Algorithms for NLP

Parsing I

Yulia Tsvetkov – CMU

Slides: Ivan Titov – University of Edinburgh, Taylor Berg-Kirkpatrick – CMU/UCSD, Dan Klein – UC Berkeley
Ambiguity

- I saw a girl with a telescope
The move followed a round of similar increases by other lenders, reflecting a continuing decline in that market.
A Supervised ML Problem

- Data for parsing experiments:
  - Penn WSJ Treebank = 50,000 sentences with associated trees
  - Usual set-up: 40,000 training, 2,400 test

Canadian Utilities had 1988 revenue of $1.16 billion, mainly from its natural gas and electric utility businesses in Alberta, where the company serves about 800,000 customers.
Outline

- Syntax: intro, CFGs, PCFGs
- CFGs: Parsing
- PCFGs: Parsing
- Parsing evaluation
Syntax
Syntax

- The study of the patterns of formation of sentences and phrases from word

- my dog  Pron N
- the dog   Det N
- the cat   Det N
- the large cat Det Adj N
- the black cat Det Adj N
- ate a sausage V Det N
Syntax

- The study of the patterns of formation of sentences and phrases from word
  - Borders with semantics and morphology sometimes blurred

_Afyonkarahisarlılaştırabildiklerimizdenmişsinizcesineee_

in Turkish means

"as if you are one of the people that we thought to be originating from Afyonkarahisar" [wikipedia]
The process of predicting syntactic representations

Syntactic Representations

Different types of syntactic representations are possible, for example:

Constuent (a.k.a. phrase-structure) tree
Constituent trees

- Internal nodes correspond to phrases
  - $S$ – a sentence
  - NP (Noun Phrase): My dog, a sandwich, lakes, ...
  - VP (Verb Phrase): ate a sausage, barked, …
  - PP (Prepositional phrases): with a friend, in a car, …

- Nodes immediately above words are PoS tags (aka preterminals)
  - PN – pronoun
  - D – determiner
  - V – verb
  - N – noun
  - P – preposition
Bracketing notation

- It is often convenient to represent a tree as a bracketed sequence

(S
  (NP (PN My) (N Dog))
  (VP (V ate)
    (NP (D a) (N sausage))
  )
)
The process of predicting syntactic representations

Syntactic Representations

- Different types of syntactic representations are possible, for example:

**Constituent (a.k.a. phrase-structure) tree**

**Dependency tree**
Dependency trees

- Nodes are words (along with PoS tags)
- Directed arcs encode syntactic dependencies between them
- Labels are types of relations between the words
  - poss – possesive
  - dobj – direct object
  - nsub - subject
  - det - determiner
Recovering shallow semantics

- Some semantic information can be (approximately) derived from syntactic information
  - Subjects (nsubj) are (often) agents ("initiator / doers for an action")
  - Direct objects (dobj) are (often) patients ("affected entities")
Recovering shallow semantics

- Some semantic information can be (approximately) derived from syntactic information
  - Subjects (nsubj) are (often) agents ("initiator / doers for an action")
  - Direct objects (dobj) are (often) patients ("affected entities")
- But even for agents and patients consider:
  - Mary is baking a cake in the oven
  - A cake is baking in the oven
- In general it is not trivial even for the most shallow forms of semantics
  - E.g., consider prepositions: *in* can encode direction, position, temporal information, …
Constituent and dependency representations

- Constituent trees can (potentially) be converted to dependency trees

- Dependency trees can (potentially) be converted to constituent trees
Constituent trees

- **Internal nodes correspond to phrases**
  - **S** – a sentence
  - **NP** (Noun Phrase): My dog, a sandwich, lakes,
  - **VP** (Verb Phrase): ate a sausage, barked, …
  - **PP** (Prepositional phrases): with a friend, in a car, …

- **Nodes immediately above words are PoS tags (aka preterminals)**
  - PN – pronoun
  - D – determiner
  - V – verb
  - N – noun
  - P – preposition
Constituency Tests

- How do we know what nodes go in the tree?

- Classic constituency tests:
  - Substitution by *proform*
  - Movement
    - Clefting
    - Preposing
    - Passive
  - Modification
  - Coordination/Conjunction
  - Ellipsis/Deletion
Conflicting Tests

- Constituency isn’t always clear
  - Units of transfer:
    - think about ~ penser à
    - talk about ~ hablar de
  - Phonological reduction:
    - I will go → I’ll go
    - I want to go → I wanna go
    - a le centre → au centre
CFGs
## Context Free Grammar (CFG)

<table>
<thead>
<tr>
<th>Grammar (CFG)</th>
<th>Lexicon</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROOT → S</td>
<td>NN → interest</td>
</tr>
<tr>
<td>S → NP VP</td>
<td>NNS → raises</td>
</tr>
<tr>
<td>NP → DT NN</td>
<td>VBP → interest</td>
</tr>
<tr>
<td>NP → NN NNS</td>
<td>VBZ → raises</td>
</tr>
<tr>
<td></td>
<td>...</td>
</tr>
</tbody>
</table>

- **Other grammar formalisms:** LFG, HPSG, TAG, CCG…
( (S (NP-SBJ The move)
  (VP followed
   (NP (NP a round)
    (PP of
     (NP (NP similar increases)
      (PP by
       (NP other lenders))
     (PP against
      (NP Arizona real estate loans))))))
,
  (S-ADV (NP-SBJ *)
   (VP reflecting
    (NP (NP a continuing decline)
     (PP-LOC in
      (NP that market))))))
.)
 CFGs

\[
S \rightarrow NP \ VF
\]

\[
NP \rightarrow NP \ PF
\]

\[
NP \rightarrow D \ N
\]

\[
NP \rightarrow PN
\]

\[
PP \rightarrow P \ NF
\]

\[
VP \rightarrow V
\]

\[
VP \rightarrow V \ NF
\]

\[
VP \rightarrow VP \ PF
\]

\[
PN \rightarrow I
\]

\[
V \rightarrow saw
\]

\[
V \rightarrow ate
\]

\[
P \rightarrow with
\]

\[
P \rightarrow in
\]

\[
D \rightarrow a
\]

\[
D \rightarrow the
\]

\[
N \rightarrow girl
\]

\[
N \rightarrow telescope
\]

\[
N \rightarrow sandwich
\]
CFGs

\[ S \rightarrow NP \ VF \]
\[ VP \rightarrow V \]
\[ VP \rightarrow V \ NF \]
\[ VP \rightarrow VP \ PF \]
\[ NP \rightarrow NP \ PF \]
\[ NP \rightarrow D \ N \]
\[ PP \rightarrow P \ NF \]
\[ N \rightarrow girl \]
\[ N \rightarrow telescope \]
\[ N \rightarrow sandwich \]
\[ PN \rightarrow I \]
\[ V \rightarrow saw \]
\[ V \rightarrow ate \]
\[ P \rightarrow with \]
\[ P \rightarrow in \]
\[ D \rightarrow a \]
\[ D \rightarrow the \]
CFGs

