Natural Language Models and Interfaces

BSc Artificial Intelligence

Lecturer: Wilker Aziz
Institute for Logic, Language, and Computation

2018, week 5, lecture a

NLMI

Trees and grammars

Context-free grammars

Probabilistic context-free grammars

Modelling language so far

Bag-of-word models (or unigram LMs)

- ignore word order entirely

Modelling language so far

Bag-of-word models (or unigram LMs)

- ignore word order entirely

n-gram models

- capture a shortened fixed-length history

Modelling language so far

Bag-of-word models (or unigram LMs)

- ignore word order entirely

n-gram models

- capture a shortened fixed-length history

HMM models

- capture a shortened fixed-length history
- by abstracting away from word form through word classes

Long-distance dependencies

The form of one word often depends on (agrees with) another even when arbitrarily long material intervenes

- Sam sleeps soundly
- Dogs sleep soundly

Adapted from T. Deoskar

Long-distance dependencies

The form of one word often depends on (agrees with) another even when arbitrarily long material intervenes

- Sam sleeps soundly
- Dogs sleep soundly
- Sam, who is my cousin, sleeps soundly
- Dogs often play around my house and then sleep soundly

Adapted from T. Deoskar

Long-distance dependencies

The form of one word often depends on (agrees with) another even when arbitrarily long material intervenes

- Sam sleeps soundly
- Dogs sleep soundly
- Sam, who is my cousin, sleeps soundly
- Dogs often play around my house and then sleep soundly
- Sam, the man with red hair who is my cousin, sleeps soundly

Adapted from T. Deoskar

Long-distance dependencies

The form of one word often depends on (agrees with) another even when arbitrarily long material intervenes

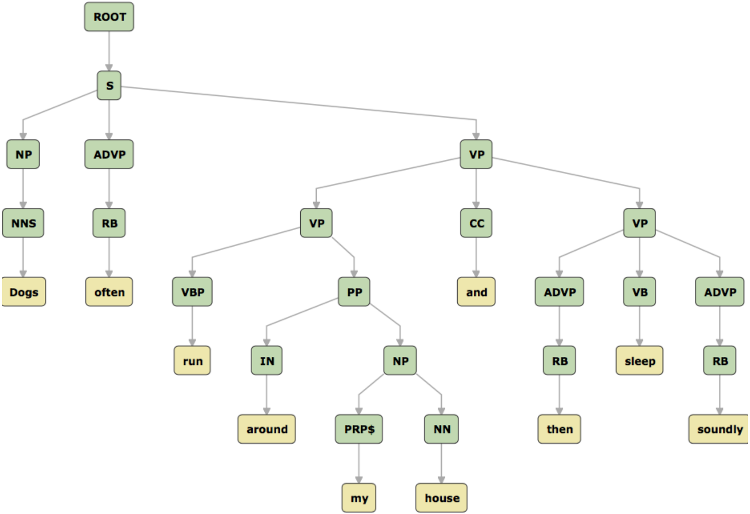
- Sam sleeps soundly
- Dogs sleep soundly
- Sam, who is my cousin, sleeps soundly
- Dogs often play around my house and then sleep soundly
- Sam, the man with red hair who is my cousin, sleeps soundly

We want models that can capture these dependencies

- and are less sensitive to distance in linear order

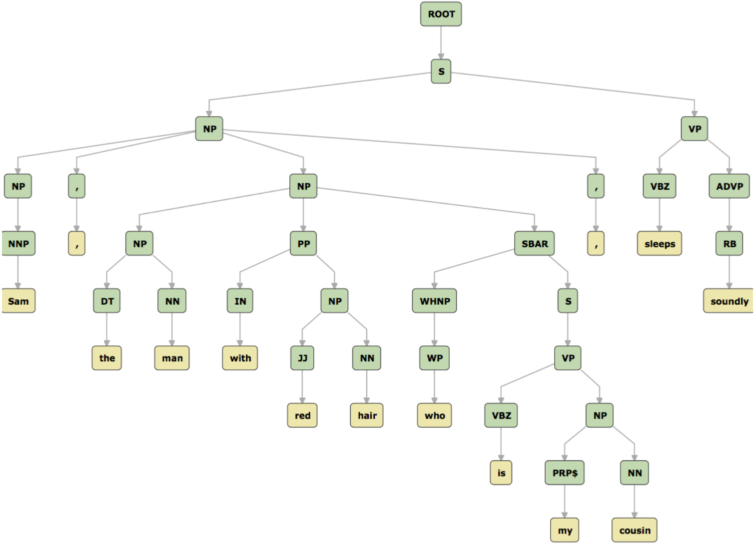
Adapted from T. Deoskar

What if we organise words and phrases in a tree?



CoreNLP

What if we organise words and phrases in a tree?



CoreNLP

Phrases

Words are organised into groups (phrases) which function as a unit

- POS categories indicate which words are substitutable.
  e.g., substituting adjectives
  - I saw a red cat
  - I saw a former cat
  - I saw a sleepy cat
- Phrasal categories indicate which phrases are substitutable
  e.g., substituting noun phrase
  - Dogs sleep soundly
  - My next-door neighbours sleep soundly
  - Green ideas sleep soundly

Phrasal categories: noun phrase (NP), verb phrase (VP), prepositional phrase (PP), etc.

Adapted from T. Deoskar

Heads and Phrases

The class that a word belongs to is closely linked to the name of the phrase it customarily appears in.

- In a X-phrase (e.g. NP), the key occurrence of X (e.g. N) is called the head.

Adapted from T. Deoskar

Heads and Phrases

The class that a word belongs to is closely linked to the name of the phrase it customarily appears in.

- In a X-phrase (e.g. NP), the key occurrence of X (e.g. N) is called the head.

English NPs are commonly of the form

- (Det) Adj* **Noun** (PP — RelClause)*
  NP: the angry duck that tried to bite me ; **head: duck**

Adapted from T. Deoskar

Heads and Phrases

The class that a word belongs to is closely linked to the name of the phrase it customarily appears in.

- In a X-phrase (e.g. NP), the key occurrence of X (e.g. N) is called the head.

English NPs are commonly of the form

- (Det) Adj* **Noun** (PP — RelClause)*
  NP: the angry duck that tried to bite me ; **head: duck**

VPs are commonly of the form

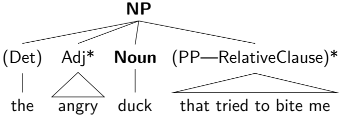
- (Aux) Adv* **Verb** Arg* Adjunct*
  VP: usually eats pasta for dinner ; **head : eat**

Adapted from T. Deoskar

Heads and Phrases

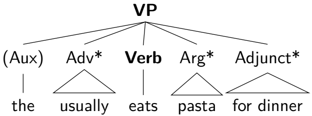
English NPs are commonly of the form

- (Det) Adj* **Noun** (PP — RelClause)*



VPs are commonly of the form

- (Aux) Adv* **Verb** Arg* Adjunct*



Theories of Syntax

A theory of syntax should explain which sentences are well-formed (grammatical) and which are not

- *well-formed* is distinct from *meaningful*.
- Example from Chomsky
  Colorless green ideas sleep furiously

Adapted from T. Deoskar

Theories of Syntax

A theory of syntax should explain which sentences are well-formed (grammatical) and which are not

- *well-formed* is distinct from *meaningful*.
- Example from Chomsky
  Colorless green ideas sleep furiously
- However, the reason we care about syntax is mainly for interpreting meaning

Adapted from T. Deoskar

Desirable properties of a grammar

Chomsky specified two properties that make a grammar "interesting and satisfying"

- It should be a finite specification of the strings of the language, rather than a list of its sentences.
- It should be revealing, in allowing strings to be associated with meaning (semantics) in a systematic way.

