NLP1 2023/24

Modelling Syntactic Structure

Lecturer: Wilker Aziz
(week 2, lecture b)

Where are we at?

- What makes NLP hard
- Text classification
- Language modelling
- Sequence labelling
- **Syntactic parsing**

Modelling language so far

- Bag-of-words (NB, unigram LM)
  - ignore word order entirely
- Markov models
  - memorise valid phrases, but they are short
- HMM
  - capture shallow syntactic patterns through adjacent word classes
  - no semantic dependency amongst words

?

Outline

- **Trees and grammars**
- Context-free grammars (CFGs)
- Probabilistic CFGs
- Evaluation
- Limitation and extensions

Phrase Categories

Much like we abstracted from words to their (syntactic) categories, we can abstract from phrases to their syntactic categories.

Mentimeter

Substitute the adjective in *What a lazy/ADJ cat!*

28 responses

Mentimeter

[NP My dogs] sleep soundly

19 responses

I	My boyfriend	His turtles
My thoughts	My cats	The kids
Rich people	No one	I'd love to

5 19

Mentimeter

[$_{NP}$ My dogs] sleep soundly

19 responses

My boyfriends	the children	Volcanoes
I	The voices in my head	Giant gorillas
My chonky cat	The Erics	My dead grandma

5

19

Mentimeter

[$_{\text{NP}}$ My dogs] sleep soundly

19 responses

My cats

5

19

Mentimeter

Generalising POS categories

POS categories indicate which words are substitutable: **I saw a ___/ADJ cat.**

Phrasal categories indicate which *phrases* are substitutable: **[NP ___] sleep soundly.**

