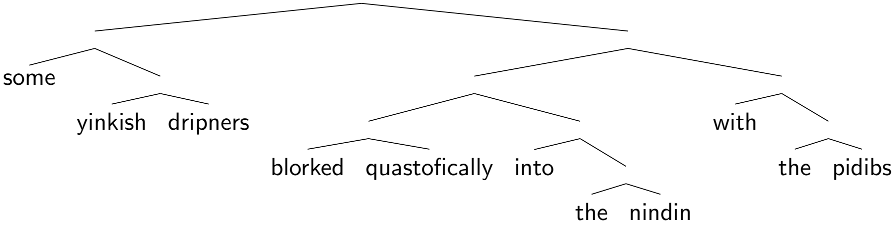
Overview of Natural Language Processing Part II & ACS L390

Lecture 5: Phrase Structure and Structured Prediction

Weiwei Sun

Department of Computer Science and Technology
University of Cambridge

Michaelmas 2024/25



Words are organized into nested blocks

Lecture 5: Phrase Structure and Structured Prediction

1. Phrase structure
3. Structured prediction
4. Probabilistic Context-free grammars
5. Neural parameterisation

Phrase Structure

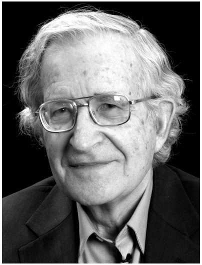
Interview of Noam Chomsky by Lex Fridman

(1) a. the guy who fixed the car very carefully packed his tools

b. very carefully, the guy who fixed the car packed his tools

c. *very carefully, the guy who fixed the car is tall

I think the deepest property of language and puzzling property that's been discovered is what is sometimes called structure dependence. [...] Linear closeness is an easy computation, but here you're doing a much more, what looks like a more complex computation.



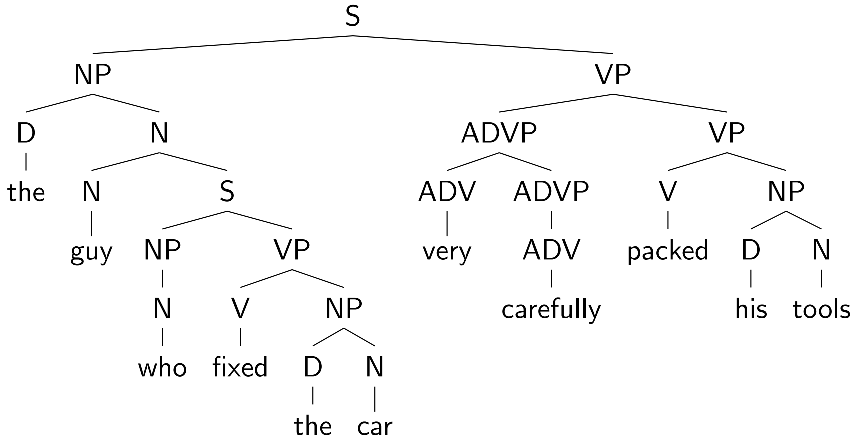
Noam Chomsky: Language, Cognition, and Deep Learning

www.youtube.com/watch?v=cMscNuSUy0I

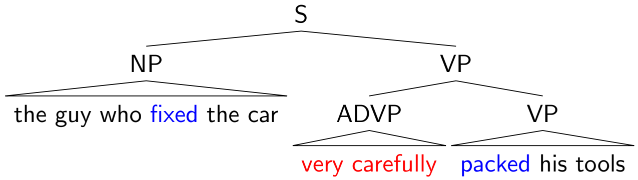
Constituency (phrase structure)

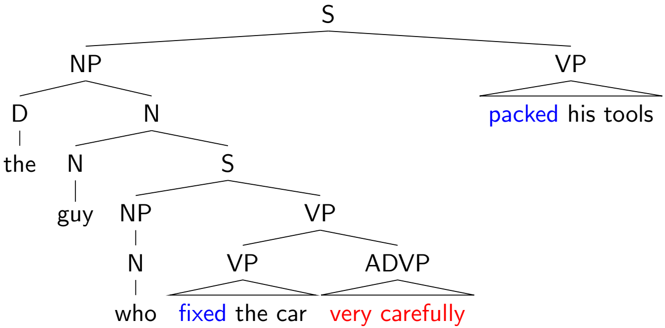
The basic idea

Phrase structure organizes words into *nested constituents*, which can be represented as **a tree**.



