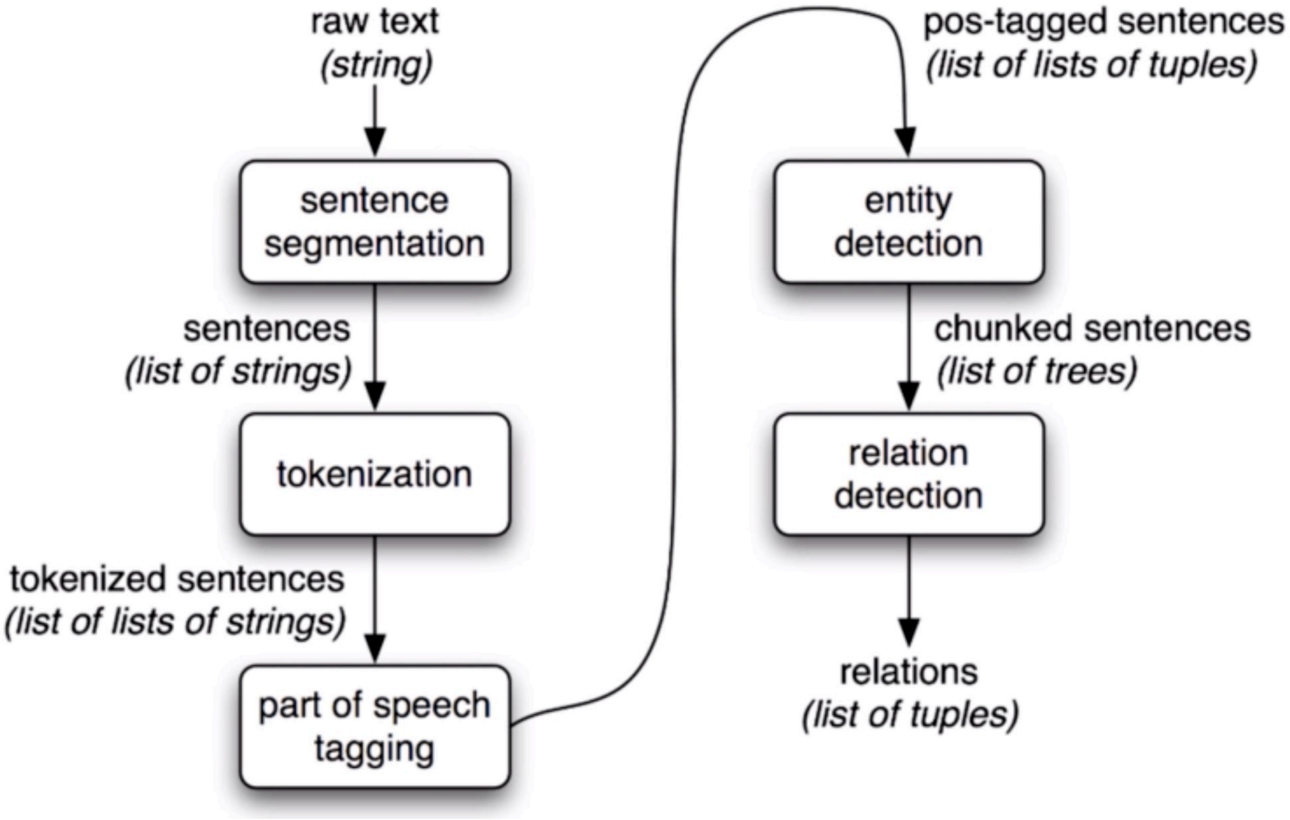
Natural Language Processing

Unit 5

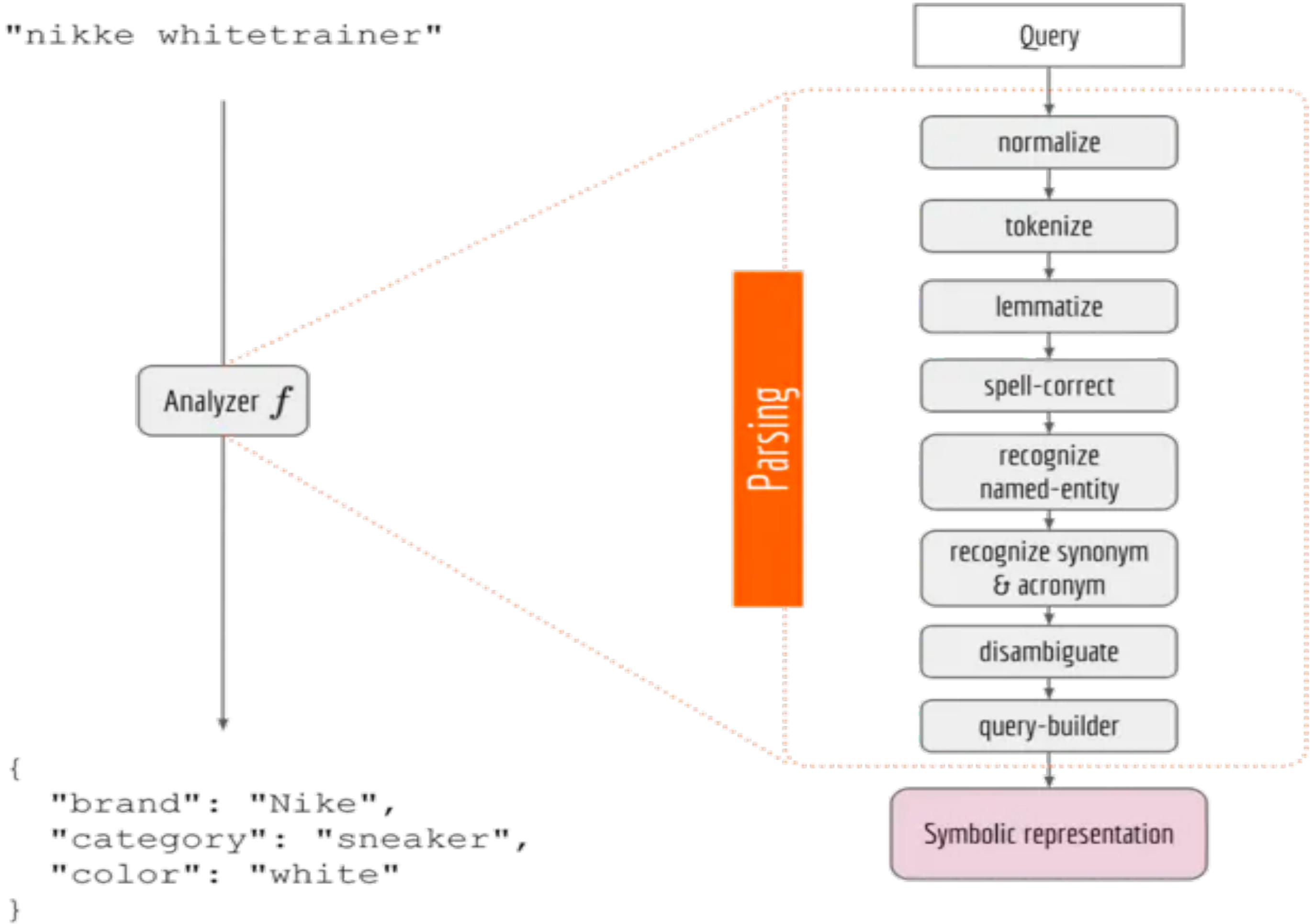
Processing Text

Faizan Javed
DataScience @ SMU

NLP pipelines



NLP pipelines



```
"nikke whitetrainer"
```

```
{
  "brand": "Nike",
  "category": "sneaker",
  "color": "white"
}
```