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

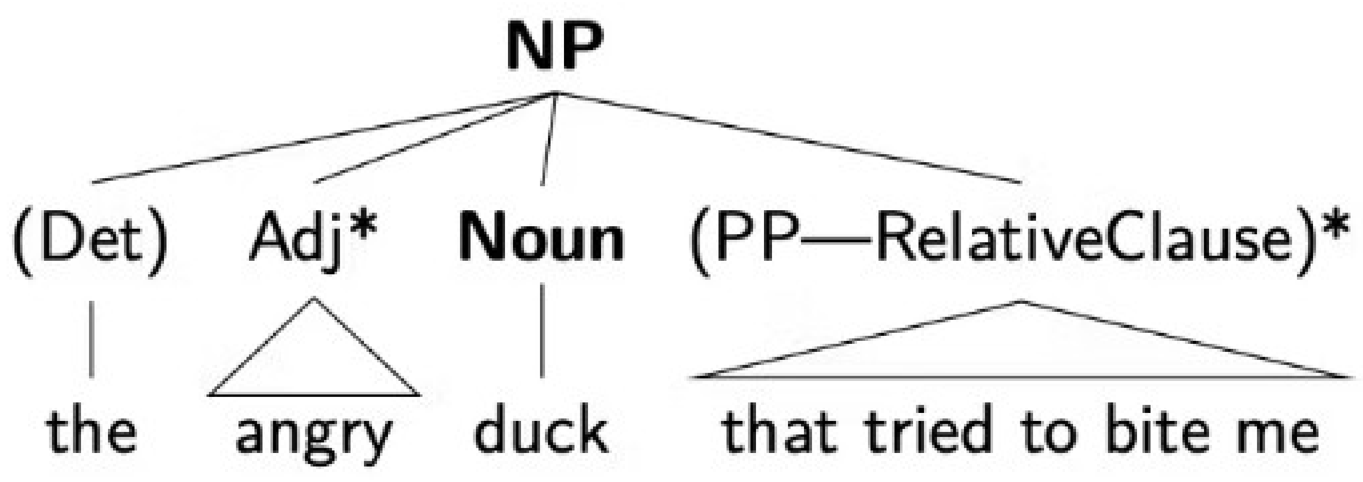
1

Heads and Phrases

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

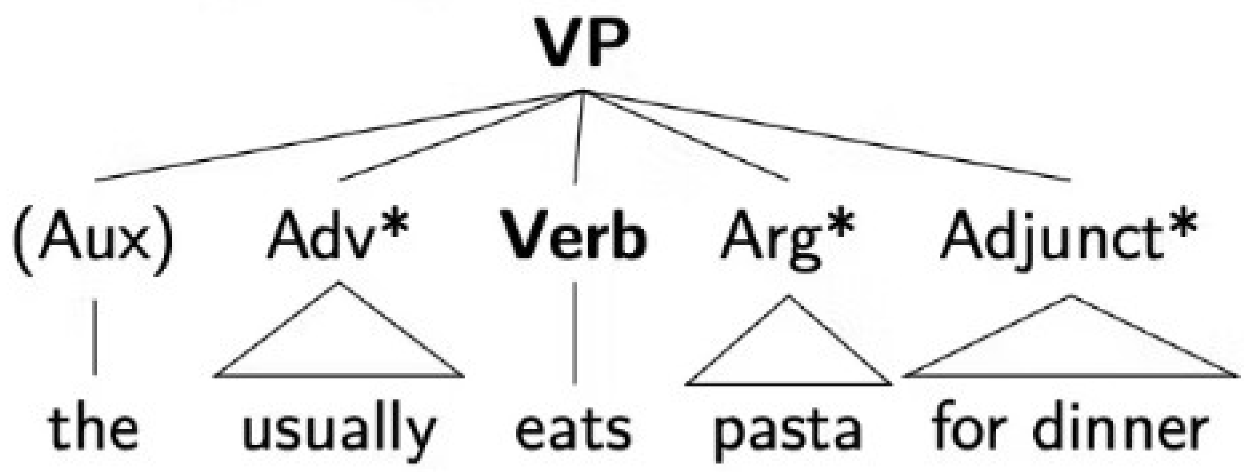
English NPs are commonly of the form

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



VPs are commonly of the form

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



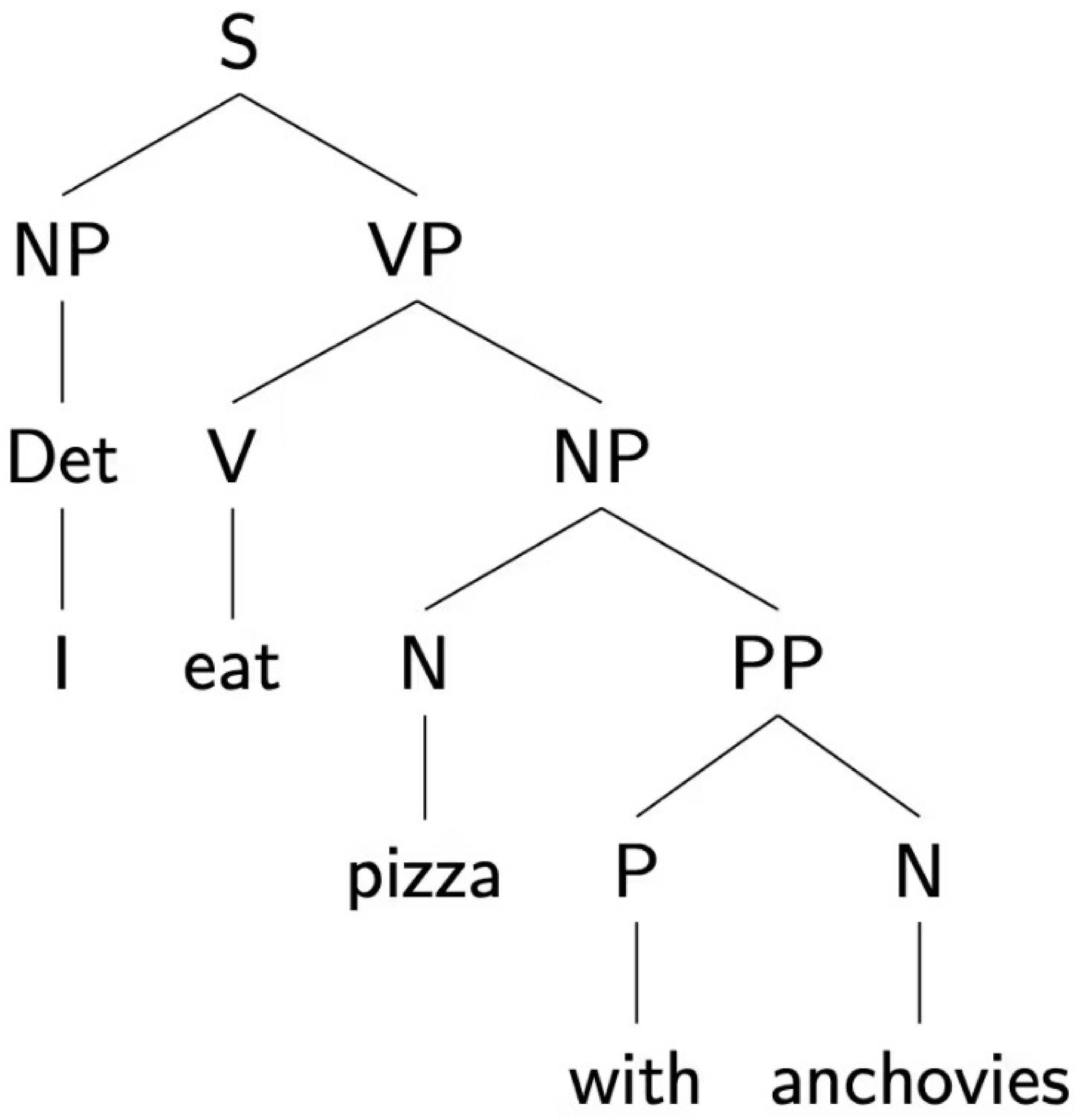
Constituency

Harry the Horse	a high-class spot such as Mindy's
the Broadway coppers	the reason he comes into the Hot Box
they	three parties from Brooklyn

three parties from Brooklyn *arrive*. . .
a high-class spot such as Mindy's *attracts*. . .
the Broadway coppers *love*. . .
they *sit*

Syntactic constituency is the idea that groups of words can behave as single units, or constituents.

One evidence for their existence is that they appear in similar syntactic environments (e.g., noun phrases tend to appear before a verb).



Nesting

Constituents can be hierarchically embedded in other constituents.

You can view the result as a tree-like structure.

2

Theory of Syntax

Explains which sentences are well-formed and which are not.

A good theory should be a **finite** specification of the strings of the language (as opposed to an infinite list of sentences).

It should also provide a good interface for semantic processing.

Mentimeter

Outline

- Trees and grammars
- **Context-free grammars (CFGs)**
- Probabilistic CFGs
- Evaluation
- Limitation and extensions

Context-Free Grammar (CFG)

A rewriting system with two types of *symbols* and a set of *symbol-rewriting rules*.

Symbols

Terminals (or *constants*): words

Nonterminals (or *variables*): word and phrasal categories.

Rules

$\mathrm{X} \to \beta$ where X is a nonterminal, and β is any string of terminal and nonterminal symbols.

