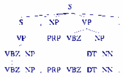
CS 294-5: Statistical Natural Language Processing

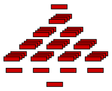
Parsing: PCFGs

Dan Klein

Learning vs. Inference

- There are two aspects to parsing:
  - Learning: designing a good grammar.
    - Coverage
    - Ambiguity resolution
    - Smoothing
  - Inference: parsing with a given grammar.
    - Runtime
    - Memory load
    - Exact or approximate / pruning?

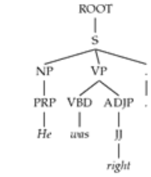




- Today we're only concerned with learning.

Treebank Parsing in 20 sec

- Need a PCFG for broad coverage parsing.
- Can take a grammar right off the trees (doesn't work well):



ROOT → S 1

S → NP VP . 1

NP → PRP 1

VP → VBD ADJP 1

…..

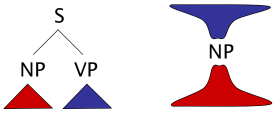
- Better results by enriching the grammar (e.g., lexicalization).
- We'll show that lexicalization isn't necessary for high-performance parsing.

PCFGs and Independence

- The symbols in a PCFG define independence assumptions:

S → NP VP

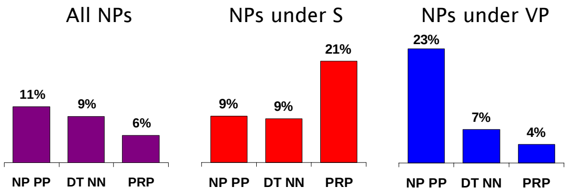
NP → DT NN



- At any node, the material inside that node is independent of the material outside that node, given the label of that node.
- Any information that statistically connects behavior inside and outside a node must flow through that node.

Non-Independence I

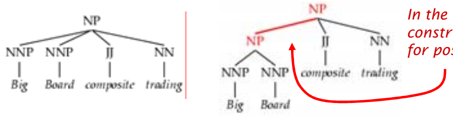
- Independence assumptions are often too strong.



- Example: the expansion of an NP is highly dependent on the parent of the NP (i.e., subjects vs. objects).

Non-Independence II

- Who cares?
  - NB, HMMs, all make false assumptions!
  - For generation, consequences would be obvious.
  - For parsing, does it impact accuracy?
- Symptoms of overly strong assumptions:
  - Rewrites get used where they don't belong.
  - Rewrites get used too often or too rarely.

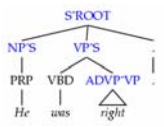


In the PTB, this construction is for possessives

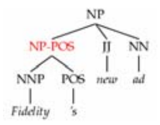
Breaking Up the Symbols

- We can relax independence assumptions by encoding dependencies into the PCFG symbols:

Parent annotation
[Johnson 98]



Marking
possessive NPs



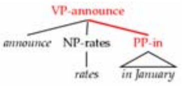
- What are the most useful features to encode?

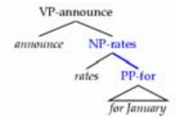
Annotations

- Annotations split the grammar categories into sub-categories.
- Conditioning on history vs. annotating
  - P(NP^S → PRP) is a lot like P(NP → PRP | S)
  - P(NP-POS → NNP POS) isn't history conditioning.
- Feature grammars vs. annotation
  - Can think of a symbol like NP^NP-POS as NP [parent:NP, +POS]
- After parsing with an annotated grammar, the annotations are then stripped for evaluation.

The Lexicalization Hammer

- Lexical heads important for certain classes of ambiguities (e.g., PP attachment):
- Lexicalizing grammar creates a mucl larger grammar.
  - Sophisticated smoothing needed
  - Smarter parsing algorithms
  - More data needed
- How necessary is lexicalization?
  - Bilexical vs. monolexical selection
  - Closed vs. open class lexicalization





Unlexicalized PCFGs

- What do we mean by an "unlexicalized" PCFG?
  - Grammar rules are not systematically specified down to the level of lexical items
    - NP-stocks is not allowed
    - NP^S-CC is fine
  - Closed vs. open class words (NP^S-the)
    - Long tradition in linguistics of using function words as features or markers for selection
    - Contrary to the bilexical idea of semantic heads
    - Open-class selection really a proxy for semantics
- Honesty checks:
  - Number of symbols: keep the grammar very small
  - No smoothing: over-annotating is a real danger