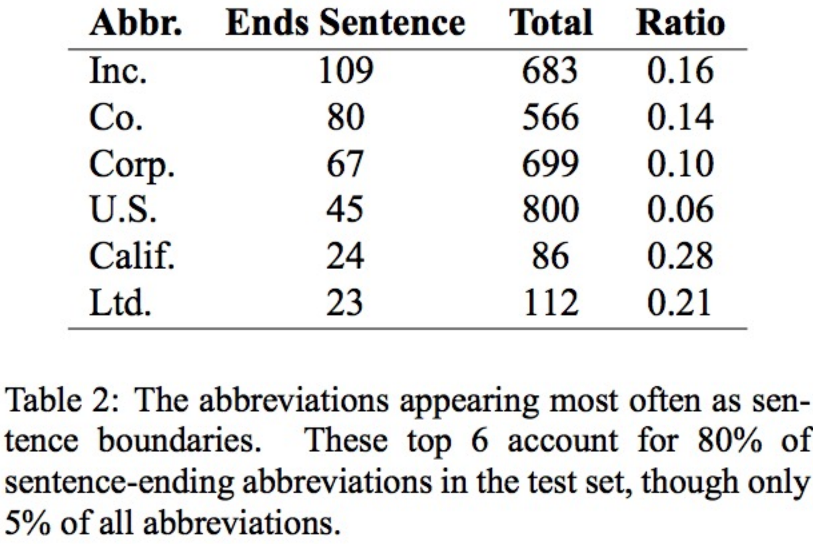
Sentence Segmentation

Sentence Boundary Detection and the Problem with the U.S., acl 2009 https://www.aclweb.org/anthology/N09-2061/

Process of deciding where sentences end and begin

Period (.) is the default sentence segmentation

Simple Examples,
U.S. Corporate
Mr. Kolmogrov
Kidding !
Christoper K. Columbus
For e.g.
"Are you kidding ?" asked the senator.
aftermath@gmail.cs.uva.edu



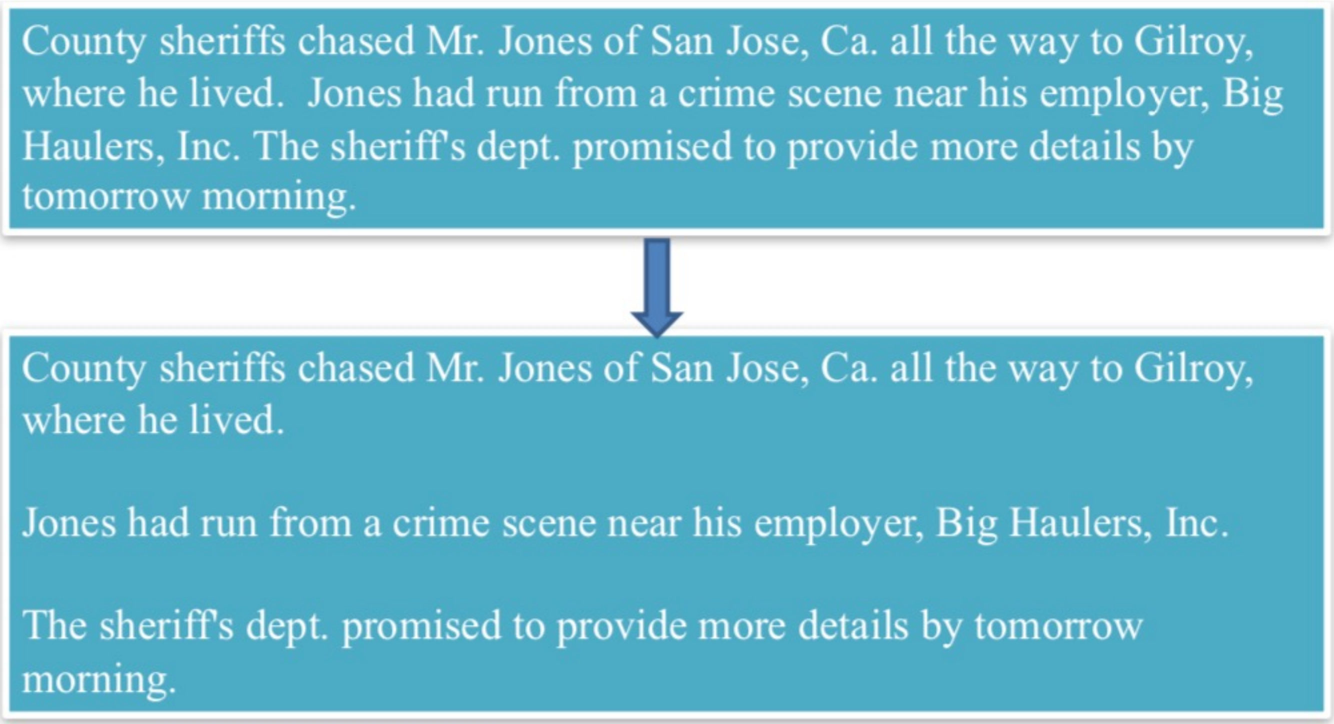
Abbr.	Ends Sentence	Total	Ratio
Inc.	109	683	0.16
Co.	80	566	0.14
Corp.	67	699	0.10
U.S.	45	800	0.06
Calif.	24	86	0.28
Ltd.	23	112	0.21