Example

Nonterminals: S, NP, VP, PP, Pron, N, V, P

Terminals: *I, eat, pizza, with, anchovies*

S $\rightarrow$ NP VP

NP $\rightarrow$ Pron

NP $\rightarrow$ N

NP $\rightarrow$ NP PP

PP $\rightarrow$ P NP

VP $\rightarrow$ V

VP $\rightarrow$ VP NP

VP $\rightarrow$ VP PP

Pron $\rightarrow$ *I*

N $\rightarrow$ *pizza*

N $\rightarrow$ *anchovies*

P $\rightarrow$ *with*

V $\rightarrow$ eat

Mentimeter

CFG Formally

Σ is a finite set of terminal symbols

$\mathcal{V}$ is a finite set of nonterminal symbols, with a distinguished start symbol $\mathrm{S} \in \mathcal{V}$ and $\mathcal{V} \cap \Sigma = \emptyset$

$\mathcal{R}$ is a finite set of rules of the kind: $\mathrm{X} \rightarrow \beta$ with $\mathrm{X} \in \mathcal{V}$ and $\beta \in (\Sigma \cup \mathcal{V})^*$.

A CFG is the tuple $\langle \Sigma, \mathcal{V}, \mathrm{S}, \mathcal{R} \rangle$

CFG terminology

- Arity (length of rule's RHS)
  - unary: $\mathrm{A} \rightarrow \mathrm{B}$
  - binary: $\mathrm{X} \rightarrow \mathrm{B\ C}$
  - n-ary: $\mathrm{X} \rightarrow \mathrm{X}_1, \ldots, \mathrm{X}_n$
  - if the longest rule has arity a, we say the grammar has arity a

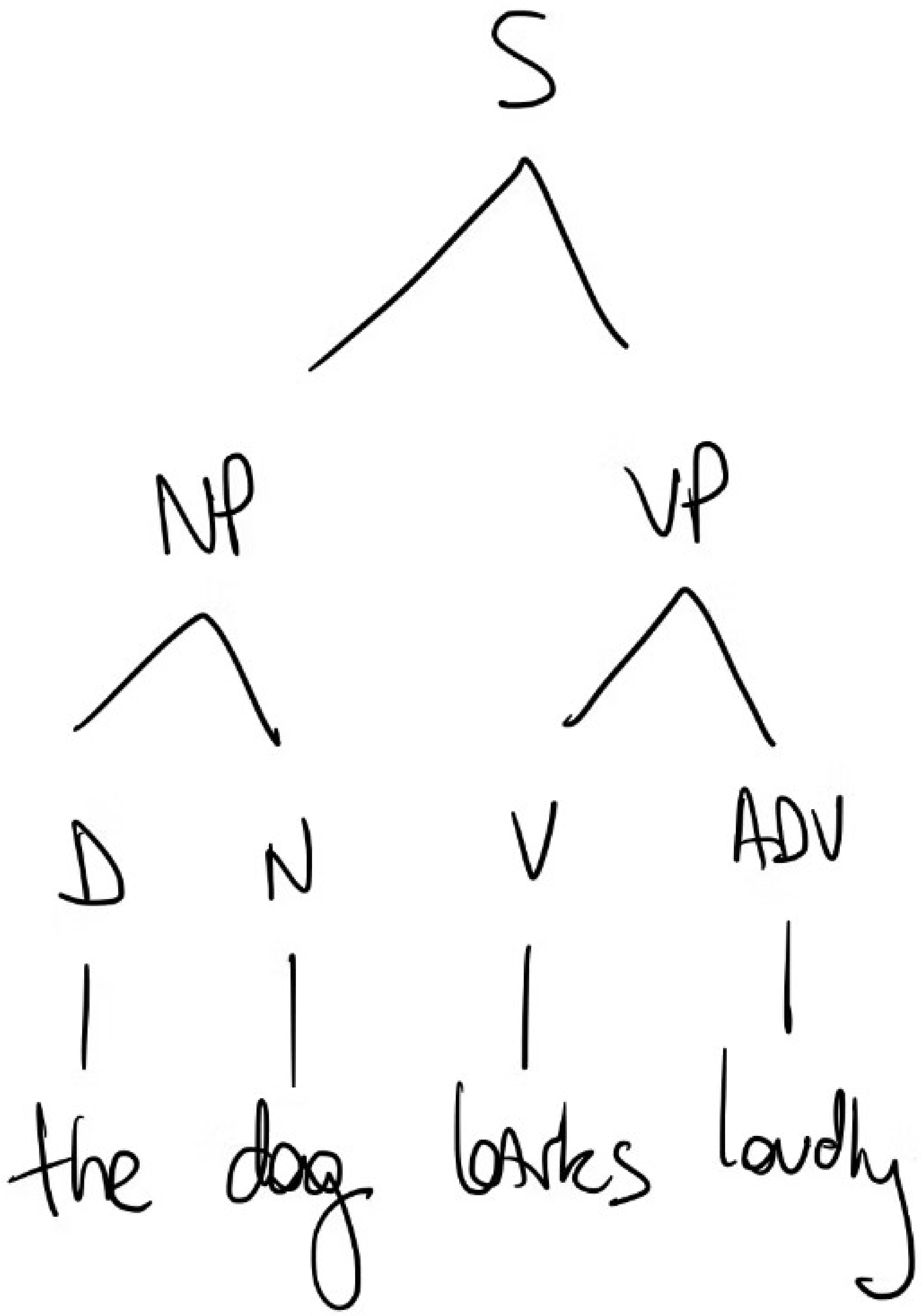
Mentimeter

Derivation

A **derivation** is a sequence of strings: we start with the start symbol $\langle \mathrm{S} \rangle$, then recursively rewrite the leftmost nonterminal X by application of a rule $\mathrm{X} \rightarrow \beta \in \mathcal{R}$ until only terminals remain.

