]	$Verb \rightarrow want$	[.40]
$Nom \rightarrow Noun$	[.75]	$Aux \rightarrow can$	[.40]
$Nom \rightarrow Noun\ Nom$	[.20]	$Aux \rightarrow does$	[.30]
$Nom \rightarrow Proper\text{-}Noun\ Nom$	[.05]	$Aux \rightarrow do$	[.30]
$VP \rightarrow Verb$	[.55]	$Proper\text{-}Noun \rightarrow TWA$	[.40]
$VP \rightarrow Verb\ NP$	[.40]	$Proper\text{-}Noun \rightarrow Denver$	[.40]
$VP \rightarrow Verb\ NP\ NP$	[.05]	$Pronoun \rightarrow you$ [.40] | I [.60]	

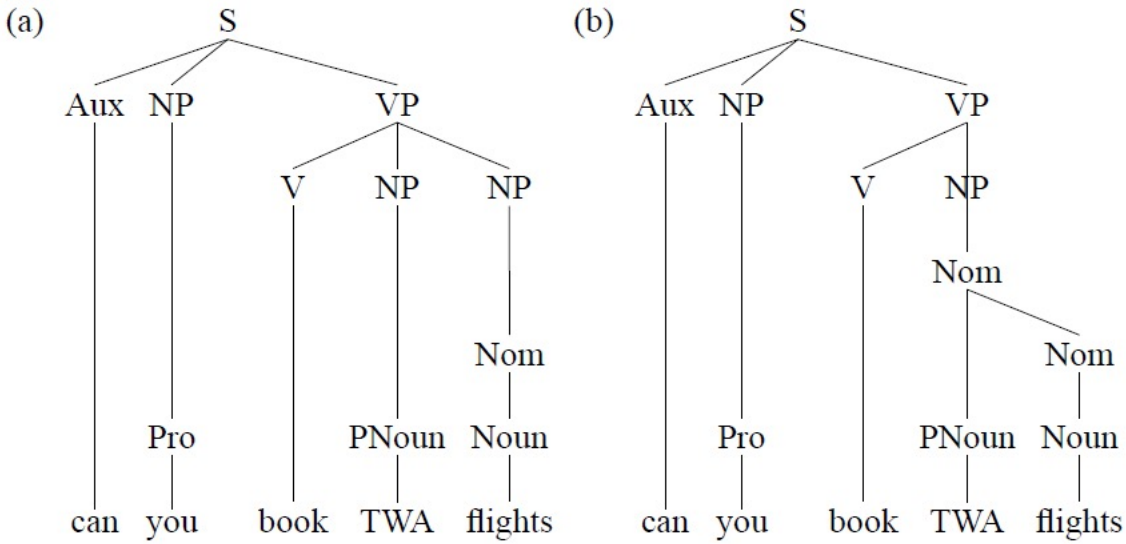
Figure 12.1 A PCFG; a probabilistic augmentation of the miniature English grammar and lexicon in Figure 10.2. These probabilities are not based on a corpus; they were made up merely for expository purposes.

$$P(T, S) = \prod_{n \in T} p(r(n))$$

The Probability of a Parse Tree

The joint probability of a particular parse *T* and sentence *S*, is defined as the product of the probabilities of all the rules *r* used to expand each node *n* in the parse tree:

An Example—Disambiguation



An Example—Disambiguation

- Consider the productions for each parse:

Rules			P	Rules			P
S	→	Aux NP VP	.15	S	→	Aux NP VP	.15
NP	→	Pro	.40	NP	→	Pro	.40
VP	→	V NP NP	.05	VP	→	V NP	.40
NP	→	Nom	.05	NP	→	Nom	.05
NP	→	PNoun	.35	Nom	→	PNoun Nom	.05
Nom	→	Noun	.75	Nom	→	Noun	.75
Aux	→	Can	.40	Aux	→	Can	.40
NP	→	Pro	.40	NP	→	Pro	.40
Pro	→	you	.40	Pro	→	you	.40
Verb	→	book	.30	Verb	→	book	.30
PNoun	→	TWA	.40	Pnoun	→	TWA	.40
Noun	→	flights	.50	Noun	→	flights	.50

Probabilities of the Two Parses

$$P(T_l) = .15 * .40 * .05 * .05 * .35 * .75 * .40 * .40 * .40 * .30 * .40 * .50 = 1.5 \times 10^{-6}$$

book flights for (on behalf of) TWA

$$P(T_r) = .15 * .40 * .40 * .05 * .05 * .75 * .40 * .40 * .40 * .30 * .40 * .50 = 1.7 \times 10^{-6}$$

book flights that are on TWA

We favor the tree on the right in disambiguation because it has a higher probability.

