COMP5046: Statistical Phrase Structure Parsing

Joel Nothman
joel.nothman@sydney.edu.au

School of Information Technologies
University of Sydney

2018-05-22

Part I

Probabilistic Context Free Grammars

Preposition attachment ambiguity

I saw the girl on the hill with the telescope

Coordination ambiguity

I do not like green eggs and bark

Coordination ambiguity

I do not like green eggs and bark

I do not like green (eggs and bark)

I do not like (green eggs and bark)

I do not (like green eggs and bark)

I (do not like green eggs and bark)

Coordination ambiguity

I do not like green eggs and bark

Note: this case also relies on homonymy resulting in a part of speech ambiguity

Noun compound ambiguity

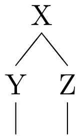
Natural language processing is tricky

Ambiguity

- (almost) any useful natural language grammar is ambiguous
- and most wide-coverage grammar is **highly ambiguous**
- we must choose the correct parse amongst many
- we do this using **statistical models** of language
- and select the most probable parse

Context-free Phrase Structure Grammar

- A CFG consists of a list of *rewrite rules* or *production rules*
- Each rule is of the form $\langle X \longrightarrow Y\ Z \rangle$ (need not be binary)
- A string is considered part of the language if it reduces to an S symbol spanning the whole sentence
- Each production can be written as a branch in a tree:



- Can find all parses efficiently with CKY or Earley

Probabilistic CFGs

- how do we handle ambiguous sentences?
  $\Longrightarrow$ **probabilities!**
- how can we assign probabilities to parse trees?
- assign probabilities to individual production rules:
  $P(N_i \longrightarrow \zeta_j | N_i)$ abbrev. to $P(N_i \longrightarrow \zeta_j)$

PCFG Assumptions

- **Place Invariance** $P(N_i \longrightarrow \zeta_j)$ is the same no matter where in the sentence the rule is being applied
- **Context Freeness** P does not depend on words dominated by the subtree
- **Ancestor Freeness** P does not depend on parent/ancestor nodes in tree
- given a sentence S and a tree T:

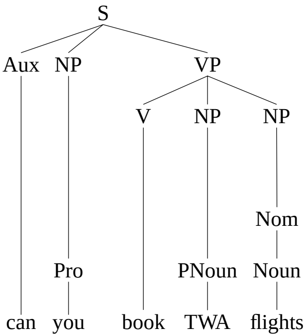
$$P(T, S) = \prod_{N_i \longrightarrow \zeta_j \in T} P(N_i \longrightarrow \zeta_j) \tag{1}$$

An PCFG example grammar

$S \rightarrow NP\ VP$	[.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]	

Jurafsky and Martin, Figure 12.1

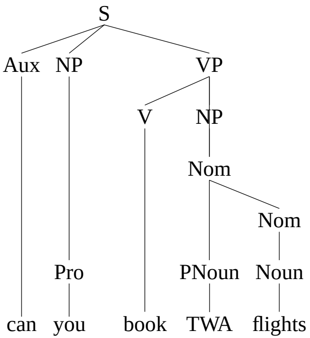
An PCFG example parse



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

Jurafsky and Martin, Figure 12.2(a)

An PCFG example parse



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

Jurafsky and Martin, Figure 12.2(b)

Example probabilities

- 12.2(a)

$$\begin{array}{lll} VP \longrightarrow V\ NP\ NP & 0.05 \\ NP \longrightarrow PNoun & 0.35 \\ NP \longrightarrow Nom & 0.05 \end{array}$$

- $0.05 \times 0.35 \times 0.05 = 0.000875$

- 12.2(b)

$$\begin{array}{lll} VP \longrightarrow V\ NP & 0.40 \\ NP \longrightarrow Nom & 0.05 \\ Nom \longrightarrow PNoun\ Nom & 0.05 \end{array}$$