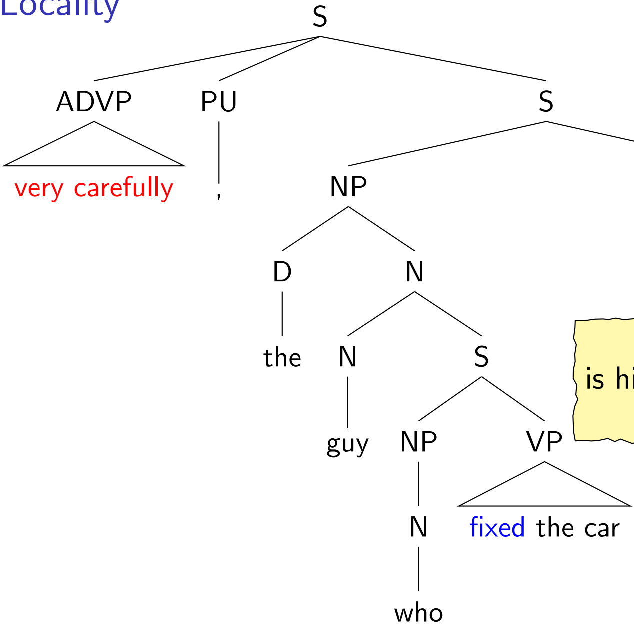
Different structures, different meaning





Results by a cool parser: `http://erg.delph-in.net/logon`

Locality



very carefully
is hierarchically closer to
packed

Applications of parsing

Modern parsers are quite accurate

For some languages, automatic syntactic parsing is good enough to help in a range of NLP tasks

- Machine translation
- Information extraction
- Grammar checking
- etc.

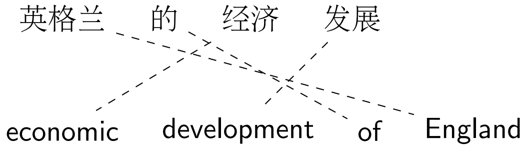
Applications of parsing

Modern parsers are quite accurate

For some languages, automatic syntactic parsing is good enough to help in a range of NLP tasks

- Machine translation
- Information extraction
- Grammar checking
- etc.

Translate “英格兰的经济发展” into English



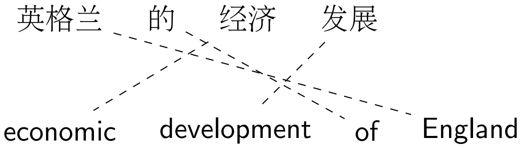
Applications of parsing

Modern parsers are quite accurate

For some languages, automatic syntactic parsing is good enough to help in a range of NLP tasks

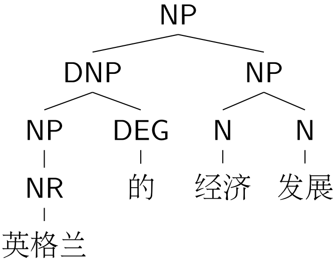
- Machine translation
- Information extraction
- Grammar checking
- etc.

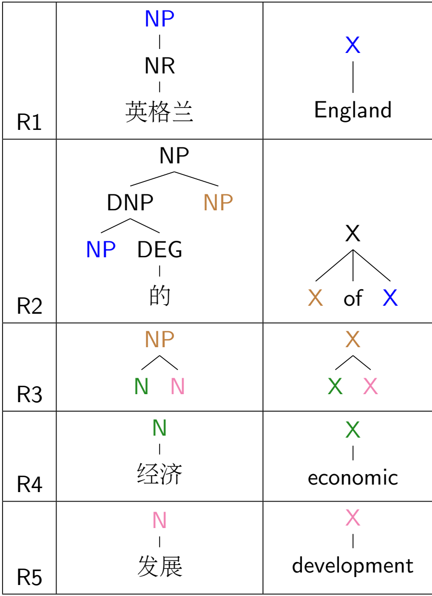
Translate “英格兰的经济发展” into English



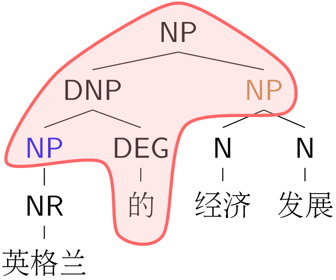
finite-state transducer?