What Can You Do With a PCFG?

- Just as with CFGs, PCFGs can be used for both parsing and generation, but they have advantages in both areas:
  - **Parsing**
    - CFGs are good for “precision” parsers
    - PCFGs are good for robust parsers
    - PCFGs can disambiguate
  - **Generation**
    - CFGs generate grammatical sentences
    - PCFGs can generate more natural sentences

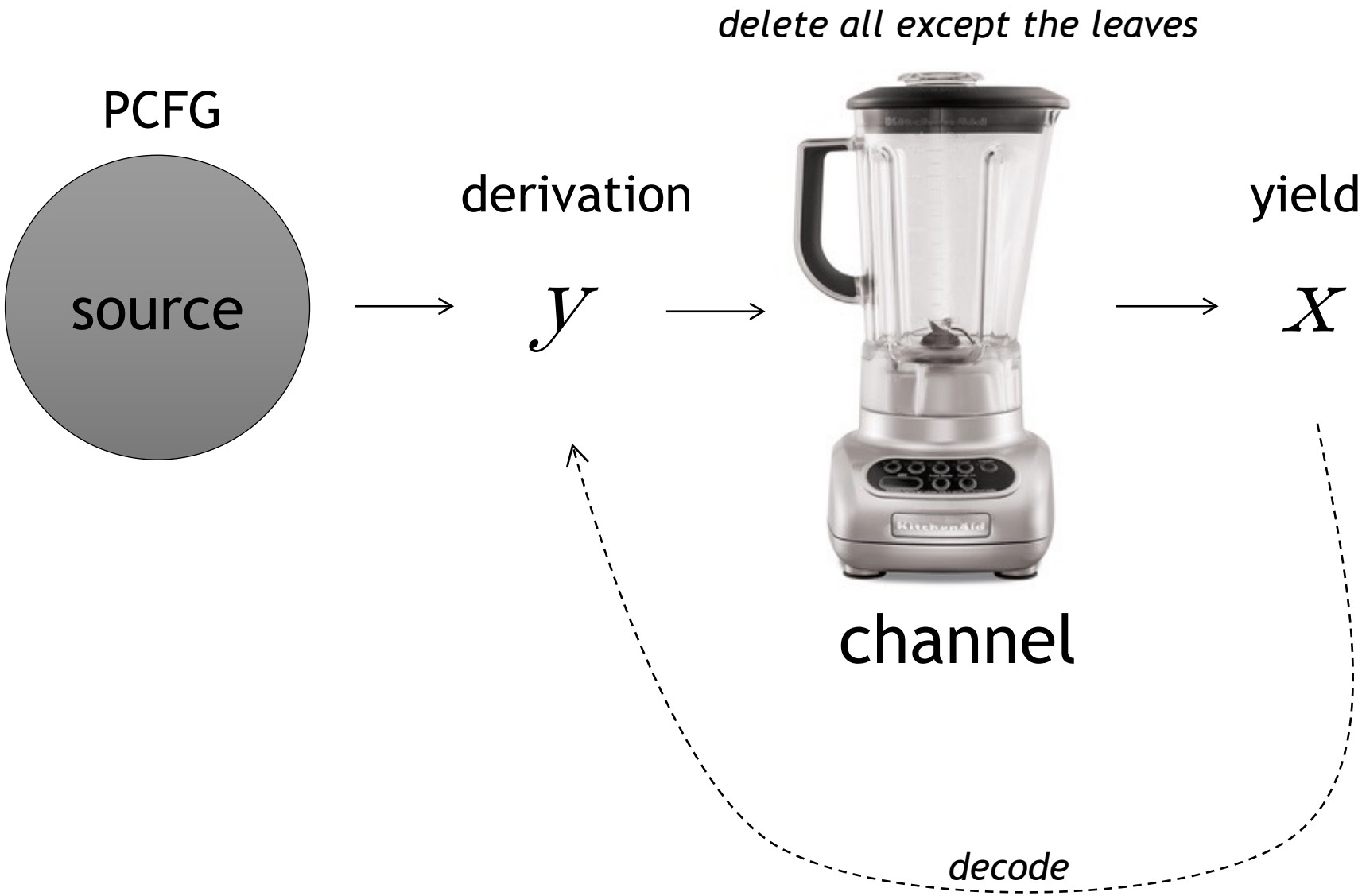
Where Do the Probabilities in PCFGs Come From?