- $0.40 \times 0.05 \times 0.05 = 0.001$

Training PCFGs

- can be trained like MM and HMM for sequence tagging
- if an annotated corpus is available read counts off
  for relative frequencies (Maximum Likelihood Estimate)
- for unannotated data use *Inside-Outside* algorithm
  which is a tree generalisation of Forward-Backward for HMMs
- apply smoothing

Finding the most probable PCFG parse

- similar to Viterbi algorithm for finding best tag sequence
- but with CKY chart
- computed recursively from bottom up
- initial step $P(N_i \longrightarrow w_j)$
- calculate product of child probabilities at parent position
- choose maximum for each parent category:
  $\forall i, \operatorname{argmax}_j P(N_i \rightarrow w_j)$

- See NLTK chart parser demos

Probabilistic CKY

```
function CYK(words,grammar) returns The most probable parse
                                    and its probability

 Create and clear π[num_words,num_words,num_nonterminals]

 # base case
 for i ← 1 to num_words
    for A ← 1 to num_nonterminals
       if (A → w_i) is in grammar then
          π [i,i,A] ← P(A → w_i)

 # recursive case
 for span ← 2 to num_words
    for begin ← 1 to num_words − span + 1
       end ← begin + span − 1
       for m = begin to end − 1
          for A = 1 to num_nonterminals
          for B = 1 to num_nonterminals
          for C = 1 to num_nonterminals
             prob = π [begin,m,B] × π [m + 1,end,C] × P(A→B C)
             if (prob > π [begin,end,A]) then
                π [begin,end,A] = prob
                back[begin,end,A] = {m,B,C}
 return build_tree(back[1,num_words,1]), π [1,num_words,1]
```

Jurafsky and Martin, Figure 12.3 (corrected)

Problems with Probabilistic PCFGs

- bias towards smaller trees
- bias to include non-terminals with fewer alternative expansions
- PCFGs don't account for word co-occurrences, i.e. plausibility
- in fact, there's lots they don't take into account
- much is addressed by Collins' parser and more recent work:
  - lexicalised production rules with backoff
  - annotated non-terminals
  - discriminative reranking

Comparative parsers

	CFG	PCFG	Dep
Is x a valid sentence?	✓	?	?
Estimate the probability of x		✓	
Identify best parse under ambiguity		✓	✓
Extract constituents (phrases, clauses, etc.)	✓	✓	
Extract useful paths between words	?	?	✓
Parse in under $O(n^2)$			✓

Part II

Annotated PCFGs and Reranking

Undergeneration and overgeneration

- Context-free phrase-structure grammar is a language for writing grammars
- A perfect grammar will not *under*- or *over*-generate
  - …This is really really hard!
- $\langle NP \longrightarrow DT\ JJ \rangle$, e.g. ‘The elderly’

Undergeneration and overgeneration

- Context-free phrase-structure grammar is a language for writing grammars
- A perfect grammar will not *under*- or *over*-generate
  - ...This is really really hard!
- $\langle NP \longrightarrow DT\ JJ \rangle$, e.g. 'The elderly'
  - ...But not *an elderly* :(
- In practice, all grammars are at least a little leaky

Making a grammar less leaky

- Most words have their own usage kinks
- Conflating words into categories invites problems
  - DT for *the*, *a*, *an*, *this*, *that*...
  - Small class, but not all behave the same!
  - e.g. the dogs vs. a dogs
  - e.g. the old vs. an old
- A basic PCFG trained on PTB is underspecified
- E.g. grammars are better if structure is driven by the words

  This is called *lexicalisation*

Lexicalisation: letting words rule

- Phrase-structure grammars assign *strings*, e.g. DT
  - Each string is an arbitrary identifier
  - Meaning of the strings determined by their usage in the grammar
  - $\langle NP \longrightarrow DT\ NN \rangle$ means a DT is 'a class of words that occur before a noun as part of an NP'
  - $\langle DT \longrightarrow the/a/an \rangle$ means a DT is 'a class of words including the, a and an
- Instead of strings, we can assign structured objects
- This helps us to declare more at once, making finer-grained, word-appropriate distinctions