An example: Tree-to-string transduction





An example: Tree-to-string transduction

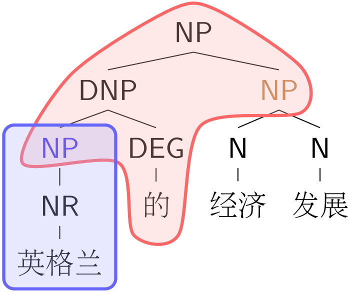


NP of NP

▷R2

R1	NP NR 英格兰	X England
R2	NP DNP NP NP DEG 的	X X of X
R3	NP N N	X X X
R4	N 经济	X economic
R5	N 发展	X development

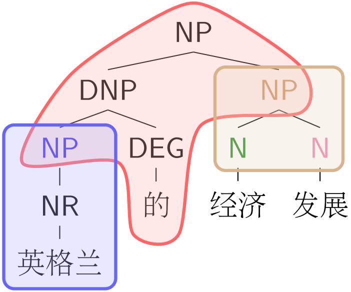
An example: Tree-to-string transduction



NP of NP ▷R2
NP of England ▷R1

R1	NP NR 英格兰	X England
R2	NP DNP NP NP DEG 的	X X of X
R3	NP N N	X X X
R4	N 经济	X economic
R5	N 发展	X development

An example: Tree-to-string transduction



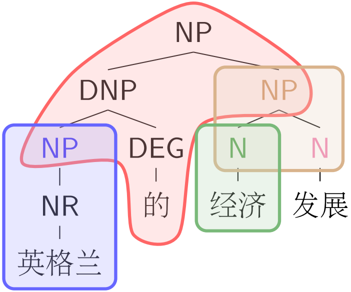
R1	NP NR 英格兰	X England
R2	NP DNP NP NP DEG 的	X X of X
R3	NP N N	X X X
R4	N 经济	X economic
R5	N 发展	X development

NP of NP ▷R2

NP of England ▷R1

N N of England ▷R3

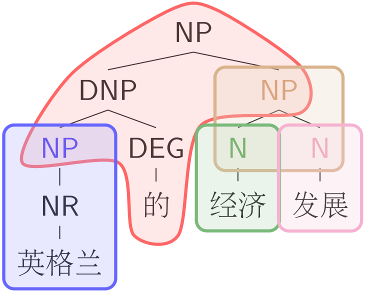
An example: Tree-to-string transduction



NP of NP ▷R2
NP of England ▷R1
N N of England ▷R3
economic N of England ▷R4

R1	NP NR 英格兰	X England
R2	NP DNP NP NP DEG 的	X X of X
R3	NP N N	X X X
R4	N 经济	X economic
R5	N 发展	X development

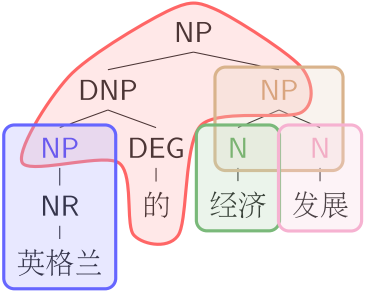
An example: Tree-to-string transduction



NP of NP ▷R2
NP of England ▷R1
N N of England ▷R3
economic N of England ▷R4
economic development of England ▷R5

R1	NP NR 英格兰	X England
R2	NP DNP NP NP DEG 的	X X of X
R3	NP N N	X X X
R4	N 经济	X economic
R5	N 发展	X development

An example: Tree-to-string transduction



R1	NP NR 英格兰	X England
R2	NP DNP NP NP DEG 的	X X of X
R3	NP N N	X X X
R4	N 经济	X economic
R5	N 发展	X development

NP of NP ▷R2
NP of England ▷R1
N N of England ▷R3
economic N of England ▷R4
economic development of England ▷R5

recursive
form transformation

Structured Prediction

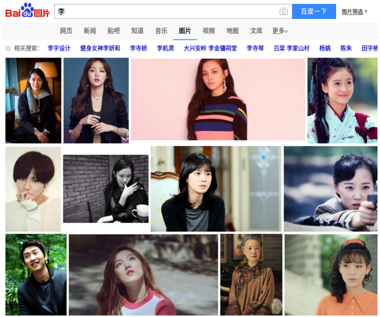
pre-lecture: watch this video

www.youtube.com/watch?v=bjUwSHGsG9o

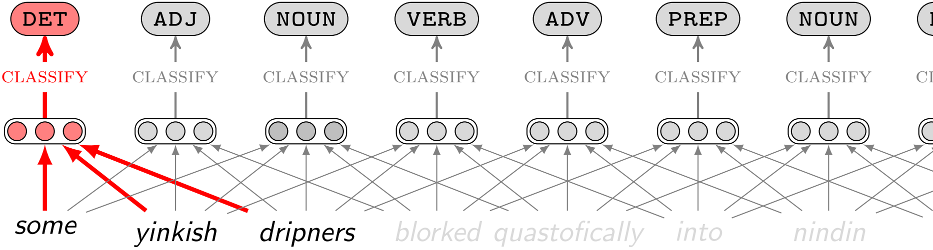
Muhammad Li

Howard: Who's Muhammad Li?

