

Syntax - Introduction

Pawan Goyal

CSE, IIT Kharagpur

Week 5: Lecture 1

What is Syntax?

NPTEL

What is Syntax?

- Refers to the way words are arranged together, and the relationship between them.

NPTEL

What is Syntax?

- Refers to the way words are arranged together, and the relationship between them.
- **Language Models:** Importance of modeling word order

NPTEL

What is Syntax?

- Refers to the way words are arranged together, and the relationship between them.
- **Language Models:** Importance of modeling word order
- **POS categories:** An equivalence class for words

NPTEL

What is Syntax?

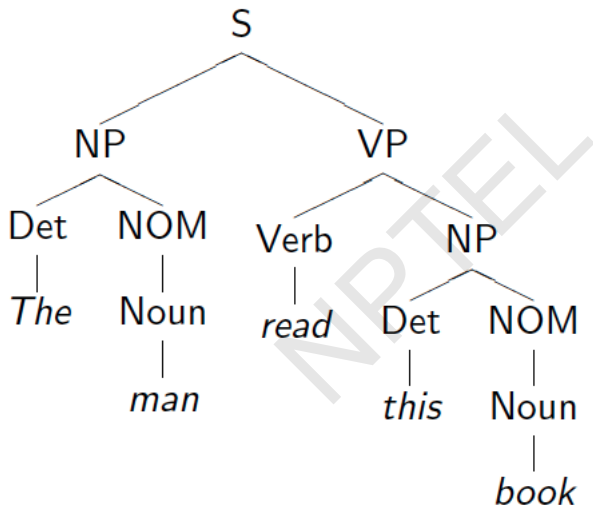
- Refers to the way words are arranged together, and the relationship between them.
- **Language Models:** Importance of modeling word order
- **POS categories:** An equivalence class for words
- More complex notions: constituency, grammatical relations, subcategorization etc.

What is Syntax?

- Refers to the way words are arranged together, and the relationship between them.
- **Language Models:** Importance of modeling word order
- **POS categories:** An equivalence class for words
- More complex notions: constituency, grammatical relations, subcategorization etc.



Syntax Tree: Example



Defining the notions: Constituency

Constituent

A group of words acts as a single unit - phrases, clauses etc.

NPTEL

Defining the notions: Constituency

Constituent

A group of words acts as a single unit - phrases, clauses etc.

Part of Speech - "Substitution Test"

The {sad, intelligent, green, fat, ...} one is in the corner.

Defining the notions: Constituency

Constituent

A group of words acts as a single unit - phrases, clauses etc.

Part of Speech - "Substitution Test"

The {sad, intelligent, green, fat, ...} one is in the corner.

Constituency: Noun Phrase

- *Kermit the frog*
- *they*
- *December twenty-sixth*
- *the reason he is running for president*

Constituent Phrases

Usually named based on the word that heads the constituent:

<i>the man from Amherst</i>	is a Noun Phrase (NP) because the head man is a noun
<i>extremely clever</i>	is an Adjective Phrase (AP) because the head clever is an adjective
<i>down the river</i>	is a Prepositional Phrase (PP) because the head down is a preposition
<i>killed the rabbit</i>	is a Verb Phrase (VP) because the head killed is a verb

Constituent Phrases

Usually named based on the word that heads the constituent:

<i>the man from Amherst</i>	is a Noun Phrase (NP) because the head <i>man</i> is a noun
<i>extremely clever</i>	is an Adjective Phrase (AP) because the head <i>clever</i> is an adjective
<i>down the river</i>	is a Prepositional Phrase (PP) because the head <i>down</i> is a preposition
<i>killed the rabbit</i>	is a Verb Phrase (VP) because the head <i>killed</i> is a verb

Words can also act as phrases

Joe grew potatoes

Constituent Phrases

Usually named based on the word that heads the constituent:

<i>the man from Amherst</i>	is a Noun Phrase (NP) because the head <i>man</i> is a noun
<i>extremely clever</i>	is an Adjective Phrase (AP) because the head <i>clever</i> is an adjective
<i>down the river</i>	is a Prepositional Phrase (PP) because the head <i>down</i> is a preposition
<i>killed the rabbit</i>	is a Verb Phrase (VP) because the head <i>killed</i> is a verb

Words can also act as phrases

Joe grew potatoes

Joe and *potatoes* are both nouns and noun phrases

Constituent Phrases

Usually named based on the word that heads the constituent:

<i>the man from Amherst</i>	is a Noun Phrase (NP) because the head <i>man</i> is a noun
<i>extremely clever</i>	is an Adjective Phrase (AP) because the head <i>clever</i> is an adjective
<i>down the river</i>	is a Prepositional Phrase (PP) because the head <i>down</i> is a preposition
<i>killed the rabbit</i>	is a Verb Phrase (VP) because the head <i>killed</i> is a verb

Words can also act as phrases

Joe grew potatoes

Joe and *potatoes* are both nouns and noun phrases

Compare with: *The man from Amherst grew beautiful russet potatoes.*

Constituent Phrases

Usually named based on the word that heads the constituent:

<i>the man from Amherst</i>	is a Noun Phrase (NP) because the head <i>man</i> is a noun
<i>extremely clever</i>	is an Adjective Phrase (AP) because the head <i>clever</i> is an adjective
<i>down the river</i>	is a Prepositional Phrase (PP) because the head <i>down</i> is a preposition
<i>killed the rabbit</i>	is a Verb Phrase (VP) because the head <i>killed</i> is a verb

Words can also act as phrases

Joe grew potatoes

Joe and *potatoes* are both nouns and noun phrases

Compare with: *The man from Amherst grew beautiful russet potatoes.*

Joe appears in a place that a larger noun phrase could have been.

Evidence that constituency exists

They appear in similar environments

NPTEL

Evidence that constituency exists

They appear in similar environments

Kermit the frog comes on stage

They come to Massachusetts every summer

December twenty-sixth comes after Christmas

The reason he is running for president comes out only now.

But not each individual word in the constituent

*The comes out... *is comes out... *for comes out...

Evidence that constituency exists

They appear in similar environments

Kermit the frog comes on stage

They come to Massachusetts every summer

December twenty-sixth comes after Christmas

The reason he is running for president comes out only now.

But not each individual word in the constituent

*The comes out... *is comes out... *for comes out...

Can be placed in a number of different locations

Evidence that constituency exists

They appear in similar environments

Kermit the frog comes on stage

They come to Massachusetts every summer

December twenty-sixth comes after Christmas

The reason he is running for president comes out only now.

But not each individual word in the constituent

*The comes out... *is comes out... *for comes out...

Can be placed in a number of different locations

Constituent = Prepositional phrase: On December twenty-sixth

On December twenty-sixth I'd like to fly to Florida.

I'd like to fly on December twenty-sixth to Florida.

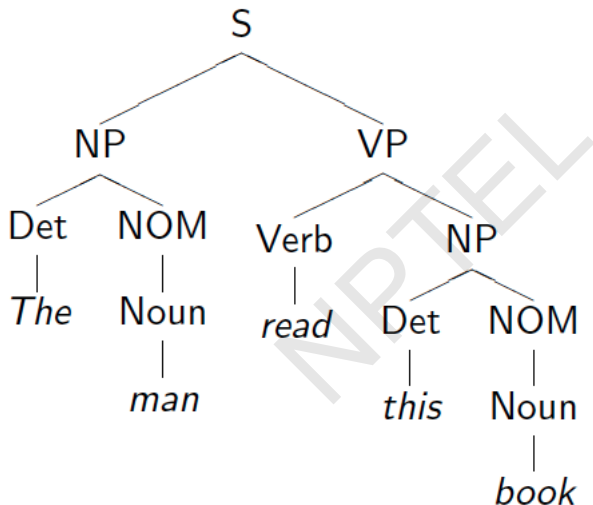
I'd like to fly to Florida on December twenty-sixth.

But not split apart

*On December I'd like to fly twenty-sixth to Florida.

*On I'd like to fly December twenty-sixth to Florida.

Modeling Constituency: what tool do we need?



Modeling Constituency

NPTEL

Modeling Constituency

Context-free grammar

The most common way of modeling constituency

NPTEL

Modeling Constituency

Context-free grammar

The most common way of modeling constituency

Consists of production Rules

These rules express the ways in which the symbols of the language can be grouped and ordered together

Modeling Constituency

Context-free grammar

The most common way of modeling constituency

Consists of production Rules

These rules express the ways in which the symbols of the language can be grouped and ordered together

Example

Noun phrase can be composed of either a ProperNoun or a determiner (Det) followed by a Nominal; a Nominal can be more than one nouns

Modeling Constituency

Context-free grammar

The most common way of modeling constituency

Consists of production Rules

These rules express the ways in which the symbols of the language can be grouped and ordered together

Example

Noun phrase can be composed of either a ProperNoun or a determiner (Det) followed by a Nominal; a Nominal can be more than one nouns

NP → Det Nominal

NP → ProperNoun

Nominal → Noun | Noun Nominal

NPTEL

CFG: $G = (T, N, S, R)$

- T : set of terminals
- N : set of non-terminals
 - ▶ For NLP, we distinguish out a set $P \subset N$ of pre-terminals, which always rewrite as terminals
- S : start symbol
- R : Rules/productions of the form $X \rightarrow \gamma$, $X \in N$ and $\gamma \in (T \cup N)^*$

CFG: $G = (T, N, S, R)$

- T : set of terminals
- N : set of non-terminals
 - ▶ For NLP, we distinguish out a set $P \subset N$ of pre-terminals, which always rewrite as terminals
- S : start symbol
- R : Rules/productions of the form $X \rightarrow \gamma$, $X \in N$ and $\gamma \in (T \cup N)^*$

Terminals and pre-terminals

Terminals mainly correspond to words in the language while pre-terminals mainly correspond to POS categories

CFG: $G = (T, N, S, R)$

- T : set of terminals
- N : set of non-terminals
 - ▶ For NLP, we distinguish out a set $P \subset N$ of pre-terminals, which always rewrite as terminals
- S : start symbol
- R : Rules/productions of the form $X \rightarrow \gamma$, $X \in N$ and $\gamma \in (T \cup N)^*$

Terminals and pre-terminals

Terminals mainly correspond to words in the language while pre-terminals mainly correspond to POS categories

Example

$NP \rightarrow \text{Det Nominal}$

$NP \rightarrow \text{ProperNoun}$

$\text{Nominal} \rightarrow \text{Noun} \mid \text{Noun Nominal}$

Example

$NP \rightarrow \text{Det Nominal}$

$NP \rightarrow \text{ProperNoun}$

$\text{Nominal} \rightarrow \text{Noun} \mid \text{Noun Nominal}$

Now, these can be combined with other rules, that express facts about a lexicon.