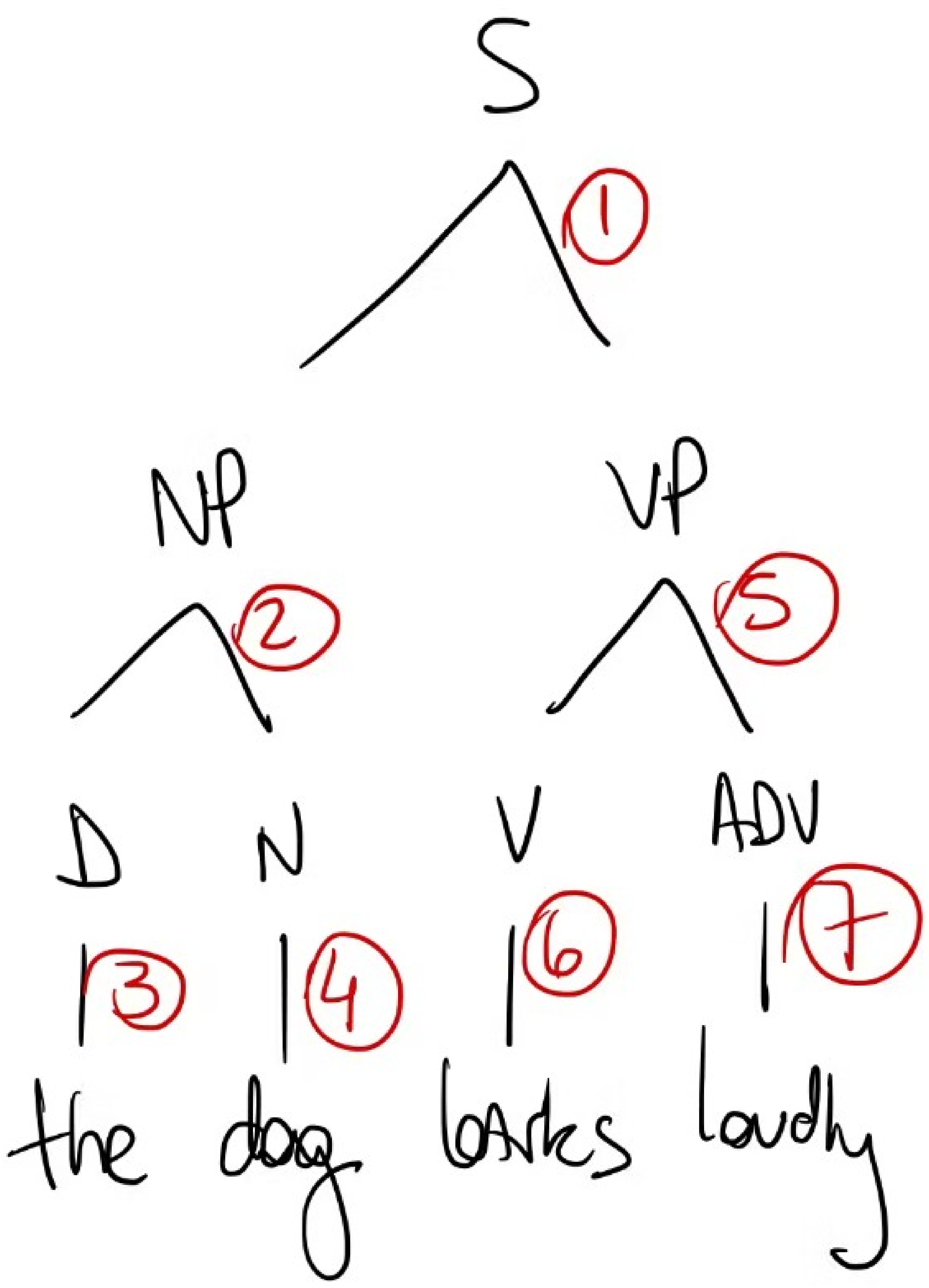
If a string $w_1 \cdots w_n$ is derivable from S we write: $\mathrm{S} \stackrel{*}{\Rightarrow} w_1 \cdots w_n$.

We call the string $w_1 \cdots w_n$ the **yield** of the derivation.



Example

- $\langle \underline{\text{S}} \rangle$
- $\langle \underline{\text{NP}}\ \text{VP} \rangle$
- $\langle \underline{\text{D}}\ \text{N VP} \rangle$
- $\langle \text{the}\ \underline{\text{N}}\ \text{VP} \rangle$
- $\langle \text{the dog}\ \underline{\text{VP}} \rangle$
- $\langle \text{the dog}\ \underline{\text{V}}\ \text{ADV} \rangle$
- $\langle \text{the dog barks}\ \underline{\text{ADV}} \rangle$
- $\langle \text{the dog barks loudly} \rangle$



Sequence of rule applications from $\langle S \rangle$

- S ⟶ NP VP
- NP ⟶ D N
- D ⟶ the
- N ⟶ dog
- VP ⟶ V ADV
- V ⟶ barks
- ADV ⟶ loudly

2

Derive "dogs chase cat":

13 responses

1 2 12 5 4 6 2 9	1, 2, 11, 5, 6, 2, 10	1, 11, 5, 4, 7, 2, 9
1, 2, 12, 5, 4, 6, 2, 9	1, 2, 12, 5, 4, 7, 10	1 , 2 , 12, 5, 4, 7, 2, 10
1 2 12 5 4 6 2 9	1, 2, 12, 5,4, 6, 2, 9	S -> NP VP -> N VP -> dogs VP -> dogs VP NP -> dogs V NP -> dogs chase NP -> dogs chase N -> dogs chase cat

13

Derive "dogs chase cat":