Table 2: The abbreviations appearing most often as sentence boundaries. These top 6 account for 80% of sentence-ending abbreviations in the test set, though only 5% of all abbreviations.

“Vanilla” approach that handles majority of the cases (95%):

(a) If it's a period, it ends a sentence.
(b) If the preceding token is in the hand-compiled list of abbreviations, then it doesn't end a sentence.
(c) If the next token is capitalized, then it ends a sentence.

Sentence Segmentation



County sheriffs chased Mr. Jones of San Jose, Ca. all the way to Gilroy, where he lived. Jones had run from a crime scene near his employer, Big Haulers, Inc. The sheriff's dept. promised to provide more details by tomorrow morning.

County sheriffs chased Mr. Jones of San Jose, Ca. all the way to Gilroy, where he lived.

Jones had run from a crime scene near his employer, Big Haulers, Inc.

The sheriff's dept. promised to provide more details by tomorrow morning.

Word tokenization

Token: word – sequence of characters

Tokenization: convert a string of characters into a sequence of tokens

Separator: tabs, spaces, line breaks

Punctuations, hyphens and apostrophes

Example:

I can't believe that the Carolina Hurricanes won the 2005-2006 Stanley Cup.

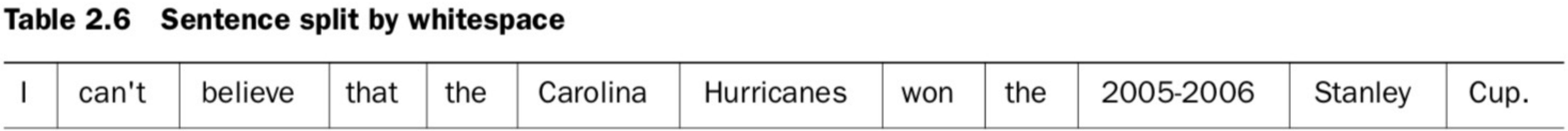


Table 2.6 Sentence split by whitespace

I	can't	believe	that	the	Carolina	Hurricanes	won	the	2005-2006	Stanley	Cup.

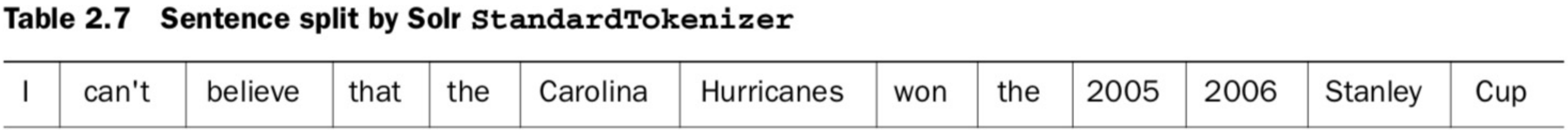


Table 2.7 Sentence split by Solr `StandardTokenizer`

I	can't	believe	that	the	Carolina	Hurricanes	won	the	2005	2006	Stanley	Cup

Lucene StandardTokenizer takes into account acronyms and punctuation (also removed period – search use case)