Example

$NP \rightarrow \text{Det Nominal}$

$NP \rightarrow \text{ProperNoun}$

$\text{Nominal} \rightarrow \text{Noun} \mid \text{Noun Nominal}$

Now, these can be combined with other rules, that express facts about a lexicon.

$\text{Det} \rightarrow \text{a}$

$\text{Det} \rightarrow \text{the}$

$\text{Noun} \rightarrow \text{flight}$

Example

$NP \rightarrow \text{Det Nominal}$

$NP \rightarrow \text{ProperNoun}$

$\text{Nominal} \rightarrow \text{Noun} \mid \text{Noun Nominal}$

Now, these can be combined with other rules, that express facts about a lexicon.

$\text{Det} \rightarrow \text{a}$

$\text{Det} \rightarrow \text{the}$

$\text{Noun} \rightarrow \text{flight}$

Can you identify the terminal, non-terminals and preterminals?

CFG as a generator

NP → Det Nominal

NP → ProperNoun

Nominal → Noun | Noun Nominal

Det → a

Det → the

Noun → flight

NPTEL

CFG as a generator

NP → Det Nominal

NP → ProperNoun

Nominal → Noun | Noun Nominal

Det → a

Det → the

Noun → flight

Generating 'a flight':

NPTTEL

CFG as a generator

$NP \rightarrow \text{Det Nominal}$

$NP \rightarrow \text{ProperNoun}$

$\text{Nominal} \rightarrow \text{Noun} \mid \text{Noun Nominal}$

$\text{Det} \rightarrow \text{a}$

$\text{Det} \rightarrow \text{the}$

$\text{Noun} \rightarrow \text{flight}$

Generating 'a flight':

NP

CFG as a generator

$NP \rightarrow \text{Det Nominal}$

$NP \rightarrow \text{ProperNoun}$

$\text{Nominal} \rightarrow \text{Noun} \mid \text{Noun Nominal}$

$\text{Det} \rightarrow \text{a}$

$\text{Det} \rightarrow \text{the}$

$\text{Noun} \rightarrow \text{flight}$

Generating 'a flight':

$NP \rightarrow \text{Det Nominal}$

NPTTEL

CFG as a generator

$NP \rightarrow \text{Det Nominal}$

$NP \rightarrow \text{ProperNoun}$

$\text{Nominal} \rightarrow \text{Noun} \mid \text{Noun Nominal}$

$\text{Det} \rightarrow \text{a}$

$\text{Det} \rightarrow \text{the}$

$\text{Noun} \rightarrow \text{flight}$

Generating 'a flight':

$NP \rightarrow \text{Det Nominal}$

$\rightarrow \text{Det Noun}$

CFG as a generator

$NP \rightarrow \text{Det Nominal}$

$NP \rightarrow \text{ProperNoun}$

$\text{Nominal} \rightarrow \text{Noun} \mid \text{Noun Nominal}$

$\text{Det} \rightarrow \text{a}$

$\text{Det} \rightarrow \text{the}$

$\text{Noun} \rightarrow \text{flight}$

Generating 'a flight':

$NP \rightarrow \text{Det Nominal}$

$\rightarrow \text{Det Noun} \rightarrow \text{a Noun}$

CFG as a generator

$NP \rightarrow \text{Det Nominal}$

$NP \rightarrow \text{ProperNoun}$

$\text{Nominal} \rightarrow \text{Noun} \mid \text{Noun Nominal}$

$\text{Det} \rightarrow \text{a}$

$\text{Det} \rightarrow \text{the}$

$\text{Noun} \rightarrow \text{flight}$

Generating 'a flight':

$NP \rightarrow \text{Det Nominal}$

$\rightarrow \text{Det Noun} \rightarrow \text{a Noun} \rightarrow \text{a flight}$

CFG as a generator

$NP \rightarrow \text{Det Nominal}$

$NP \rightarrow \text{ProperNoun}$

$\text{Nominal} \rightarrow \text{Noun} \mid \text{Noun Nominal}$

$\text{Det} \rightarrow \text{a}$

$\text{Det} \rightarrow \text{the}$

$\text{Noun} \rightarrow \text{flight}$

Generating 'a flight':

$NP \rightarrow \text{Det Nominal}$

$\rightarrow \text{Det Noun} \rightarrow \text{a Noun} \rightarrow \text{a flight}$

- Thus a CFG can be used to randomly generate a series of strings

CFG as a generator

$NP \rightarrow \text{Det Nominal}$

$NP \rightarrow \text{ProperNoun}$

$\text{Nominal} \rightarrow \text{Noun} \mid \text{Noun Nominal}$

$\text{Det} \rightarrow \text{a}$

$\text{Det} \rightarrow \text{the}$

$\text{Noun} \rightarrow \text{flight}$

Generating 'a flight':

$NP \rightarrow \text{Det Nominal}$

$\rightarrow \text{Det Noun} \rightarrow \text{a Noun} \rightarrow \text{a flight}$

- Thus a CFG can be used to randomly generate a series of strings
- This sequence of rule expansions is called a derivation of the string of words, usually represented as a tree

A CFG defines a formal language = set of all sentences (string of words) that can be derived by the grammar

A CFG defines a formal language = set of all sentences (string of words) that can be derived by the grammar

- Sentences in this set are said to be **grammatical**
- Sentences outside this set are said to be **ungrammatical**

Recursive Definition

- $PP \rightarrow \text{Prep NP}$
- $NP \rightarrow \text{Noun PP}$

Recursive Definition

- PP → Prep NP
- NP → Noun PP

Example Sentence

[_SThe mailman ate his [_{NP} lunch [_{PP} with his friend [_{PP} from the cleaning staff [_{PP} of the building [_{PP} at the intersection [_{PP} on the north end [_{PP} of town]]]]]]].

What does Context stand for in CFG?

NPTTEL

What does *Context* stand for in CFG?

- The notion of *context* has nothing to do with the ordinary meaning of word context in language

NPTEL

What does Context stand for in CFG?

- The notion of *context* has nothing to do with the ordinary meaning of word context in language
- All it really means is that the non-terminal on the left-hand side of a rule is out there all by itself (free of context)

What does Context stand for in CFG?

- The notion of *context* has nothing to do with the ordinary meaning of word context in language
- All it really means is that the non-terminal on the left-hand side of a rule is out there all by itself (free of context)

$A \rightarrow BC$

- I can rewrite A as B followed by C regardless of the context in which A is found
- Or when I see a B followed by a C , I can infer an A regardless of the surrounding context

Syntax -Parsing I

Pawan Goyal

CSE, IIT Kharagpur

Week 5: Lecture 2

Grammar Rewrite Rules

$S \rightarrow NP VP$

$S \rightarrow Aux NP VP$

$S \rightarrow VP$

$NP \rightarrow Det NOM$

$NOM \rightarrow Noun$

$NOM \rightarrow Noun NOM$

$VP \rightarrow Verb$

$VP \rightarrow Verb NP$

$Det \rightarrow that \mid this \mid a \mid the$

$Noun \rightarrow book \mid flight \mid meal \mid man$

$Verb \rightarrow book \mid include \mid read$

$Aux \rightarrow does$

NPTEL

Grammar Rewrite Rules

$S \rightarrow NP VP$

$S \rightarrow Aux NP VP$

$S \rightarrow VP$

$NP \rightarrow Det NOM$

$NOM \rightarrow Noun$

$NOM \rightarrow Noun NOM$

$VP \rightarrow Verb$

$VP \rightarrow Verb NP$

$Det \rightarrow that \mid this \mid a \mid the$

$Noun \rightarrow book \mid flight \mid meal \mid man$

$Verb \rightarrow book \mid include \mid read$

$Aux \rightarrow does$

$S \rightarrow NP VP$

$\rightarrow Det NOM VP$

$\rightarrow The NOM VP$

$\rightarrow The Noun VP$

$\rightarrow The man VP$

$\rightarrow The man Verb NP$

$\rightarrow The man read NP$

$\rightarrow The man read Det NOM$

$\rightarrow The man read this NOM$

$\rightarrow The man read this Noun$

$\rightarrow The man read this book$

S → NP VP
→ Det NOM VP
→ *The* NOM VP
→ *The* Noun VP
→ *The man* VP
→ *The man* Verb NP
→ *The man read* NP
→ *The man read* Det NOM
→ *The man read this* NOM
→ *The man read this* Noun
→ *The man read this book*

Parse Tree

S \rightarrow NP VP

\rightarrow Det NOM VP

\rightarrow *The* NOM VP

\rightarrow *The* Noun VP

\rightarrow *The man* VP

\rightarrow *The man* Verb NP

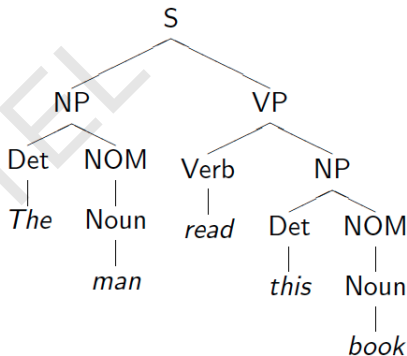
\rightarrow *The man* read NP

\rightarrow *The man* read Det NOM

\rightarrow *The man* read *this* NOM

\rightarrow *The man* read *this* Noun

\rightarrow *The man* read *this book*



What is Parsing?

- The process of taking a string and a grammar and returning all possible parse trees for that string

NPTEL

What is Parsing?

- The process of taking a string and a grammar and returning all possible parse trees for that string
- That is, find all trees, whose root is the start symbol S , which cover exactly the words in the input

What is Parsing?