Adapted from T. Deoskar

Desirable properties of a grammar

Chomsky specified two properties that make a grammar "interesting and satisfying"

- It should be a finite specification of the strings of the language, rather than a list of its sentences.
- It should be revealing, in allowing strings to be associated with meaning (semantics) in a systematic way.

We can add another desirable property

- It should capture structural and distributional properties of the language
  e.g. where heads of phrases are located
  e.g. how a sentence transforms into a question
  e.g. which phrases can move around the sentence

Adapted from T. Deoskar

Desirable properties of a grammar

- Context-free grammars (CFGs) provide a pretty good approximation
- Some features of NLs are more easily captured using *mildly context-sensitive* grammars
  - Combinatory Categorial Grammar (CCG)
  - Lexicalised Tree Adjoining Grammar (LTAG)

Adapted from T. Deoskar

A small fragment of English

Let's say we want to capture in a grammar the structural and distributional properties that give rise to sentences like this:

A duck walked in the park.	NP,V,PP
The man walked with a duck.	NP,V,PP
You made a duck.	Pro,V,NP
You made her duck.	? Pro,V,NP
A man with a telescope saw you.	NP,PP,V,Pro
A man saw you with a telescope.	NP,V,Pro,PP
You saw a man with a telescope.	Pro,V,NP,PP

- write lexical rules that generate the words appearing in them
- write grammatical rules that generate these phrase structures

Adapted from T. Deoskar

Grammar for the small fragment of English

Grammar G1 generates the sentences on the previous slide:

Grammatical rules	**Lexical rules**
S → NP VP	Det → a | the | her (determiners)
NP → Det N	N → man | park | duck | telescope (nouns)
NP → Det N PP	Pro → you (pronoun)
NP → Pro	V → saw | walked | made (verbs)
VP → V NP PP	Prep → in | with | for (prepositions)
VP → V NP	
VP → V	
PP → Prep NP	

Does G1 produce a finite or an infinite number of sentences?

Adapted from T. Deoskar

Recursion

Recursion in a grammar makes it possible to generate an infinite number of sentences

- Direct recursion: a non-terminal on the LHS of a rule also appears on its RHS
  VP $\rightarrow$ VP Conj VP
  Conj $\rightarrow$ and — or
- Indirect recursion: some non-terminal can be expanded (in several steps) to a sequence of symbols containing that non-terminal
  NP $\rightarrow$ Det N PP
  PP $\rightarrow$ Prep NP

NLMI

Context-Free Grammar

A rewriting system with two types of symbols

- Terminals (or *constants*): words
- Nonterminals (or *variables*): phrasal categories
  e.g. S, NP, VP
  with S being the start symbol

Rules of the form X $\rightarrow \beta$
where β is any string of nonterminals and terminals
indicate that X can be replaced by β anywhere where X occurs

CFG example

-

S $\rightarrow$ NP VP	**(Sentences)**
NP $\rightarrow$ D N | Pro | PropN	**(Noun phrases)**
D $\rightarrow$ PosPro | Art | NP 's	**(Determiners)**
VP $\rightarrow$ Vi | Vt NP | Vp NP VP	**(Verb phrases)**
Pro $\rightarrow$ i | we | you | he | she | him | her	**(Pronouns)**
PosPro $\rightarrow$ my | our | your | his | her	**(Possessive pronouns)**
PropN $\rightarrow$ Robin | Jo	**(Proper nouns)**
Art $\rightarrow$ a | an | the	**(Articles)**
N $\rightarrow$ man | duck | saw | park | telescope	**(Nouns)**
Vi $\rightarrow$ sleep | run | duck	**(Intransitive verbs)**
Vt $\rightarrow$ eat | break | see | saw	**(Transitive verbs)**
Vp $\rightarrow$ see | saw | heard	**(Verbs with NP VP args)**

CFGs formally

Let Σ be a finite set of terminal symbols (aka words)
e.g. generically we write $x \in \Sigma$

CFGs formally

Let Σ be a finite set of terminal symbols (aka words)
e.g. generically we write $\mathsf{x} \in \Sigma$

Let $\mathcal{V}$ be a finite set of nonterminal symbols (aka variables)
where $\Sigma \cup \mathcal{V} = \emptyset$ and $\mathsf{S} \in \mathcal{V}$ is a distinguished start symbol

CFGs formally

Let Σ be a finite set of terminal symbols (aka words)
e.g. generically we write $x \in \Sigma$

Let $\mathcal{V}$ be a finite set of nonterminal symbols (aka variables)
where $\Sigma \cup \mathcal{V} = \emptyset$ and $S \in \mathcal{V}$ is a distinguished start symbol

Let $\beta \in (\Sigma \cup \mathcal{V})^*$ be a (possibly empty) string of terminal and nonterminal symbols
and let ϵ denote the empty string

CFGs formally

Let Σ be a finite set of terminal symbols (aka words)
e.g. generically we write $\mathsf{x} \in \Sigma$

Let $\mathcal{V}$ be a finite set of nonterminal symbols (aka variables)
where $\Sigma \cup \mathcal{V} = \emptyset$ and $\mathsf{S} \in \mathcal{V}$ is a distinguished start symbol

Let $\beta \in (\Sigma \cup \mathcal{V})^*$ be a (possibly empty) string of terminal and nonterminal symbols
and let ϵ denote the empty string

Let $\mathcal{R} \subseteq \mathcal{V} \times (\Sigma \cup \mathcal{V})^*$ be a finite set of rules of the form
$\mathsf{X} \to \beta$ where $\mathsf{X} \in \mathcal{V}$ and $\beta \in (\Sigma \cup \mathcal{V})^*$

CFGs formally

Let Σ be a finite set of terminal symbols (aka words)
e.g. generically we write $\mathsf{x} \in \Sigma$

Let $\mathcal{V}$ be a finite set of nonterminal symbols (aka variables)
where $\Sigma \cup \mathcal{V} = \emptyset$ and $\mathsf{S} \in \mathcal{V}$ is a distinguished start symbol

Let $\beta \in (\Sigma \cup \mathcal{V})^*$ be a (possibly empty) string of terminal and nonterminal symbols
and let ϵ denote the empty string

Let $\mathcal{R} \subseteq \mathcal{V} \times (\Sigma \cup \mathcal{V})^*$ be a finite set of rules of the form
$\mathsf{X} \to \beta$ where $\mathsf{X} \in \mathcal{V}$ and $\beta \in (\Sigma \cup \mathcal{V})^*$

A CFG is the tuple $\mathfrak{G} = \langle \Sigma, \mathcal{V}, \mathsf{S}, \mathcal{R} \rangle$