- From a **tree bank**

$$P(\alpha \rightarrow \beta|\alpha) = \frac{\text{Count}(\alpha \rightarrow \beta)}{\sum_\gamma \text{Count}(\alpha \rightarrow \gamma)} = \frac{\text{Count}(\alpha \rightarrow \beta)}{\text{Count}(\alpha)}$$

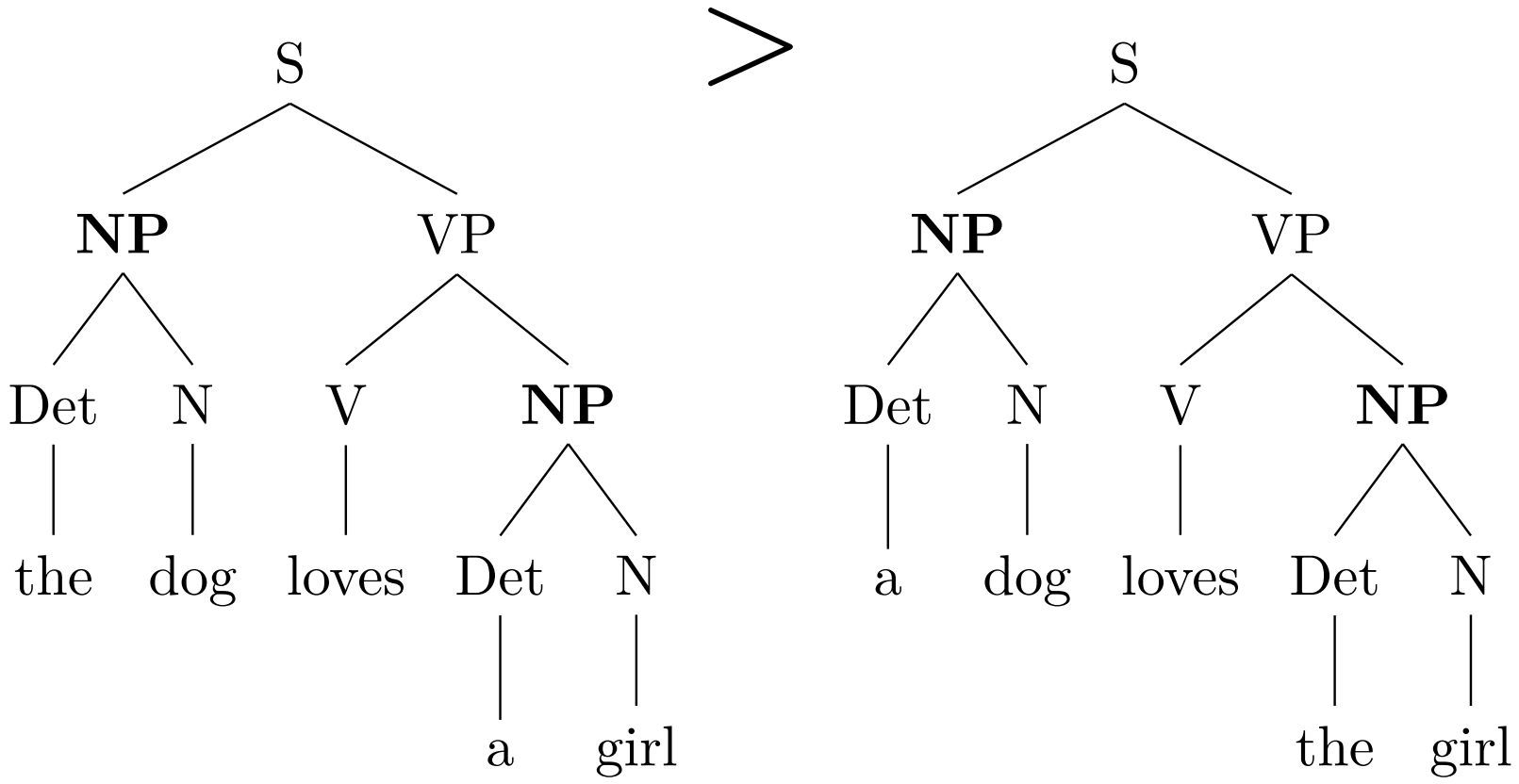
- From a **corpus**
  - Parse the corpus with your CFG
  - Count the rules for each parse
  - Normalize
  - But wait, most sentences are ambiguous!
    - "Keep a separate count for each parse of a sentence and weigh each partial count by the probability of the parse it appears in."