- The process of taking a string and a grammar and returning all possible parse trees for that string
- That is, find all trees, whose root is the start symbol S , which cover exactly the words in the input

What are the constraints? “book that flight”

- There must be three leaves, *book*, *that* and *flight*
- The tree must have one root, the start symbol S

What is Parsing?

- The process of taking a string and a grammar and returning all possible parse trees for that string
- That is, find all trees, whose root is the start symbol S , which cover exactly the words in the input

What are the constraints? “book that flight”

- There must be three leaves, *book*, *that* and *flight*
- The tree must have one root, the start symbol S
- Give rise to two search strategies: *top-down* (goal-oriented) and *bottom-up* (data-directed)

Grammar

S → **NP VP**

S → **Aux NP VP**

S → **VP**

NP → **Pronoun**

NP → **Proper-Noun**

NP → **Det Nominal**

Nominal → **Noun**

Nominal → **Nominal Noun**

Nominal → **Nominal PP**

VP → **Verb**

VP → **Verb NP**

VP → **VP PP**

PP → **Prep NP**

Lexicon

Det → **the | a | that | this**

Noun → **book | flight | meal | money**

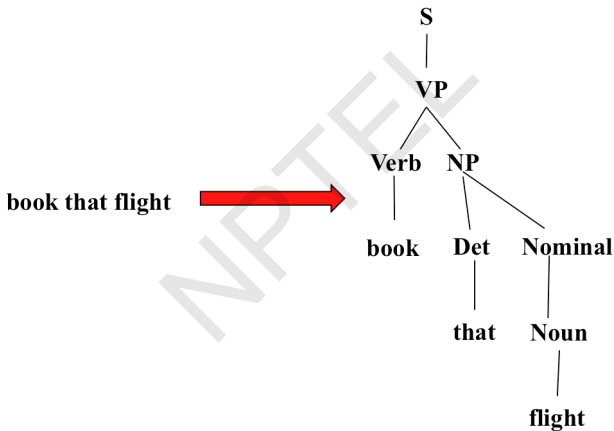
Verb → **book | include | prefer**

Pronoun → **I | he | she | me**

Proper-Noun → **Houston | NWA**

Aux → **does**

Prep → **from | to | on | near | through**



- Searches for a parse tree by trying to build upon the root node S down to the leaves

NPTEL

Top-Down Parsing

- Searches for a parse tree by trying to build upon the root node S down to the leaves
- Start by assuming that the input can be derived by the designated start symbol S

Top-Down Parsing

- Searches for a parse tree by trying to build upon the root node S down to the leaves
- Start by assuming that the input can be derived by the designated start symbol S
- Find all trees that can start with S , by looking at the grammar rules with S on the left-hand side

Top-Down Parsing

- Searches for a parse tree by trying to build upon the root node S down to the leaves
- Start by assuming that the input can be derived by the designated start symbol S
- Find all trees that can start with S , by looking at the grammar rules with S on the left-hand side
- Trees are grown downward until they eventually reach the POS categories at the bottom

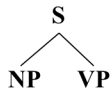
Top-Down Parsing

- Searches for a parse tree by trying to build upon the root node S down to the leaves
- Start by assuming that the input can be derived by the designated start symbol S
- Find all trees that can start with S , by looking at the grammar rules with S on the left-hand side
- Trees are grown downward until they eventually reach the POS categories at the bottom
- Trees whose leaves fail to match the words in the input can be rejected

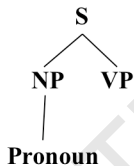
S

NPTEL

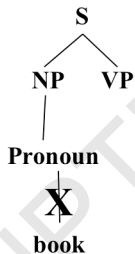
Top-Down Parsing



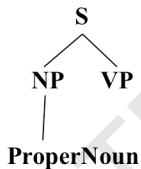
Top-Down Parsing



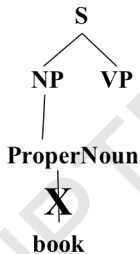
Top-Down Parsing



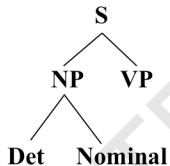
Top-Down Parsing



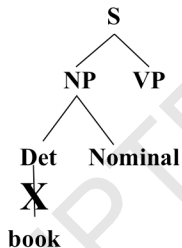
Top-Down Parsing



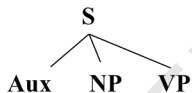
Top-Down Parsing



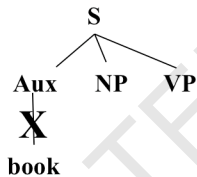
Top-Down Parsing



Top-Down Parsing



Top-Down Parsing



Top-Down Parsing

S
|
VP

NPTEL

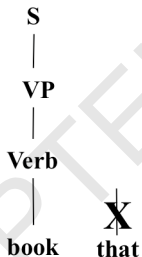
Top-Down Parsing

S
|
VP
|
Verb

Top-Down Parsing

S
|
VP
|
Verb
|
book

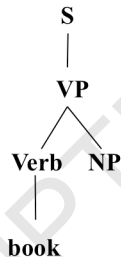
Top-Down Parsing



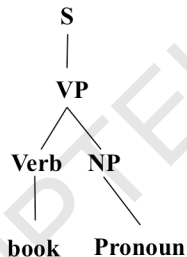
Top-Down Parsing



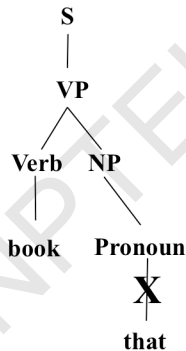
Top-Down Parsing



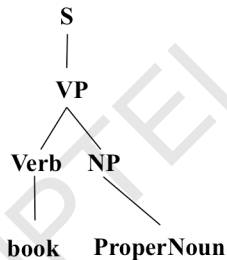
Top-Down Parsing



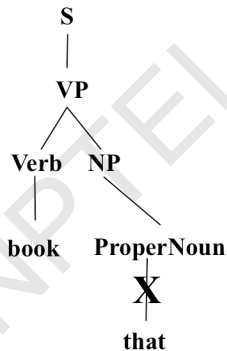
Top-Down Parsing



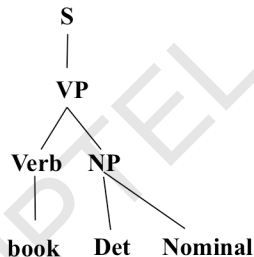
Top-Down Parsing



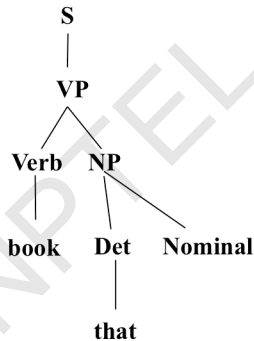
Top-Down Parsing



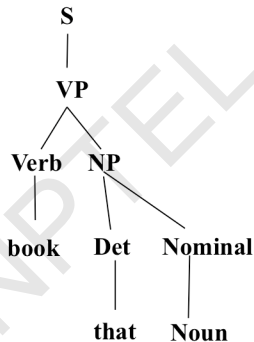
Top-Down Parsing



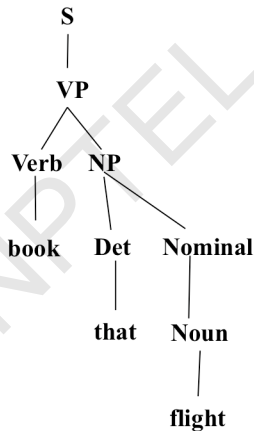
Top-Down Parsing



Top-Down Parsing



Top-Down Parsing



- The parser starts with the words of the input, and tries to build trees from the words up, by applying rules from the grammar one at a time
- Parser looks for the places in the parse-in-progress where the right-hand-side of some rule might fit.