CFG terminology

The number of symbols on the RHS is the arity of the rule

- unary: $\mathsf{X} \to \mathsf{Y}$
- binary: $\mathsf{X} \to \mathsf{Y}\,\mathsf{Z}$
- n-ary: $\mathsf{X} \to \mathsf{X}_1 \cdots \mathsf{X}_n$
- if the longest rule has arity a we say the grammar has arity a

CFG terminology

The number of symbols on the RHS is the arity of the rule

- unary: $\mathsf{X} \to \mathsf{Y}$
- binary: $\mathsf{X} \to \mathsf{Y}\,\mathsf{Z}$
- n-ary: $\mathsf{X} \to \mathsf{X}_1 \cdots \mathsf{X}_n$
- if the longest rule has arity a we say the grammar has arity a

Pre-terminal rules

- unary rules such as $\mathsf{X} \to \mathsf{x}$
  where $\mathsf{x} \in \Sigma$ is a terminal

CFG terminology

The number of symbols on the RHS is the arity of the rule

- unary: $\mathsf{X} \to \mathsf{Y}$
- binary: $\mathsf{X} \to \mathsf{Y}\,\mathsf{Z}$
- n-ary: $\mathsf{X} \to \mathsf{X}_1 \cdots \mathsf{X}_n$
- if the longest rule has arity a we say the grammar has arity a

Pre-terminal rules

- unary rules such as $\mathsf{X} \to \mathsf{x}$
  where $\mathsf{x} \in \Sigma$ is a terminal

How many?

- pre-terminal rules?

CFG terminology

The number of symbols on the RHS is the arity of the rule

- unary: X $\rightarrow$ Y
- binary: X $\rightarrow$ Y Z
- n-ary: X $\rightarrow$ $\mathsf{X}_1 \cdots \mathsf{X}_n$
- if the longest rule has arity a we say the grammar has arity a

Pre-terminal rules

- unary rules such as X $\rightarrow$ x
  where x $\in \Sigma$ is a terminal

How many?

- pre-terminal rules? $O(|\mathcal{V}| \times |\Sigma|)$

CFG terminology

The number of symbols on the RHS is the arity of the rule

- unary: X $\rightarrow$ Y
- binary: X $\rightarrow$ Y Z
- n-ary: X $\rightarrow$ X$_1 \cdots$ X$_n$
- if the longest rule has arity a we say the grammar has arity a

Pre-terminal rules

- unary rules such as X $\rightarrow$ x
  where x $\in \Sigma$ is a terminal

How many?

- pre-terminal rules? $O(|\mathcal{V}| \times |\Sigma|)$
- phrase rules?

CFG terminology

The number of symbols on the RHS is the arity of the rule

- unary: $\mathsf{X} \to \mathsf{Y}$
- binary: $\mathsf{X} \to \mathsf{Y}\,\mathsf{Z}$
- n-ary: $\mathsf{X} \to \mathsf{X}_1 \cdots \mathsf{X}_n$
- if the longest rule has arity a we say the grammar has arity a

Pre-terminal rules

- unary rules such as $\mathsf{X} \to \mathsf{x}$
  where $\mathsf{x} \in \Sigma$ is a terminal

How many?

- pre-terminal rules? $O(|\mathcal{V}| \times |\Sigma|)$
- phrase rules? $O(|\mathcal{V}| \times |\Sigma \cup \mathcal{V}|^a)$

Derivation

We can use CFGs to derive strings

A derivation is a sequence of strings

- we start from the string $\langle \mathsf{S} \rangle$
- and at each step we rewrite the leftmost nonterminal X by application of a rule $\mathsf{X} \rightarrow \beta$
- until only terminals remain
  which we denote $\mathsf{S} \overset{*}{\Rightarrow} \mathsf{x}_1 \cdots \mathsf{x}_n$

Derivation

We can use CFGs to derive strings

A derivation is a sequence of strings

- we start from the string $\langle \mathsf{S} \rangle$
- and at each step we rewrite the leftmost nonterminal X by application of a rule $\mathsf{X} \rightarrow \beta$
- until only terminals remain
  which we denote $\mathsf{S} \overset{*}{\Rightarrow} \mathsf{x}_1 \cdots \mathsf{x}_n$

Example

1. $\langle \mathsf{S} \rangle$

Derivation

We can use CFGs to derive strings

A derivation is a sequence of strings

- we start from the string $\langle \mathsf{S} \rangle$
- and at each step we rewrite the leftmost nonterminal X by application of a rule $\mathsf{X} \to \beta$
- until only terminals remain
  which we denote $\mathsf{S} \overset{*}{\Rightarrow} \mathsf{x}_1 \cdots \mathsf{x}_n$

Example

1. $\langle \mathsf{S} \rangle$
2. $\langle \mathsf{NP\ VP} \rangle$

Derivation

We can use CFGs to derive strings

A derivation is a sequence of strings

- we start from the string $\langle \mathrm{S} \rangle$
- and at each step we rewrite the leftmost nonterminal X by application of a rule $\mathrm{X} \to \beta$
- until only terminals remain
  which we denote $\mathrm{S} \overset{*}{\Rightarrow} \mathrm{x}_1 \cdots \mathrm{x}_n$

Example

1. $\langle \mathrm{S} \rangle$
2. $\langle \mathrm{NP\ VP} \rangle$
3. $\langle \mathrm{D\ N\ VP} \rangle$

Derivation

We can use CFGs to derive strings

A derivation is a sequence of strings

- we start from the string $\langle \mathsf{S} \rangle$
- and at each step we rewrite the leftmost nonterminal X by application of a rule $\mathsf{X} \rightarrow \beta$
- until only terminals remain
  which we denote $\mathsf{S} \overset{*}{\Rightarrow} \mathsf{x}_1 \cdots \mathsf{x}_n$

Example

1. $\langle \mathsf{S} \rangle$
2. $\langle \mathsf{NP\ VP} \rangle$
3. $\langle \mathsf{D\ N\ VP} \rangle$
4. $\langle \mathsf{the\ N\ VP} \rangle$

Derivation

We can use CFGs to derive strings

A derivation is a sequence of strings

- we start from the string $\langle \mathsf{S} \rangle$
- and at each step we rewrite the leftmost nonterminal X by application of a rule $\mathsf{X} \rightarrow \beta$
- until only terminals remain
  which we denote $\mathsf{S} \stackrel{*}{\Rightarrow} \mathsf{x}_1 \cdots \mathsf{x}_n$

Example

1. $\langle$S$\rangle$
2. $\langle$NP VP$\rangle$
3. $\langle$D N VP$\rangle$
4. $\langle$the N VP$\rangle$
5. $\langle$the dog VP$\rangle$

Derivation

We can use CFGs to derive strings

A derivation is a sequence of strings

- we start from the string $\langle \text{S} \rangle$
- and at each step we rewrite the leftmost nonterminal X by application of a rule $\text{X} \rightarrow \beta$
- until only terminals remain
  which we denote $\text{S} \overset{*}{\Rightarrow} \text{x}_1 \cdots \text{x}_n$

Example

1. $\langle \text{S} \rangle$
2. $\langle \text{NP VP} \rangle$
3. $\langle \text{D N VP} \rangle$
4. $\langle \text{the N VP} \rangle$
5. $\langle \text{the dog VP} \rangle$
6. $\langle \text{the dog V} \rangle$