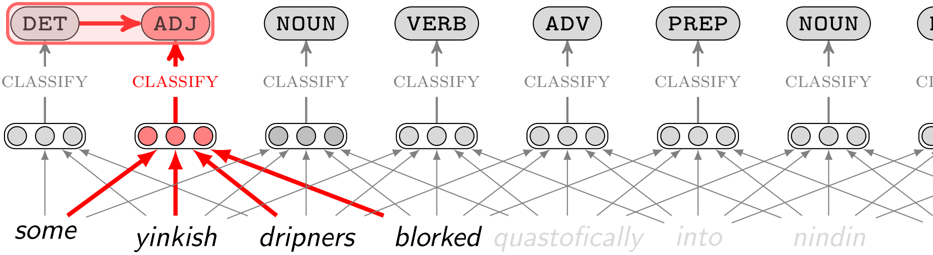
Sheldon: Muhammad is the most common first name in the world, Li, the most common surname. As I didn't know the answer, I thought that gave me a mathematical edge.



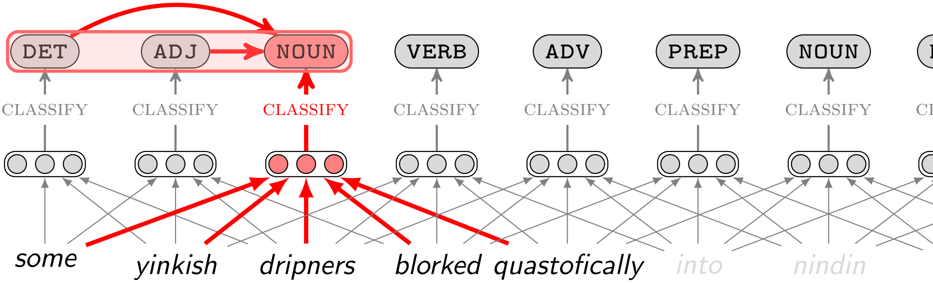
POS tagging and prediction



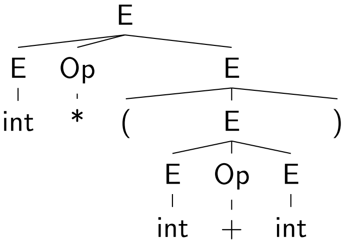
POS tagging and prediction

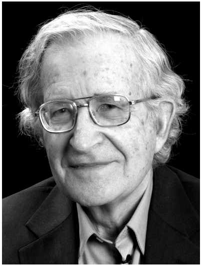


POS tagging and prediction



Two perspectives $\approx$ Possible vs Probable





[...] Therefore the true logic for this world is the calculus of probabilities, which takes account of the magnitude of the probability which is, or ought to be, in a reasonable man's mind.

Linguistic structure prediction

As a structured prediction problem

- Search space: Is this analysis possible? ▷CFG (today)
- Measurement: Is this analysis *good*? ▷PCFG (today)

$$\mathbf{y}^*(\mathbf{x}; \theta) = \operatorname*{arg\,max}_{\mathbf{y} \in \mathcal{Y}(\mathbf{x})} \textsc{Score}(\mathbf{x}, \mathbf{y})$$

- Decoding: find the analysis that obtains the highest score
- Parameter estimation: find good parameters

Linguistic structure prediction

As a structured prediction problem

- Search space: Is this analysis possible? ▷CFG (today)
- Measurement: Is this analysis *good*? ▷PCFG (today)

$$\mathbf{y}^*(\mathbf{x}; \boxed{\theta}) = \boxed{\arg\max}_{\boxed{\mathbf{y} \in \mathcal{Y}(\mathbf{x})}} \boxed{\textsc{Score}(\mathbf{x}, \mathbf{y})}$$

- Decoding: find the analysis that obtains the highest score
- Parameter estimation: find good parameters

Linguistic structure prediction

As a structured prediction problem

- Search space: Is this analysis possible? ▷CFG (today)
- Measurement: Is this analysis *good*? ▷PCFG (today)

$$\mathbf{y}^*(\mathbf{x}; \theta) = \arg\max_{\mathbf{y} \in \mathcal{Y}(\mathbf{x})} \textsc{Score}(\mathbf{x}, \mathbf{y})$$

- Decoding: find the analysis that obtains the highest score
- Parameter estimation: find good parameters

Linguistic structure prediction

As a structured prediction problem

- Search space: Is this analysis possible? ▷CFG (today)
- Measurement: Is this analysis *good*? ▷PCFG (today)

$$\mathbf{y}^*(\mathbf{x}; \theta) = \operatorname*{arg\,max}_{\mathbf{y} \in \mathcal{Y}(\mathbf{x})} \textsc{Score}(\mathbf{x}, \mathbf{y})$$