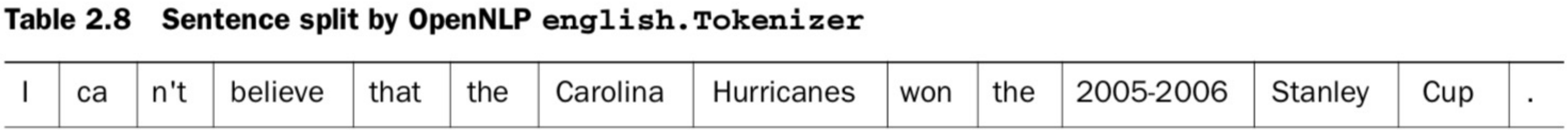


Table 2.8 Sentence split by OpenNLP `english.Tokenizer`

I	ca	n't	believe	that	the	Carolina	Hurricanes	won	the	2005-2006	Stanley	Cup	.

Punctuation is maintained and the contraction *can't* has been split up.
OpenNLP is using tokenization as a precursor to performing grammatical processing.
For this type of application, punctuation is useful for helping identify clause boundaries
and *can* and *not* have distinct grammatical roles.

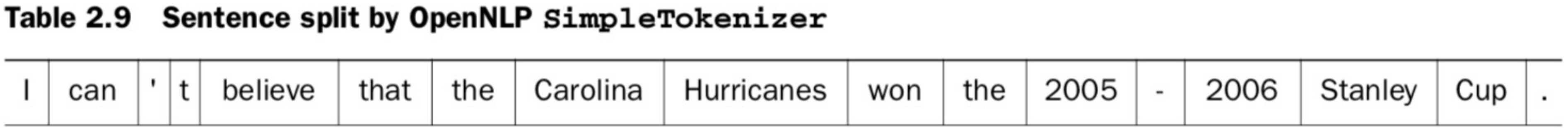


Table 2.9 Sentence split by OpenNLP `SimpleTokenizer`

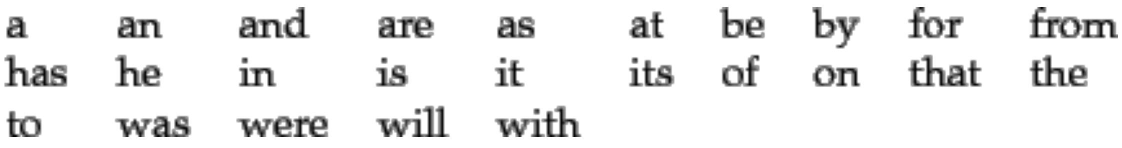
I	can	'	t	believe	that	the	Carolina	Hurricanes	won	the	2005	-	2006	Stanley	Cup	.

Named entity use case (splitting up by token class: alphabetic, numeric, whitespace, et al)
Splitting the date up allows application to identify each component as a date

- Small words (one or two characters) can be important in some queries, usually in combinations with other words. For example, xp, ma, pm, ben e king, el paso, master p, gm, j lo, world war II.[11]
- Both hyphenated and non-hyphenated forms of many words are common. In some cases the hyphen is not needed. For example, e-bay, wal-mart, active-x, cd-rom, t-shirts. At other times, hyphens should be considered either as part of the word or a word separator. For example, winston-salem, mazda rx-7, e-cards, pre-diabetes, t-mobile, spanish-speaking.
- Special characters are an important part of the tags, URLs, code, and other important parts of documents that must be correctly tokenized.
- Capitalized words can have different meaning from lowercase words. For example, "Bush" and "Apple".
- Apostrophes can be a part of a word, a part of a possessive, or just a mistake. For example, rosie o'donnell, can't, don't, 80's, 1890's, men's straw hats, master's degree, england's ten largest cities, shriner's.
- Numbers can be important, including decimals. For example, nokia 3250, top 10 courses, united 93, quicktime 6.5 pro, 92.3 the beat, 288358 (yes, this was a real query; it's a patent number).
- Periods can occur in numbers, abbreviations (e.g., "I.B.M.", "Ph.D."), URLs, ends of sentences, and other situations.