\[ S \rightarrow NP \ VF \]
\[ NP \rightarrow D\ N \]
\[ NP \rightarrow PN \]
\[ VP \rightarrow V \]
\[ VP \rightarrow V \ NF \]
\[ VP \rightarrow VP \ PF \]
\[ PN \rightarrow I \]
\[ PN \rightarrow I \]
\[ V \rightarrow saw \]
\[ V \rightarrow ate \]
\[ P \rightarrow with \]
\[ P \rightarrow in \]
\[ D \rightarrow a \]
\[ D \rightarrow the \]
CFGs

\[ S \rightarrow NP \ VF \]
\[ VP \rightarrow V \]
\[ VP \rightarrow V \ NP \]
\[ VP \rightarrow VP \ PF \]
\[ NP \rightarrow NP \ PF \]
\[ NP \rightarrow D \ N \]
\[ NP \rightarrow PN \]
\[ PP \rightarrow P \ NF \]

\[ N \rightarrow girl \]
\[ N \rightarrow telescope \]
\[ N \rightarrow sandwich \]
\[ PN \rightarrow I \]
\[ V \rightarrow saw \]
\[ V \rightarrow ate \]
\[ P \rightarrow with \]
\[ P \rightarrow in \]
\[ D \rightarrow a \]
\[ D \rightarrow the \]
CFGs

\[ S \rightarrow NP \ VF \]
\[ VP \rightarrow V \]
\[ VP \rightarrow V \ NF \]
\[ VP \rightarrow VP \ PF \]
\[ NP \rightarrow NP \ PF \]
\[ NP \rightarrow D \ N \]
\[ NP \rightarrow PN \]
\[ PP \rightarrow P \ NF \]

\[ N \rightarrow girl \]
\[ N \rightarrow telescope \]
\[ N \rightarrow sandwich \]
\[ PN \rightarrow I \]
\[ V \rightarrow saw \]
\[ V \rightarrow ate \]
\[ P \rightarrow with \]
[\[ P \rightarrow in \]
\[ D \rightarrow a \]
\[ D \rightarrow the \]
CFGs

\[
S \rightarrow NP \ VF \\
N \rightarrow girl \\
N \rightarrow telescope \\
VP \rightarrow V \\
N \rightarrow sandwich \\
VP \rightarrow V \ NF \\
PN \rightarrow I \\
VP \rightarrow VP \ PF \\
V \rightarrow saw \\
P \rightarrow with \\
NP \rightarrow NP \ PF \\
P \rightarrow in \\
PN \rightarrow D \ N \\
D \rightarrow a \\
NP \rightarrow PN \\
P \rightarrow the \\
PP \rightarrow P \ NP \\
\]

\[
S \rightarrow NP \ VF \\
NP \rightarrow PN \\
I \rightarrow saw \\
PN \rightarrow I \\
VP \rightarrow VP \\
P \rightarrow with \\
PP \rightarrow P \\
NP \rightarrow PN \\
PP \rightarrow P \ NF \\
\]

Diagram of a CFG: 

- **S**: The start symbol, which can be replaced by NP followed by VF.
- **NP**: Can be replaced by PN followed by NF or by D followed by N.
- **VP**: Can be replaced by a verb (V) or another VP.
- **PP**: Can be replaced by a preposition (P) followed by NF.
- **I**: Represents the verb "saw".
- **N**: Represents a noun, with rules for specific nouns like "girl" and "sandwich".
- **D**: Represents a determiner, with rules for specific determiners like "a" and "the".

The diagram shows the hierarchical structure of phrases and their transformations based on the rules of the CFG.
CFGs

\[ S \rightarrow NP \ VP \]

\[ V \rightarrow \text{eat} \]

\[ V \rightarrow \text{ate} \]

\[ N \rightarrow \text{girl} \]

\[ N \rightarrow \text{telescope} \]

\[ N \rightarrow \text{sandwich} \]

\[ PN \rightarrow I \]

\[ V \rightarrow \text{saw} \]

\[ V \rightarrow \text{ate} \]

\[ P \rightarrow \text{with} \]

\[ P \rightarrow \text{in} \]

\[ D \rightarrow a \]

\[ D \rightarrow the \]
CFGs

\[ S \rightarrow NP \ VF \]
\[ VP \rightarrow V \]
\[ VP \rightarrow VP \ NP \]
\[ VP \rightarrow V \ NP \]
\[ VP \rightarrow VP \ PF \]
\[ NP \rightarrow NP \ PF \]
\[ NP \rightarrow D \ N \]
\[ NP \rightarrow PN \]
\[ PP \rightarrow P \ NP \]
\[ PP \rightarrow P \ NF \]

\[ N \rightarrow girl \]
\[ N \rightarrow telescope \]
\[ N \rightarrow sandwich \]
\[ PN \rightarrow I \]
\[ V \rightarrow saw \]
\[ V \rightarrow ate \]
\[ P \rightarrow with \]
\[ P \rightarrow in \]
\[ D \rightarrow a \]
\[ D \rightarrow the \]
A context-free grammar is a 4-tuple $<N, T, S, R>$

- $N$ : the set of non-terminals
  - Phrasal categories: S, NP, VP, ADJP, etc.
  - Parts-of-speech (pre-terminals): NN, JJ, DT, VB

- $T$ : the set of terminals (the words)

- $S$ : the start symbol
  - Often written as ROOT or TOP
  - *Not* usually the sentence non-terminal $S$

- $R$ : the set of rules
  - Of the form $X \rightarrow Y_1 Y_2 \ldots Y_k$, with $X, Y_i \in N$
  - Examples: $S \rightarrow NP \ VP$, $VP \rightarrow VP \ CC \ VP$
  - Also called rewrites, productions, or local trees
An example grammar

\[ N = \{ S, VP, NP, PP, N, V, PN, P \} \]
\[ T = \{ girl, telescope, sandwich, I, saw, ate, with, in, a, the \} \]
\[ S = \{ S \} \]
\[ R \]

\[ S \rightarrow NP \quad VP \]
\[ VP \rightarrow V \]
\[ VP \rightarrow V \quad NP \]
\[ VP \rightarrow VP \quad PF \]
\[ NP \rightarrow NP \quad PF \]
\[ NP \rightarrow D \quad N \]
\[ NP \rightarrow PN \]
\[ PP \rightarrow P \quad NF \]

Called **Inner rules**

Preterminal rules

\[ N \rightarrow girl \]
\[ N \rightarrow telescope \]
\[ N \rightarrow sandwich \]
\[ PN \rightarrow I \]
\[ V \rightarrow saw \]
\[ V \rightarrow ate \]
\[ P \rightarrow with \]
\[ P \rightarrow in \]
\[ D \rightarrow a \]
\[ D \rightarrow the \]
Why context-free?