Bottom-Up Parsing

book

that

flight

Bottom-Up Parsing

Noun

|
book

that

flight

Bottom-Up Parsing

Nominal

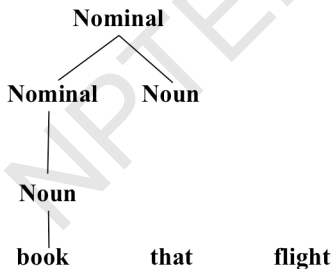
Noun

book

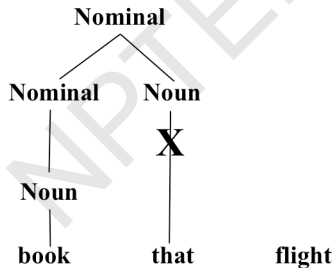
that

flight

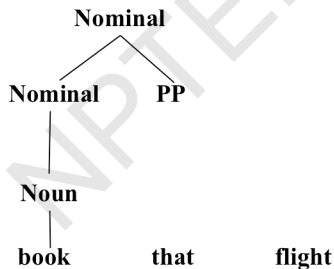
Bottom-Up Parsing

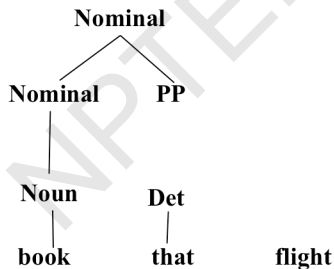


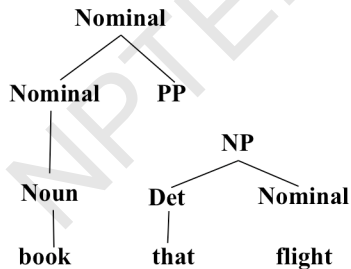
Bottom-Up Parsing



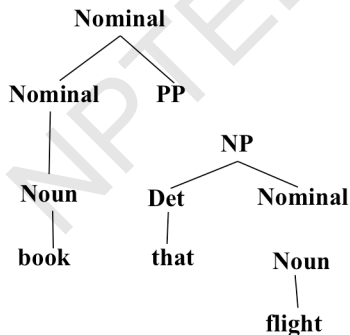
Bottom-Up Parsing



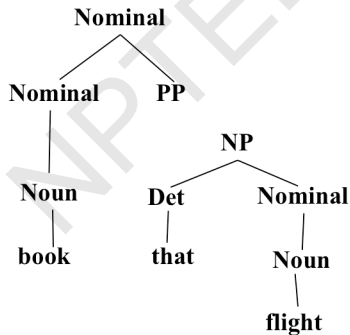


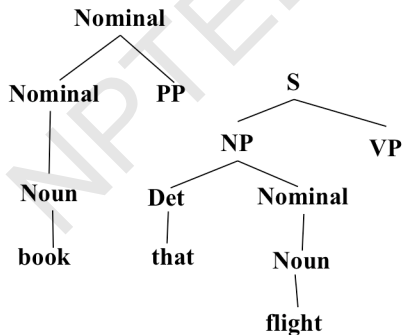


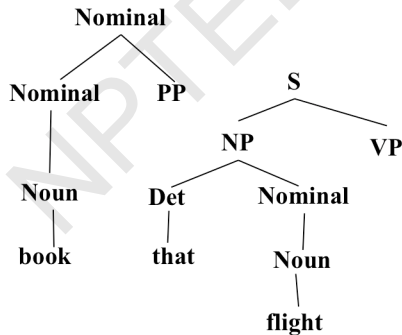
Bottom-Up Parsing



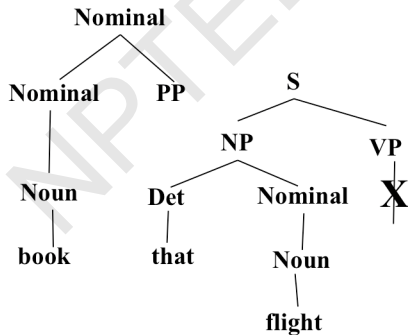
Bottom-Up Parsing



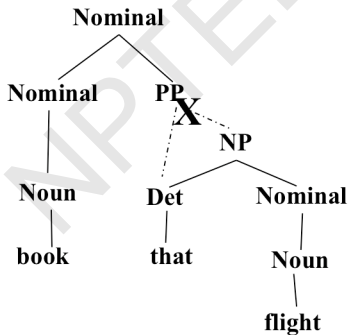


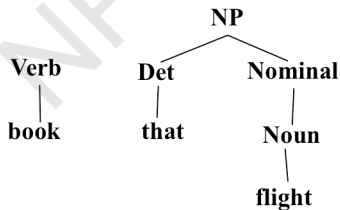


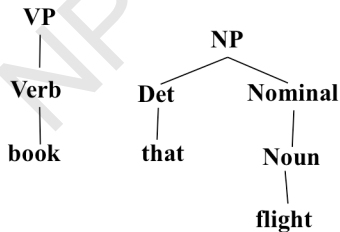
Bottom-Up Parsing



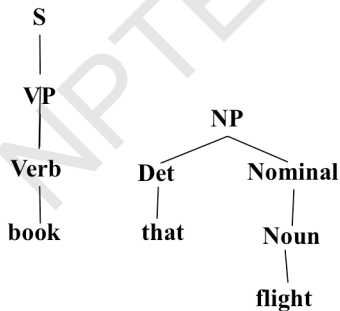
Bottom-Up Parsing



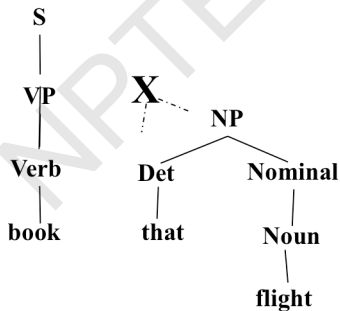




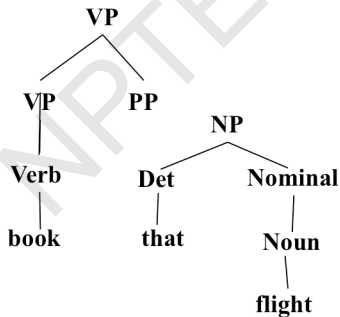
Bottom-Up Parsing



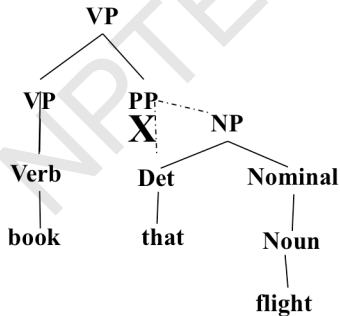
Bottom-Up Parsing



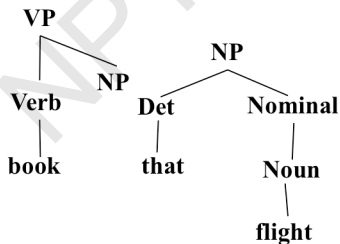
Bottom-Up Parsing

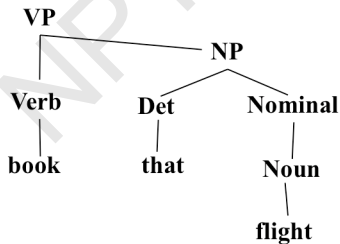


Bottom-Up Parsing

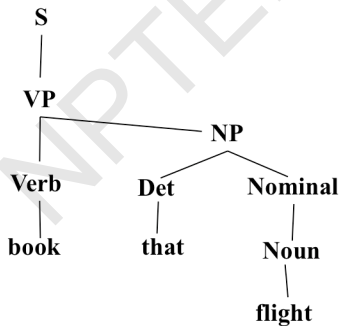


Bottom-Up Parsing





Bottom-Up Parsing



Top-Down vs. Bottom-Up

NPTEL

Top-Down vs. Bottom-Up

- Top down never explores options that will not lead to a full parse, but can explore many options that never connect to the actual sentence.

Top-Down vs. Bottom-Up

- Top down never explores options that will not lead to a full parse, but can explore many options that never connect to the actual sentence.
- Bottom up never explores options that do not connect to the actual sentence but can explore options that can never lead to a full parse.

Top-Down vs. Bottom-Up

- Top down never explores options that will not lead to a full parse, but can explore many options that never connect to the actual sentence.
- Bottom up never explores options that do not connect to the actual sentence but can explore options that can never lead to a full parse.
- Relative amounts of wasted search depend on how much the grammar branches in each direction.

Dynamic Programming Parsing

NPTEL

- To avoid extensive repeated work, must cache intermediate results, i.e. completed phrases.

- To avoid extensive repeated work, must cache intermediate results, i.e. completed phrases.
- Caching (memoizing) critical to obtaining a polynomial time parsing (recognition) algorithm for CFGs.

- To avoid extensive repeated work, must cache intermediate results, i.e. completed phrases.
- Caching (memoizing) critical to obtaining a polynomial time parsing (recognition) algorithm for CFGs.
- Dynamic programming algorithms based on both top-down and bottom-up search can achieve $O(n^3)$ recognition time where n is the length of the input string.

- CKY (Cocke-Kasami-Younger) algorithm: bottom-up, requires normalizing the grammar

- CKY (Cocke-Kasami-Younger) algorithm: bottom-up, requires normalizing the grammar
- Earley Parser - top-down, does not require normalizing grammar, more complex

- CKY (Cocke-Kasami-Younger) algorithm: bottom-up, requires normalizing the grammar
- Earley Parser - top-down, does not require normalizing grammar, more complex
- More generally, *chart parsers* retain completed phrases in a chart and can combine top-down and bottom-up searches.

- Grammar must be converted to Chomsky normal form (CNF) in which all productions must have
 - ▶ Either, exactly two non-terminals on the RHS
 - ▶ Or, 1 terminal symbol on the RHS

- Grammar must be converted to Chomsky normal form (CNF) in which all productions must have
 - ▶ Either, exactly two non-terminals on the RHS
 - ▶ Or, 1 terminal symbol on the RHS
- Parse bottom-up storing phrases formed from all substrings in a triangular table (chart)

Original Grammar

S → **NP VP**

S → **Aux NP VP**

S → **VP**

NP → **Pronoun**

NP → **Proper-Noun**

NP → **Det Nominal**

Nominal → **Noun**

Nominal → **Nominal Noun**

Nominal → **Nominal PP**

VP → **Verb**

VP → **Verb NP**

VP → **VP PP**

PP → **Prep NP**

Pronoun → **I | he | she | me**

Noun → **book | flight | meal | money**

Verb → **book | include | prefer**

Proper-Noun → **Houston | NWA**

Original Grammar

$S \rightarrow NP VP$
 $S \rightarrow Aux NP VP$
 $S \rightarrow VP$
 $NP \rightarrow Pronoun$
 $NP \rightarrow Proper-Noun$
 $NP \rightarrow Det Nominal$
 $Nominal \rightarrow Noun$
 $Nominal \rightarrow Nominal Noun$
 $Nominal \rightarrow Nominal PP$
 $VP \rightarrow Verb$
 $VP \rightarrow Verb NP$
 $VP \rightarrow VP PP$
 $PP \rightarrow Prep NP$
 $Pronoun \rightarrow I \mid he \mid she \mid me$
 $Noun \rightarrow book \mid flight \mid meal \mid money$
 $Verb \rightarrow book \mid include \mid prefer$
 $Proper-Noun \rightarrow Houston \mid NWA$

Chomsky Normal Form

$S \rightarrow NP VP$
 $S \rightarrow X1 VP$
 $X1 \rightarrow Aux NP$
 $S \rightarrow book \mid include \mid prefer$
 $S \rightarrow Verb NP$
 $S \rightarrow VP PP$
 $NP \rightarrow I \mid he \mid she \mid me$
 $NP \rightarrow Houston \mid NWA$
 $NP \rightarrow Det Nominal$
 $Nominal \rightarrow book \mid flight \mid meal \mid money$
 $Nominal \rightarrow Nominal Noun$
 $Nominal \rightarrow Nominal PP$
 $VP \rightarrow book \mid include \mid prefer$
 $VP \rightarrow Verb NP$
 $VP \rightarrow VP PP$
 $PP \rightarrow Prep NP$
 $Pronoun \rightarrow I \mid he \mid she \mid me$
 $Noun \rightarrow book \mid flight \mid meal \mid money$
 $Verb \rightarrow book \mid include \mid prefer$
 $Proper-Noun \rightarrow Houston \mid NWA$

Syntax -CKY, PCFGs

Pawan Goyal

CSE, IIT Kharagpur

Week 5: Lecture 3

CKY Algorithm

- Let n be the number of words in the input. Think about $n + 1$ lines separating them, numbered 0 to n .
- x_{ij} will denote the words between line i and j
- We build a table so that x_{ij} contains all the possible non-terminal spanning for words between line i and j .
- We build the Table bottom-up.