13 responses

1, 2, 12, 5, 4, 6, 2, 9

1 2 12 5 4 6 2 9

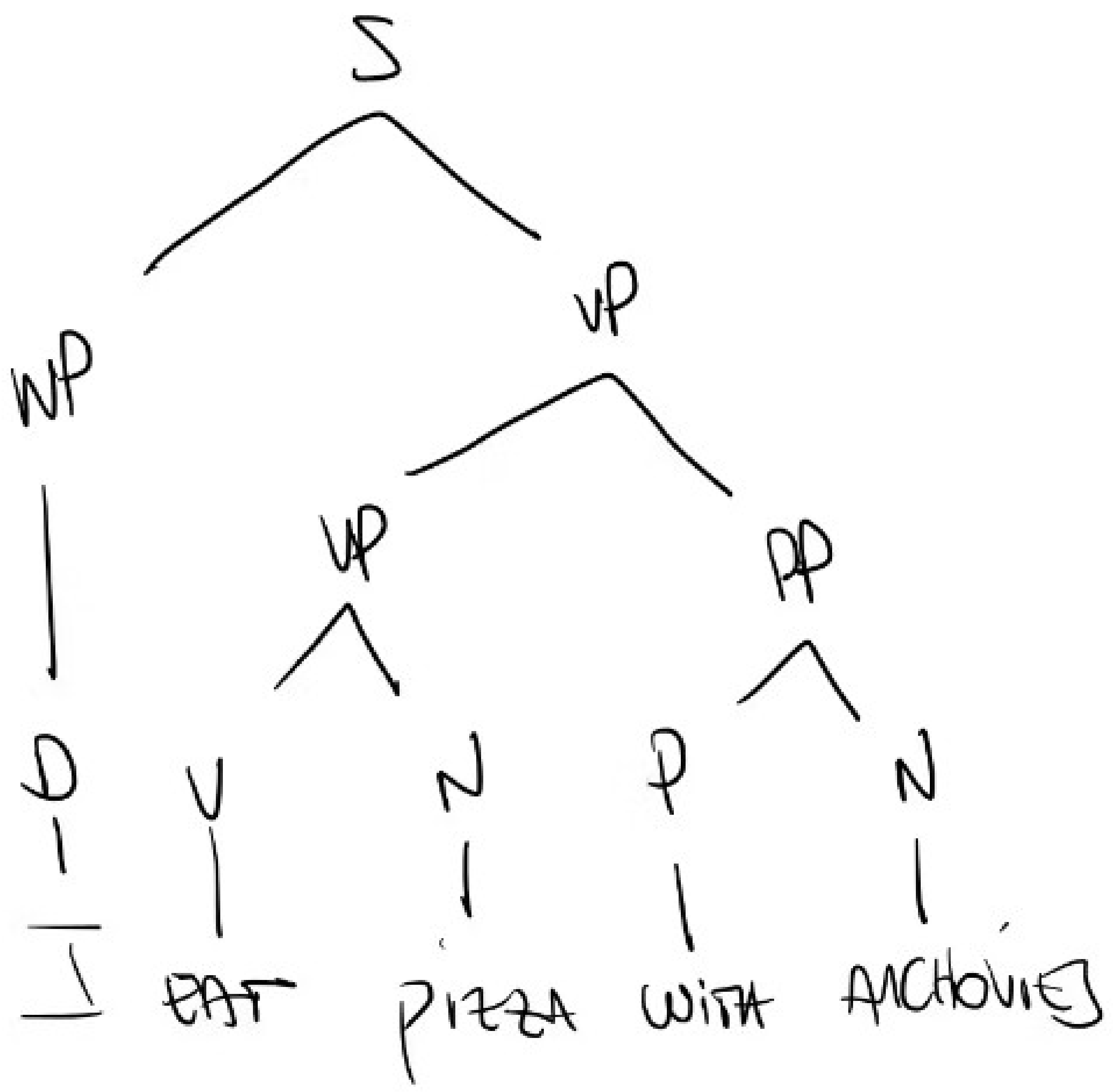
1,2,12,5,

1 2 12 5 6 2 9

13

Mentimeter

What is the meaning of the derivation on the left?