What can be a sub-tree is only affected by what the phrase type is (VP) but not the context.
Why context-free?

What can be a sub-tree is only affected by what the phrase type is (VP) but not the context.

Not grammatical.
Here, the coarse VP and NP categories cannot enforce subject-verb agreement in number resulting in the coordination ambiguity.

"Bark" can refer both to a noun or a verb.

This tree would be ruled out if the context would be somehow captured (subject-verb agreement).
Ambiguities
Why parsing is hard?  Ambiguity

- Prepositional phrase attachment ambiguity
Put the block in the box on the table in the kitchen

3 prepositional phrases, 5 interpretations:

- Put the block ((in the box on the table) in the kitchen)
- Put the block (in the box (on the table in the kitchen))
- Put ((the block in the box) on the table) in the kitchen.
- Put (the block (in the box on the table)) in the kitchen.
- Put (the block in the box) (on the table in the kitchen)
Put the block in the box on the table in the kitchen

- 3 prepositional phrases, 5 interpretations:
  - Put the block ((in the box on the table) in the kitchen)
  - Put the block (in the box (on the table in the kitchen))
  - Put ((the block in the box) on the table) in the kitchen.
  - Put (the block (in the box on the table)) in the kitchen.
  - Put (the block in the box) (on the table in the kitchen)

- A general case:

\[
C_{at_n} = \binom{2n}{n} - \binom{2n}{n-1} \sim \frac{4^n}{n^{3/2}\sqrt{\pi}}
\]

1, 2, 5, 14, 42, 132, 429, 1430, 4862, 16796, 58786, …
Canadian Utilities had 1988 revenue of $1.16 billion, mainly from its natural gas and electric utility businesses in Alberta, where the company serves about 800,000 customers.
Syntactic Ambiguities I

- Prepositional phrases:
  They cooked the beans in the pot on the stove with handles.

- Particle vs. preposition:
  The puppy tore up the staircase.

- Complement structures
  The tourists objected to the guide that they couldn’t hear.
  She knows you like the back of her hand.

- Gerund vs. participial adjective
  Visiting relatives can be boring.
  Changing schedules frequently confused passengers.
Syntactic Ambiguities II

- Modifier scope within NPs
  impractical design requirements
  plastic cup holder

- Multiple gap constructions
  The chicken is ready to eat.
  The contractors are rich enough to sue.

- Coordination scope:
  Small rats and mice can squeeze into holes or cracks in the wall.
- **Dark ambiguities**: most analyses are shockingly bad (meaning, they don’t have an interpretation you can get your mind around)

  This analysis corresponds to the correct parse of
  
  “This is panic buying!”

- **Unknown words and new usages**
- **Solution**: We need mechanisms to focus attention on the best ones, probabilistic techniques do this
How to Deal with Ambiguity?

- We want to score all the derivations to encode how plausible they are.

*Put the block in the box on the table in the kitchen*
PCFGs
A context-free grammar is a tuple <N, T, S, R>

- N : the set of non-terminals
  - Phrasal categories: S, NP, VP, ADJP, etc.
  - Parts-of-speech (pre-terminals): NN, JJ, DT, VB
- T : the set of terminals (the words)
- S : the start symbol
  - Often written as ROOT or TOP
  - Not usually the sentence non-terminal S
- R : the set of rules
  - Of the form $X \rightarrow Y_1 Y_2 \ldots Y_k$, with $X, Y_i \in N$
  - Examples: $S \rightarrow NP \; VP$, $VP \rightarrow VP \; CC \; VP$
  - Also called rewrites, productions, or local trees

A PCFG adds:
- A top-down production probability per rule $P(Y_1 Y_2 \ldots Y_k \mid X)$
PCFGs

Associate probabilities with the rules:

\[ p(X \rightarrow \alpha) \]

\[ \forall X \rightarrow \alpha \in R : \quad 0 \leq p(X \rightarrow \alpha) \leq 1 \]

\[ \forall X \in N : \quad \sum_{\alpha : X \rightarrow \alpha \in R} p(X \rightarrow \alpha) = 1 \]

\[ S \rightarrow NP \ VF \quad 1.0 \quad (NP \ A \ girl) \ (VP \ ate \ a \ sandwich) \]

\[ VP \rightarrow V \quad 0.2 \]

\[ VP \rightarrow V \ NF \quad 0.4 \quad (VP \ ate) \ (NP \ a \ sandwich) \]

\[ VP \rightarrow VP \ PF \quad 0.4 \quad (VP \ saw \ a \ girl) \ (PP \ with \ …) \]

\[ NP \rightarrow NP \ PF \quad 0.3 \quad (NP \ a \ girl) \ (PP \ with \ ….) \]

\[ NP \rightarrow D \ N \quad 0.5 \quad (D \ a) \ (N \ sandwich) \]

\[ NP \rightarrow PN \quad 0.2 \]

\[ PP \rightarrow P \ NF \quad 1.0 \quad (P \ with) \ (NP \ with \ a \ sandwich) \]

Now we can score a tree as a product of probabilities corresponding to the used rules.

\[ \hat{N} \rightarrow girl \quad 0.2 \]

\[ N \rightarrow telescope \quad 0.7 \]

\[ N \rightarrow sandwich \quad 0.1 \]

\[ PN \rightarrow I \quad 1.0 \]

\[ V \rightarrow saw \quad 0.5 \]

\[ V \rightarrow ate \quad 0.5 \]

\[ P \rightarrow with \quad 0.6 \]

\[ P \rightarrow in \quad 0.4 \]

\[ D \rightarrow a \quad 0.3 \]

\[ D \rightarrow the \quad 0.7 \]
PCFGs

\[
S \rightarrow NP \ VP \ 1.0 \\
VP \rightarrow V \ 0.2 \\
NP \rightarrow NP \ PF \ 0.3 \\
NP \rightarrow D \ N \ 0.5 \\
NP \rightarrow PN \ 0.2 \\
PP \rightarrow P \ NF \ 1.0
\]

\[
N \rightarrow girl \ 0.2 \\
N \rightarrow telescope \ 0.7 \\
N \rightarrow sandwich \ 0.1 \\
PN \rightarrow I \ 1.0 \\
V \rightarrow saw \ 0.5 \\
V \rightarrow ate \ 0.5 \\
P \rightarrow with \ 0.6 \\
P \rightarrow in \ 0.4 \\
D \rightarrow a \ 0.3 \\
D \rightarrow the \ 0.7
\]

\[p(T) = \]
PCFGs

\[
p(T) = 1.0 \times
\]

\[
S \rightarrow NP \ VF 1.0
\]

\[
VP \rightarrow V 0.2
\]

\[
VP \rightarrow V \ NP 0.4
\]

