## Hand video

- http://www.youtube.com/watch?v=-KxjVlaLBmk

# ADVANCED PARSING

David Kauchak
CS457 – Spring 2011

*some slides adapted from Dan Klein*

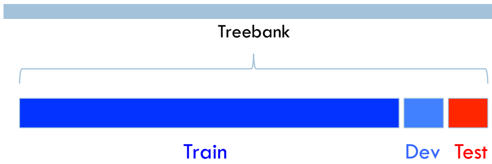
## Admin

- Assignment 2 grades e-mailed
- Assignment 3?
- Survey
  - Thanks for the feedback
  - NLP within AI

## Parsing evaluation

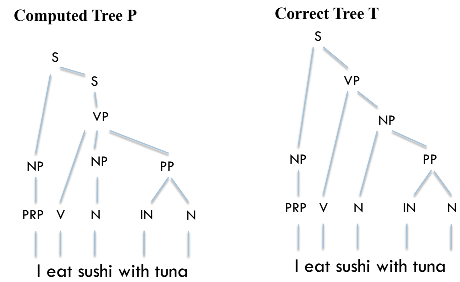
- You've constructed a parser
- You want to know how good it is
- Ideas?

## Parsing evaluation




- Learn a model using the training set
- Parse the test set without looking at the "correct" trees
- Compare our generated parse tree to the "correct" tree

## Comparing trees

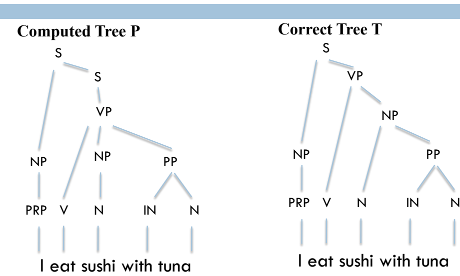



Ideas?

## Comparing trees

- Idea 1: see if the trees match exactly
  - Problems?
    - Will have a low number of matches (people often disagree)
    - Doesn't take into account getting it *almost* right
- Idea 2: compare the constituents

## Comparing trees




How many constituents match?

How can we turn this into a score?

## Evaluation measures

- Precision

$$\frac{\text{\# of correct constituents}}{\text{\# of constituents in the computed tree}}$$

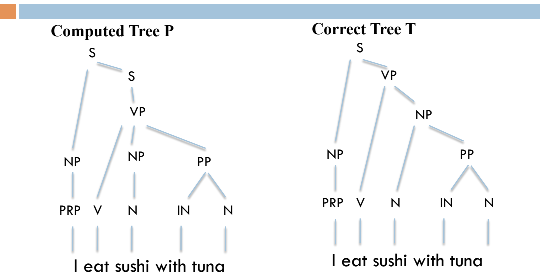
- Recall

$$\frac{\text{\# of correct constituents}}{\text{\# of constituents in the correct tree}}$$

- F1

$$\frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

## Comparing trees




# Constituents: 11 # Correct Constituents: 9 # Constituents: 10

Precision: 9/11 Recall: 9/10 F1: 0.857

## Parsing evaluation

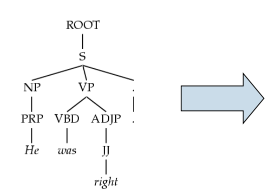
- Corpus: Penn Treebank, WSJ

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

- Parsing has been fairly standardized to allow for easy comparison between systems

## Treebank PCFGs

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



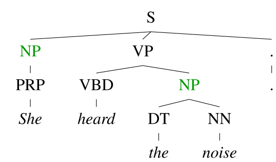

ROOT → S
S → NP VP .
NP → PRP
VP → VBD ADJP
…..

| Model | F1 |
|---|---|
| Baseline | 72.0 |

## Generic PCFG Limitations

- PCFGs do not use any information about where the current constituent is in the tree
- PCFGs do not rely on specific words or concepts, only general structural disambiguation is possible (e.g. prefer to attach PPs to Nominals)
- MLE estimates are not always the best

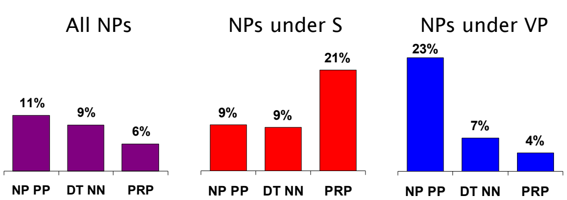
## Conditional Independence?




