

COMP/LING 345

From Natural Language to Data Science

Tokenization and Regular Expressions

Siva Reddy



Credits: Some Slides are adapted from Dan Jurafsky's CS124 course

Outline

- Tokenization
- Regular expressions
- Corpus Query Language
- Tree Regular Expressions

Tokenization

- Most text processing starts with tokenization
 - Tokenize text into words and sentences
 - Normalize tokens into meaningful words

C'mon man, @LoriLightfoot, it's a peaceful #protest!!!

identifying meaningful units

Tokenization

- Most text processing starts with tokenization
 - Tokenize text into words and sentences
 - Normalize tokens into meaningful words

C'mon man, @LoriLightfoot, it's a peaceful #protest!!!

C'm on man , @LoriLightfoot , it 's a peaceful #protest !!!

Tokenization

- Most text processing starts with tokenization
 - Tokenize text into words and sentences
 - Normalize tokens into meaningful words

C'mon man, @LoriLightfoot, it's a peaceful #protest!!!

C'm on man , @LoriLightfoot , it 's a peaceful #protest !!!

Come on man , @LoriLightfoot , it is a peaceful #protest !!!

We will mainly deal with English language here. Tokenization is hard for languages like Chinese, Thai.

We will mainly deal with English language here. Tokenization is hard for languages like Chinese, Thai.

ตัวอย่าง|การเขียน|ภาษาไทย

Word count

Types are the distinct words in a corpus,
whereas **tokens** are the running words.

total number of words

- How many words are in the above sentence?
 - **Types**: 11 (13 with punctuation included)
 - **Tokens**: 14 (16 with punctuation)

Word count

Types are the distinct words in a corpus,
whereas **tokens** are the running words.

- How many words are in the above sentence?
 - **Types:** 11 (13?)
 - **Tokens:** 14 (16?)

Corner cases

- I lived in **San Fransisco.** *entity names*
 - Ideally *San Fransisco* should be treated as one word
- **Apostrophe's are complicated**
 - **Looks like a single word but syntactically two words**
 - c'mon -> **c'm** on -> come on
 - don't -> do **n't** -> do not
 - you'll -> you **'ll** -> you will*} need to create rules*