\[
VP \rightarrow VP \ PF 0.4
\]

\[
NP \rightarrow NP \ PF 0.3
\]

\[
NP \rightarrow D \ N 0.5
\]

\[
NP \rightarrow PN 0.2
\]

\[
PP \rightarrow P \ NF 1.0
\]

\[
N \rightarrow girl 0.2
\]

\[
N \rightarrow telescope 0.7
\]

\[
N \rightarrow sandwich 0.1
\]

\[
PN \rightarrow I 1.0
\]

\[
V \rightarrow saw 0.5
\]

\[
V \rightarrow ate 0.5
\]

\[
P \rightarrow with 0.6
\]

\[
P \rightarrow in 0.4
\]

\[
D \rightarrow a 0.3
\]

\[
D \rightarrow the 0.7
\]
PCFGs

\[ S \rightarrow NP \ VF \ 1.0 \]
\[ VP \rightarrow V \ 0.2 \]
\[ VP \rightarrow V \ NF \ 0.4 \]
\[ VP \rightarrow VP \ PF \ 0.4 \]
\[ NP \rightarrow NP \ PF \ 0.3 \]
\[ NP \rightarrow D \ N \ 0.5 \]
\[ NP \rightarrow PN \ 0.2 \]
\[ PP \rightarrow P \ NF \ 1.0 \]
\[ N \rightarrow girl \ 0.2 \]
\[ N \rightarrow telescope \ 0.7 \]
\[ N \rightarrow sandwich \ 0.1 \]
\[ PN \rightarrow I \ 1.0 \]
\[ V \rightarrow saw \ 0.5 \]
\[ V \rightarrow ate \ 0.5 \]
\[ P \rightarrow with \ 0.6 \]
\[ P \rightarrow in \ 0.4 \]
\[ D \rightarrow a \ 0.3 \]
\[ D \rightarrow the \ 0.7 \]

\[ p(T) = 1.0 \times 0.2 \times \]
PCFGs

\[ S \rightarrow NP \ VF \ 1.0 \]
\[ VP \rightarrow V \ 0.2 \]
\[ VP \rightarrow V \ NF \ 0.4 \]
\[ VP \rightarrow VP \ PF \ 0.4 \]
\[ NP \rightarrow NP \ PF \ 0.3 \]
\[ NP \rightarrow D \ N \ 0.5 \]
\[ NP \rightarrow PN \ 0.2 \]
\[ PP \rightarrow P \ NF \ 1.0 \]
\[ N \rightarrow girl \ 0.2 \]
\[ N \rightarrow telescope \ 0.7 \]
\[ N \rightarrow sandwich \ 0.1 \]
\[ PN \rightarrow I \ 1.0 \]
\[ V \rightarrow saw \ 0.5 \]
\[ V \rightarrow ate \ 0.5 \]
\[ P \rightarrow with \ 0.6 \]
\[ P \rightarrow in \ 0.4 \]
\[ D \rightarrow a \ 0.3 \]
\[ D \rightarrow the \ 0.7 \]

\[ p(T) = 1.0 \times 0.2 \times 1.0 \times \]
PCFGs

$S \to NP \ VF\ 1.0$

$VP \to V\ 0.2$

$VP \to V \ NF\ 0.4$

$VP \to VP \ PF\ 0.4$

$NP \to NP \ PF\ 0.3$

$NP \to D\ N\ 0.5$

$NP \to PN\ 0.2$

$PP \to P \ NF\ 1.0$

$N \to girl\ 0.2$

$N \to telescope\ 0.7$

$N \to sandwich\ 0.1$

$PN \to I\ 1.0$

$V \to saw\ 0.5$

$V \to ate\ 0.5$

$P \to with\ 0.6$

$P \to in\ 0.4$

$D \to a\ 0.3$

$D \to the\ 0.7$

$p(T) = 1.0 \times 0.2 \times 1.0 \times 0.4 \times$
PCFGs

\[ p(T) = 1.0 \times 0.2 \times 1.0 \times 0.4 \times 0.5 \times \]
PCFGs

\[ p(T) = 1.0 \times 0.2 \times 1.0 \times 0.4 \times 0.5 \times 0.3 \times \]

\[
S \rightarrow NP \ VF \ 1.0
\]

\[
N \rightarrow \text{girl} \ 0.2
\]

\[
N \rightarrow \text{telescope} \ 0.7
\]

\[
V \rightarrow \text{ate} \ 0.5
\]

\[
P \rightarrow \text{with} \ 0.6
\]

\[
P \rightarrow \text{in} \ 0.4
\]

\[
P \rightarrow \text{the} \ 0.7
\]
PCFGs

\[
p(T) = 1.0 \times 0.2 \times 1.0 \times 0.4 \times 0.5 \times 0.3 \times 0.5 \times 0.3 \times 0.2 \times 1.0 \times 0.6 \times 0.5 \times 0.3 \times 0.7 = 2.26 \times 10^{-5}
\]
PCFG Estimation
ML estimation

- A treebank: a collection sentences annotated with constituent trees

- An estimated probability of a rule (maximum likelihood estimates)

\[ p(X \rightarrow \alpha) = \frac{C(X \rightarrow \alpha)}{C(X)} \]

  - The number of times the rule used in the corpus
  - The number of times the nonterminal X appears in the treebank

- Smoothing is helpful
  - Especially important for preterminal rules
We defined a distribution over production rules for each nonterminal. Our goal was to define a distribution over parse trees. Unfortunately, not all PCFGs give rise to a proper distribution over trees, i.e. the sum over probabilities of all trees the grammar can generate may be less than 1: \( \sum_T P(T) < 1 \). Good news: any PCFG estimated with the maximum likelihood procedure are always proper (Chi and Geman, 98).
Penn Treebank: peculiarities

- Wall street journal: around 40,000 annotated sentences, 1,000,000 words
  - Fine-grained part of speech tags (45), e.g., for verbs
    - VBD: Verb, past tense
    - VBG: Verb, gerund or present participle
    - VBP: Verb, present (non-3\textsuperscript{rd} person singular)
    - VBZ: Verb, present (3\textsuperscript{rd} person singular)
    - MD: Modal

- Flat NPs (no attempt to disambiguate NP attachment)
CKY Parsing
Parsing

- Parsing is search through the space of all possible parses
  - e.g., we may want either any parse, all parses or the highest scoring parse (if PCFG):
    \[
    \text{arg max } P(T) \\
    T \in G(x)
    \]

- Bottom-up:
  - One starts from words and attempt to construct the full tree

- Top-down
  - Start from the start symbol and attempt to expand to get the sentence
CKY algorithm (aka CYK)

- Cocke-Kasami-Younger algorithm
  - Independently discovered in late 60s / early 70s

- An efficient bottom up parsing algorithm for (P)CFGs
  - can be used both for the recognition and parsing problems
  - Very important in NLP (and beyond)