- Decoding: find the analysis that obtains the highest score
- Parameter estimation: find good parameters

Linguistic structure prediction

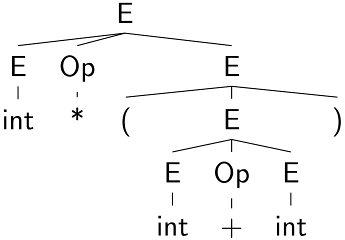
As a structured prediction problem

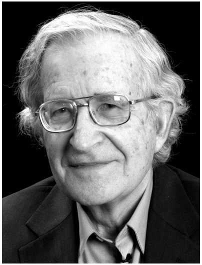
- Search space: Is this analysis possible? ▷CFG (today)
- Measurement: Is this analysis *good*? ▷PCFG (today)

$$\mathbf{y}^*(\mathbf{x}; \boxed{\theta}) = \boxed{\arg\max}_{\boxed{\mathbf{y} \in \mathcal{Y}(\mathbf{x})}} \boxed{\textsc{Score}(\mathbf{x}, \mathbf{y})}$$

- Decoding: find the analysis that obtains the highest score
- Parameter estimation: find good parameters

Context-Free Grammar





Formal grammars

Formally specify a grammar that can generate all and only the acceptable sentences of a natural language.

Formal grammars

Formally specify a grammar that can generate all and only the acceptable sentences of a natural language.

A grammar G consists of the following components:

1. A finite set Σ of terminal symbols.
2. A finite set N of nonterminal symbols that is disjoint from Σ.
3. A distinguished nonterminal symbol that is the START symbol.
4. A finite set R of production rules, each rule of the form

   $$(\Sigma \cup N)^+ \rightarrow (\Sigma \cup N)^*$$

   Each production rule maps from one string of symbols to another.

Context-Free Grammars

1. N: variables
2. Σ: terminals
3. R: productions

$$A \rightarrow (N \cup \Sigma)^*$$

$$A \in N$$

4. S: START

A linguistic example (1)

$N = \{\mathsf{S}, \mathsf{NP}, \mathsf{VP}, \mathsf{AdjP}, \mathsf{AdvP}\} \cup \{\mathsf{N}, \mathsf{Adj}, \mathsf{Adv}\}$

$\Sigma = \{$*colorless*, *green*, *ideas*, *sleep*, *furiously*$\}$

R

S→NP VP	NP→AdjP NP
VP→VP AdvP	
VP→V	NP→N
AdvP→Adv	AdjP→Adj

Adj→*colorless*	Adj→*green*
N→*ideas*	V→*sleep*
Adv→*furiously*	

$S = \mathsf{S}$

A linguistic example (1)

$N = \{\mathsf{S}, \mathsf{NP}, \mathsf{VP}, \mathsf{AdjP}, \mathsf{AdvP}\} \cup \{\mathsf{N}, \mathsf{Adj}, \mathsf{Adv}\}$

$\Sigma = \{$*colorless*, *green*, *ideas*, *sleep*, *furiously*$\}$

R

S→NP VP	NP→AdjP NP
VP→VP AdvP	
VP→V	NP→N
AdvP→Adv	AdjP→Adj

Adj→*colorless*	Adj→*green*
N→*ideas*	V→*sleep*
Adv→*furiously*	

$S = \mathsf{S}$

We can derive the structure of a string.

A linguistic example (1)

$N = \{\mathsf{S}, \mathsf{NP}, \mathsf{VP}, \mathsf{AdjP}, \mathsf{AdvP}\} \cup \{\mathsf{N}, \mathsf{Adj}, \mathsf{Adv}\}$

$\Sigma = \{$*colorless*, *green*, *ideas*, *sleep*, *furiously*$\}$

R

S→NP VP	NP→AdjP NP
VP→VP AdvP	
VP→V	NP→N
AdvP→Adv	AdjP→Adj

Adj→*colorless*	Adj→*green*
N→*ideas*	V→*sleep*
Adv→*furiously*	

$S = \mathsf{S}$

We can derive the structure of a string.