- Not every NP expansion can fill every NP slot
  - A grammar with symbols like "NP" won't be context-free
  - Statistically, conditional independence too strong

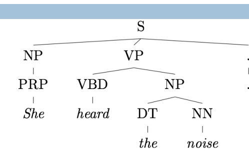
## Non-Independence

- Independence assumptions are often too strong



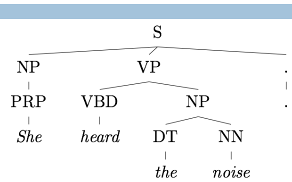

- Example: the expansion of an NP is highly dependent on the parent of the NP (i.e., subjects vs. objects).
- Also: the subject and object expansions are correlated

## Grammar Refinement



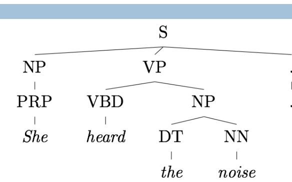

- PCFG would treat these two NPs the same… but they're not!
- We can't exchange them: "the noise heard she"
- Idea: expand/refine our grammar
- Challenges:
  - Must refine in ways that facilitate disambiguation
  - Must trade-offs between too little and too much refinement. Concerns?
    - Too much refinement -> sparsity problems
    - To little -> can't discriminate (PCFG)

## Grammar Refinement



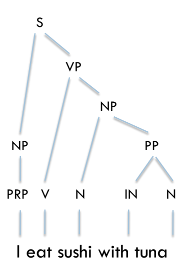

Ideas?

## Grammar Refinement




- Structure Annotation [Johnson '98, Klein&Manning '03]
  - Differentiate constituents based on their local context
- Lexicalization [Collins '99, Charniak '00]
  - Differentiate constituents based on the spanned words
- Constituent splitting [Matsuzaki et al. 05, Petrov et al. '06]
  - Cluster/group words into sub-constituents

## Less independence

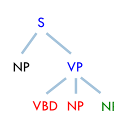



S -> NP VP
NP -> PRP
PRP -> I
VP -> V NP
V -> eat
NP -> N PP
N -> sushi
PP -> IN N
IN -> with
N -> tuna

We're making a strong independence assumption here!

## Markovization

- Except for the root node, every node in a parse tree has:
  - A vertical history/context
  - A horizontal history/context

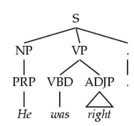



Traditional PCFGs use the full horizontal context and a vertical context of 1

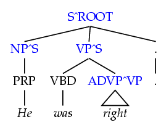
## Vertical Markovization

- Vertical Markov order: rewrites depend on past $k$ ancestor nodes.
- Order 1 is most common: aka parent annotation

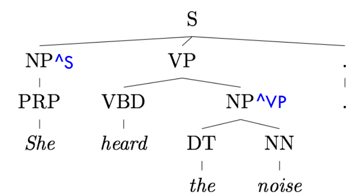
Order 1



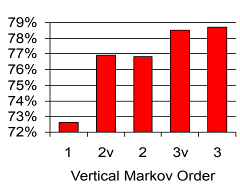

Order 2

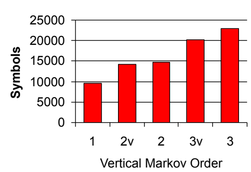



## Allows us to make finer grained distinctions




## Vertical Markovization



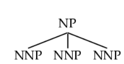




F1 performance

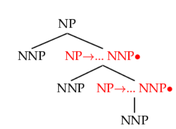
\# of non-terminals

## Horizontal Markovization

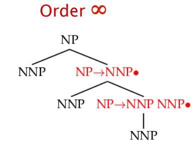
- Horizontal Markov order: rewrites depend on past $k$ ancestor nodes
- Order 1 is most common: condition on a single sibling



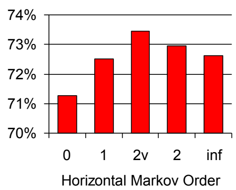

Order 1

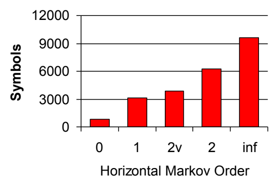



Order ∞



## Horizontal Markovization

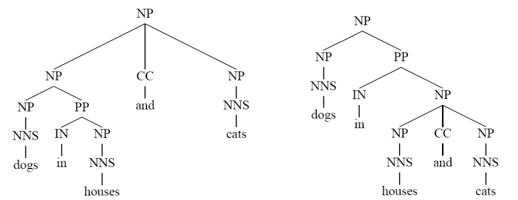



F1 performance

\# of non-terminals

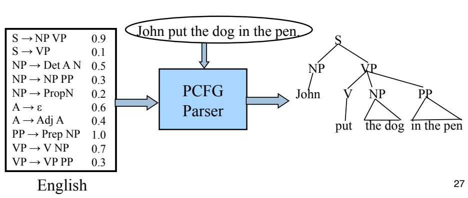
## Problems with PCFGs




- What's different between basic PCFG scores here?