- We will start with the non-probabilistic version
Constraints on the grammar

- The basic CKY algorithm supports only rules in the Chomsky Normal Form (CNF):

  \[ C \rightarrow x \]

  \[ C \rightarrow C_1 C_2 \]

  - **Unary preterminal** rules (generation of words given PoS tags):
    
    \[ N \rightarrow \text{telescope} \quad D \rightarrow \text{the} \]

  - **Binary inner** rules:
    
    \[ S \rightarrow NP VF \quad NP \rightarrow D \quad N \]
Constraints on the grammar

- The basic CKY algorithm supports only rules in the Chomsky Normal Form (CNF):

  \[ C \rightarrow x \]

  \[ C \rightarrow C_1 C_2 \]

- Any CFG can be converted to an equivalent CNF
  - Equivalent means that they define the same language
  - However (syntactic) trees will look differently
  - It is possible to address it by defining such transformations that allows for easy reverse transformation
Transformation to CNF form

What one need to do to convert to CNF form

- Get rid of unary rules: $C \rightarrow C_1$
- Get rid of N-ary rules: $C \rightarrow C_1 C_2 \ldots C_n \ (n > 2)$

Not a problem, as our CKY algorithm will support unary rules

Crucial to process them, as required for efficient parsing
Consider \[ NP \rightarrow DT \ NNP \ VBG \ NN \]

- How do we get a set of binary rules which are equivalent?
Consider

\[ NP \rightarrow DT \ NNP \ VBG \ NN \]

\[
\text{NP}
\]
\[
\text{DT} \quad \text{NNP} \quad \text{VBG} \quad \text{NN}
\]
\[
\begin{align*}
\text{the} & \quad \text{Dutch} & \quad \text{publishing} & \quad \text{group}
\end{align*}
\]

How do we get a set of binary rules which are equivalent?

\[ NP \rightarrow DT \ X \]
\[ X \rightarrow NNP \ Y \]
\[ Y \rightarrow VBG \ NN \]
Transformation to CNF form: binarization

- Consider

\[ NP \rightarrow DT \ NNP \ VBG \ NN \]

\[
\text{NP} \\
\downarrow \\
\text{DT} \quad \text{NNP} \quad \text{VBG} \quad \text{NN} \\
\downarrow \quad \downarrow \quad \downarrow \quad \downarrow \\
\text{the} \quad \text{Dutch} \quad \text{publishing} \quad \text{group}
\]

- How do we get a set of binary rules which are equivalent?

\[
NP \rightarrow DT \ X \\
X \rightarrow NNP \ Y \\
Y \rightarrow VBG \ NN
\]

- A more systematic way to refer to new non-terminals

\[
NP \rightarrow DT \ @NP|DT \\
@NP|DT \rightarrow NNP \ @NP|DT.NNP \\
@NP|DT.NNP \rightarrow VBG \ NN
\]
Transformation to CNF form: binarization

- Instead of binarizing tuples we can binarize trees on preprocessing:

```
NP
  |   
  v   
DT   NNP  VBG  NN
  the  Dutch  publishing  group
```

Also known as lossless Markovization in the context of PCFGs

```
NP
  |   
  v   
DT  @NP->_DT
  the

NP
  |   
  v   
DT  @NP->_DT_NNP
  the

NP
  |   
  v   
NNP  @NP->_DT_NNP
  Dutch

NP
  |   
  v   
VBG  NN
  publishing  group
```

Can be easily reversed on postprocessing
 CKY: Parsing task

- We are given
  - a grammar $<N, T, S, R>$
  - a sequence of words $w = (w_1, w_2, \ldots, w_n)$

- Our goal is to produce a parse tree for $w$
CKY: Parsing task

- We are given
  - a grammar \( <N, T, S, R> \)
  - a sequence of words \( w = (w_1, w_2, \ldots, w_n) \)
- Our goal is to produce a parse tree for \( w \)
- We need an easy way to refer to substrings of \( w \)

\( \text{span } (i, j) \) refers to words between fenceposts \( i \) and \( j \)
Parsing one word

\[ C \rightarrow w_i \]
Parsing one word

\[ C \rightarrow w_i \]
Parsing one word

$C \rightarrow w_i$

covers all words
between $i - 1$ and $i$
Parsing longer spans

$C \rightarrow C_1 \ C_2$

Check through all C1, C2, mid

covers all words btw min and mid  covers all words btw mid and max
Parsing longer spans

\[ C \rightarrow C_1 \quad C_2 \]

Check through all C1, C2, mid

covers all words btw min and mid

covers all words btw mid and max
Parsing longer spans

covers all words between $\text{min}$ and $\text{max}$
## CKY in action

<table>
<thead>
<tr>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

### Preterminal rules

- **S → NP VP**

### Inner rules

- **VP → M V**
- **VP → V**
- **NP → N**
- **NP → N NP**

### Terminal rules

- **N → can**
- **N → lead**
- **N → poison**
- **M → can**
- **M → must**
- **V → poison**
- **V → lead**
Preterminal rules

<table>
<thead>
<tr>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

max = 1 max = 2 max = 3

min = 0

min = 1

min = 2

Chart (aka parsing triangle)

Inner rules

\[ S \rightarrow NP \; VP \]

\[ VP \rightarrow M \; V \]

\[ VP \rightarrow V \]

\[ NP \rightarrow N \]

\[ NP \rightarrow N \; NP \]

Preterminal rules

\[ N \rightarrow can \]

\[ N \rightarrow lead \]

\[ N \rightarrow poison \]

\[ M \rightarrow can \]

\[ M \rightarrow must \]

\[ V \rightarrow poison \]

\[ V \rightarrow lead \]
Preterminal rules

<table>
<thead>
<tr>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

Inner rules

\[ S \rightarrow NP \ VP \]

\[ VP \rightarrow M \ V \]

\[ NP \rightarrow N \]

\[ NP \rightarrow N \ NP \]

\[ N \rightarrow can \]

\[ N \rightarrow lead \]

\[ N \rightarrow poison \]

\[ M \rightarrow can \]

\[ M \rightarrow must \]

\[ V \rightarrow poison \]

\[ V \rightarrow lead \]
$S \rightarrow NP \ VP$

$VP \rightarrow M \ V$

$NP \rightarrow N$

$NP \rightarrow N \ NP$

$N \rightarrow can$

$N \rightarrow lead$

$N \rightarrow poison$

$M \rightarrow can$

$M \rightarrow must$

$V \rightarrow poison$

$V \rightarrow lead$
\[
S \rightarrow NP \ VP \\
VP \rightarrow M \ V \\
NP \rightarrow N \\
NP \rightarrow N \ NP \\
N \rightarrow can \\
N \rightarrow lead \\
N \rightarrow poison \\
M \rightarrow can \\
M \rightarrow must \\
V \rightarrow poison \\
V \rightarrow lead
\]
<table>
<thead>
<tr>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