Derivation

We can use CFGs to derive strings

A derivation is a sequence of strings

- we start from the string $\langle \mathsf{S} \rangle$
- and at each step we rewrite the leftmost nonterminal X by application of a rule $\mathsf{X} \to \beta$
- until only terminals remain

  which we denote $\mathsf{S} \overset{*}{\Rightarrow} \mathsf{x}_1 \cdots \mathsf{x}_n$

Example

1. $\langle$S$\rangle$
2. $\langle$NP VP$\rangle$
3. $\langle$D N VP$\rangle$
4. $\langle$the N VP$\rangle$
5. $\langle$the dog VP$\rangle$
6. $\langle$the dog V$\rangle$
7. $\langle$the dog barks$\rangle$

Derivation - examples

Example

1. $\langle S \rangle$

Example

Derivation - examples

Example

1. $\langle \text{S} \rangle$
2. $\langle \text{NP VP} \rangle$

Example

Derivation - examples

Example

1. $\langle \text{S} \rangle$
2. $\langle \text{NP VP} \rangle$
3. $\langle \text{D N VP} \rangle$

Example

Derivation - examples

Example

1. $\langle \text{S} \rangle$
2. $\langle \text{NP VP} \rangle$
3. $\langle \text{D N VP} \rangle$
4. $\langle \text{the N VP} \rangle$

Example

Derivation - examples

Example

1. $\langle \text{S} \rangle$
2. $\langle \text{NP VP} \rangle$
3. $\langle \text{D N VP} \rangle$
4. $\langle \text{the N VP} \rangle$
5. $\langle \text{the dog VP} \rangle$

Example

Derivation - examples

Example

1. $\langle$S$\rangle$
2. $\langle$NP VP$\rangle$
3. $\langle$D N VP$\rangle$
4. $\langle$the N VP$\rangle$
5. $\langle$the dog VP$\rangle$
6. $\langle$the dog V$\rangle$

Example

Derivation - examples

Example

1. $\langle$S$\rangle$
2. $\langle$NP VP$\rangle$
3. $\langle$D N VP$\rangle$
4. $\langle$the N VP$\rangle$
5. $\langle$the dog VP$\rangle$
6. $\langle$the dog V$\rangle$
7. $\langle$the dog runs$\rangle$

Example

Derivation - examples

Example

1. $\langle \text{S} \rangle$
2. $\langle \text{NP VP} \rangle$
3. $\langle \text{D N VP} \rangle$
4. $\langle \text{the N VP} \rangle$
5. $\langle \text{the dog VP} \rangle$
6. $\langle \text{the dog V} \rangle$
7. $\langle \text{the dog runs} \rangle$

Example

1. $\langle \text{S} \rangle$

Derivation - examples

Example

1. $\langle \text{S} \rangle$
2. $\langle \text{NP VP} \rangle$
3. $\langle \text{D N VP} \rangle$
4. $\langle \text{the N VP} \rangle$
5. $\langle \text{the dog VP} \rangle$
6. $\langle \text{the dog V} \rangle$
7. $\langle \text{the dog runs} \rangle$

Example

1. $\langle \text{S} \rangle$
2. $\langle \text{NP VP} \rangle$

Derivation - examples

Example

1. $\langle \text{S} \rangle$
2. $\langle \text{NP VP} \rangle$
3. $\langle \text{D N VP} \rangle$
4. $\langle \text{the N VP} \rangle$
5. $\langle \text{the dog VP} \rangle$
6. $\langle \text{the dog V} \rangle$
7. $\langle \text{the dog runs} \rangle$

Example

1. $\langle \text{S} \rangle$
2. $\langle \text{NP VP} \rangle$
3. $\langle \text{N VP} \rangle$

Derivation - examples

Example

1. $\langle$S$\rangle$
2. $\langle$NP VP$\rangle$
3. $\langle$D N VP$\rangle$
4. $\langle$the N VP$\rangle$
5. $\langle$the dog VP$\rangle$
6. $\langle$the dog V$\rangle$
7. $\langle$the dog runs$\rangle$

Example

1. $\langle$S$\rangle$
2. $\langle$NP VP$\rangle$
3. $\langle$N VP$\rangle$
4. $\langle$cats VP$\rangle$

Derivation - examples

Example

1. $\langle S \rangle$
2. $\langle NP\ VP \rangle$
3. $\langle D\ N\ VP \rangle$
4. $\langle \text{the}\ N\ VP \rangle$
5. $\langle \text{the dog}\ VP \rangle$
6. $\langle \text{the dog}\ V \rangle$
7. $\langle \text{the dog runs} \rangle$

Example

1. $\langle S \rangle$
2. $\langle NP\ VP \rangle$
3. $\langle N\ VP \rangle$
4. $\langle \text{cats}\ VP \rangle$
5. $\langle \text{cats}\ V \rangle$

Derivation - examples

Example

1. $\langle$S$\rangle$
2. $\langle$NP VP$\rangle$
3. $\langle$D N VP$\rangle$
4. $\langle$the N VP$\rangle$
5. $\langle$the dog VP$\rangle$
6. $\langle$the dog V$\rangle$
7. $\langle$the dog runs$\rangle$

Example

1. $\langle$S$\rangle$
2. $\langle$NP VP$\rangle$
3. $\langle$N VP$\rangle$
4. $\langle$cats VP$\rangle$
5. $\langle$cats V$\rangle$
6. $\langle$cats run$\rangle$

Derivation - more examples

1. $\langle \mathsf{S} \rangle$

Derivation - more examples

1. $\langle \mathsf{S} \rangle$
2. $\langle \mathsf{NP\ VP} \rangle$

Derivation - more examples

1. $\langle$S$\rangle$
2. $\langle$NP VP$\rangle$
3. $\langle$NP CC NP VP$\rangle$

Derivation - more examples

1. $\langle \text{S} \rangle$
2. $\langle \text{NP VP} \rangle$
3. $\langle \text{NP CC NP VP} \rangle$
4. $\langle \text{N CC NP VP} \rangle$

Derivation - more examples

1. $\langle \text{S} \rangle$
2. $\langle \text{NP VP} \rangle$
3. $\langle \text{NP CC NP VP} \rangle$
4. $\langle \text{N CC NP VP} \rangle$
5. $\langle \text{cats CC NP VP} \rangle$

Derivation - more examples

1. $\langle$S$\rangle$
2. $\langle$NP VP$\rangle$
3. $\langle$NP CC NP VP$\rangle$
4. $\langle$N CC NP VP$\rangle$
5. $\langle$cats CC NP VP$\rangle$
6. $\langle$cats and NP VP$\rangle$

Derivation - more examples

1. $\langle$S$\rangle$
2. $\langle$NP VP$\rangle$
3. $\langle$NP CC NP VP$\rangle$
4. $\langle$N CC NP VP$\rangle$
5. $\langle$cats CC NP VP$\rangle$
6. $\langle$cats and NP VP$\rangle$
7. $\langle$cats and N VP$\rangle$

Derivation - more examples

1. $\langle$S$\rangle$
2. $\langle$NP VP$\rangle$
3. $\langle$NP CC NP VP$\rangle$
4. $\langle$N CC NP VP$\rangle$
5. $\langle$cats CC NP VP$\rangle$
6. $\langle$cats and NP VP$\rangle$
7. $\langle$cats and N VP$\rangle$
8. $\langle$cats and dogs VP$\rangle$