anchovies make good tools for eating

18

anchovies make good pizza toppings

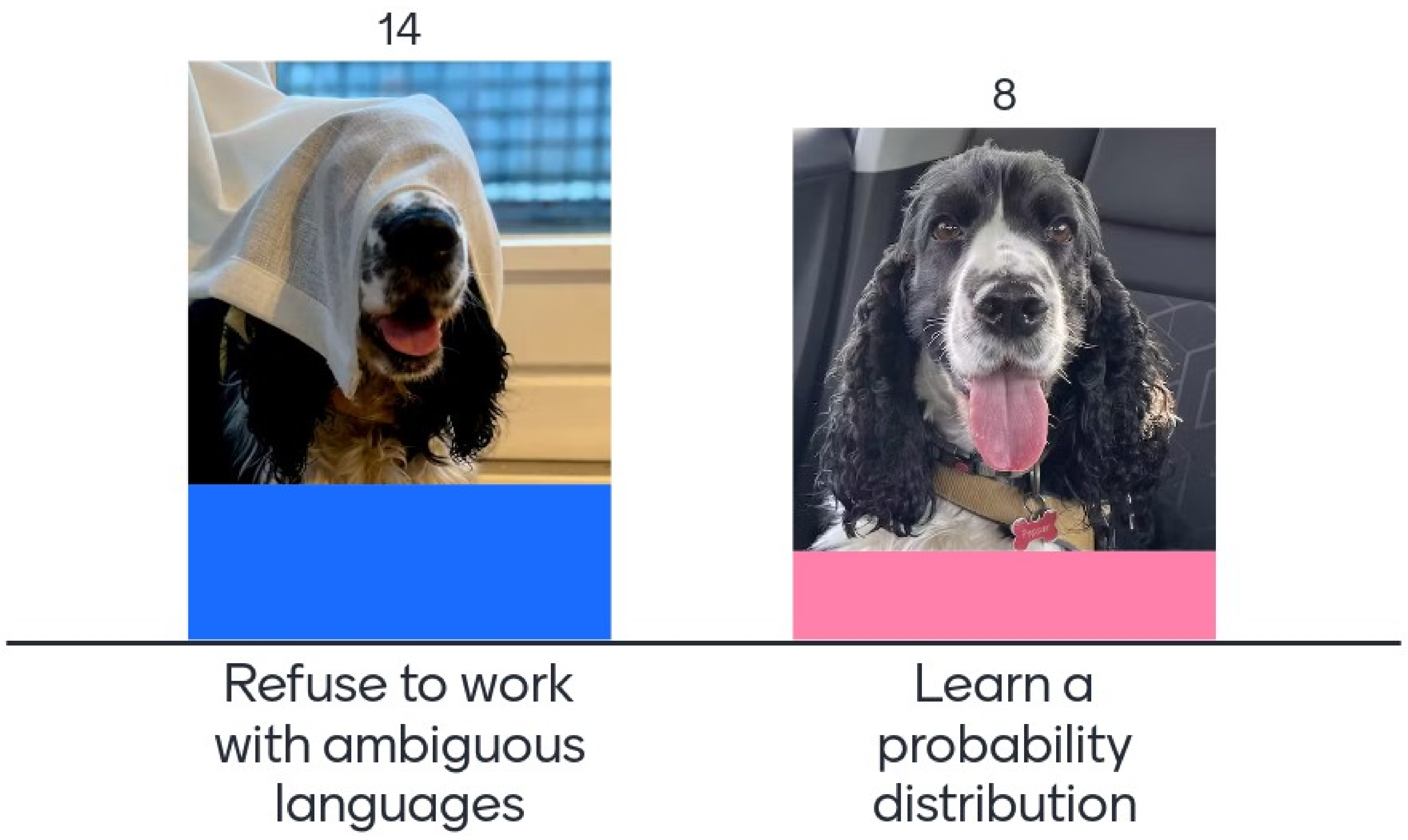
3

4

21

Mentimeter

How do you want to deal with (structural) ambiguity?



2

22

Outline

- Trees and grammars
- Context-free grammars (CFGs)
- **Probabilistic CFGs**
- Evaluation
- Limitations and extensions

Probabilistic Context-Free Grammars

A probability distribution over the space of all derivations (including their yields) supported by a grammar.

Assigning Probability to a Derivation

A random derivation $D = \langle R_1, \ldots, R_M \rangle$ is a sequence of M random rule applications. A valid derivation rewrites S into a sequence of random words $X = \langle W_1, \ldots, W_L \rangle$.

We can assign probability mass to $r_{1:m}$ via chain rule

$$P_D(r_{1:m}) = \prod_{j=1}^{m} P_{R|H}(r_j | r_{<j})$$

1

A Markov model over *steps* in a derivation

A random derivation $D = \langle R_1, \ldots, R_M \rangle$ is a sequence of M random rule applications. A random rule is a pair (N, S) of a random LHS nonterminal and a RHS string.

We can assign probability mass to $r_{1:m}$ via chain rule under a Markov assumption:

$$P_D(r_{1:m}) \overset{\text{ind.}}{=} \prod_{j=1}^{m} P_R(r_j) = \prod_{j=1}^{m} P_{S|N}(\beta_j | v_j)$$

with $v_j \in \mathcal{V}$ and $\beta_j \in (\mathcal{V} \cup \Sigma)^*$

Note: pretend every derivation starts with $r_0 = \text{BoS} \rightarrow \text{S}$.

Mentimeter

Generative Story

1. Start with $D = \langle \mathrm{S} \rangle$

2. If all symbols in D are terminal, stop. Else, go to (3).

3. Condition on the left-most nonterminal symbol v in the derivation, and draw a RHS string β with probability $P_{S|V}(\beta|v)$, replace v in D by β. Repeat from (2).

This corresponds to a **depth-first** expansion of nonterminals. See my commented Colab demo.

Parameterisation

If we can rewrite a nonterminal variable v into K different ways, associate with v a distribution:

$$S|V = v \sim \mathrm{Categorical}(\theta_{v\to\beta_1}, \ldots, \theta_{v\to\beta_K})$$

Examples:

$$S|V = \mathrm{S} \sim \mathrm{Categorical}(\theta_{\mathrm{S}\to\mathrm{NP\ VP}}, \theta_{\mathrm{S}\to\mathrm{VP}})$$

$$S|V = \mathrm{N} \sim \mathrm{Categorical}(\theta_{\mathrm{N}\to\mathrm{cat}}, \theta_{\mathrm{N}\to\mathrm{dog}}, \theta_{\mathrm{N}\to\mathrm{bird}})$$

The pmf assigns mass $p(r_{1:m}; \theta) = \prod_{j=1}^{m} \theta_{v_j\to\beta_j}$

to a derivation

$$r_{1:m} = \langle v_1 \to \beta_1, \ldots, v_m \to \beta_m \rangle$$

Mentimeter

Given a dataset, give the MLE for $\theta_{v \to \beta}$

4 responses

count(theta_v, beta) / count(theta_v)

Proportion of v's which map to beta in the training set

Count(v -> b) / count(v)

count(v, beta)/count(v)

1

4

Estimation via MLE

As always,

$$\theta_{v\to\beta} = \frac{\text{count}(v \to \beta)}{\sum_{v\to\gamma\in\mathcal{R}} \text{count}(v \to \gamma)}$$