CKY Algorithm

- Let n be the number of words in the input. Think about $n + 1$ lines separating them, numbered 0 to n .
- x_{ij} will denote the words between line i and j
- We build a table so that x_{ij} contains all the possible non-terminal spanning for words between line i and j .
- We build the Table bottom-up.

Home Exercise

Use CKY algorithm to find the parse tree for “Book the flight through Houston” using the CNF form shown in the previous slide.

CKY for CFG

a 1	pilot 2	likes 3	flying 4	planes 5

$S \rightarrow NP VP$
 $VP \rightarrow VBG NNS$
 $VP \rightarrow VBZ VP$
 $VP \rightarrow VBZ NP$
 $NP \rightarrow DT NN$
 $NP \rightarrow JJ NNS$
 $DT \rightarrow a$
 $NN \rightarrow pilot$
 $VBZ \rightarrow likes$
 $VBG \rightarrow flying$
 $JJ \rightarrow flying$
 $NNS \rightarrow planes$

CKY for CFG

a 1	pilot 2	likes 3	flying 4	planes 5
Determiner DT	Noun Phrase NP	-	-	S S
	Noun Singular NN	-	-	-
		VBZ Ver, 3rd Person, Singular Present	-	Verb Phrase VP VP
			Adjective JJ Verb, Gerund or Past Participle VBG	NP VP
				Noun Plural NNS

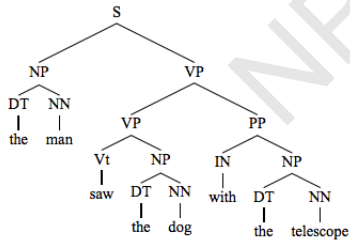
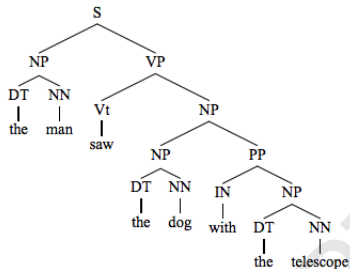
Sentence

$$\begin{aligned}
 S &\rightarrow NP VP \\
 VP &\rightarrow VBG NNS \\
 VP &\rightarrow VBZ VP \\
 VP &\rightarrow VBZ NP \\
 NP &\rightarrow DT NN \\
 NP &\rightarrow JJ NNS \\
 DT &\rightarrow a \\
 NN &\rightarrow pilot \\
 VBZ &\rightarrow likes \\
 VBG &\rightarrow flying \\
 JJ &\rightarrow flying \\
 NNS &\rightarrow planes
 \end{aligned}$$

Vt: Transitive Verb

IN: Prepositions (in, on, at, by, with) or Subordinating Conjunctions (because, although, if)

What about Ambiguities?



Probabilistic Context-free grammars (PCFGs)

PCFG: $G = (T, N, S, R, P)$

- T : set of terminals
- N : set of non-terminals
 - ▶ For NLP, we distinguish out a set $P \subset N$ of pre-terminals, which always rewrite as terminals
- S : start symbol
- R : Rules/productions of the form $X \rightarrow \gamma$, $X \in N$ and $\gamma \in (T \cup N)^*$

Probabilistic Context-free grammars (PCFGs)

PCFG: $G = (T, N, S, R, P)$

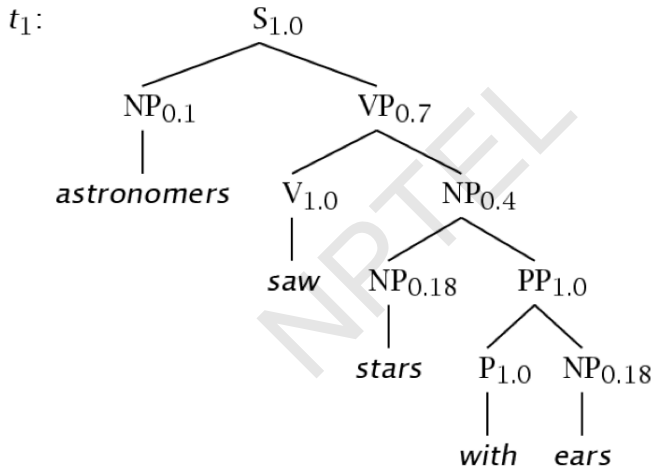
- T : set of terminals
- N : set of non-terminals
 - ▶ For NLP, we distinguish out a set $P \subset N$ of pre-terminals, which always rewrite as terminals
- S : start symbol
- R : Rules/productions of the form $X \rightarrow \gamma$, $X \in N$ and $\gamma \in (T \cup N)^*$
- $P(R)$ gives the probability of each rule.

$$\forall X \in N, \sum_{X \rightarrow \gamma \in R} P(X \rightarrow \gamma) = 1$$

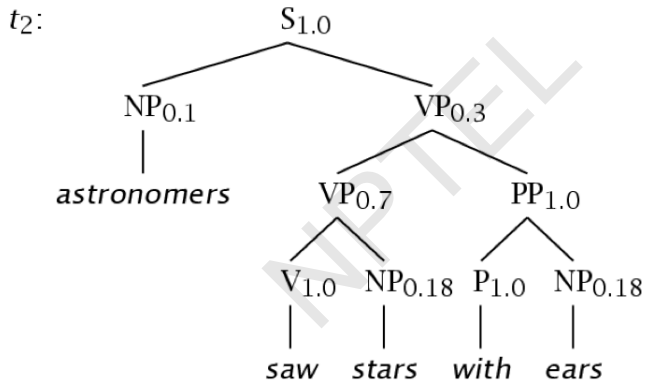
A Simple PCFG (in CNF)

S	→	NP VP	1.0	NP	→	NP PP	0.4
VP	→	V NP	0.7	NP	→	<i>astronomers</i>	0.1
VP	→	VP PP	0.3	NP	→	<i>ears</i>	0.18
PP	→	P NP	1.0	NP	→	<i>saw</i>	0.04
P	→	<i>with</i>	1.0	NP	→	<i>stars</i>	0.18
V	→	<i>saw</i>	1.0	NP	→	<i>telescope</i>	0.1

Example Trees



Example Trees



Probability of trees and strings

- $P(t)$: The probability of tree is the product of the probabilities of the rules used to generate it
- $P(w_{1n})$: The probability of the string is the sum of the probabilities of the trees which have that string as their yield

Tree and String probabilities

NPTTEL

Tree and String probabilities

w_{15} = *astronomers saw stars with ears*

$$\begin{aligned}P(t_1) &= 1.0 * 0.1 * 0.7 * 1.0 * 0.4 * 0.18 \\&\quad * 1.0 * 1.0 * 0.18 \\&= 0.0009072\end{aligned}$$

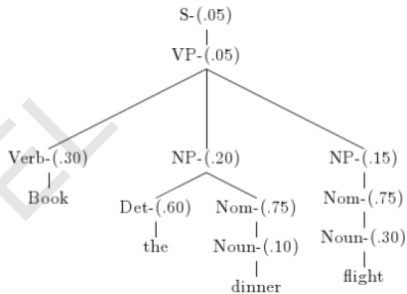
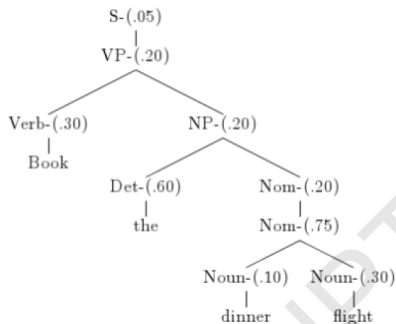
$$\begin{aligned}P(t_2) &= 1.0 * 0.1 * 0.3 * 0.7 * 1.0 * 0.18 \\&\quad * 1.0 * 1.0 * 0.18 \\&= 0.0006804\end{aligned}$$

$$\begin{aligned}P(w_{15}) &= P(t_1) + P(t_2) \\&= 0.0009072 + 0.0006804 \\&= 0.0015876\end{aligned}$$

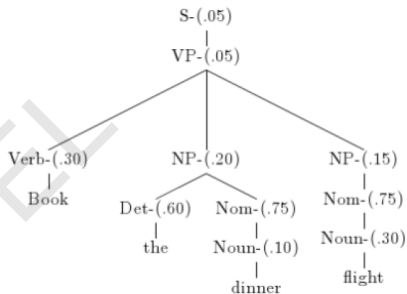
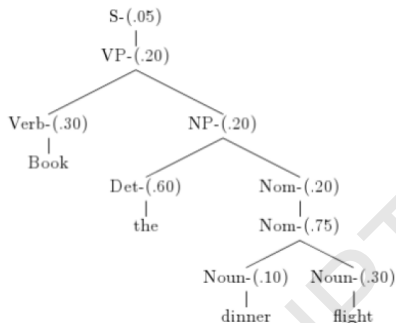
“Book the dinner flight”

NPTEL

“Book the dinner flight”



“Book the dinner flight”



Probabilities

- Parse tree 1: $.05 \times .20 \times .30 \times .20 \times .60 \times .20 \times .75 \times .10 \times .30 = 1.62 \times 10^{-6}$
- Parse tree 2: $.05 \times .05 \times .30 \times .20 \times .60 \times .75 \times .10 \times .15 \times .75 \times .30 = 2.28 \times 10^{-7}$

Features of PCFGs

- As the number of possible trees for a given input grows, a PCFG gives some idea of the plausibility of a particular parse

NPTEL

Features of PCFGs

- As the number of possible trees for a given input grows, a PCFG gives some idea of the plausibility of a particular parse
- *But* the probability estimates are based purely on structural factors, and do not factor in lexical co-occurrence. Thus, PCFG does not give a very good idea of the plausibility of the sentence.