Stop words

Extremely common “function” words that are of little value in helping select documents matching a user need - **how are they discovered?**



a	an	and	are	as	at	be	by	for	from
has	he	in	is	it	its	of	on	that	the
to	was	were	will	with					

They are application specific – may add some content words to the list e.g., Mr., Mrs., Dr., etc

Should you always filter stop words?

“President of the United States” vs ”President United States”, “The Who”

Originally done to save space in the index, but some newer search engines no longer remove stop words since they can help in more advanced queries (e.g., Google BERT)

“Can you get medicine for someone pharmacy” would have resulted in general information about filling prescriptions at pharmacies.

With BERT, Google pays particular attention to the preposition **“for”.** Google now understands that you’re asking if a person can pick up someone else’s prescription from a pharmacy.

“2019 Brazil traveler to USA need a visa” would have resulted in information for USA citizens traveling to Brazil. Google would have ignored the word **“to”.**

With BERT, Google understands that the searcher wants information about visa requirements for Brazilians traveling to the USA.

Stemming and Lemmatization

Inflections: syntactic differences between words

Toy vs Toys , Universe vs University

Stemming: reducing a word to its root form

Lemmatization: uses lexical knowledge bases such as WordNet to obtain word base forms

Porter stemmer examples:
apple, apples --> appl
meanness, meaning --> mean

Stemming and Lemmatization

Algorithmic stemmer: uses a program to decide whether two words are related (based on word suffixes)

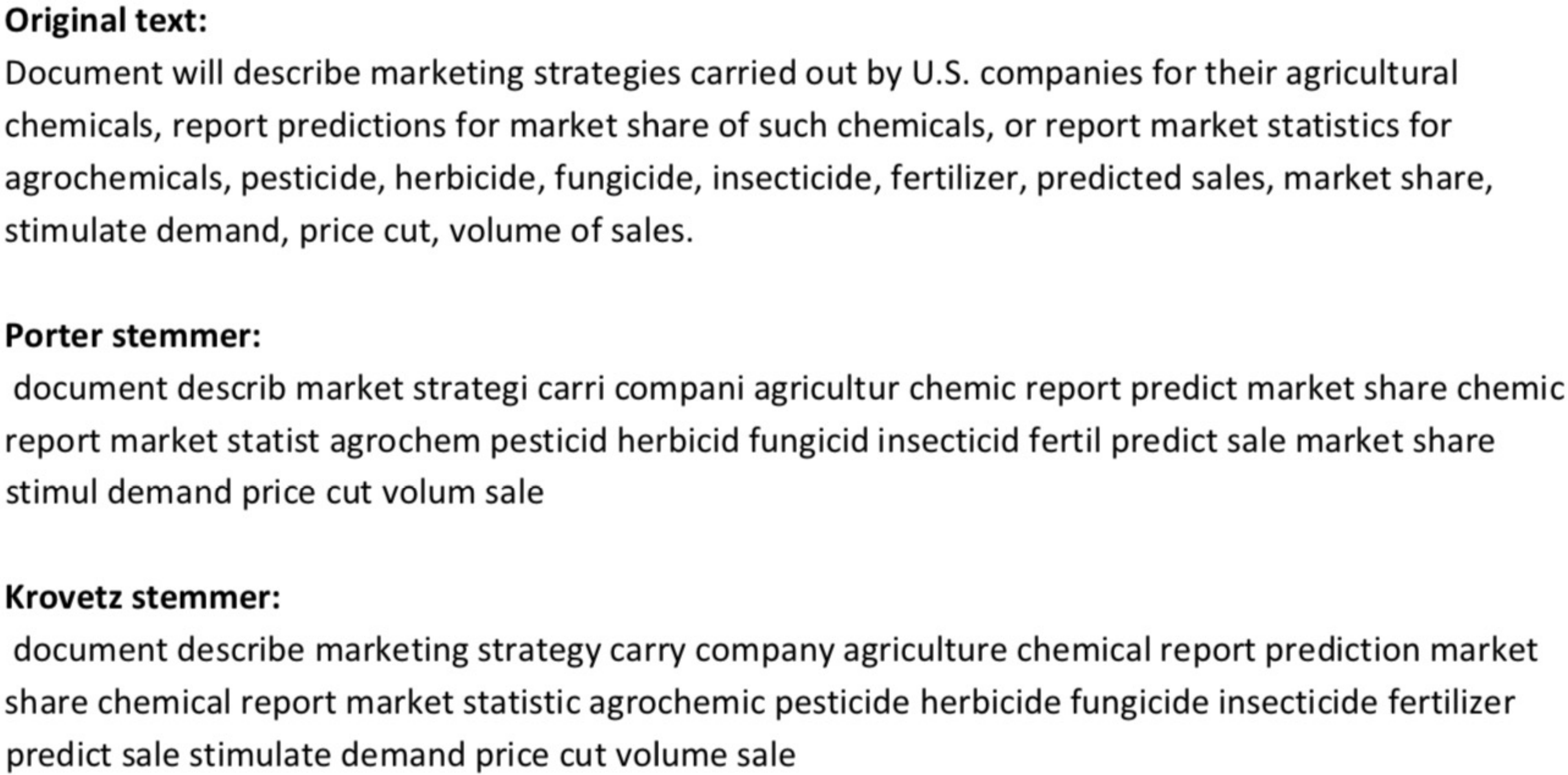
Dictionary-based stemmer: relies on pre-created dictionaries of related terms to store term relationships