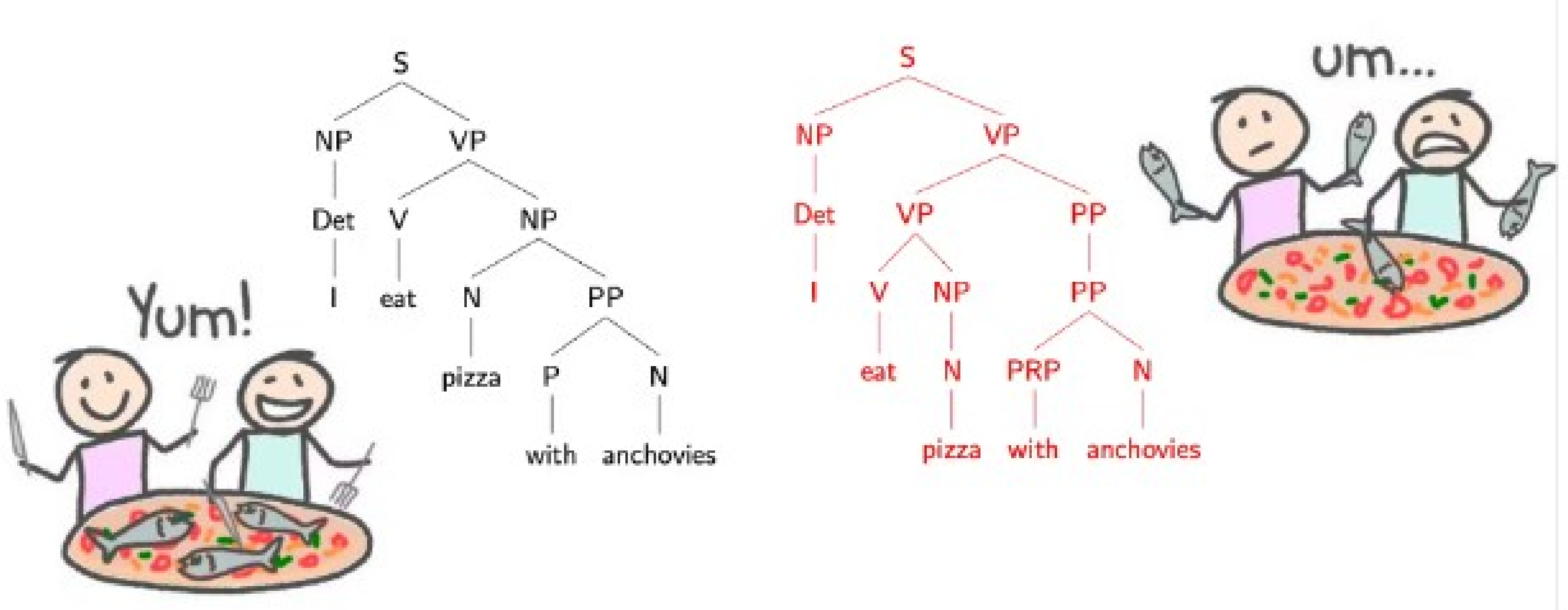
Outline

- Trees and grammars
- Context-free grammars (CFGs)
- Probabilistic CFGs
- **Evaluation**
- Limitations and extensions

2

Mentimeter

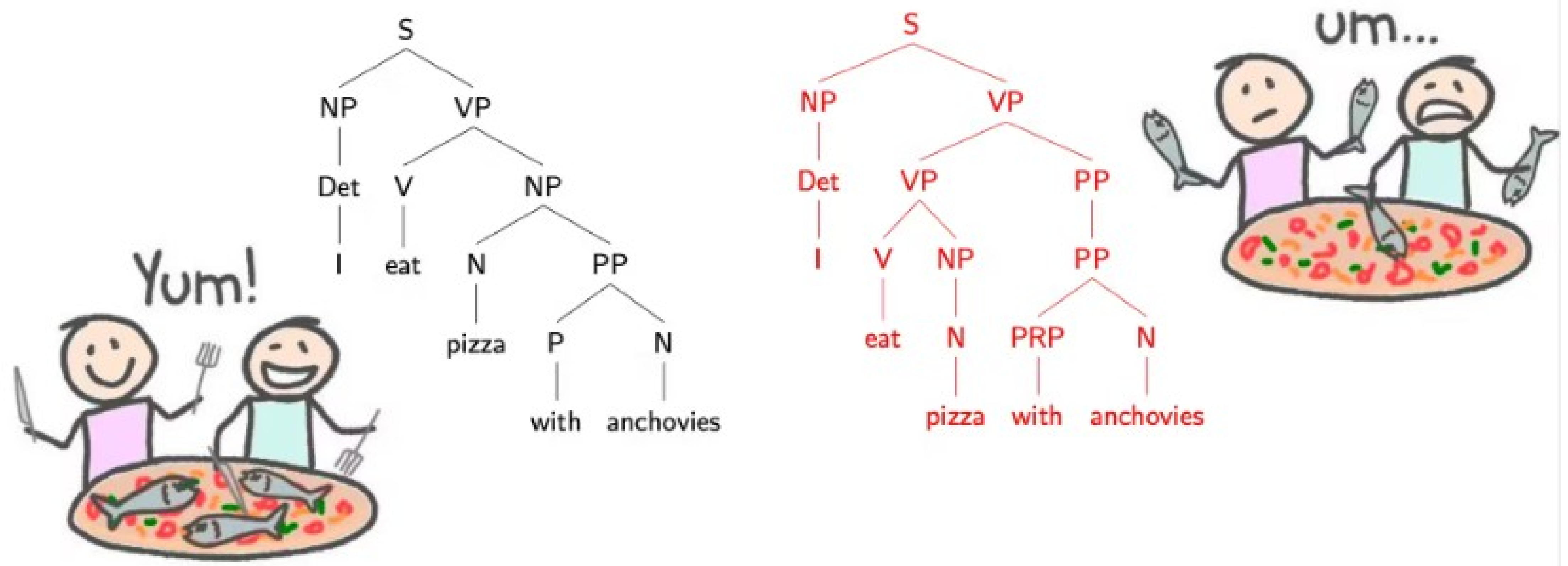
Probability of a Sentence



Probability of the correct parse	Average probability	Marginal probability
0	0	0
✗	✗	✓

1

Probability of a Sentence



Due to structural ambiguities, there are potentially many derivations for any one sentence $w_{1:l}$.