Derivation - more examples

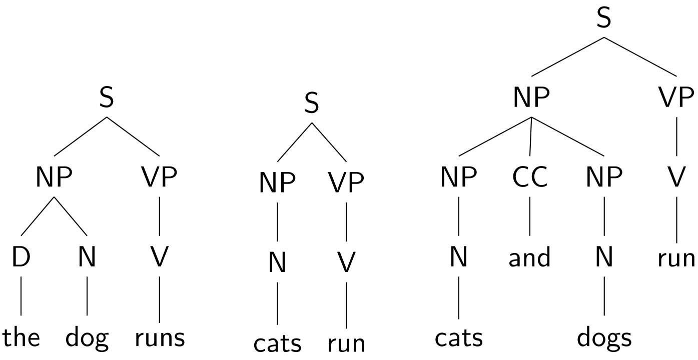
1. $\langle$S$\rangle$
2. $\langle$NP VP$\rangle$
3. $\langle$NP CC NP VP$\rangle$
4. $\langle$N CC NP VP$\rangle$
5. $\langle$cats CC NP VP$\rangle$
6. $\langle$cats and NP VP$\rangle$
7. $\langle$cats and N VP$\rangle$
8. $\langle$cats and dogs VP$\rangle$
9. $\langle$cats and dogs V$\rangle$

Derivation - more examples

1. $\langle$S$\rangle$
2. $\langle$NP VP$\rangle$
3. $\langle$NP CC NP VP$\rangle$
4. $\langle$N CC NP VP$\rangle$
5. $\langle$cats CC NP VP$\rangle$
6. $\langle$cats and NP VP$\rangle$
7. $\langle$cats and N VP$\rangle$
8. $\langle$cats and dogs VP$\rangle$
9. $\langle$cats and dogs V$\rangle$
10. $\langle$cats and dogs run$\rangle$

Parse trees

Parse trees compactly represent derivations

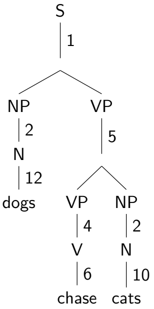


Derivation: a sequence of rule applications

A derivation can be seen as a sequence of rule applications $\langle r_1, \ldots, r_m \rangle$

- starts from S
- and after m steps yields a string $\mathrm{yield}(r_1^m) = x_1^n$
- the sequence can be read off of a tree by a depth-first traversal

Derivations as trees



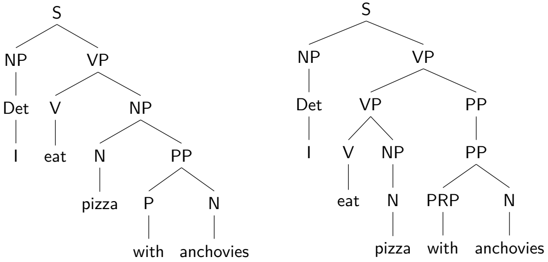
1. S → NP VP
2. NP → N
3. NP → D N
4. VP → V
5. VP → VP NP
6. V → chase
7. V → chases
8. D → the
9. N → cat
10. N → cats
11. N → dog
12. N → dogs

Sequence of rule applications (depth-first traversal)
$\langle r_1 = 1, r_2 = 2, r_3 = 12, r_4 = 5, r_5 = 4, r_6 = 6, r_7 = 2, r_8 = 10 \rangle$

The sentence is the yield of the derivation

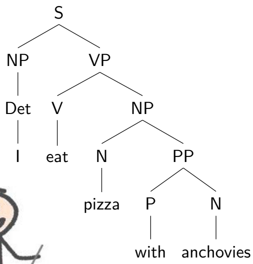
Structural ambiguity

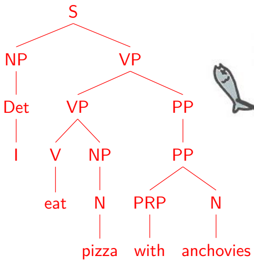
Different structure leads to different interpretation

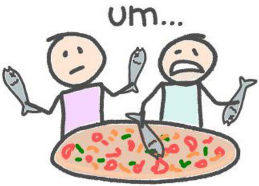


Structural ambiguity

Different structure leads to different interpretation

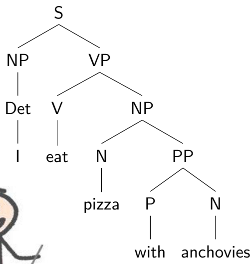


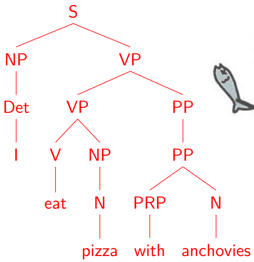


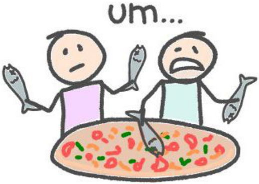


Structural ambiguity

Different structure leads to different interpretation







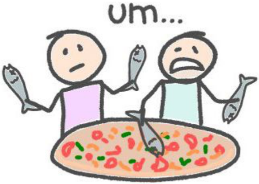
How should we deal with this?

Structural ambiguity

Different structure leads to different interpretation

[S [NP [Det I]] [VP [V eat] [NP [N pizza] [PP [P with] [N anchovies]]]]]

[S [NP [Det I]] [VP [VP [V eat] [NP [N pizza]]] [PP [PP [PRP with] [N anchovies]]]]]



How should we deal with this? Probabilities!

NLMI

Trees and grammars

Context-free grammars

Probabilistic context-free grammars

Probability distributions over derivations

We define a random derivation D

- a sequence $\langle R_1, \ldots, R_m \rangle$ of random rule applications
- where R is a random variable indexing rules of the grammar

The probability over a sequence of m rules can be written

$$P_{D|M}(r_1^m|m) = \underbrace{\prod_{i=1}^{m} P_{R_i|R_{<i}}(r_i|r_{<i})}_{\text{chain rule}}$$

Probability distributions over derivations

We define a random derivation D

- a sequence $\langle R_1, \ldots, R_m \rangle$ of random rule applications
- where R is a random variable indexing rules of the grammar

The probability over a sequence of m rules can be written

$$P_{D|M}(r_1^m|m) = \underbrace{\prod_{i=1}^{m} P_{R_i|R_{<i}}(r_i|r_{<i})}_{\text{chain rule}}$$

$$\approx \underbrace{\prod_{i=1}^{m} P_R(r_i)}_{\text{independence assumption}}$$

Conditional independence

A rule rewrites a LHS nonterminal symbol into a RHS string

- A rule $R : v \to \beta$ corresponds to a random pair (LHS, RHS)
- LHS corresponds to a random nonterminal symbol $v \in \mathcal{V}$
- RHS corresponds to a random sequence of terminals and nonterminals $\beta \in (\Sigma \cup \mathcal{V})^a$

Then we re-write the probability of a derivation as

$$P_{D|M}(r_1^m|m) = P_D(\langle \underbrace{(v_1, \beta_1)}_{r_1}, \ldots, \underbrace{(v_m, \beta_m)}_{r_m} \rangle | m)$$