PCFGs as a Noisy Channel



Problems with PCFGs

Subject NPs are more likely to have *the*; Object NPs are more likely to have *a*

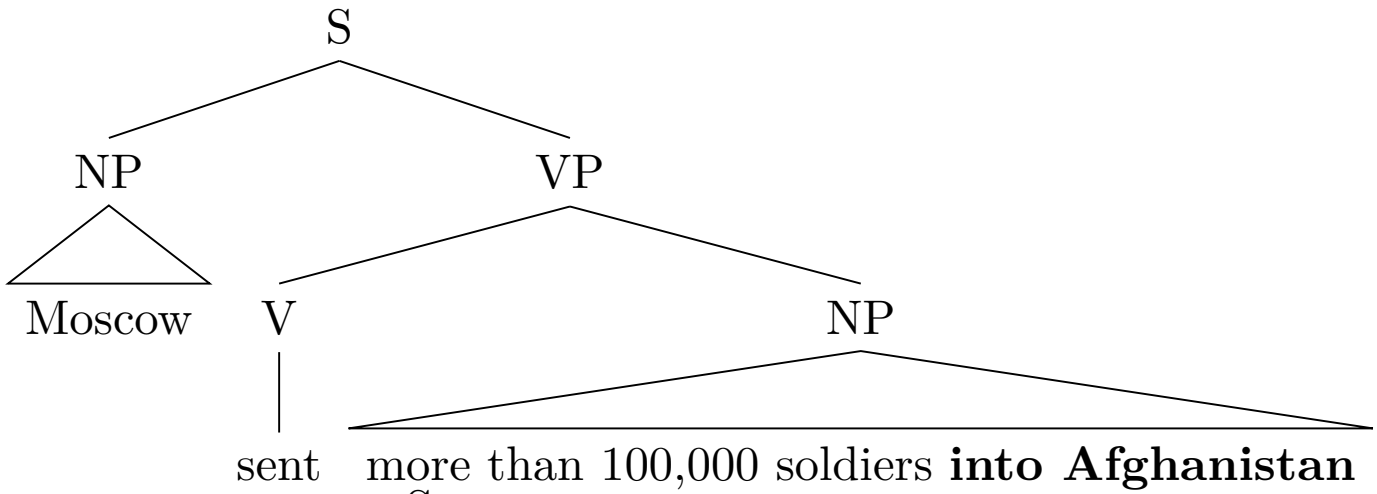


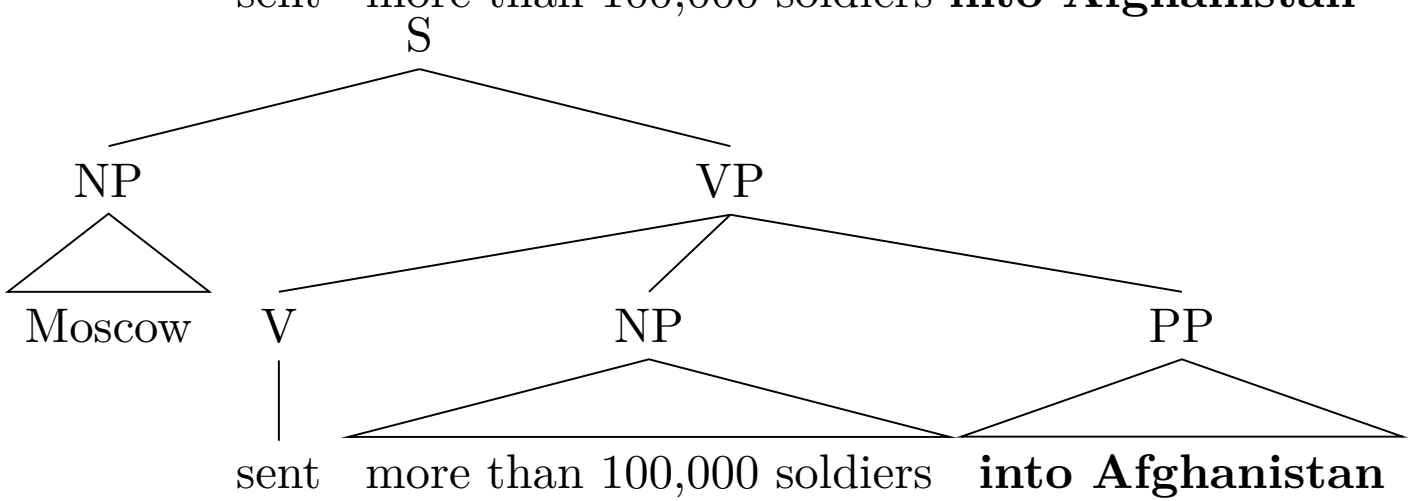
An example of a structural dependency!