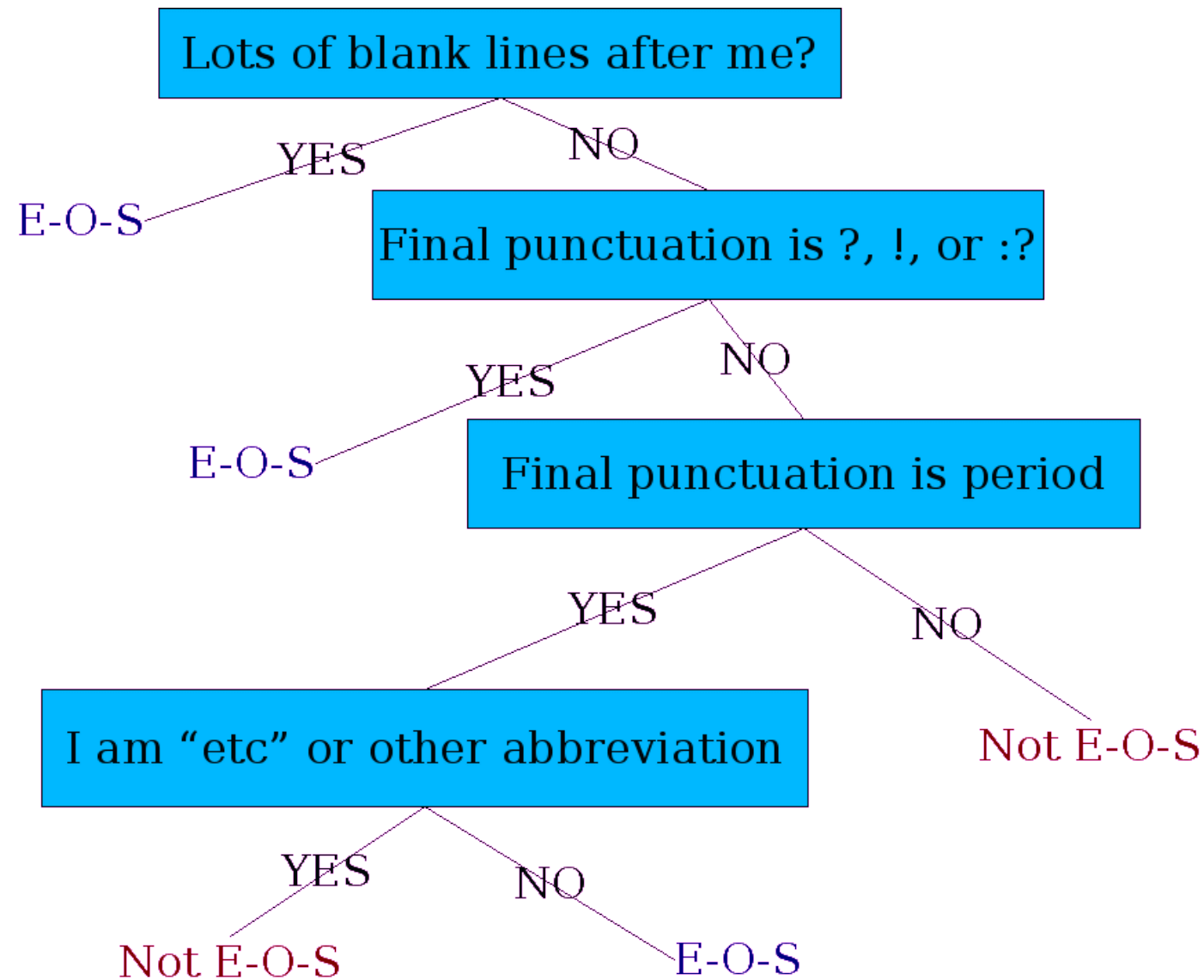
Sentence boundary

- ., ?, ! are commonly used for sentence boundaries in English.
- But it isn't simple; "." in e.g. is not the end of the sentence.
- Mr. Ms. Dr. Prof. end with . but are not sentence end markers.

special cases

doesn't mark the end of a sentence in every case

Sentence boundary



Regular expressions

*With one rule
you can match
lots of things*

- A formal language for specifying text strings
- How can we search for any of these?
 - woodchuck
 - Woodchuck
 - woodchucks
 - Woodchucks
 - woodchuckss
 - Woodchuckss
 -



Regular Expressions: Disjunctions

- Letters inside square brackets []

Pattern	Matches
[wW]oodchuck	Woodchuck, woodchuck
[1234567890]	Any digit

Regular Expressions: Disjunctions

- Letters inside square brackets []

Pattern	Matches
<code>[wW]oodchuck</code>	Woodchuck, woodchuck
<code>[1234567890]</code>	Any digit

- Ranges

Pattern	Matches	
<code>[A-Z]</code>	An upper case letter	<u>D</u> renched Blossoms
<code>[a-z]</code>	A lower case letter	<u>m</u> y beans were impatient
<code>[0-9]</code>	A single digit	Chapter <u>1</u> : Down the Rabbit

Regular expressions: ? * + .

Pattern	Matches	
colou?r	Optional previous char	<u>color</u> <u>colour</u>
oo*h!	0 or more of previous char	<u>oh!</u> <u>ooh!</u> <u>oooh!</u> <u>ooooh!</u>
o+h!	1 or more of previous char	<u>oh!</u> <u>ooh!</u> <u>oooh!</u> <u>ooooh!</u>
baa+		<u>baa</u> <u>baaa</u> <u>baaaa</u> <u>baaaaa</u>
beg.n		<u>begin</u> <u>begun</u> <u>begun</u> <u>beg3n</u>

Exercise

How can we search for any of these?

- woodchuck
- Woodchuck
- woodchucks
- Woodchucks
- woodchuckss
- Woodchuckss
-

Exercise

How can we search for any of these?