Algorithmic stemmer examples:
e.g.1, **suffix-s** stemmer: any word ending with ”s” is plural.
cakes → cake, dogs→ dog; I → is (?)
can’t detect **century** and **centuries**.
Can be extended to handle suffixes such as -ing, -ed..

e.g. 2, Porter Stemmer; contains several steps
Step 1a:

- Replace sses by ss (e.g., stresses → stress).
- Delete s if the preceding word part contains a vowel not immediately before the s (e.g., gaps → gap but gas → gas).
- Replace ied or ies by i if preceded by more than one letter, otherwise by ie (e.g., ties → tie, cries → cri).
- If suffix is us or ss do nothing (e.g., stress → stress).

Stemming and Lemmatization



Original text:

Document will describe marketing strategies carried out by U.S. companies for their agricultural chemicals, report predictions for market share of such chemicals, or report market statistics for agrochemicals, pesticide, herbicide, fungicide, insecticide, fertilizer, predicted sales, market share, stimulate demand, price cut, volume of sales.

Porter stemmer:

document describ market strategi carri compani agricultur chemic report predict market share chemic report market statist agrochem pesticid herbicid fungicid insecticid fertil predict sale market share stimul demand price cut volum sale

Krovetz stemmer:

document describe marketing strategy carry company agriculture chemical report prediction market share chemical report market statistic agrochemic pesticide herbicide fungicide insecticide fertilizer predict sale stimulate demand price cut volume sale

Categorizing and Tagging words

Universal Part-of-Speech Tagset

Tag	Meaning	English Examples
ADJ	adjective	*new, good, high, special, big, local*
ADP	adposition	*on, of, at, with, by, into, under*
ADV	adverb	*really, already, still, early, now*
CONJ	conjunction	*and, or, but, if, while, although*
DET	determiner, article	*the, a, some, most, every, no, which*
NOUN	noun	*year, home, costs, time, Africa*
NUM	numeral	*twenty-four, fourth, 1991, 14:24*
PRT	particle	*at, on, out, over per, that, up, with*
PRON	pronoun	*he, their, her, its, my, I, us*
VERB	verb	*is, say, told, given, playing, would*
.	punctuation marks	*. , ; !*
X	other	*ersatz, esprit, dunno, gr8, univeristy*

Why is POS tagging hard?