Vanilla PCFGs Cannot Capture Structural Dependencies

- (P)CFG rules apply independently
  - S → **NP** VP
  - VP → V **NP**
  - Same rules apply to all **NP**s, regardless of context
  - *the* more common in subjects than *a*

Vanilla PCFGs Cannot Capture Lexical Dependencies





Probabilistic Lexicalized context free grammars

The Word that Defines/Constrains the Rest of a Constituent Is the **Head**

- Noun phrases (NPs) are headed by nouns
- Verb phrases (VPs) are headed by verbs
- In lexicalized grammars/trees we augment the label with the head

A Example of a Lexicalized Tree

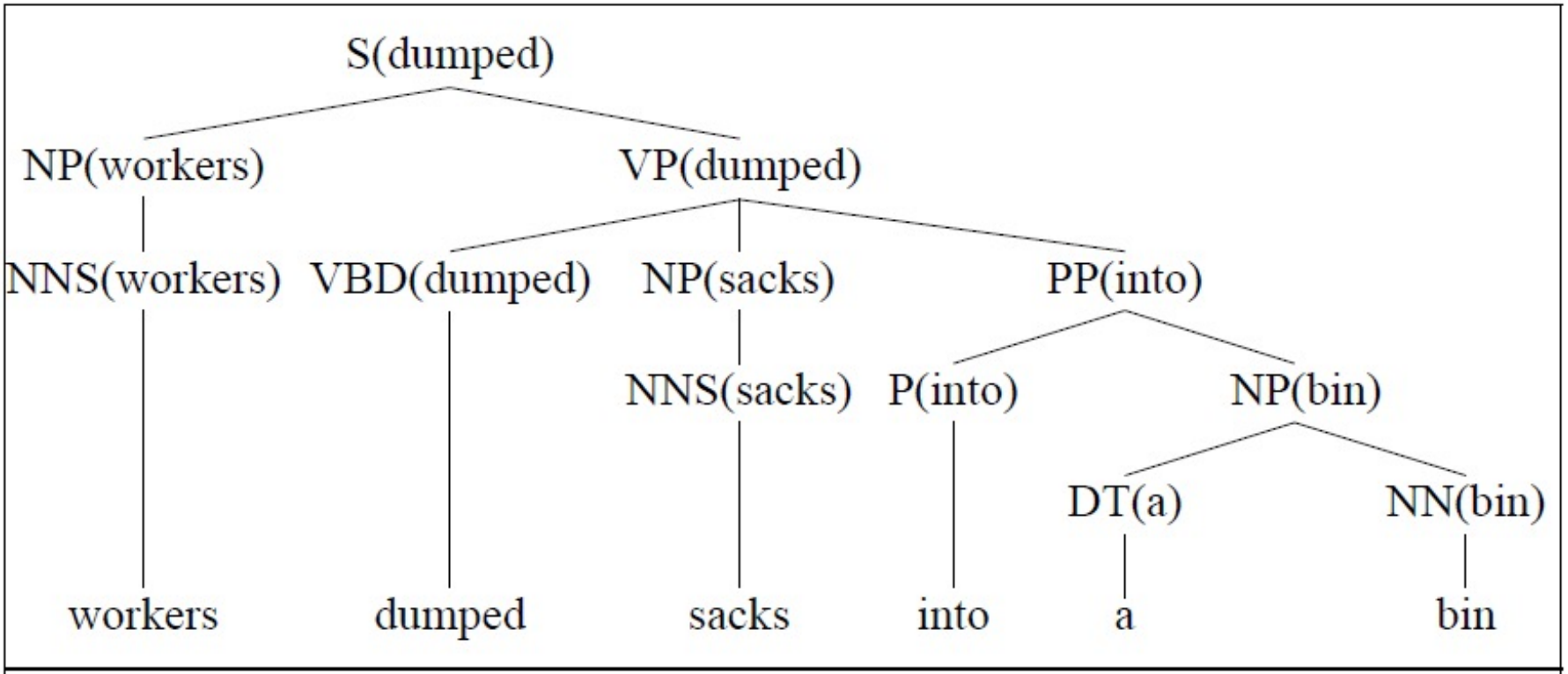


Figure 12.5 A lexicalized tree from Collins (1999).

Lexicalized Grammars

- Fancy lexicalized grammars exist (LFG, HPSG, etc.)
- We keep it simple
- **A PCFG with more rules**

Lexicalized Grammars

- Viewed in this way...
  - Lexicalized grammars are huge (perhaps impractically huge)
  - However, they can be used with the same algorithms we have already learned
- Lexicalized grammars require more training data than vanilla PCFGs, but they can capture probabilistic patterns that PCFGs could never capture
- In most contemporary applications, lexicalized grammars are chosen over vanilla PCFGs

Questions?