- woodchuck
- Woodchuck
- woodchucks
- Woodchucks
- woodchuckss
- Woodchuckss
-

`[wW]oodchucks*`

Exercise

[Exercise \(Click me\)](#)

Regular Expressions: Negation in Disjunction

- Negations `[^Ss]`
 - Carat means negation only when first in []

Pattern	Matches	
<code>[^A-Z]</code>	Not an upper case letter	Oyfn pripetchik
<code>[^Ss]</code>	Neither 'S' nor 's'	<u>I</u> have no exquisite reason"
<code>[^e^]</code>	Neither e nor ^	Look <u>h</u> ere
<code>a^b</code>	The pattern a carat b	Look up <u>a^b</u> now

[Exercise \(click me\)](#)

Regular Expressions: More Disjunction

- Woodchucks is another name for groundhog!
- The pipe | for disjunction



Pattern	Matches
<code>groundhog woodchuck</code>	
<code>(yours mine)? truly</code>	<code>truly, yours truly,</code> <code>mine truly</code>
<code>a b c</code>	<code>= [abc]</code>
<code>[gG]roundhog [Ww]oodchuck</code>	

Exercise

- Which of the following matches regexp `/(very)+(smart)?(tall|kind) man/`
 - very smart man
 - smart tall man
 - very very smart kind man
 - very very very tall man

Exercise

- Find me all instances of the word “the” in a text.
- Expression: **the**

Exercise

- Find me all instances of the word “the” in a text.
- Expression: **the**
 - misses capitalized examples

Exercise

- Find me all instances of the word “the” in a text.
- Expression: `the`
 - misses capitalized examples
- Expression: `[Tt]he`

Exercise

- Find me all instances of the word “the” in a text.
- Expression: `the`
 - misses capitalized examples
- Expression: `[Tt]he`
 - Incorrectly returns **other** or **theology**

Exercise

- Find me all instances of the word “the” in a text.
- Expression: `the`
 - misses capitalized examples
- Expression: `[Tt]he`
 - Incorrectly returns **other** or **theology**
- Expression: `[^a-zA-Z][Tt]he[^a-zA-Z]`

Drawbacks of regular expressions

- Patterns could become ugly 😞
- Find all forms of run using regular expression: `[Rr]un`, `[Rr]unning`, `[Rr]an`, `[Rr]uns`
 - `[Rr](uns?|an|unning)`

Drawbacks of regular expressions

- Patterns could become ugly 😬
- Find all forms of run using regular expression: `[Rr]un`, `[Rr]unning`, `[Rr]an`, `[Rr]uns`
 - `[Rr](uns?|an|unning)`

Word	↓ Frequency
run	239,819
running	157,181
runs	79,247
ran	48,365
Run	10,406
Running	6,021
Runs	810
Ran	653

Drawbacks of regular expressions

- Patterns could become ugly 😬
- Find all forms of run using regular expression: `[Rr]un`, `[Rr]unning`, `[Rr]an`, `[Rr]uns`
 - `[Rr](uns?|an|unning)`
- Becomes uglier for words like `eat`
- Linguistic knowledge comes to our rescue

Word	↓ Frequency
run	239,819
running	157,181
runs	79,247
ran	48,365
Run	10,406
Running	6,021
Runs	810
Ran	653

Linguistic knowledge: lemma

- Lemma is the canonical form of different word forms (also known as root, headword)

Joe Biden be run for president of the United States .
Joe Biden is running for president of the United States .

lemmatizer

[1] CQL: <https://www.sketchengine.eu/documentation/corpus-querying/>

Corpus Query Language

- Corpus Query Language (CQL)
allows us to use regular
expressions with linguistic
knowledge [1]
- Find all word forms of **run**:
`[word="[Rr](uns?|an|unning)"]`

[1] CQL: <https://www.sketchengine.eu/documentation/corpus-querying/>

Corpus Query Language

- Corpus Query Language (CQL)
allows us to use regular
expressions with linguistic
knowledge [1]

- Find all word forms of **run**:
`[word="[Rr](uns?|an|unning)"]`

- Find all word forms of **run**:
`[lemma="run"]`

same results

[1] CQL: <https://www.sketchengine.eu/documentation/corpus-querying/>

Corpus Query Language

- Corpus Query Language (CQL) allows us to use regular expressions with linguistic knowledge [1]
- Find all word forms of **run**:
[word="[Rr](uns?|an|unning)"]
- Find all word forms of **run**:
[lemma="run"]

Word	↓ Frequency
run	239,819
running	145,217
runs	79,247
ran	48,365
Running	4,204
Run	3,670
Ran	179
Runs	166
RUN	124