Preterminal rules

Inner rules

\[
S \rightarrow NP \ VP
\]

\[
VP \rightarrow M \ V
\]

\[
VP \rightarrow V
\]

\[
NP \rightarrow N
\]

\[
NP \rightarrow N \ NP
\]

\[
N \rightarrow can
\]

\[
N \rightarrow lead
\]

\[
N \rightarrow poison
\]

\[
M \rightarrow can
\]

\[
M \rightarrow must
\]

\[
V \rightarrow poison
\]

\[
V \rightarrow lead
\]
Preterminal rules

<table>
<thead>
<tr>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

max = 1  max = 2  max = 3

min = 0

min = 1

min = 2

S

Inner rules

Preterminal rules

$S \rightarrow NP \ VP$

$VP \rightarrow M \ V$

$VP \rightarrow V$

$NP \rightarrow N$

$NP \rightarrow N \ NP$

$N \rightarrow can$

$N \rightarrow lead$

$N \rightarrow poison$

$M \rightarrow can$

$M \rightarrow must$

$V \rightarrow poison$

$V \rightarrow lead$
Preterminal rules

<table>
<thead>
<tr>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

Inner rules

\[
S \rightarrow NP \ VP
\]

\[
VP \rightarrow M \ V
\]

\[
VP \rightarrow V
\]

\[
NP \rightarrow N
\]

\[
NP \rightarrow N \ NP
\]

Preterminal rules

\[
N \rightarrow can
\]

\[
N \rightarrow lead
\]

\[
N \rightarrow poison
\]

\[
M \rightarrow can
\]

\[
M \rightarrow must
\]

\[
V \rightarrow poison
\]

\[
V \rightarrow lead
\]
<table>
<thead>
<tr>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

Preterminal rules

\[ S \rightarrow NP \ VP \]

\[ VP \rightarrow M \ V \]
\[ VP \rightarrow V \]

\[ NP \rightarrow N \]
\[ NP \rightarrow N \ NP \]

Inner rules

\[ N \rightarrow can \]
\[ N \rightarrow lead \]
\[ N \rightarrow poison \]

Preterminal rules

\[ M \rightarrow can \]
\[ M \rightarrow must \]

\[ V \rightarrow poison \]
\[ V \rightarrow lead \]
Preterminal rules

\[ S \rightarrow NP \ VP \]

Inner rules

\[ VP \rightarrow M \ V \]
\[ VP \rightarrow V \]

\[ NP \rightarrow N \]
\[ NP \rightarrow N \ NP \]

Preterminal rules

\[ N \rightarrow can \]
\[ N \rightarrow lead \]
\[ N \rightarrow poison \]

\[ M \rightarrow can \]
\[ M \rightarrow must \]

\[ V \rightarrow poison \]
\[ V \rightarrow lead \]
Preterminal rules

<table>
<thead>
<tr>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

max = 1    max = 2    max = 3

min = 0

1 \( N, V \)

min = 1

2 \( N, M \)

min = 2

3 \( N, V \)

Inner rules

\[
S \rightarrow NP \ VP
\]

\[
VP \rightarrow M \ V
\]

\[
VP \rightarrow V
\]

\[
NP \rightarrow N
\]

\[
NP \rightarrow N \ NP
\]

Preterminal rules

\[
N \rightarrow can
\]

\[
N \rightarrow lead
\]

\[
N \rightarrow poison
\]

\[
M \rightarrow can
\]

\[
M \rightarrow must
\]

\[
V \rightarrow poison
\]

\[
V \rightarrow lead
\]
Check about unary rules
Preterminal rules

<table>
<thead>
<tr>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
</tr>
</tbody>
</table>

max = 1  max = 2  max = 3

min = 0  min = 1  min = 2

1. $N, V$
   $NP, VP$

2. $N, M$
   $NP$

3. $N, V$
   $NP, VP$

4. $?$

Inner rules

$S \rightarrow NP \ VP$

$VP \rightarrow M \ V$

$VP \rightarrow V$

$NP \rightarrow N$

$NP \rightarrow N \ NP$

Preterminal rules

$N \rightarrow can$

$N \rightarrow lead$

$N \rightarrow poison$

$M \rightarrow can$

$M \rightarrow must$

$V \rightarrow poison$

$V \rightarrow lead$
**Preterminal rules**

<table>
<thead>
<tr>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>(min = 0)</th>
<th>(min = 1)</th>
<th>(min = 2)</th>
<th>max = 1</th>
<th>max = 2</th>
<th>max = 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N, V$</td>
<td>$N, M$</td>
<td>$N, V$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$NP, VP$</td>
<td>$NP$</td>
<td>$NP, VP$</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

$S \rightarrow NP \ VP$

$VP \rightarrow M \ V$

$VP \rightarrow V$

$NP \rightarrow N$

$NP \rightarrow N \ NP$

$N \rightarrow can$

$N \rightarrow lead$

$N \rightarrow poison$

$M \rightarrow can$

$M \rightarrow must$

$V \rightarrow poison$

$V \rightarrow lead$
\[
S \rightarrow NP \ VP
\]

\[
VP \rightarrow M \ V
\]

\[
VP \rightarrow V
\]

\[
NP \rightarrow N
\]

\[
NP \rightarrow N \ NP
\]

\[
N \rightarrow can
\]

\[
N \rightarrow lead
\]

\[
N \rightarrow poison
\]

\[
M \rightarrow can
\]

\[
M \rightarrow must
\]

\[
V \rightarrow poison
\]

\[
V \rightarrow lead
\]
Check about unary rules: no unary rules here
## Preterminal rules

<table>
<thead>
<tr>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>min = 0</th>
<th>min = 1</th>
<th>min = 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>

- **1**: $N, V$
- **2**: $N, M$
- **3**: $N, V$
- **4**: $NP$
- **5**: ?