## Example of Importance of Lexicalization

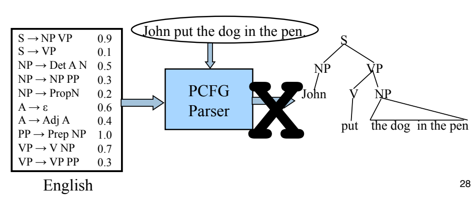
- A general preference for attaching PPs to NPs rather than VPs can be learned by a vanilla PCFG
- But the desired preference can depend on specific words



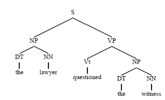



## Example of Importance of Lexicalization

- A general preference for attaching PPs to NPs rather than VPs can be learned by a vanilla PCFG
- But the desired preference can depend on specific words



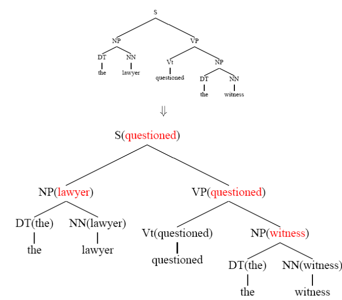
## Lexicalized Trees



How could we lexicalize the grammar/tree?

## Lexicalized Trees

- Add "headwords" to each phrasal node
  - Syntactic vs. semantic heads
  - Headship not in (most) treebanks
  - Usually *use head rules*, e.g.:
    - NP:
      - Take leftmost NP
      - Take rightmost N*
      - Take rightmost JJ
      - Take right child
    - VP:
      - Take leftmost VB*
      - Take leftmost VP
      - Take left child




## Lexicalized PCFGs?

- Problem: we now have to estimate probabilities like

$$VP(put) \rightarrow VBD(put)\ NP(dog)\ PP(in)$$

- How would we estimate the probability of this rule?

$$\frac{\text{Count}(VP(put) \rightarrow VBD(put)\ NP(dog)\ PP(in))}{\text{Count}(VP\ (put))}$$

- Never going to get these automically off of a treebank
- Ideas?

## One approach

- Combine this with some of the markovization techniques we saw
- Collins' (1999) parser
  - Models productions based on context to the left and the right of the head daughter.
    - LHS $\rightarrow L_n L_{n-1} \ldots L_1 H\ R_1 \ldots R_{m-1} R_m$
  - First generate the head (H) and then repeatedly generate left ($L_i$) and right ($R_i$) context symbols until the symbol STOP is generated.

## Sample Production Generation

VPput → VBDput NPdog PPin

Note: Penn treebank tends to have fairly flat parse trees that produce long productions.

| VPput → | STOP | VBDput | NPdog | PPin | STOP |
|---|---|---|---|---|---|
| | $L_1$ | H | $R_1$ | $R_2$ | $R_3$ |

$P_L$(STOP | VPput) * $P_H$(VBD | Vpput)*
$P_R$(NPdog | VPput)*
$P_R$(PPin | VPput) * $P_R$(STOP | PPin)

## Estimating Production Generation Parameters

- Estimate $P_H$, $P_L$, and $P_R$ parameters from treebank data

$$P_R(\text{PPin} \mid \text{VPput}) = \frac{\text{Count(PPin right of head in a VPput production)}}{\text{Count(symbol right of head in a VPput-VBD)}}$$

$$P_R(\text{NPdog} \mid \text{VPput}) = \frac{\text{Count(NPdog right of head in a VPput production)}}{\text{Count(symbol right of head in a VPput)}}$$

- Smooth estimates by combining with simpler models conditioned on just POS tag or no lexical info

$$smP_R(\text{PPin} \mid \text{VPput-}) = \lambda_1 P_R(\text{PPin} \mid \text{VPput}) + (1-\lambda_1)(\lambda_2 P_R(\text{PPin} \mid \text{VPVBD}) + (1-\lambda_2) P_R(\text{PPin} \mid \text{VP}))$$

## Problems with lexicalization

- We've solved the estimation problem
- There's also the issue of performance
- Lexicalization causes the size of the number of grammar rules to explode!
- Our parsing algorithms take too long too finish

- Ideas?

## Pruning during search

- We can no longer keep all possible parses around
- We can no longer guarantee that we actually return the most likely parse
- Beam search [Collins 99]
  - In each cell only keep the **K** most likely hypothesis
  - Disregard constituents over certain spans (e.g. punctuation)
  - F1 of 88.6!