[1] CQL: <https://www.sketchengine.eu/documentation/corpus-querying/>

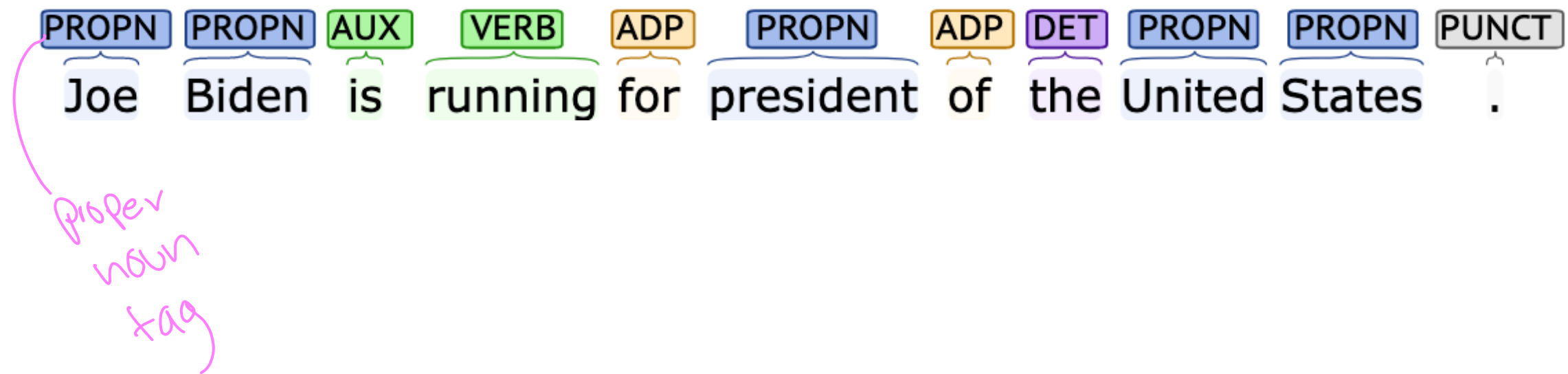
Linguistic knowledge:

Syntactic category

- Words can be classified into different syntactic categories (also known as part of speech)
 - noun, verb, adjective, adverb, determiner, adposition,

Linguistic knowledge: Syntactic category

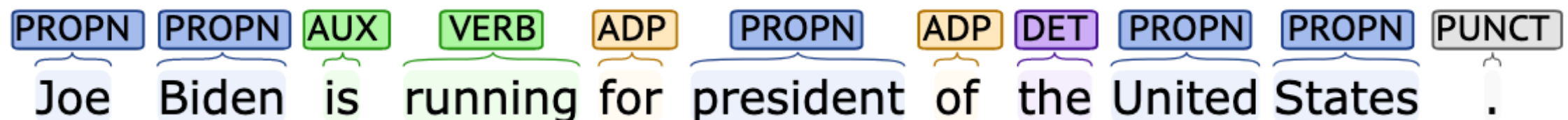
- Words can be classified into different syntactic categories (also known as part of speech)
 - noun, verb, adjective, adverb, determiner, adposition,
- Universal Dependencies Tagset



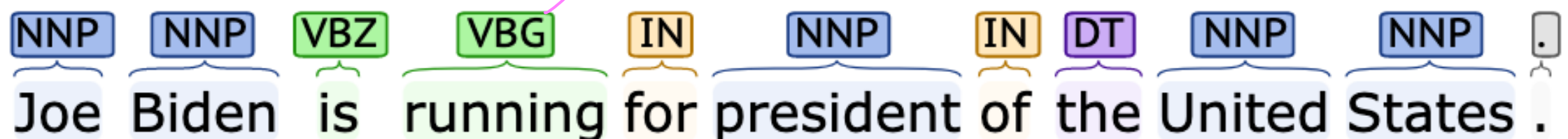
Linguistic knowledge: Syntactic category

- Words can be classified into different syntactic categories (also known as part of speech)
 - noun, verb, adjective, adverb, determiner, adposition,

- Universal Dependencies Tagset



- Penn Treebank Tagset



Linguistic knowledge: Syntactic category


Linguistic knowledge: Syntactic category

- Find all the instances of **run** as noun.