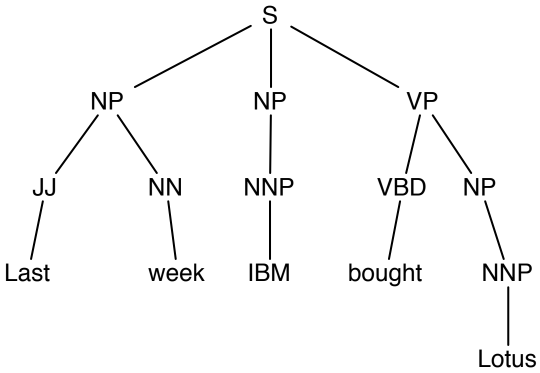
Preposition Phrase (PP) attachment

- I saw the girl on the hill. . .
  - . . . with the binoculars.
  - . . . with the dress.
  - . . . with the church.
- these attachment choices require knowledge of the noun
- I saw the girl on the hill. . .
  - . . . with the telescope.
  - . . . through the telescope.
  - . . . beside the highway.
- these attachment choices require knowledge of the preposition

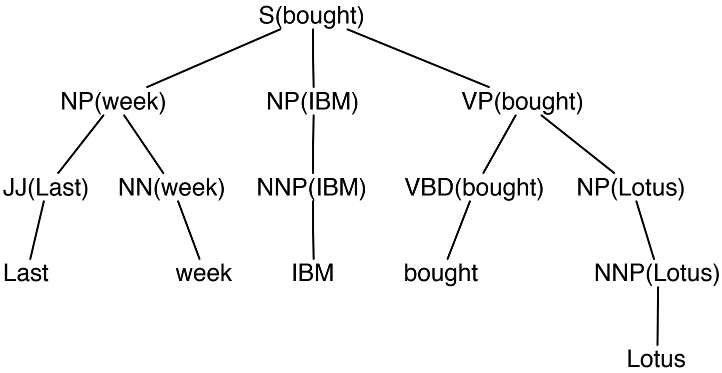
Lexicalised PCFGs

- we need more information than non-terminals:
  - to constrain the grammar
  - to model attachment better
- add information about the words (*lexicalise*)
- which words to choose? **heads**
- add this information to the rules (expand the rule set)

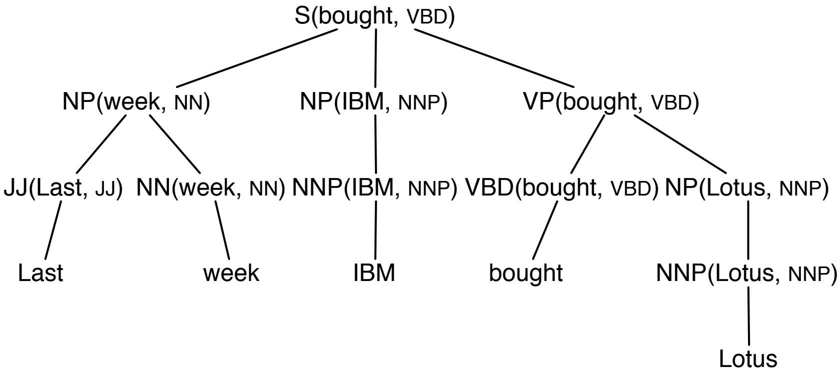
Lexicalised PCFG example



Lexicalised PCFG example



Lexicalised PCFG example



Lexicalised PCFG rules

$$\text{S} \longrightarrow \text{NP NP VP}$$

$$\downarrow$$

$$\text{S(bought)} \longrightarrow \text{NP(week) NP(IBM) VP(bought)}$$

$$\downarrow$$

$$\text{S(bought, } \textsc{vbd}\text{)} \longrightarrow \text{NP(week, } \textsc{nn}\text{) NP(IBM, } \textsc{nnp}\text{) VP(bought, } \textsc{vbd}\text{)}$$