- As the number of possible trees for a given input grows, a PCFG gives some idea of the plausibility of a particular parse
- *But* the probability estimates are based purely on structural factors, and do not factor in lexical co-occurrence. Thus, PCFG does not give a very good idea of the plausibility of the sentence.
- Real text tends to have grammatical mistakes. PCFG avoids this problem by ruling out nothing, but by giving implausible sentences a low probability

- As the number of possible trees for a given input grows, a PCFG gives some idea of the plausibility of a particular parse
- *But* the probability estimates are based purely on structural factors, and do not factor in lexical co-occurrence. Thus, PCFG does not give a very good idea of the plausibility of the sentence.
- Real text tends to have grammatical mistakes. PCFG avoids this problem by ruling out nothing, but by giving implausible sentences a low probability
- In practice, a PCFG is a worse language model for English than an n-gram model

Features of PCFGs

- As the number of possible trees for a given input grows, a PCFG gives some idea of the plausibility of a particular parse
- *But* the probability estimates are based purely on structural factors, and do not factor in lexical co-occurrence. Thus, PCFG does not give a very good idea of the plausibility of the sentence.
- Real text tends to have grammatical mistakes. PCFG avoids this problem by ruling out nothing, but by giving implausible sentences a low probability
- In practice, a PCFG is a worse language model for English than an n-gram model
- All else being equal, the probability of a smaller tree is greater than a larger tree

Important Questions?

Let W_{1m} be a sentence, G a grammar, t a parse tree

NPTTEL

Important Questions?

Let W_{1m} be a sentence, G a grammar, t a parse tree

- What is the most likely parse of sentence?

$$\operatorname{argmax}_t P(t|w_{1m}, G)$$

Important Questions?

Let W_{1m} be a sentence, G a grammar, t a parse tree

- What is the most likely parse of sentence?

$$\operatorname{argmax}_t P(t|w_{1m}, G)$$

- What is the probability of a sentence?

$$P(w_{1m}|G)$$

Important Questions?

Let W_{1m} be a sentence, G a grammar, t a parse tree

- What is the most likely parse of sentence?

$$\operatorname{argmax}_t P(t|w_{1m}, G)$$

- What is the probability of a sentence?

$$P(w_{1m}|G)$$

- How to learn the rule probabilities in the grammar G ?

PCFGs - Inside-outside probabilities

Pawan Goyal

CSE, IIT Kharagpur

Week 5: Lecture 4

How to find the most likely parse?: CKY for PCFG

NPTEL

How to find the most likely parse?: CKY for PCFG

a 1	pilot 2	likes 3	flying 4	planes 5

$S \rightarrow NP VP$ [1.0]
 $VP \rightarrow VBG NNS$ [0.1]
 $VP \rightarrow VBZ VP$ [0.1]
 $VP \rightarrow VBZ NP$ [0.3]
 $NP \rightarrow DT NN$ [0.3]
 $NP \rightarrow JJ NNS$ [0.4]
 $DT \rightarrow a$ [0.3]
 $NN \rightarrow pilot$ [0.1]
 $VBZ \rightarrow likes$ [0.4]
 $VBG \rightarrow flying$ [0.5]
 $JJ \rightarrow flying$ [0.1]
 $NNS \rightarrow planes$ [.34]

CKY for PCFG

a 1	pilot 2	likes 3	flying 4	planes 5
DT [0.3]	NP [.009]	-	-	S [1.4688x10 ⁻⁵] S [6.12x10 ⁻⁶]
	NN [0.1]	-	-	-
		VBZ [0.4]	-	VP [.001632] VP [.00068]
			JJ [0.1] VBG [0.5]	NP [.0136] VP [.017]
				NNS [.34]

$S \rightarrow NP VP$ [1.0]
 $VP \rightarrow VBG NNS$ [0.1]
 $VP \rightarrow VBZ VP$ [0.1]
 $VP \rightarrow VBZ NP$ [0.3]
 $NP \rightarrow DT NN$ [0.3]
 $NP \rightarrow JJ NNS$ [0.4]
 $DT \rightarrow a$ [0.3]
 $NN \rightarrow pilot$ [0.1]
 $VBZ \rightarrow likes$ [0.4]
 $VBG \rightarrow flying$ [0.5]
 $JJ \rightarrow flying$ [0.1]
 $NNS \rightarrow planes$ [.34]

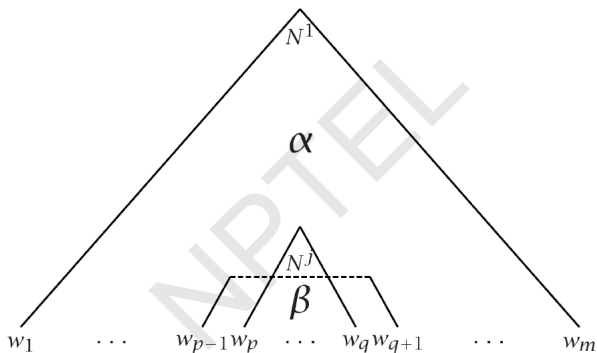
$$0.009 \times 0.00068 \times 1.0 = 6.12 \times 10^{-6}$$

$$P(w_{1m}|G)$$

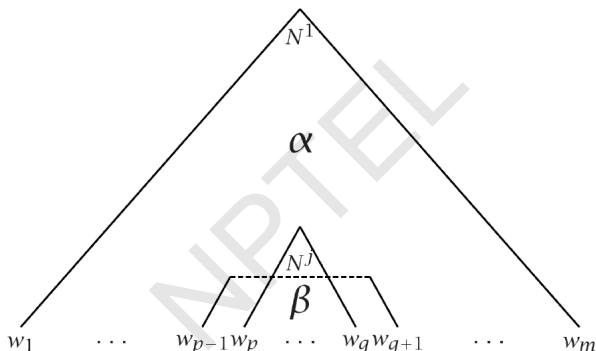
$$P(w_{1m}|G)$$

- In general, simply summing the probabilities of all possible parse trees is not an efficient way to calculate the string probability
- We use *inside algorithm*, a dynamic programming algorithm based on inside probabilities.

Inside and Outside Probabilities



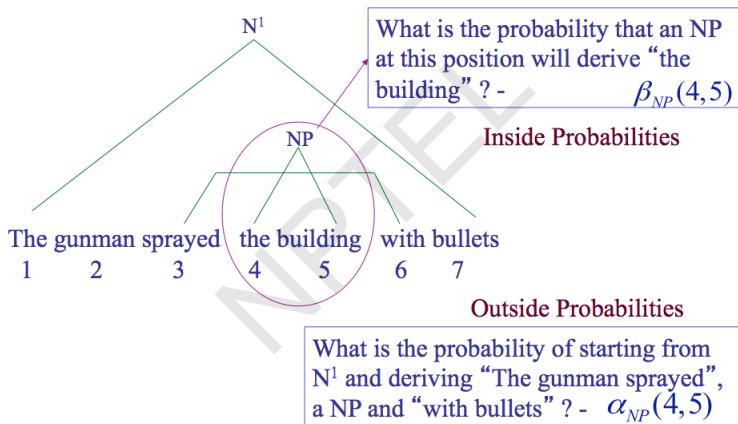
Inside and Outside Probabilities



Outside: $\alpha_j(p, q) = P(w_{1(p-1)}, N^j_{pq}, w_{(q+1)m} | G)$

Inside: $\beta_j(p, q) = P(w_{pq} | N^j_{pq}, G)$

Inside-outside probabilities

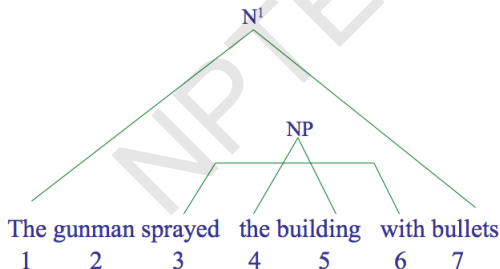


Inside-outside probabilities

$\alpha_{NP}(4,5)$ for "the building"

$= P(\text{The gunman sprayed, } NP_{4,5}, \text{ with bullets} \mid G)$

$\beta_{NP}(4,5)$ for "the building" $= P(\text{the building} \mid NP_{4,5}, G)$



$$\beta_j(p, q) = P(w_{pq} | N_{pq}^j, G)$$

NPTTEL

Inside Probabilities: Base Step

$$\beta_j(p, q) = P(w_{pq} | N_{pq}^j, G)$$

Base case

$$\begin{aligned}\beta_j(k, k) &= P(w_{kk} | N_{kk}^j, G) \\ &= P(N^j \rightarrow w_k | G)\end{aligned}$$

Base case for pre-terminals only

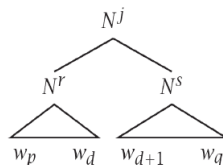
E.g., suppose $N^j = NN$ is being considered and $NN \rightarrow \text{building}$ is one of the rules with probability 0.5

$$\beta_{NN}(5, 5) = P(\text{building} | NN_{5,5}, G) = P(NN_{5,5} \rightarrow \text{building} | G)$$

Inside Probabilities: Induction Step

Assuming Chomsky Normal Form, the first rule must be of the form $N^j \rightarrow N^r N^s$

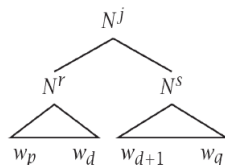
$$\beta_j(p, q) = \sum_{r,s} \sum_{d=p}^{q-1} P(N^j \rightarrow N^r N^s) \beta_r(p, d) \beta_s(d+1, q)$$



Inside Probabilities: Induction Step

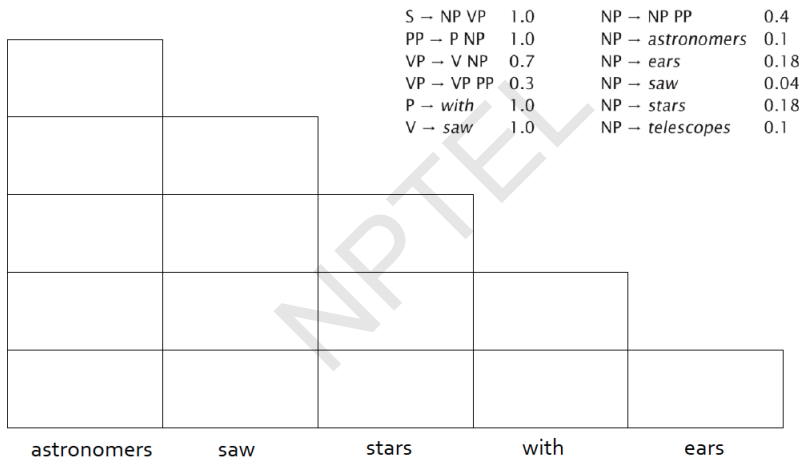
Assuming Chomsky Normal Form, the first rule must be of the form $N^j \rightarrow N^r N^s$

$$\beta_j(p, q) = \sum_{r,s} \sum_{d=p}^{q-1} P(N^j \rightarrow N^r N^s) \beta_r(p, d) \beta_s(d+1, q)$$