*Super hard
with
regular regex*

Linguistic knowledge:

Syntactic category

- Find all the instances of **run** as noun.
- [lemma="run" & tag="N.*"]


Linguistic knowledge:

Syntactic category

- Find all the instances of **run** as noun.
- `[lemma="run" & tag="N.*"]`

r high-fives or walk-off home	run	hugs. </s><s> Give us reasc
n scored a whopping 13,288	runs	in Test matches and 10,889
in Test matches and 10,889	runs	in One-Day Internationals. <
all reached mid-March on a	<u>run</u>	of three defeats in their last
at counts almost as a strong	run	. </s><s> In 13 matches con
/Ps, it's time for that 75-year	run	to end. </s><s> "Looking ba
ing stretched their unbeaten	run	in all competitions to 15 gar
th minute from a jinking solo	run	capped by a crisp low shot i
ed just 53 more passes than	runs	</s><s> Running backs Re

Exercise

main noun + modifier followed by other things

- Find all the noun phrases containing **virus**

$[tag = "DT"]$ $[tag = "J.* | N.*"]$ $[lemma = "virus"]$
determiner nouns or adjectives head noun

the corona virus

a deadly virus

the novel virus

the viruses

a new virus

the new virus

this deadly virus

Exercise

- Find all the noun phrases containing **virus**

the corona virus

a deadly virus

the novel virus

the viruses

a new virus

the new virus

this deadly virus

Exercise

- Find all the noun phrases containing **virus**
- `[tag="D.*"] [tag="N.*|J.*"]{0,3} [lemma="virus"]`

Handwritten notes:
noun adjective modifier
determiner 0-3 modifiers

the corona virus

a deadly virus

the novel virus

the viruses

a new virus

the new virus

this deadly virus

Exercise

- What are some things coronavirus has impacted?

Exercise

- What are some things coronavirus has impacted?
- [lemma="coronavirus"]
[]{0,3} [lemma="impact"]
[]{0,3} [tag="N.*"]

Exercise

- What are some things coronavirus has impacted?
- [lemma="coronavirus"]
[{}{0,3} lemma="impact"]
[{}{0,3} tag="N.*"]



Word	↓ Frequency
economy	135
industry	57
business	53
people	49
businesses	41
demand	31
communities	31
market	30
health	29
finances	28
world	27

Linguistic knowledge: beyond sequential regular expressions

- Corpus query language works sequentially.
- Even simple queries like finding subjects of a verb require complex patterns (often with many false positives).

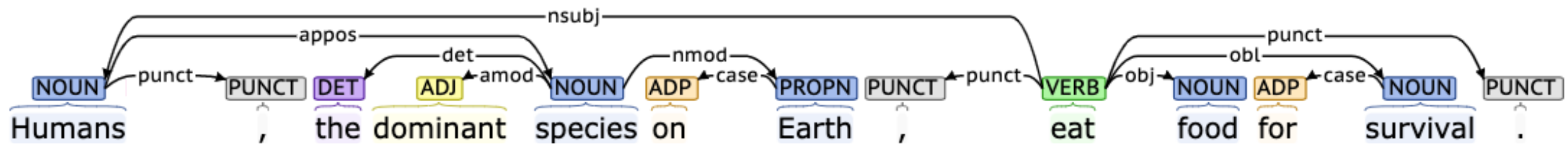
Humans eat food for survival.

Humans, the dominant species on Earth, eat food for survival

The food eaten by humans ...