Conditional independence

A rule rewrites a LHS nonterminal symbol into a RHS string

- A rule $R : v \to \beta$ corresponds to a random pair $(\mathrm{LHS}, \text{RHS})$
- LHS corresponds to a random nonterminal symbol $v \in \mathcal{V}$
- RHS corresponds to a random sequence of terminals and nonterminals $\beta \in (\Sigma \cup \mathcal{V})^a$

Then we re-write the probability of a derivation as

$$P_{D|M}(r_1^m|m) = P_D(\langle \underbrace{(v_1, \beta_1)}_{r_1}, \ldots, \underbrace{(v_m, \beta_m)}_{r_m} \rangle | m)$$

$$= \prod_{i=1}^{m} P_R(r_i) = \prod_{i=1}^{m} P_R(v_i \to \beta_i)$$

Conditional independence

A rule rewrites a LHS nonterminal symbol into a RHS string

- A rule $R : v \to \beta$ corresponds to a random pair $(\mathrm{LHS}, \mathsf{RHS})$
- LHS corresponds to a random nonterminal symbol $v \in \mathcal{V}$
- RHS corresponds to a random sequence of terminals and nonterminals $\beta \in (\Sigma \cup \mathcal{V})^a$

Then we re-write the probability of a derivation as

$$
\begin{aligned}
P_{D|M}(r_1^m|m) &= P_D(\langle \underbrace{(v_1, \beta_1)}_{r_1}, \ldots, \underbrace{(v_m, \beta_m)}_{r_m} \rangle | m) \\
&= \prod_{i=1}^{m} P_R(r_i) = \prod_{i=1}^{m} P_R(v_i \to \beta_i) \\
&= \prod_{i=1}^{m} P_{\mathrm{RHS}|\mathrm{LHS}}(\beta_i | v_i)
\end{aligned}
$$

Factorisation

We make each factor $P_{\mathrm{RHS}|\mathrm{LHS}=v}$ a distribution $\mathrm{Cat}(\boldsymbol{\theta}^{(v)})$

- $0 < \theta_{v \to \beta} < 1$
- $\sum_\beta \theta_{v \to \beta} = 1$

Factorisation

We make each factor $P_{\mathrm{RHS}|\mathrm{LHS}=v}$ a distribution $\mathrm{Cat}(\boldsymbol{\theta}^{(v)})$

- $0 < \theta_{v \to \beta} < 1$
- $\sum_{\beta} \theta_{v \to \beta} = 1$

What's the support of $P_{\mathrm{RHS}|\mathrm{LHS}=v}$?

Factorisation

We make each factor $P_{\text{RHS}|\text{LHS}=v}$ a distribution $\text{Cat}(\boldsymbol{\theta}^{(v)})$

- $0 < \theta_{v \to \beta} < 1$
- $\sum_\beta \theta_{v \to \beta} = 1$

What's the support of $P_{\text{RHS}|\text{LHS}=v}$?

- The set of all RHS strings $(\Sigma \cup \mathcal{V})^a$

Factorisation

We make each factor $P_{\text{RHS}|\text{LHS}=v}$ a distribution $\text{Cat}(\boldsymbol{\theta}^{(v)})$

- $0 < \theta_{v\to\beta} < 1$
- $\sum_\beta \theta_{v\to\beta} = 1$

What's the support of $P_{\text{RHS}|\text{LHS}=v}$?

- The set of all RHS strings $(\Sigma \cup \mathcal{V})^a$

Notation guideline

- $P_R(v \to \beta) = P_{\text{RHS}|\text{LHS}}(\beta|v) = \theta_{v\to\beta}$ or $P_R(r) = \theta_r$
  e.g. $P_R(\mathsf{S} \to \mathsf{NP\ VP}) = P_{\text{RHS}|\mathsf{LHS}}(\mathsf{NP\ VP}|\mathsf{S}) = \theta_{\mathsf{S}\to\mathsf{NP\ VP}}$

Factorisation

We make each factor $P_{\text{RHS}|\text{LHS}=v}$ a distribution $\text{Cat}(\boldsymbol{\theta}^{(v)})$

- $0 < \theta_{v \to \beta} < 1$
- $\sum_{\beta} \theta_{v \to \beta} = 1$

What's the support of $P_{\text{RHS}|\text{LHS}=v}$?

- The set of all RHS strings $(\Sigma \cup \mathcal{V})^{a}$

Notation guideline

- $P_R(v \to \beta) = P_{\text{RHS}|\text{LHS}}(\beta|v) = \theta_{v \to \beta}$ or $P_R(r) = \theta_r$
  e.g. $P_R(\mathsf{S} \to \mathsf{NP\ VP}) = P_{\text{RHS}|\mathsf{LHS}}(\mathsf{NP\ VP}|\mathsf{S}) = \theta_{\mathsf{S} \to \mathsf{NP\ VP}}$

How many parameters to represent P_R?

Factorisation

We make each factor $P_{\mathrm{RHS}|\mathrm{LHS}=v}$ a distribution $\mathrm{Cat}(\boldsymbol{\theta}^{(v)})$

- $0 < \theta_{v\to\beta} < 1$
- $\sum_\beta \theta_{v\to\beta} = 1$

What's the support of $P_{\mathrm{RHS}|\mathrm{LHS}=v}$?

- The set of all RHS strings $(\Sigma \cup \mathcal{V})^a$

Notation guideline

- $P_R(v \to \beta) = P_{\mathrm{RHS}|\mathrm{LHS}}(\beta|v) = \theta_{v\to\beta}$ or $P_R(r) = \theta_r$
  e.g. $P_R(\mathsf{S} \to \mathsf{NP\ VP}) = P_{\mathrm{RHS}|\mathsf{LHS}}(\mathsf{NP\ VP}|\mathsf{S}) = \theta_{\mathsf{S}\to\mathsf{NP\ VP}}$

How many parameters to represent P_R?

- One cpd per LHS, thus $O(|\mathcal{V}| \times |\Sigma \cup \mathcal{V}|^a)$

Probability of a derivation

A simple product over m terms

$$P_{D|M}(r_1^m|m) = \prod_{i=1}^{m} P_R(v_i \to \beta_i)$$

Probability of a derivation

A simple product over m terms

$$
\begin{aligned}
P_{D|M}(r_1^m|m) &= \prod_{i=1}^{m} P_R(v_i \to \beta_i) \\
&= \prod_{i=1}^{m} P_{\mathrm{RHS|LHS}}(\beta_i|v_i)
\end{aligned}
$$

Probability of a derivation

A simple product over m terms

$$
\begin{aligned}
P_{D|M}(r_1^m|m) &= \prod_{i=1}^{m} P_R(v_i \to \beta_i) \\
&= \prod_{i=1}^{m} P_{\mathrm{RHS}|\mathrm{LHS}}(\beta_i|v_i) \\
&= \prod_{i=1}^{m} \mathrm{Cat}(\beta_i|\boldsymbol{\theta}^{(v_i)})
\end{aligned}
$$