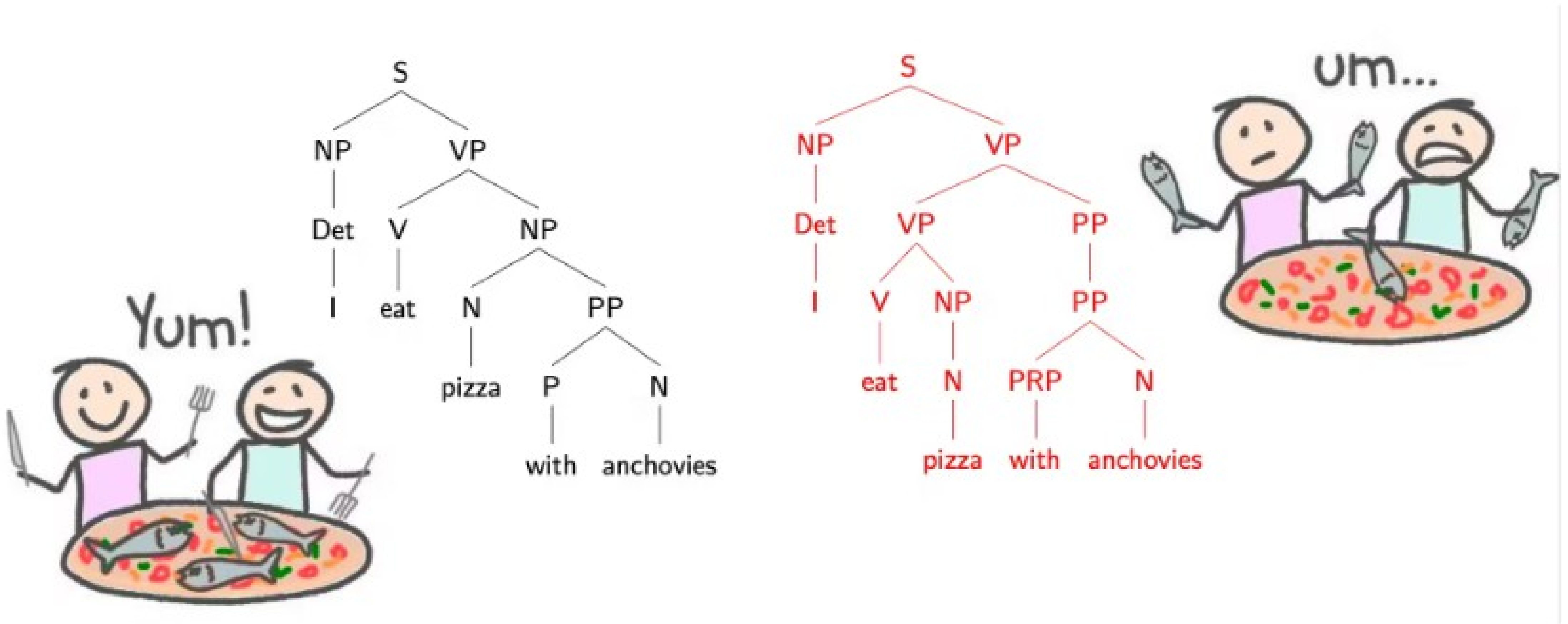
The PCFG assigns **marginal probability**

$$P_X(w_{1:l}) = \sum_{r_{1:m}} P_D(r_{1:m}) \times [\text{yield}(r_{1:m}) = w_{1:l}]$$

equal to the sum of the probabilities of all derivations whose yield is the sentence.

Evaluation as an LM

Assess perplexity of model using marginal probability of sentences in heldout dataset of valid sentences.



Evaluation as a Syntactic Parser

Obtain the **most probable derivation** subject to its yield being the sentence we want to parse:

$$\arg\max_{r_{1:m}} P_D(r_{1:m}) \times [\text{yield}(r_{1:m}) = w_{1:l}]$$

Typically report the constituent label precision, recall, F. See section 17.8.

Mentimeter

Parse Forest

The key to both uses of PCFG (as an LM and as a parser) is to find all derivations of a given sentence $w_{1:l}$. We call the set of all trees that derive $\mathrm{S} \overset{*}{\Rightarrow} w_{1:l}$ a **parse forest** for $w_{1:l}$.

We typically work with binary-branching trees (arity=2), then the number of trees for a sentence of L words is the Catalan number

$$C_L = \frac{(2L)!}{(L+1)!L!}.$$

1

CKY

But to find the sum of probabilities or the maximum probability we do not need to **enumerate** the trees, we can exploit the Markov assumption in yet another dynamic programme: read Section 17.6.

The CKY algorithm is a packed representation of all trees. It can be used to find marginal probability (Inside algorithm) and maximum probability (Viterbi algorithm).

Outline

- Trees and grammars
- Context-free grammars (CFGs)
- Probabilistic CFGs
- Evaluation
- **Limitations and extensions**

Limitations of PCFGs

Waiting for responses ...

Limitations

Limited use of linguistic context due to generative formulation.

The context-free assumption is not enough in general: some linguistic constructions violate it.

Dynamic programming for PCFGs takes time that is cubic in sentence length.

Extensions

Like the HMM, the PCFG is a generative model. If all we care about is a mechanism to predict parse trees, then we can use a conditional model and employ rich features. Examples: transition-based parsers, CRF parsers.

We may care about relations between words, more so than constituency, for that we develop **dependency grammars**. Optional reading: Chapter 18.

Ask me anything

0 questions
0 upvotes

What Next

- Check Colab demo
- Study CKY: Section 17.6
- There are various very good CKY videos online
  - I will pick some I like or record one for you
- P&P1
- Lexical semantics and word embeddings