Dependency trees

- Dependency trees indicate the syntactic relationship between words



<http://corenlp.run/>

<http://stanza.run/>

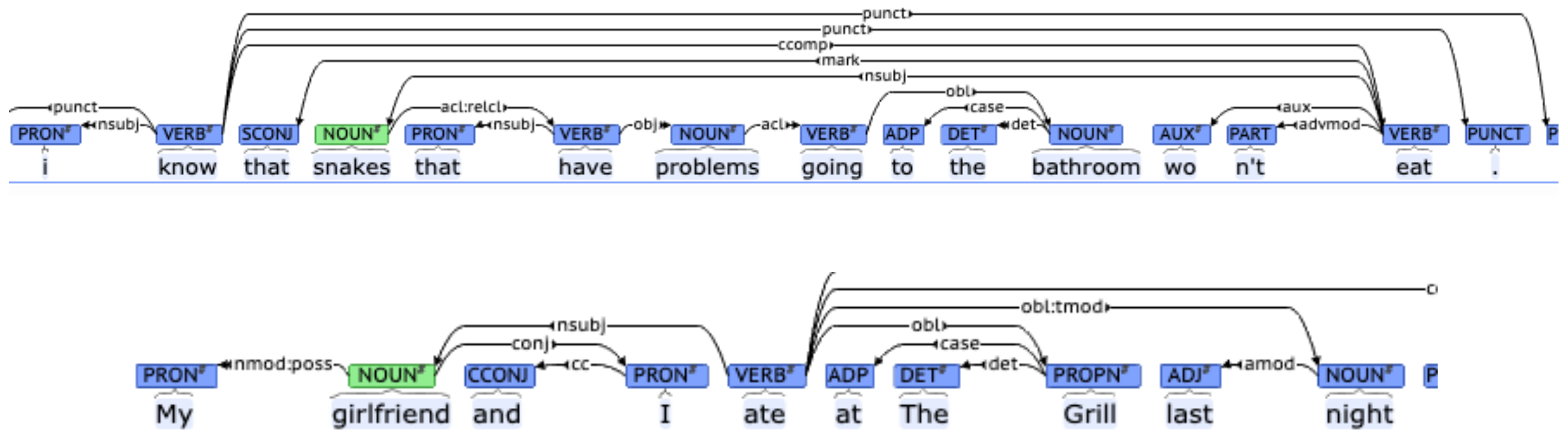
Tree regular expressions

- Tree Regular expressions allow us to run regular expressions on tree structures.

Symbol	Description	Symbol	Description
$A < B$	A is the parent of B	$A << B$	A is an ancestor of B
$A \$ B$	A and B are sisters	$A \$+ B$	B is next sister of A
$A <_i B$	B is i^{th} child of A	$A <: B$	B is only child of A
$A <<\# B$	A on head path of B	$A <<- B$	B is rightmost descendent
$A .. B$	A precedes B in depth-first traversal of tree		
$A <+(C) B$	A dominates B via unbroken chain of Cs		

Exercise

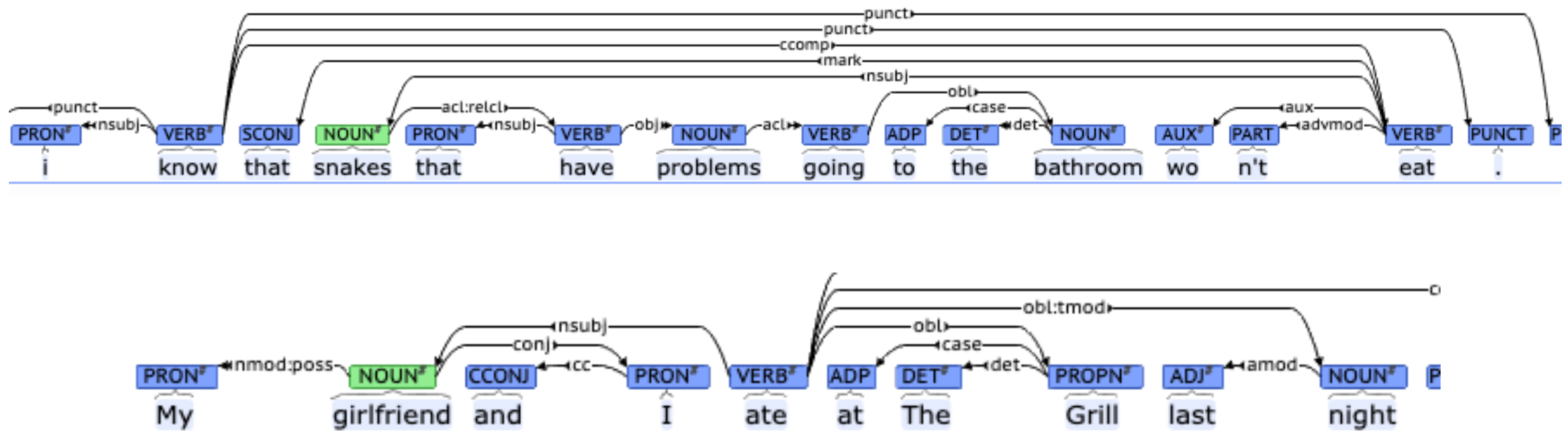
- Find all the subjects of “eat”



Use <https://corenlp.run>

Exercise

- Find all the subjects of “eat”
- `{pos:/N.*}/ <nsbj {word:/eat/}`



Use <https://corenlp.run>