- Consider different splits of the words - indicated by d
E.g., *the huge building*
- Consider different non-terminals to be used in the rule:
E.g., $NP \rightarrow DT NN$, $NP \rightarrow DT NNS$

Calculation of inside probabilities



Calculation of inside probabilities

	1	2	3	4	5
1	$\beta_{NP} = 0.1$		$\beta_S = 0.0126$		$\beta_S = 0.0015876$
2		$\beta_{NP} = 0.04$ $\beta_V = 1.0$	$\beta_{VP} = 0.126$		$\beta_{VP} = 0.015876$
3			$\beta_{NP} = 0.18$		$\beta_{NP} = 0.01296$
4				$\beta_P = 1.0$	$\beta_{PP} = 0.18$
5					$\beta_{NP} = 0.18$
	<i>astronomers</i>	<i>saw</i>	<i>stars</i>	<i>with</i>	<i>ears</i>

Outside Probabilities

Compute top-down (after inside probabilities)

NPTEL

Outside Probabilities

Compute top-down (after inside probabilities)

Base Case

NPTEL

Outside Probabilities

Compute top-down (after inside probabilities)

Base Case

$$\alpha_1(1, m) = 1$$

$$\alpha_j(1, m) = 0, j \neq 1$$

Outside Probabilities

Compute top-down (after inside probabilities)

Base Case

$$\alpha_1(1, m) = 1$$

$$\alpha_j(1, m) = 0, j \neq 1$$

Induction

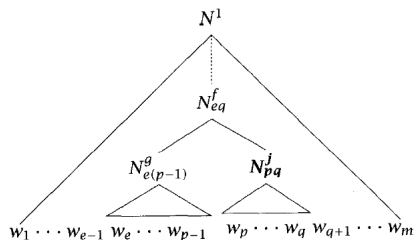
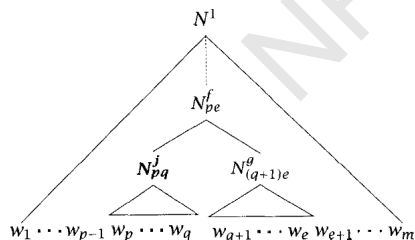
Outside Probabilities

Compute top-down (after inside probabilities)

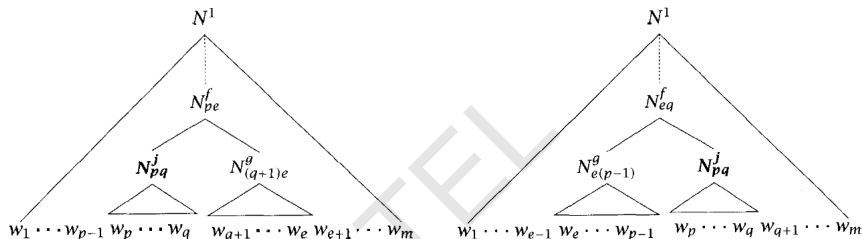
Base Case

$$\alpha_1(1, m) = 1$$
$$\alpha_j(1, m) = 0, j \neq 1$$

Induction



Outside Probabilities: Induction



$$\begin{aligned}
 \alpha_j(p, q) = & \sum_{f, g} \sum_{e=q+1}^m \alpha_f(p, e) P(N^f \rightarrow N^j N^g) \beta_g(q+1, e) \\
 & + \sum_{f, g} \sum_{e=1}^{p-1} \alpha_f(e, q) P(N^f \rightarrow N^g N^j) \beta_g(e, p-1)
 \end{aligned}$$

Product of inside-outside probabilities

$$\alpha_j(p, q)\beta_j(p, q) = P(w_{1(p-1)}, N_{pq}^j, w_{(q+1)m}|G)P(w_{pq}|N_{pq}^j, G) = P(w_{1m}, N_{pq}^j|G)$$

NPTEL

Product of inside-outside probabilities

$$\alpha_j(p, q)\beta_j(p, q) = P(w_{1(p-1)}, N_{pq}^j, w_{(q+1)m} | G) P(w_{pq} | N_{pq}^j, G) = P(w_{1m}, N_{pq}^j | G)$$

The probability of the sentence and that there is some consistent spanning from word p to q is given by:

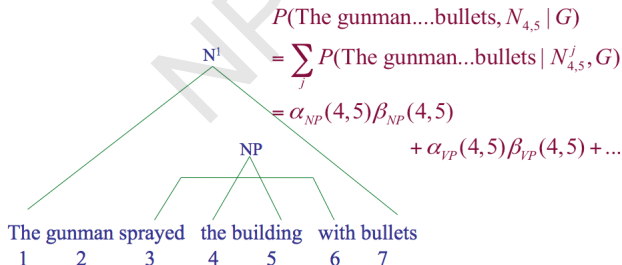
$$P(w_{1m}, N_{pq} | G) = \sum \alpha_j(p, q)\beta_j(p, q) = P(N_1 \rightarrow w_{1m}, N_{pq} \rightarrow w_{pq} | G)$$

Product of inside-outside probabilities

$$\alpha_j(p, q)\beta_j(p, q) = P(w_{1(p-1)}, N_{pq}^j, w_{(q+1)m} | G) P(w_{pq} | N_{pq}^j, G) = P(w_{1m}, N_{pq}^j | G)$$

The probability of the sentence and that there is some consistent spanning from word p to q is given by:

$$P(w_{1m}, N_{pq} | G) = \sum \alpha_j(p, q)\beta_j(p, q) = P(N_1 \rightarrow w_{1m}, N_{pq} \rightarrow w_{pq} | G)$$



Inside-outside probabilities

Pawan Goyal

CSE, IIT Kharagpur

Week 5: Lecture 5

How to get the rule probabilities

Parsed Training Data

You can count!

$$\hat{P}(N^j \rightarrow \delta) = \frac{C(N^j \rightarrow \delta)}{\sum_{\gamma} C(N^j \rightarrow \gamma)}$$

How to get the rule probabilities

Parsed Training Data

You can count!

$$\hat{P}(N^j \rightarrow \delta) = \frac{C(N^j \rightarrow \delta)}{\sum_{\gamma} C(N^j \rightarrow \gamma)}$$

But what if the training data is not available?

i.e. gold standard parse is not known.

How to get the rule probabilities

Parsed Training Data

You can count!

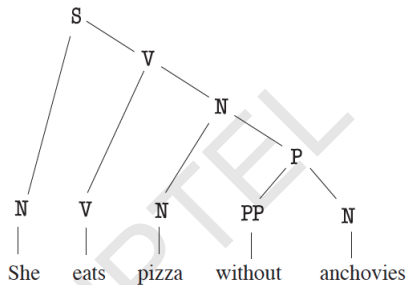
$$\hat{P}(N^j \rightarrow \delta) = \frac{C(N^j \rightarrow \delta)}{\sum_{\gamma} C(N^j \rightarrow \gamma)}$$

But what if the training data is not available?

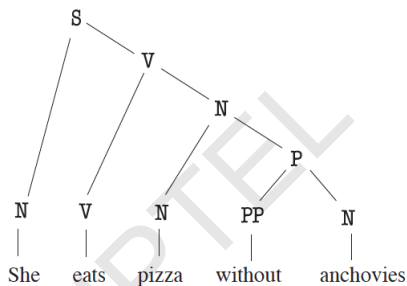
i.e. gold standard parse is not known.

- Underlying CFG is known and we are given a set of sentences
- For each sentence, we can find out all the possible parses
- *Maximize the likelihood of the sentences in the data under the PCFG constraints*

Example data



Example data



Rules of the form $A \rightarrow BC$

$S \rightarrow N V$

$V \rightarrow V N$

$N \rightarrow N P$

$P \rightarrow PP N$

Rules of the form $A \rightarrow w$

$N \rightarrow \text{She}$

$V \rightarrow \text{eats}$

$N \rightarrow \text{pizza}$

$PP \rightarrow \text{without}$

$N \rightarrow \text{anchovies.}$

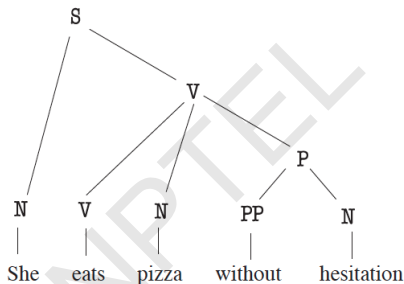
Example data

Is any other parse possible for *She eats pizza without anchovies* syntactically?

NPTEL

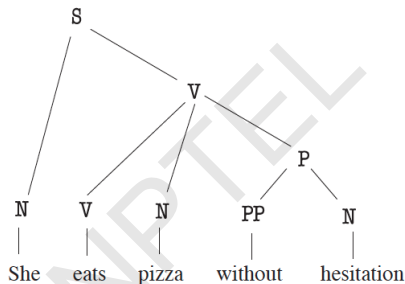
Example data

Is any other parse possible for *She eats pizza without anchovies* syntactically?
Consider *She eats pizza without hesitation*



Example data

Is any other parse possible for *She eats pizza without anchovies* syntactically?
Consider *She eats pizza without hesitation*



New Context-free rules:

$$V \rightarrow V N P$$
$$N \rightarrow \text{hesitation} .$$

Estimating the model parameters

We need to find probabilities such as

- $\phi(S \rightarrow N \ V)$
- $\phi(N \rightarrow pizza)$

NPTEL

Estimating the model parameters

We need to find probabilities such as

- $\phi(S \rightarrow N V)$
- $\phi(N \rightarrow \text{pizza})$

Requirements

For each non-terminal A , the derivation probabilities sum up to 1

$$\sum_{\alpha} \phi(A \rightarrow \alpha) = 1$$

Estimating the model parameters

We need to find probabilities such as

- $\phi(S \rightarrow N V)$
- $\phi(N \rightarrow pizza)$

Requirements

For each non-terminal A , the derivation probabilities sum up to 1

$$\sum_{\alpha} \phi(A \rightarrow \alpha) = 1$$

For the example grammar:

$$\begin{aligned}\phi(N \rightarrow N P) + \phi(N \rightarrow pizza) + \phi(N \rightarrow anchovies) &+ \\ &+ \phi(N \rightarrow hesitation) + \phi(N \rightarrow She) = 1 \\ \phi(V \rightarrow V N) + \phi(V \rightarrow V N P) + \phi(V \rightarrow eats) &= 1\end{aligned}$$

$$\begin{aligned}\phi(S \rightarrow N V) &= 1 \\ \phi(P \rightarrow PP N) &= 1 \\ \phi(PP \rightarrow without) &= 1\end{aligned}$$

Likelihood computation

W_1 = “She eats pizza without anchovies”

W_2 = “She eats pizza without hesitation”.