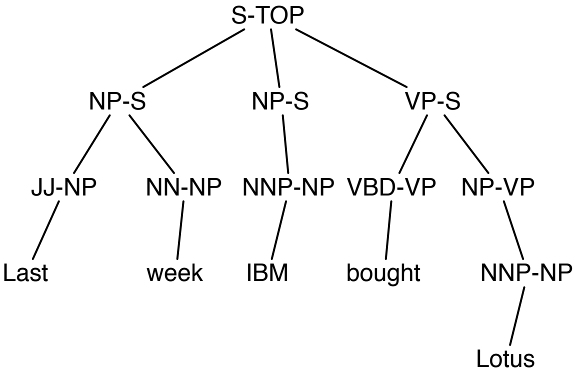
Problems with Lexicalised PCFGs

- massive increase in the number of rules
- potential reduction in coverage
- increased **sparsity** is the biggest problem
  - backoff or interpolation models
  - smoothing
- heads still don't provide all the info we need
  - e.g. preposition + head noun for PPs

Ancestor freeness assumption is wrong

- e.g. choice of NP expansion varies dramatically with parent
- overall NP $\longrightarrow$ NP PP is preferred
- under S, NP $\longrightarrow$ PRP is preferred
- we can annotate the nodes with their parent (Johnson, 1998)
- and/or their grandparent

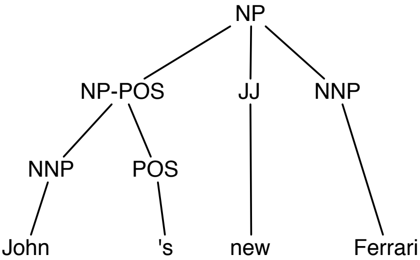
Parent annotation (Johnson, 1998)



Annotation in general

- perhaps the Penn Treebank labels are too coarse
- we can choose to split certain labels (e.g. with parent)
- heading towards feature-based grammars

Possessives Example



Unlexicalised PCFGs (Klein and Manning, 2003)

- add the minimum annotation to reach lexicalised performance
- annotation with open class words not allowed
- lots of small annotations, including annotation of:
  - parents
  - splitting % into a separate POS tag
  - adding -AUX to auxiliary verbs
  - adding Penn Treebank annotations (NP-TMP is child of S)
  - possessive NPs
  - VPs distinguished: finite, infinitive and gerund
  - depth in tree
- or split rules automatically (Petrov and Klein)

Generative and discriminative models

- **joint** or **generative** models
  - model **both** observed instance (i.e. outcome) and classification
  - as if class **generated** instance
  - requires modelling probability of an observation
  - language models, Naïve Bayes, hidden markov models, PCFGs
  - trained using (smoothed) Maximum Likelihood Estimate
- **conditional** or **discriminative** models
  - model classification **given** observed instance
  - may incorporate arbitrary features as evidence
  - Maximum Entropy, SVMs, perceptrons
  - trained using (smoothed) conditional Maximum Likelihood Estimate

Discriminative reranking

- PCFGs are a generative model:
  model the joint probability of a sentence and its parse
- have to model dependencies explicitly

$\therefore$ use a PCFG to generate n best parses ($n = 50?$)

- then re-score them with a *discriminative* model
- training: find feature weights scoring correct parses higher than alternatives
- similar methods are successful in information retrieval

The power of reranking

- features (evidence) derived from sentence and parse
- not constrained to local production decisions
  - features to prefer right-branching trees
  - features to prefer heavy constituents at end
  - features to require subject-verb agreement
- can incorporate multiple formalisms/theories/parsers
  - e.g. compare proposed phrase structure parse to dependency parser output
- features can be expensive to compute relative to initial score
- state-of-the art parsing $\approx 91\%$ F_1 over PTB constituencies

Take away

- A PCFG adds probabilities to each production rule in the grammar
- Defines a language model:
- probability of sentence = product of probability of production rules
- May decide among ambiguous structures
- Chart parsing as for CFGs, like Viterbi for sequences
- Discriminative reranking can help make generative models perform better