Probability of a derivation

A simple product over m terms

$$
\begin{aligned}
P_{D|M}(r_1^m|m) &= \prod_{i=1}^{m} P_R(v_i \to \beta_i) \\
&= \prod_{i=1}^{m} P_{\mathrm{RHS}|\mathrm{LHS}}(\beta_i|v_i) \\
&= \prod_{i=1}^{m} \mathrm{Cat}(\beta_i|\boldsymbol{\theta}^{(v_i)}) \\
&= \prod_{i=1}^{m} \theta_{v_i \to \beta_i}
\end{aligned}
$$

where r_i corresponds to $v_i \to \beta_i$

Probabilistic CFG Language Model

Generative story

Probabilistic CFG Language Model

Generative story

1. Generate a sentence length $N \sim P_N$

Probabilistic CFG Language Model

Generative story

1. Generate a sentence length $N \sim P_N$
2. Generate a derivation length $M|n \sim P_{M|N}$

Probabilistic CFG Language Model

Generative story

1. Generate a sentence length $N \sim P_N$
2. Generate a derivation length $M|n \sim P_{M|N}$
3. Generate a derivation $D|m \sim P_{D|M}$

Probabilistic CFG Language Model

Generative story

1. Generate a sentence length $N \sim P_N$
2. Generate a derivation length $M|n \sim P_{M|N}$
3. Generate a derivation $D|m \sim P_{D|M}$
4. Generate a sentence $S|r_1^m, n, m \sim P_{S|DNM}$

Probabilistic CFG Language Model

Generative story

1. Generate a sentence length $N \sim P_N$
2. Generate a derivation length $M|n \sim P_{M|N}$
3. Generate a derivation $D|m \sim P_{D|M}$
4. Generate a sentence $S|r_1^m, n, m \sim P_{S|DNM}$

$$P_{SD}(x_1^n, r_1^m) = P_N(n) P_{M|N}(m|n) P_{D|M}(r_1^m) P_{S|DNM}(x_1^n|r_1^m)$$

Probabilistic CFG Language Model

Generative story

1. Generate a sentence length $N \sim P_N$
2. Generate a derivation length $M|n \sim P_{M|N}$
3. Generate a derivation $D|m \sim P_{D|M}$
4. Generate a sentence $S|r_1^m, n, m \sim P_{S|DNM}$

$$P_{SD}(x_1^n, r_1^m) = P_N(n)P_{M|N}(m|n)P_{D|M}(r_1^m)P_{S|DNM}(x_1^n|r_1^m)$$

But note that (4) is deterministic

$$P_{S|DNM}(x_1^n|r_1^m) = \begin{cases} 1 & \text{if } \mathrm{yield}(r_1^m) = x_1^n \\ 0 & \text{otherwise} \end{cases}$$

Probability of a sentence

Joint distribution

$$P_{SD}(x_1^n, r_1^m) = P_N(n) P_{M|N}(m|n) P_{D|M}(r_1^m) P_{S|DNM}(x_1^n | r_1^m)$$

Probability of a sentence

Joint distribution

$$P_{SD}(x_1^n, r_1^m) = P_N(n)P_{M|N}(m|n)P_{D|M}(r_1^m)P_{S|DNM}(x_1^n|r_1^m)$$

Marginal

$$P_S(x_1^n) = \sum_{r_1^m} P_N(n)P_{M|N}(m|n)P_{D|M}(r_1^m)P_{S|DNM}(x_1^n|r_1^m)$$

Probability of a sentence

Joint distribution

$$P_{SD}(x_1^n, r_1^m) = P_N(n)P_{M|N}(m|n)P_{D|M}(r_1^m)P_{S|DNM}(x_1^n|r_1^m)$$

Marginal

$$\begin{aligned}P_S(x_1^n) &= \sum_{r_1^m} P_N(n)P_{M|N}(m|n)P_{D|M}(r_1^m)P_{S|DNM}(x_1^n|r_1^m)\\ &= \sum_{r_1^m} P_N(n)P_{M|N}(m|n)P_{D|M}(r_1^m) \times [\text{yield}(r_1^m) = x_1^n]\end{aligned}$$

Probability of a sentence

Joint distribution

$$P_{SD}(x_1^n, r_1^m) = P_N(n)P_{M|N}(m|n)P_{D|M}(r_1^m)P_{S|DNM}(x_1^n|r_1^m)$$

Marginal

$$\begin{aligned} P_S(x_1^n) &= \sum_{r_1^m} P_N(n)P_{M|N}(m|n)P_{D|M}(r_1^m)P_{S|DNM}(x_1^n|r_1^m) \\ &= \sum_{r_1^m} P_N(n)P_{M|N}(m|n)P_{D|M}(r_1^m) \times [\text{yield}(r_1^m) = x_1^n] \\ &= P_N(n) \sum_{r_1^m \in \mathfrak{G}(x_1^n)} P_{M|N}(m|n)P_{D|M}(r_1^m) \end{aligned}$$

where $\mathfrak{G}(x_1^n)$ is the set of derivations whose yield is x_1^n

Probability of a sentence

Typically P_N and $P_{M|N}$ are ignored (assumed uniform), then

$$P_S(x_1^n) \propto \sum_{r_1^m \in \mathfrak{G}(x_1^n)} P_{D|M}(r_1^m)$$

Probability of a sentence

Typically P_N and $P_{M|N}$ are ignored (assumed uniform), then

$$\begin{aligned} P_S(x_1^n) &\propto \sum_{r_1^m \in \mathfrak{G}(x_1^n)} P_{D|M}(r_1^m) \\ &= \sum_{r_1^m \in \mathfrak{G}(x_1^n)} \prod_{i=1}^{m} P_R(r_i) \end{aligned}$$

Probability of a sentence

Typically P_N and $P_{M|N}$ are ignored (assumed uniform), then

$$
\begin{aligned}
P_S(x_1^n) &\propto \sum_{r_1^m \in \mathfrak{G}(x_1^n)} P_{D|M}(r_1^m) \\
&= \sum_{r_1^m \in \mathfrak{G}(x_1^n)} \prod_{i=1}^{m} P_R(r_i) \\
&= \sum_{r_1^m \in \mathfrak{G}(x_1^n)} \prod_{i=1}^{m} P_R(v_i \to \beta_i)
\end{aligned}
$$

Probability of a sentence

Typically P_N and $P_{M|N}$ are ignored (assumed uniform), then

$$\begin{aligned} P_S(x_1^n) &\propto \sum_{r_1^m \in \mathfrak{G}(x_1^n)} P_{D|M}(r_1^m) \\ &= \sum_{r_1^m \in \mathfrak{G}(x_1^n)} \prod_{i=1}^{m} P_R(r_i) \\ &= \sum_{r_1^m \in \mathfrak{G}(x_1^n)} \prod_{i=1}^{m} P_R(v_i \to \beta_i) \\ &= \sum_{r_1^m \in \mathfrak{G}(x_1^n)} \prod_{i=1}^{m} \theta_{v_i \to \beta_i} \end{aligned}$$

where r_i corresponds to $v_i \to \beta_i$

Probability of a sentence

Typically P_N and $P_{M|N}$ are ignored (assumed uniform), then

$$
\begin{aligned}
P_S(x_1^n) &\propto \sum_{r_1^m \in \mathfrak{G}(x_1^n)} P_{D|M}(r_1^m) \\
&= \sum_{r_1^m \in \mathfrak{G}(x_1^n)} \prod_{i=1}^{m} P_R(r_i) \\
&= \sum_{r_1^m \in \mathfrak{G}(x_1^n)} \prod_{i=1}^{m} P_R(v_i \to \beta_i) \\
&= \sum_{r_1^m \in \mathfrak{G}(x_1^n)} \prod_{i=1}^{m} \theta_{v_i \to \beta_i}
\end{aligned}
$$

where r_i corresponds to $v_i \to \beta_i$

Challenge: to express $\mathfrak{G}(x_1^n)$ a task called parsing

Maximum likelihood estimation

We have a treebank, that is, a corpus where

- a sentence x_1^m is annotated with its CFG tree r_1^m

Our distributions $P_{\mathrm{RHS|LHS}}$ are categorical

- $\mathrm{RHS} \mid \mathrm{LHS} = v \sim \mathrm{Cat}(\boldsymbol{\theta}^{(v)})$