## Pruning with a PCFG

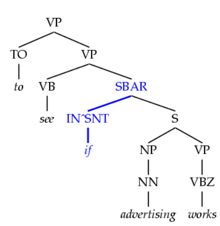
- The Charniak parser prunes using a two-pass approach [Charniak 97+]
  - First, parse with the base grammar
  - For each X:[i,j] calculate P(X|i,j,s)
    - This isn't trivial, and there are clever speed ups
  - Second, do the full CKY
    - Skip any X :[i,j] which had low (say, < 0.0001) posterior
  - Avoids almost all work in the second phase!
- F1 of 89.7!

## Tag splitting

- Lexicalization is an extreme case of splitting the tags to allow for better discrimination
- Idea: what if rather than doing it for all words, we just split some of the tags

## Tag Splits

- Problem: Treebank tags are too coarse
  - We even saw this with the variety of tagsets
- Example: Sentential, PP, and other prepositions are all marked IN
- Partial Solution:
  - Subdivide the IN tag



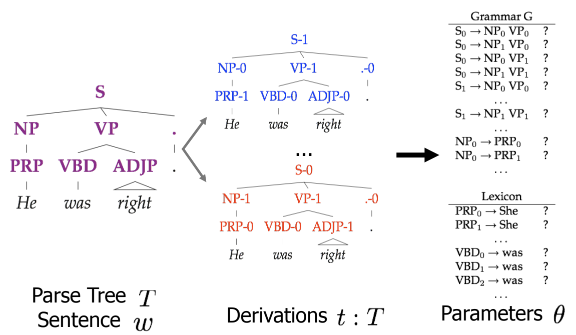

| Annotation | F1 | Size |
|---|---|---|
| Previous | 78.3 | 8.0K |
| SPLIT-IN | 80.3 | 8.1K |

## Other Tag Splits

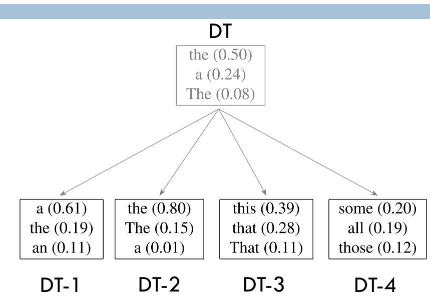
- UNARY-DT: mark demonstratives as DT^U ("the X" vs. "those")
- UNARY-RB: mark phrasal adverbs as RB^U ("quickly" vs. "very")
- TAG-PA: mark tags with non-canonical parents ("not" is an RB^VP)
- SPLIT-AUX: mark auxiliary verbs with –AUX [cf. Charniak 97]
- SPLIT-CC: separate "but" and "&" from other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

## Learning good splits: Latent Variable Grammars




## Refinement of the DT tag




## Learned Splits

- Proper Nouns (NNP):

| | | | |
|---|---|---|---|
| NNP-14 | Oct. | Nov. | Sept. |
| NNP-12 | John | Robert | James |
| NNP-2 | J. | E. | L. |
| NNP-1 | Bush | Noriega | Peters |
| NNP-15 | New | San | Wall |
| NNP-3 | York | Francisco | Street |

- Personal pronouns (PRP):

| | | | |
|---|---|---|---|
| PRP-0 | It | He | I |
| PRP-1 | it | he | they |
| PRP-2 | it | them | him |

## Learned Splits

- Relative adverbs (RBR):

| | | | |
|---|---|---|---|
| RBR-0 | further | lower | higher |
| RBR-1 | more | less | More |
| RBR-2 | earlier | Earlier | later |

- Cardinal Numbers (CD):

| | | | |
|---|---|---|---|
| CD-7 | one | two | Three |
| CD-4 | 1989 | 1990 | 1988 |
| CD-11 | million | billion | trillion |
| CD-0 | 1 | 50 | 100 |
| CD-3 | 1 | 30 | 31 |
| CD-9 | 78 | 58 | 34 |

## Final Results

| Parser | F1 ≤ 40 words | F1 all words |
|---|---|---|
| Klein & Manning ’03 | 86.3 | 85.7 |
| Matsuzaki et al. ’05 | 86.7 | 86.1 |
| Collins ’99 | 88.6 | 88.2 |
| Charniak & Johnson ’05 | 90.1 | 89.6 |
| **Petrov et. al. 06** | **90.2** | **89.7** |

## Article discussion

- Smarter Marketing and the Weak Link In Its Success
  - http://searchenginewatch.com/article/2077636/Smarter-Marketing-and-the-Weak-Link-In-Its-Success
- What are the ethics involved with tracking user interests for the purpose of advertising? Is this something you find preferable to 'blind' marketing?
- Is possible to get an accurate picture of someone's interests from their web activity? What sources would be good for doing so?
- How do you feel about websites that change content depending on the viewer? What are the implications of sites that behave this way?