### max = 1
- $N, V, NP, VP$

### max = 2
- $N, M$

### max = 3
- $NP$

## Inner rules

- $S \rightarrow NP \ VP$
- $VP \rightarrow M \ V$
- $VP \rightarrow V$
- $NP \rightarrow N$
- $NP \rightarrow N \ NP$

## Preterminal rules

- $N \rightarrow can$
- $N \rightarrow lead$
- $N \rightarrow poison$
- $M \rightarrow can$
- $M \rightarrow must$
- $V \rightarrow poison$
- $V \rightarrow lead$
Preterminal rules

<table>
<thead>
<tr>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

max = 1     max = 2     max = 3

min = 0

1. $N, V$
   $NP, VP$

min = 1

2. $N, M$
   $NP$

min = 2

3. $N, V$
   $NP, VP$

4. $NP$

5. $S, VP, NP$

Inner rules

$S \rightarrow NP \ VP$

$VP \rightarrow M \ V$
$VP \rightarrow V$

$NP \rightarrow N$
$NP \rightarrow N \ NP$

Preterminal rules

$N \rightarrow can$
$N \rightarrow lead$
$N \rightarrow poison$

$M \rightarrow can$
$M \rightarrow must$

$V \rightarrow poison$
$V \rightarrow lead$
CKY in action

<table>
<thead>
<tr>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

Preterminal rules

- $S \rightarrow NP \ VP$
- $VP \rightarrow M \ V$
- $VP \rightarrow V$

Inner rules

- $NP \rightarrow N$
- $NP \rightarrow N \ NP$
- $N \rightarrow can$
- $N \rightarrow lead$
- $N \rightarrow poison$
- $M \rightarrow can$
- $M \rightarrow must$
- $V \rightarrow poison$
- $V \rightarrow lead$

Check about unary rules: no unary rules here
<table>
<thead>
<tr>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

max = 1
max = 2
max = 3

min = 0
1. $N, V$
   $NP, VP$

min = 1
2. $N, M$
   $NP$

3. $N, V$
   $NP, VP$

4. $NP$

5. $S, VP, NP$

6. ?

$S \rightarrow NP \ VP$

$VP \rightarrow M \ V$

$VP \rightarrow V$

$NP \rightarrow N$

$NP \rightarrow N \ NP$

$N \rightarrow can$

$N \rightarrow lead$

$N \rightarrow poison$

$M \rightarrow can$

$M \rightarrow must$

$V \rightarrow poison$

$V \rightarrow lead$
Preterminal rules

Inner rules

S → NP VP

VP → M V

NP → N

NP → N NP

N → can

N → lead

N → poison

M → can

M → must

V → poison

V → lead
Preterminal rules

<table>
<thead>
<tr>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

max = 1  max = 2  max = 3

min = 0

min = 1

min = 2

mid = 1

Inner rules

$$S \rightarrow NP \ VP$$

$$VP \rightarrow M \ V$$

$$VP \rightarrow V$$

$$NP \rightarrow N$$

$$NP \rightarrow N \ NP$$

Preterminal rules

$$N \rightarrow can$$

$$N \rightarrow lead$$

$$N \rightarrow poison$$

$$M \rightarrow can$$

$$M \rightarrow must$$

$$V \rightarrow poison$$

$$V \rightarrow lead$$
Preterminal rules

<table>
<thead>
<tr>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

max = 1  max = 2  max = 3

min = 0

min = 1

min = 2

mid = 2

Inner rules

$S \rightarrow NP\ VP$

$VP \rightarrow M\ V$

$VP \rightarrow V$

$NP \rightarrow N$

$NP \rightarrow N\ NP$

Preterminal rules

$N \rightarrow can$

$N \rightarrow lead$

$N \rightarrow poison$

$M \rightarrow can$

$M \rightarrow must$

$V \rightarrow poison$

$V \rightarrow lead$
Apparently the sentence is ambiguous for the grammar: (as the grammar overgenerates)
Ambiguity

No subject-verb agreement, and poison used as an intransitive verb
CKY more formally

Chart can be represented by a Boolean 3D array $chart[\text{min}][\text{max}][\text{C}]$.

- Relevant entries have $0 < \text{min} < \text{max} \leq n$.

$chart[\text{min}][\text{max}][\text{C}] = \text{true}$ if the signature $(\text{min}, \text{max}, \text{C})$ is already added to the chart; $\text{false}$ otherwise.

Here we assume that labels ($\text{C}$) are integer indices.
for each $w_i$ from left to right

   for each preterminal rule $C \rightarrow w_i$

   chart[i - 1][i][C] = true
Implementation: binary rules

for each max from 2 to n
  
  for each min from max - 2 down to 0
    
    for each syntactic category C
      
      for each binary rule C → C₁ C₂
        
        for each mid from min + 1 to max - 1
          
          if chart[min][mid][C₁] and chart[mid][max][C₂] then

          chart[min][max][C] = true
 Unary rules

- How to integrate unary rules $C \rightarrow C_1$?
Unary rules

- How to integrate unary rules $C \rightarrow C_1$?

```plaintext
for each max from 1 to n
  for each min from max - 1 down to 0
    // First, try all binary rules as before.
...
    // Then, try all unary rules.
    for each syntactic category C
      for each unary rule $C \rightarrow C_1$
        if chart[min][max][C_1] then
          chart[min][max][C] = true
```
Unary rules

- How to integrate unary rules $C \rightarrow C_1$?

```plaintext
for each max from 1 to n
  for each min from max - 1 down to 0
    // First, try all binary rules as before.

...  

// Then, try all unary rules.

for each syntactic category C
  for each unary rule C \rightarrow C_1
    if chart[min][max][C_1] then

But we forgot something!
```
Unary closure

- What if the grammar contained 2 rules:
  \[
  A \rightarrow B \\
  B \rightarrow C
  \]

- But C can be derived from A by a chain of rules:
  \[
  A \rightarrow B \rightarrow C
  \]

- One could support chains in the algorithm but it is easier to extend the grammar, to get the **transitive closure**
  \[
  A \rightarrow B \\
  B \rightarrow C \\
  \Rightarrow \\
  A \rightarrow B \\
  B \rightarrow C \\
  \Rightarrow \\
  A \rightarrow C
  \]
Unary closure

- What if the grammar contained 2 rules:
  \[ A \rightarrow B \]
  \[ B \rightarrow C \]

- But C can be derived from A by a chain of rules:
  \[ A \rightarrow B \rightarrow C \]

- One could support chains in the algorithm but it is easier to extend the grammar, to get the transitive closure
  
  \[
  \begin{align*}
  A & \rightarrow B \\
  B & \rightarrow C \\
  \end{align*}
  \Rightarrow
  \begin{align*}
  A & \rightarrow B \\
  B & \rightarrow C \\
  A & \rightarrow C \\
  \end{align*}
  \begin{align*}
  A & \rightarrow A \\
  B & \rightarrow B \\
  C & \rightarrow C \\
  \end{align*}
  
  Convenient for programming reasons in the PCFG case
Algorithm analysis

Time complexity?

for each max from 2 to n

    for each min from max - 2 down to 0

        for each syntactic category C

            for each binary rule C -> C₁ C₂

                for each mid from min + 1 to max - 1
Algorithm analysis

Time complexity?

\[
\text{for each max from 2 to } n \\
\quad \text{for each min from } \text{max} - 2 \text{ down to 0} \\
\quad \quad \text{for each syntactic category } C \\
\quad \quad \quad \text{for each binary rule } C \rightarrow C_1 C_2 \\
\quad \quad \quad \quad \text{for each mid from } \text{min} + 1 \text{ to } \text{max} - 1
\]