S $\Rightarrow$ NP VP

A linguistic example (1)

$N = \{\mathsf{S}, \mathsf{NP}, \mathsf{VP}, \mathsf{AdjP}, \mathsf{AdvP}\} \cup \{\mathsf{N}, \mathsf{Adj}, \mathsf{Adv}\}$

$\Sigma = \{$*colorless*, *green*, *ideas*, *sleep*, *furiously*$\}$

R

S→NP VP	NP→AdjP NP
VP→VP AdvP	
VP→V	NP→N
AdvP→Adv	AdjP→Adj

Adj→*colorless*	Adj→*green*
N→*ideas*	V→*sleep*
Adv→*furiously*	

$S = \mathsf{S}$

We can derive the structure of a string.

$$\begin{aligned} \mathsf{S} &\Rightarrow \mathsf{NP\ VP} \\ &\Rightarrow \mathsf{N\ VP} \end{aligned}$$

A linguistic example (1)

$N = \{\mathsf{S}, \mathsf{NP}, \mathsf{VP}, \mathsf{AdjP}, \mathsf{AdvP}\} \cup \{\mathsf{N}, \mathsf{Adj}, \mathsf{Adv}\}$

$\Sigma = \{$*colorless*, *green*, *ideas*, *sleep*, *furiously*$\}$

R

S→NP VP	NP→AdjP NP
VP→VP AdvP	
VP→V	NP→N
AdvP→Adv	AdjP→Adj

Adj→*colorless*	Adj→*green*
N→*ideas*	V→*sleep*
Adv→*furiously*	

$S = \mathsf{S}$

We can derive the structure of a string.

S ⇒ NP VP
⇒ N VP
⇒ ideas VP
⇒ ideas VP AdvP
⇒ ideas V AdvP
⇒ ideas sleep AdvP
⇒ ideas sleep Adv
⇒ ideas sleep furiously

A linguistic example (1)

$N = \{\mathsf{S}, \mathsf{NP}, \mathsf{VP}, \mathsf{AdjP}, \mathsf{AdvP}\} \cup \{\mathsf{N}, \mathsf{Adj}, \mathsf{Adv}\}$

$\Sigma = \{$*colorless*, *green*, *ideas*, *sleep*, *furiously*$\}$

R

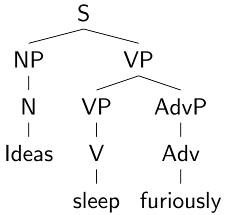
S→NP VP	NP→AdjP NP
VP→VP AdvP	
VP→V	NP→N
AdvP→Adv	AdjP→Adj

Adj→*colorless*	Adj→*green*
N→*ideas*	V→*sleep*
Adv→*furiously*	

$S = \mathsf{S}$

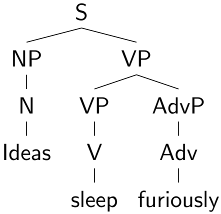
We can derive the structure of a string.

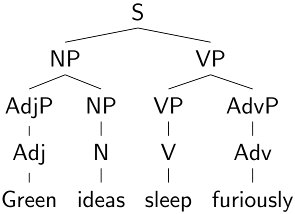
S $\Rightarrow$ NP VP
$\Rightarrow$ N VP
$\Rightarrow$ ideas VP
$\Rightarrow$ ideas VP AdvP
$\Rightarrow$ ideas V AdvP
$\Rightarrow$ ideas sleep AdvP
$\Rightarrow$ ideas sleep Adv
$\Rightarrow$ ideas sleep furiously

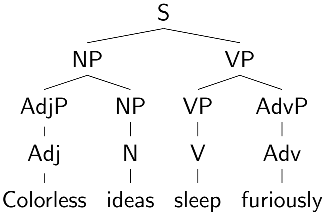


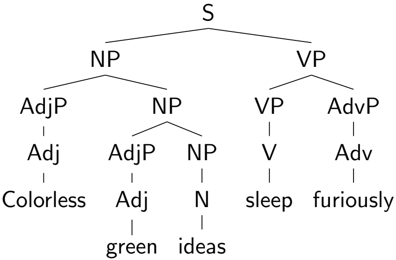
A linguistic example (2)

We can define the language of a grammar by applying the productions.









Recursion (1)

from *Inception* (https://www.imdb.com/title/tt1375666/)

recursion
place one component inside another component of the same type

Recursion (2)

Natural numbers

- $0 \leftarrow \emptyset$
- If n is a natural number, let $n+1 \leftarrow n \cup \{n\}$