Likelihood computation

W_1 = “She eats pizza without anchovies”

W_2 = “She eats pizza without hesitation”.

$$\begin{aligned} P_\phi(W_1, T_1) &= \phi(S \rightarrow N V) \phi(V \rightarrow V N) \phi(N \rightarrow N P) \times \\ &\times \phi(P \rightarrow PP N) \phi(N \rightarrow \text{She}) \phi(V \rightarrow \text{eats}) \times \\ &\times \phi(N \rightarrow \text{pizza}) \phi(PP \rightarrow \text{without}) \phi(N \rightarrow \text{anchovies}) \end{aligned}$$

$$\begin{aligned} P_\phi(W_2, T_1) &= \phi(S \rightarrow N V) \phi(V \rightarrow V N P) \phi(P \rightarrow P PP) \times \\ &\times \phi(N \rightarrow \text{She}) \phi(V \rightarrow \text{eats}) \phi(N \rightarrow \text{pizza}) \times \\ &\times \phi(PP \rightarrow \text{without}) \phi(N \rightarrow \text{hesitation}) \end{aligned}$$

Likelihood computation

$$\begin{aligned}P_{\phi}(W_1, T_2) &= \phi(S \rightarrow N V) \phi(V \rightarrow V N P) \phi(P \rightarrow P PP) \times \\&\times \phi(N \rightarrow \text{She}) \phi(V \rightarrow \text{eats}) \phi(N \rightarrow \text{pizza}) \times \\&\times \phi(PP \rightarrow \text{without}) \phi(N \rightarrow \text{anchovies})\end{aligned}$$

$$\begin{aligned}P_{\phi}(W_2, T_1) &= \phi(S \rightarrow N V) \phi(V \rightarrow V N) \phi(N \rightarrow N P) \times \\&\times \phi(P \rightarrow PP N) \phi(N \rightarrow \text{She}) \phi(V \rightarrow \text{eats}) \times \\&\times \phi(N \rightarrow \text{pizza}) \phi(PP \rightarrow \text{without}) \phi(N \rightarrow \text{hesitation})\end{aligned}$$

Likelihood computation

$$\begin{aligned}P_{\phi}(W_1, T_2) &= \phi(S \rightarrow N V) \phi(V \rightarrow V N P) \phi(P \rightarrow P PP) \times \\&\times \phi(N \rightarrow \text{She}) \phi(V \rightarrow \text{eats}) \phi(N \rightarrow \text{pizza}) \times \\&\times \phi(PP \rightarrow \text{without}) \phi(N \rightarrow \text{anchovies})\end{aligned}$$

$$\begin{aligned}P_{\phi}(W_2, T_1) &= \phi(S \rightarrow N V) \phi(V \rightarrow V N) \phi(N \rightarrow N P) \times \\&\times \phi(P \rightarrow PP N) \phi(N \rightarrow \text{She}) \phi(V \rightarrow \text{eats}) \times \\&\times \phi(N \rightarrow \text{pizza}) \phi(PP \rightarrow \text{without}) \phi(N \rightarrow \text{hesitation})\end{aligned}$$

Likelihood of the corpus

Probability of a sentence W : $P_{\phi}(W) = \sum_T P_{\phi}(W, T)$

Likelihood computation

$$\begin{aligned}P_{\phi}(W_1, T_2) &= \phi(S \rightarrow N V) \phi(V \rightarrow V N P) \phi(P \rightarrow P PP) \times \\&\times \phi(N \rightarrow She) \phi(V \rightarrow eats) \phi(N \rightarrow pizza) \times \\&\times \phi(PP \rightarrow without) \phi(N \rightarrow anchovies)\end{aligned}$$

$$\begin{aligned}P_{\phi}(W_2, T_1) &= \phi(S \rightarrow N V) \phi(V \rightarrow V N) \phi(N \rightarrow N P) \times \\&\times \phi(P \rightarrow PP N) \phi(N \rightarrow She) \phi(V \rightarrow eats) \times \\&\times \phi(N \rightarrow pizza) \phi(PP \rightarrow without) \phi(N \rightarrow hesitation)\end{aligned}$$

Likelihood of the corpus

Probability of a sentence W : $P_{\phi}(W) = \sum_T P_{\phi}(W, T)$

If the training data comprises of sentences W_1, W_2, \dots, W_N , then the likelihood is

$$L(\phi) = P_{\phi}(W_1)P_{\phi}(W_2) \cdots P_{\phi}(W_N)$$

Approach

Starting at some initial parameters ϕ , re-estimate to obtain new parameters ϕ' for which $L(\phi') \geq L(\phi)$. Repeat until convergence

Parameter Estimation

Given some rule probabilities ϕ and training corpus $W_1, W_2 \dots W_n$, the new parameters are obtained as:

$$\phi'(\mathbf{A} \rightarrow \mathbf{B} \mathbf{C}) = \frac{\text{count}(\mathbf{A} \rightarrow \mathbf{B} \mathbf{C})}{\sum_{\alpha} \text{count}(\mathbf{A} \rightarrow \alpha)}$$

$$\phi'(\mathbf{A} \rightarrow w) = \frac{\text{count}(\mathbf{A} \rightarrow w)}{\sum_{\alpha} \text{count}(\mathbf{A} \rightarrow \alpha)}$$

What is $\text{count}(\cdot)$?

Parameter Estimation

Given some rule probabilities ϕ and training corpus $W_1, W_2 \dots W_n$, the new parameters are obtained as:

$$\phi'(\mathbf{A} \rightarrow \mathbf{B} \mathbf{C}) = \frac{\text{count}(\mathbf{A} \rightarrow \mathbf{B} \mathbf{C})}{\sum_{\alpha} \text{count}(\mathbf{A} \rightarrow \alpha)}$$

$$\phi'(\mathbf{A} \rightarrow w) = \frac{\text{count}(\mathbf{A} \rightarrow w)}{\sum_{\alpha} \text{count}(\mathbf{A} \rightarrow \alpha)}$$

What is $\text{count}(\cdot)$?

$$\text{count}(\mathbf{A} \rightarrow \mathbf{B} \mathbf{C}) = \sum_{i=1}^N c_{\phi}(\mathbf{A} \rightarrow \mathbf{B} \mathbf{C}, W_i)$$

$$\text{count}(\mathbf{A} \rightarrow w) = \sum_{i=1}^N c_{\phi}(\mathbf{A} \rightarrow w, W_i)$$

Parameter Estimation

Given some rule probabilities ϕ and training corpus $W_1, W_2 \dots W_n$, the new parameters are obtained as:

$$\phi'(A \rightarrow B C) = \frac{\text{count}(A \rightarrow B C)}{\sum_{\alpha} \text{count}(A \rightarrow \alpha)}$$

$$\phi'(A \rightarrow w) = \frac{\text{count}(A \rightarrow w)}{\sum_{\alpha} \text{count}(A \rightarrow \alpha)}$$

What is $\text{count}(\cdot)$?

$$\text{count}(A \rightarrow B C) = \sum_{i=1}^N c_{\phi}(A \rightarrow B C, W_i)$$

$$\text{count}(A \rightarrow w) = \sum_{i=1}^N c_{\phi}(A \rightarrow w, W_i)$$

$c_{\phi}(A \rightarrow \alpha, W_i)$ is the expected number of times $(A \rightarrow \alpha)$ is used in generating the sentence W_i , when the rule probabilities are given by ϕ .

Computing Expected counts

Inside probabilities

The nonterminal A derives the string of words $w_i, \dots w_j$ in the sentence :

$$\beta_{ij}(A) = P_{\phi}(A \Rightarrow^* w_i \dots w_j)$$

NPTEL

Computing Expected counts

Inside probabilities

The nonterminal A derives the string of words $w_i, \dots w_j$ in the sentence :

$$\beta_{ij}(A) = P_{\phi}(A \Rightarrow^* w_i \dots w_j)$$

Outside probabilities

Beginning with the start symbol S we can derive the string

$$w_1 \dots w_{i-1} A w_{j+1} \dots w_n : \alpha_{ij}(A) = P_{\phi}(S \Rightarrow^* w_1 \dots w_{i-1} A w_{j+1} \dots w_n)$$

Computing Expected counts

Inside probabilities

The nonterminal A derives the string of words w_i, \dots, w_j in the sentence :

$$\beta_{ij}(A) = P_{\phi}(A \Rightarrow^* w_i \dots w_j)$$

Outside probabilities

Beginning with the start symbol S we can derive the string

$$w_1 \dots w_{i-1} A w_{j+1} \dots w_n : \alpha_{ij}(A) = P_{\phi}(S \Rightarrow^* w_1 \dots w_{i-1} A w_{j+1} \dots w_n)$$

Expected count

$$c_{\phi}(A \rightarrow BC, W) = \frac{\phi(A \rightarrow BC)}{P_{\phi}(W)} \sum_{1 \leq i \leq j \leq k \leq n} \alpha_{ik}(A) \beta_{ij}(B) \beta_{j+1,k}(C)$$

$$c_{\phi}(A \rightarrow w, W) = \frac{\phi(A \rightarrow w)}{P_{\phi}(W)} \sum_{1 \leq i \leq n} \alpha_{ii}(A)$$

And how to compute inside-outside probabilities

Inductively, as discussed earlier

$$\beta_{ii}(A) = \phi(A \rightarrow w_i)$$

$$\alpha_{1n}(S) = 1$$