\(O(n^3|R|)\) where \(|R|\) is the number of rules in the grammar
Practical time complexity

\[ \sim n^{3.6} \]
Probabilistic CKY
PCFGs

\[ S \rightarrow NP \ VP \ 1.0 \]
\[ VP \rightarrow V \ 0.2 \]
\[ VP \rightarrow V \ NP \ 0.4 \]
\[ VP \rightarrow VP \ PP \ 0.4 \]
\[ NP \rightarrow NP \ PP \ 0.3 \]
\[ NP \rightarrow D \ N \ 0.5 \]
\[ NP \rightarrow PN \ 0.2 \]
\[ PP \rightarrow P \ NF \ 1.0 \]

\[ N \rightarrow girl \ 0.2 \]
\[ N \rightarrow telescope \ 0.7 \]
\[ N \rightarrow sandwich \ 0.1 \]
\[ PN \rightarrow I \ 1.0 \]
\[ V \rightarrow saw \ 0.5 \]
\[ V \rightarrow ate \ 0.5 \]
\[ P \rightarrow with \ 0.6 \]
\[ P \rightarrow in \ 0.4 \]
\[ D \rightarrow a \ 0.3 \]
\[ D \rightarrow the \ 0.7 \]

\[ p(T) = 1.0 \times 0.2 \times 1.0 \times 0.4 \times 0.5 \times 0.3 \times \]
\[ 0.5 \times 0.3 \times 0.2 \times 1.0 \times 0.6 \times 0.5 \times 0.3 \times 0.7 \]
\[ = 2.26 \times 10^{-5} \]
CKY with PCFGs

- Chart is represented by a 3d array of floats:
  \[ \text{chart}[\text{min}][\text{max}][\text{label}] \]
  - It stores probabilities for the most probable subtree with a given signature

- \[ \text{chart}[0][n][S] \] will store the probability of the most probable full parse tree
Intuition

For every $C$ choose $C_1$, $C_2$ and mid such that

$$P(T_1) \times P(T_2) \times P(C \rightarrow C_1 C_2)$$

is maximal, where $T_1$ and $T_2$ are left and right subtrees.
for each \( w_i \) from left to right

for each preterminal rule \( C \rightarrow w_i \)

\[
\text{chart}[i-1][i][C] = \text{p}(C \rightarrow w_i)
\]
Implementation: binary rules

for each max from 2 to n

    for each min from max - 2 down to 0

        for each syntactic category C

            double best = undefined

            for each binary rule C -> C₁ C₂

                for each mid from min + 1 to max - 1

                    double t₁ = chart[min][mid][C₁]

                    double t₂ = chart[mid][max][C₂]

                    double candidate = t₁ * t₂ * p(C -> C₁ C₂)

                    if candidate > best then

                        best = candidate

                    chart[min][max][C] = best
Unary rules

- Similarly to CFGs: after producing scores for signatures (c, i, j), try to improve the scores by applying unary rules (and rule chains)
  - If improved, update the scores
Unary (reflexive transitive) closure

<table>
<thead>
<tr>
<th>Rule</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A \rightarrow B$</td>
<td>0.1</td>
</tr>
<tr>
<td>$B \rightarrow C$</td>
<td>0.2</td>
</tr>
</tbody>
</table>

$\Rightarrow$

<table>
<thead>
<tr>
<th>Rule</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A \rightarrow B$</td>
<td>0.1</td>
</tr>
<tr>
<td>$B \rightarrow C$</td>
<td>0.2</td>
</tr>
<tr>
<td>$A \rightarrow C$</td>
<td>$0.2 \times 0.1$</td>
</tr>
<tr>
<td>$C \rightarrow C$</td>
<td>1</td>
</tr>
<tr>
<td>$A \rightarrow A$</td>
<td>1</td>
</tr>
<tr>
<td>$B \rightarrow B$</td>
<td>1</td>
</tr>
</tbody>
</table>

Note that this is not a PCFG anymore as the rules do not sum to 1 for each parent.
Unary (reflexive transitive) closure

Note that this is not a PCFG anymore as the rules do not sum to 1 for each parent.

```
A → B  0.1
B → C  0.2

A → B  0.1
B → C  0.2
A → C  0.2 × 0.1

A → B
B → B
C → C
```

The fact that the rule is composite needs to be stored to recover the true tree.
Unary (reflexive transitive) closure

\[ A \rightarrow B \quad 0.1 \quad \Rightarrow \quad A \rightarrow B \quad 0.1 \quad A \rightarrow A \quad 1 \]
\[ B \rightarrow C \quad 0.2 \quad \Rightarrow \quad B \rightarrow C \quad 0.2 \quad B \rightarrow B \quad 1 \]
\[ A \rightarrow C \quad 0.2 \times 0.1 \quad \Rightarrow \quad C \rightarrow C \quad 1 \]

Note that this is not a PCFG anymore as the rules do not sum to 1 for each parent.

What about loops, like: \( A \rightarrow B \rightarrow A \rightarrow C \)?
Recovery of the tree

- For each signature we store backpointers to the elements from which it was built (e.g., rule and, for binary rules, midpoint)
  - start recovering from [0, n, S]

- Be careful with unary rules
  - Basically you can assume that you always used an unary rule from the closure (but it could be the trivial one \( C \rightarrow C \))
Speeding up the algorithm (approximate search)

Any ideas?
Speeding up the algorithm

- Basic pruning (roughly):
  - For every span (i,j) store only labels which have the probability at most N times smaller than the probability of the most probable label for this span
  - Check not all rules but only rules yielding subtree labels having non-zero probability

- Coarse-to-fine pruning
  - Parse with a smaller (simpler) grammar, and precompute (posterior) probabilities for each spans, and use only the ones with non-negligible probability from the previous grammar
Intrinsic evaluation:
- **Automatic**: evaluate against annotation provided by human experts (gold standard) according to some predefined measure
- **Manual**: … according to human judgment

Extrinsic evaluation: score syntactic representation by comparing how well a system using this representation performs on some task
- E.g., use syntactic representation as input for a semantic analyzer and compare results of the analyzer using syntax predicted by different parsers.
Standard evaluation setting in parsing

- Automatic intrinsic evaluation is used: parsers are evaluated against gold standard by provided by linguists
  - There is a standard split into the parts:
    - training set: used for estimation of model parameters
    - development set: used for tuning the model (initial experiments)
    - test set: final experiments to compare against previous work
Automatic evaluation of constituent parsers

- **Exact match**: percentage of trees predicted correctly
- **Bracket score**: scores how well individual phrases (and their boundaries) are identified
- **Crossing brackets**: percentage of phrases boundaries crossing

The most standard measure; we will focus on it
Brackets scores

- The most standard score is **bracket score**
- It regards a tree as a collection of brackets: \([min, max, C]\)
- The set of brackets predicted by a parser is compared against the set of brackets in the tree annotated by a linguist
- **Precision, recall and F1** are used as scores
Preview: F1 bracket score