Maximum likelihood estimation

We have a treebank, that is, a corpus where

- a sentence x_1^m is annotated with its CFG tree r_1^m

Our distributions $P_{\mathrm{RHS}|\mathrm{LHS}}$ are categorical

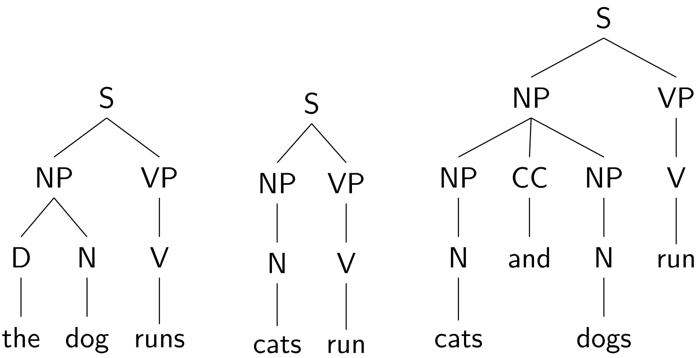
- $\mathrm{RHS} \mid \mathrm{LHS} = v \sim \mathrm{Cat}(\boldsymbol{\theta}^{(v)})$

MLE solution

$$\theta_{v \to \beta} = \frac{\mathrm{count}(v \to \beta)}{\sum_{\beta'} \mathrm{count}(v \to \beta')}$$

MLE - Example

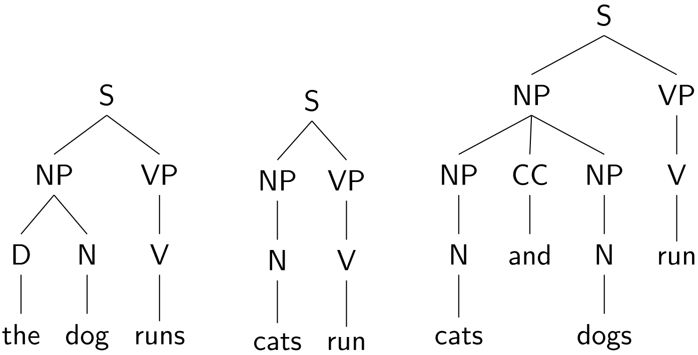
Consider the treebank



- $\theta_{\text{NP}\to\text{N}}$
- $\theta_{\text{N}\to\text{dog}}$

MLE - Example

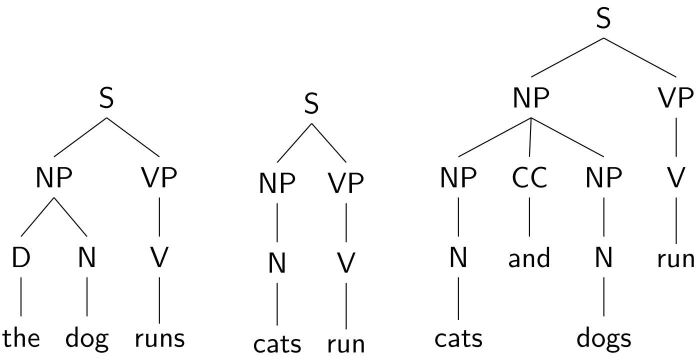
Consider the treebank



- $\theta_{\mathsf{NP}\to\mathsf{N}} = \frac{\text{count}(\mathsf{NP}\to\mathsf{N})}{\text{count}(\mathsf{NP}\to *)}$
- $\theta_{\mathsf{N}\to\mathsf{dog}}$

MLE - Example

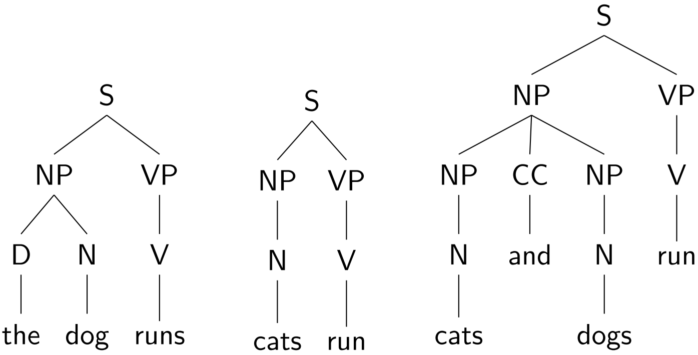
Consider the treebank



- $\theta_{\text{NP}\to\text{N}} = \frac{\text{count}(\text{NP}\to\text{N})}{\text{count}(\text{NP}\to *)} = \frac{3}{5}$
- $\theta_{\text{N}\to\text{dog}}$

MLE - Example

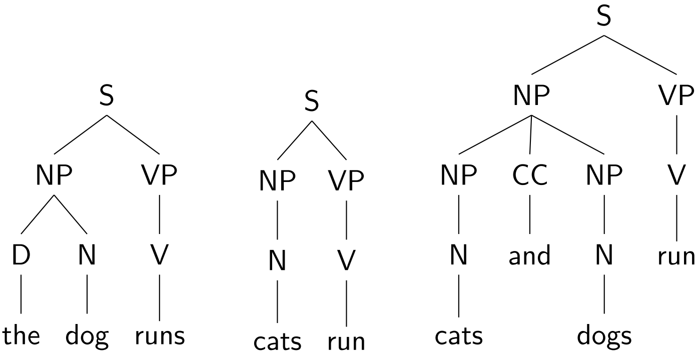
Consider the treebank



- $\theta_{\mathsf{NP}\to\mathsf{N}} = \frac{\text{count}(\mathsf{NP}\to\mathsf{N})}{\text{count}(\mathsf{NP}\to *)} = \frac{3}{5}$
- $\theta_{\mathsf{N}\to\mathsf{dog}} = \frac{\text{count}(\mathsf{N}\to\mathsf{dog})}{\text{count}(\mathsf{N}\to *)}$

MLE - Example

Consider the treebank



- $\theta_{\mathsf{NP}\to\mathsf{N}} = \frac{\text{count}(\mathsf{NP}\to\mathsf{N})}{\text{count}(\mathsf{NP}\to *)} = \frac{3}{5}$
- $\theta_{\mathsf{N}\to\mathsf{dog}} = \frac{\text{count}(\mathsf{N}\to\mathsf{dog})}{\text{count}(\mathsf{N}\to *)} = \frac{1}{4}$

References I