$0 = \emptyset$

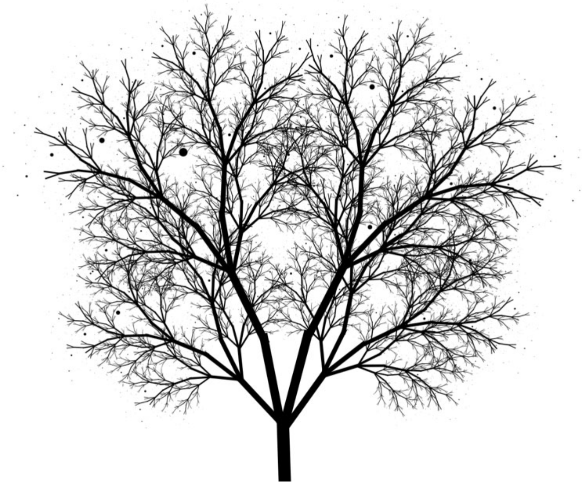
$1 = \{0\} = \{\emptyset\}$

$2 = \{0, 1\} = \{\emptyset, \{\emptyset\}\}$

$3 = \{0, 1, 2\} = \{\emptyset, \{\emptyset\}, \{\emptyset, \{\emptyset\}\}\}$

recursion
place one component inside another component of the same type

Recursion (3)



https://matthewjamestaylor.com/recursive-drawing

Recursion (4)

> *We hypothesize that FLN (faculty of language in the narrow sense) only includes recursion and is the only uniquely human component of the faculty of language.*
>
> *M Hauser, N Chomsky and W Fitch (2002)*
>
> *science.sciencemag.org/content/298/5598/1569*

(2) a. The dog bit the cat [which chased the mouse [which died]]. (right)

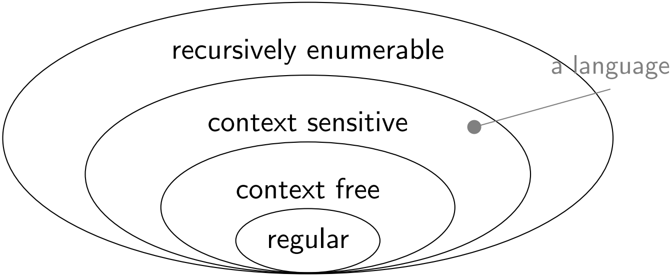
b. [[the dog] 's owner] 's friend (left)

c. The mouse [the cat [the dog bit] chased] died. (center)

Reminder: Chomsky Hierarchy

Grammar	Languages	Production rules
Type-0	Recursively enumerable	$\alpha \rightarrow \gamma$
Type-1	Context-sensitive	$\alpha A \beta \rightarrow \alpha \gamma \beta$
Type-2	Context-free	$A \rightarrow \gamma$
Type-3	Regular	$A \rightarrow a$
		$A \rightarrow aB$

$$a \in N;\ \alpha, \beta \in (N \cup \Sigma)^*, \gamma \in (N \cup \Sigma)^+$$



Where can I get a grammar?

English Treebank

- Penn Treebank = ca. 50,000 sentences with associated trees
- Usual set-up: ca. 40,000 training sentences, ca. 2,400 test sentences
- Cut all trees into 2-level subtrees.

Probabilistic Context-Free Grammars

[...] Therefore the true logic for this world is the calculus of probabilities, which takes account of the magnitude of the probability which is, or ought to be, in a reasonable man's mind.

Probabilistic CFGs

Probability of a tree t with rules $A_1 \to \beta_1, A_2 \to \beta_2, \ldots$ is

$$p(t) = \prod_{i=1}^{n} q(A_i \to \beta_i)$$

where $q(A_i \to \beta_i)$ is the probability for rule $A_i \to \beta_i$.

- When we expand A_i, how likely is it that we choose $A_i \to \beta_i$?
- For each nonterminal A_i,

$$\sum_{\beta} q(A_i \to \beta | A_i) = 1$$

- PCFG generates random derivations of CFG.
- Each event (expanding nonterminal by production rules) is statistically independent of all the others.

An example (1)

S	$\rightarrow$	NP VP	0.8
S	$\rightarrow$	Aux NP VP	0.15
S	$\rightarrow$	VP	0.05
NP	$\rightarrow$	AdjP NP	0.2
NP	$\rightarrow$	D N	0.7
NP	$\rightarrow$	N	0.1
VP	$\rightarrow$	VP AdvP	0.3
VP	$\rightarrow$	V	0.2
VP	$\rightarrow$	V NP	0.3
VP	$\rightarrow$	V NP NP	0.2
AdvP	$\rightarrow$	Adv	1.0
AdjP	$\rightarrow$	Adj	1.0

Adj	$\rightarrow$	*colorless*	0.4
Adj	$\rightarrow$	*green*	0.6
N	$\rightarrow$	*ideas*	1.0
V	$\rightarrow$	*sleep*	1.0
Adv	$\rightarrow$	*furiously*	1.0

An example (2)

S

S→NP VP 0.8

An example (2)

	S	S→NP VP	0.8
⇒	NP VP	NP→N	0.1

An example (2)