Experimental Setup

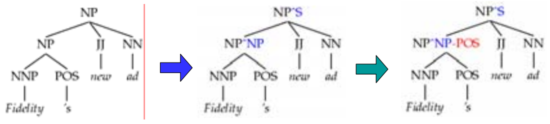
- Corpus: Penn Treebank, WSJ

Training:	sections	02-21
Development:	section	22 (first 20 files)
Test:	section	23

- Accuracy – F1: harmonic mean of per-node labeled precision and recall.
- Size – number of symbols in grammar.
  - Passive / complete symbols: NP, NP^S
  - Active / incomplete symbols: NP → NP CC •

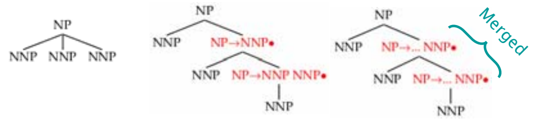
Experimental Process

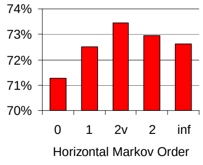
- We'll take a highly conservative approach:
  - Annotate as sparingly as possible
  - Highest accuracy with fewest symbols
  - Error-driven, manual hill-climb, adding one annotation type at a time

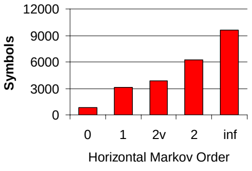


Horizontal Markovization

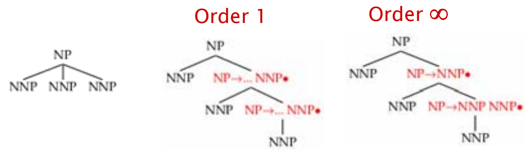
- Horizontal Markovization: Merges States

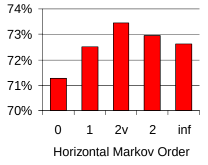


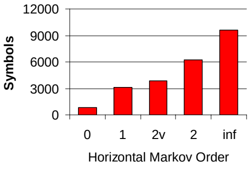




Horizontal Markovization



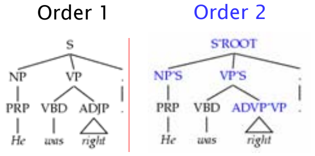


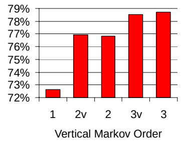


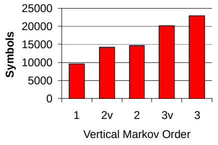
Vertical Markovization

- Vertical Markov order: rewrites depend on past *k* ancestor nodes.

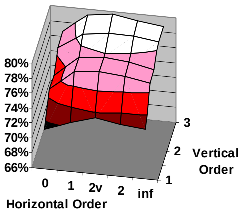
  (cf. parent annotation)

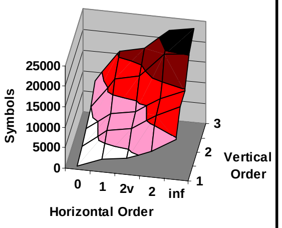






Vertical and Horizontal



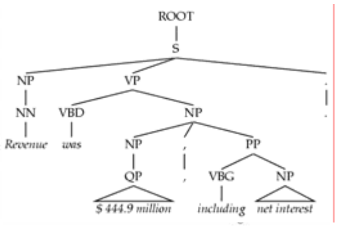


- Examples:
  - Raw treebank: v=1, h=∞
  - Johnson 98: v=2, h=∞
  - Collins 99: v=2, h=2
  - Best F1: v=3, h=2v

Model	F1	Size
Base: v=h=2v	77.8	7.5K

Unary Splits

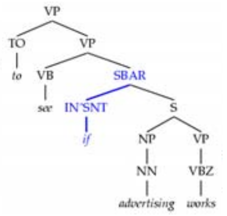
- Problem: unary rewrites used to transmute categories so a high-probability rule can be used.
- Solution: Mark unary rewrite sites with -U



Annotation	F1	Size
Base	77.8	7.5K
UNARY	78.3	8.0K

Tag Splits

- Problem: Treebank tags are too coarse.
- Example: Sentential, PP, and other prepositions are all marked IN.
- Partial Solution:
  - Subdivide the IN tag.