	S	S→NP VP	0.8
⇒	NP VP	NP→N	0.1
⇒	N VP	N→*ideas*	1.0
⇒	ideas VP	VP→VP AdvP	0.3
⇒	ideas VP AdvP	VP→V	0.2
⇒	ideas V AdvP	V→*sleep*	1.0
⇒	ideas sleep AdvP	AdvP→Adv	1.0
⇒	ideas sleep Adv	Adv→*furiously*	1.0

An example (2)

	S	S→NP VP	0.8
⇒	NP VP	NP→N	0.1
⇒	N VP	N→*ideas*	1.0
⇒	ideas VP	VP→VP AdvP	0.3
⇒	ideas VP AdvP	VP→V	0.2
⇒	ideas V AdvP	V→*sleep*	1.0
⇒	ideas sleep AdvP	AdvP→Adv	1.0
⇒	ideas sleep Adv	Adv→*furiously*	1.0

$$0.8 \times 0.1 \times 1.0 \times 0.3 \times 0.2 \times 1.0 \times 1.0 \times 1.0$$

Properties of PCFGs

- Assigns a probability to each parse-tree, allowed by the underlying CFG
- Say we have a sentence s, set of derivations for that sentence is $\mathcal{T}(s)$, as defined by a CFG. Then a PCFG assigns a probability $p(t)$ to each member of $\mathcal{T}(s)$.

- We now have a SCORE function (probability) that can ranks trees.
- The most likely parse tree for a sentence s is

$$\boxed{\arg\max_{t \in \mathcal{T}(s)} p(t)}$$

"correct" means more probable parse tree

"language" means set of grammatical sentences

Deriving a PCFG from a Treebank

Given a set of example trees (a treebank), the underlying CFG can simply be all rules seen in the corpus

Maximum Likelihood Estimates

$$q_{ML}(A \to \beta) = \frac{\textsc{count}(A \to \beta)}{\textsc{count}(A)}$$

The counts are taken from a training set of example trees.

If the training data is generated by a PCFG, then as the training data size goes to infinity, the maximum-likelihood PCFG will converge to the same distribution as the "true" PCFG.

Discriminative vs Generative

We have learned two probabilistic models:

- Log-linear: $P(Y|X)$
- PCFG: $P(X, Y)$

- Generative models can generate new data instances. It includes the distribution of the data itself, and tells you how likely a given example is.
- Discriminative models discriminate between different kinds of data instances. A discriminative model just tells you how likely a label is for a given instance.

Discriminative vs Generative

We have learned two probabilistic models:

- Log-linear: $P(Y|X)$ an example of discriminative model
- PCFG: $P(X, Y)$ an example of generative model

- Generative models can generate new data instances. It includes the distribution of the data itself, and tells you how likely a given example is.
- Discriminative models discriminate between different kinds of data instances. A discriminative model just tells you how likely a label is for a given instance.

Neural Parameterisation

Neural parameterisation

- Simple parameterisation of PCFG: $q(A_i \to \beta_i)$ is a real number.
- Alternative neural parameterisation: $q(A_i \to \beta_i)$ is a trainable NN.

Kim et al. (2019)

https://aclanthology.org/P19-1228.pdf

$$q(A \to BC) = \frac{\exp(\boldsymbol{u}_{BC}^\top \boldsymbol{w}_A)}{\sum_{B'C'} \exp(\boldsymbol{u}_{B'C'}^\top \boldsymbol{w}_A)}$$

- $A \in \Sigma$, $B, C \in \Sigma \cup N$
- $\{\boldsymbol{w}_x | x \in \Sigma \cup N\}$ and $\{\boldsymbol{u}_{xy} | x, y \in \Sigma \cup N\}$: the set of input symbol embeddings for a grammar.

Compound PCFG

$$q(A \to BC; \boldsymbol{z}) = \frac{\exp(\boldsymbol{u}_{BC}^\top [\boldsymbol{w}_A; \boldsymbol{z}])}{\sum_{B'C'} \exp(\boldsymbol{u}_{B'C'}^\top [\boldsymbol{w}_A; \boldsymbol{z}])}$$

Grammar induction

- Children acquire their grammars in a more or less unsupervised manner.
- Grammar induction is a research task in computational linguistics with the following research question: to what extent can the structure of human language be distributionally identified?
- Recent progress: Applying neural parameterisation to PCFG and estimate parameters in an unsupervised way.

Reading

D Jurafsky and J Martin. *Speech and Language Processing*.

- §17.1–§17.5, and §17.8. Context-free Grammars and Constituency Parsing. *Speech and Language Processing*. D Jurafsky and J Martin. `https://web.stanford.edu/~jurafsky/slp3/17.pdf`