Annotation	F1	Size
Previous	78.3	8.0K
SPLIT-IN	80.3	8.1K

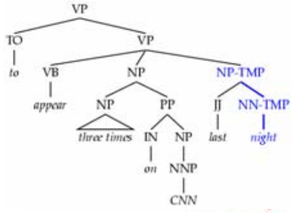
Other Tag Splits

- UNARY-DT: mark demonstratives as DT^U (“the X” vs. “those”)
- UNARY-RB: mark phrasal adverbs as RB^U (“quickly” vs. “very”)
- TAG-PA: mark tags with non-canonical parents (“not” is an RB^VP)
- SPLIT-AUX: mark auxiliary verbs with –AUX [cf. Charniak 97]
- SPLIT-CC: separate “but” and “&” from other conjunctions
- SPLIT-%: “%” gets its own tag.

F1	Size
80.4	8.1K
80.5	8.1K
81.2	8.5K
81.6	9.0K
81.7	9.1K
81.8	9.3K

Treebank Splits

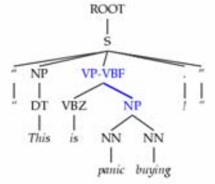
- The treebank comes with annotations (e.g., -LOC, -SUBJ, etc).
  - Whole set together hurt the baseline.
  - Some (-SUBJ) were less effective than our equivalents.
  - One in particular was very useful (NP-TMP) when pushed down to the head tag.
  - We marked gapped S nodes as well.



Annotation	F1	Size
Previous	81.8	9.3K
NP-TMP	82.2	9.6K
GAPPED-S	82.3	9.7K

Yield Splits

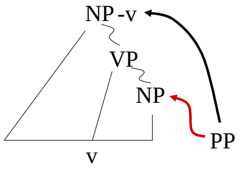
- Problem: sometimes the behavior of a category depends on something inside its future yield.
- Examples:
  - Possessive NPs
  - Finite vs. infinite VPs
  - Lexical heads!
- Solution: annotate future elements into nodes.



Annotation	F1	Size
Previous	82.3	9.7K
POSS-NP	83.1	9.8K
SPLIT-VP	85.7	10.5K

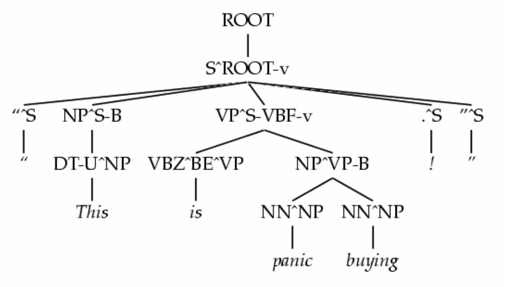
Distance / Recursion Splits

- Problem: vanilla PCFGs cannot distinguish attachment heights.
- Solution: mark a property of higher or lower sites:
  - Contains a verb.
  - Is (non)-recursive.
    - Base NPs [cf. Collins 99]
    - Right-recursive NPs



Annotation	F1	Size
Previous	85.7	10.5K
BASE-NP	86.0	11.7K
DOMINATES-V	86.9	14.1K
RIGHT-REC-NP	87.0	15.2K

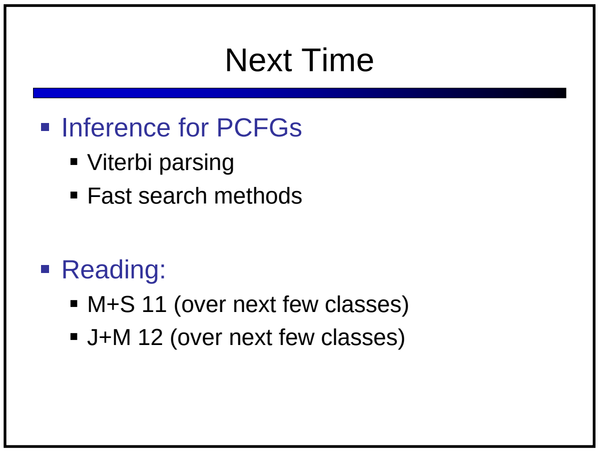
A Fully Annotated Tree



Final Test Set Results

Parser	LP	LR	F1	CB	0 CB
Magerman 95	84.9	84.6	**84.7**	1.26	56.6
Collins 96	86.3	85.8	**86.0**	1.14	59.9
Current Work	86.9	85.7	**86.3**	1.10	60.3
Charniak 97	87.4	87.5	**87.4**	1.00	62.1
Collins 99	88.7	88.6	**88.6**	0.90	67.1

- Beats “first generation” lexicalized parsers.



Next Time

- Inference for PCFGs
  - Viterbi parsing
  - Fast search methods
- Reading:
  - M+S 11 (over next few classes)
  - J+M 12 (over next few classes)