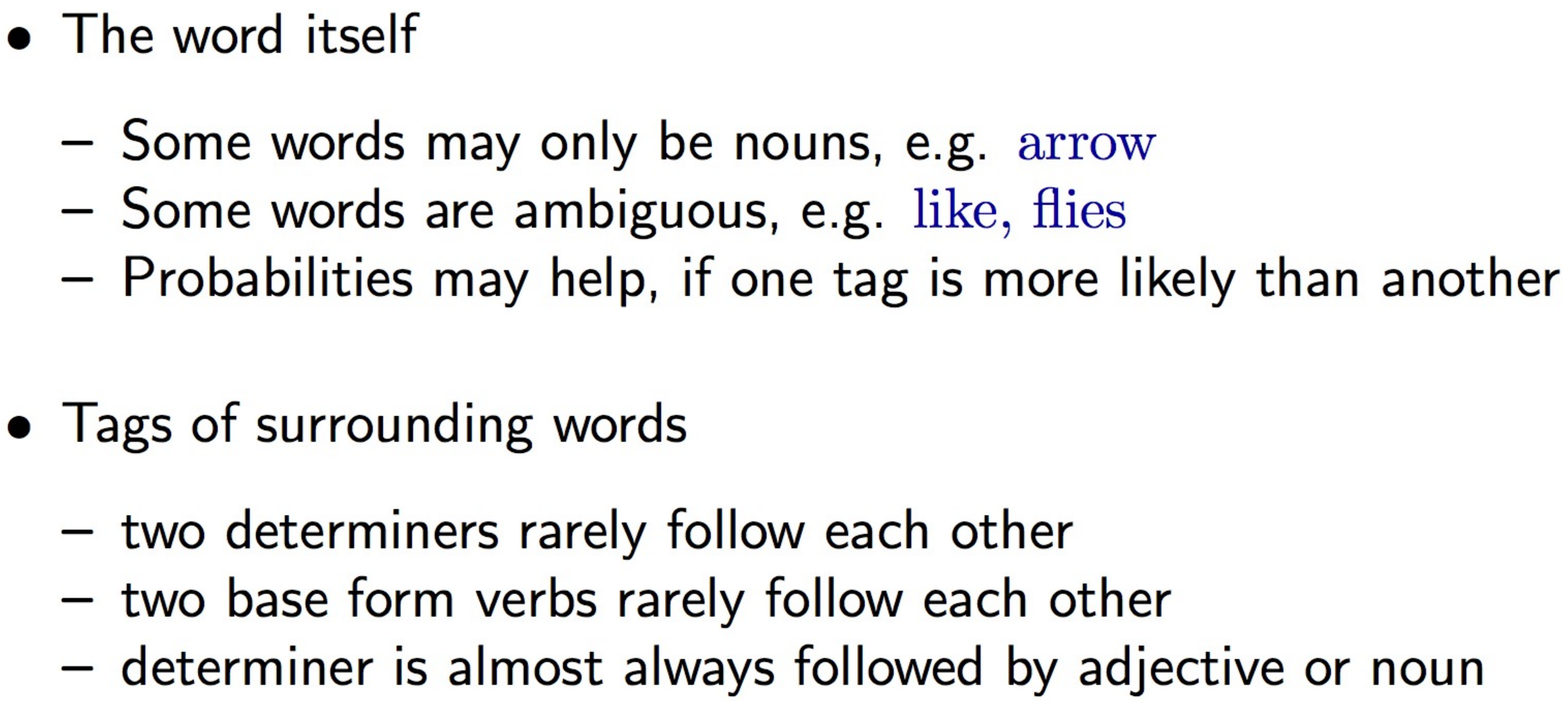
- Ambiguity:

  glass of water/NOUN vs. water/VERB the plants
  lie/VERB down vs. tell a lie/NOUN
  wind/VERB down vs. a mighty wind/NOUN (homographs)

  How about time flies like an arrow?

- Sparse data:
  - Words we haven't seen before (at all, or in this context)
  - Word-Tag pairs we haven't seen before (e.g., if we verb a noun)

Relevant knowledges for POS tagging models



- The word itself
  - Some words may only be nouns, e.g. arrow
  - Some words are ambiguous, e.g. like, flies
  - Probabilities may help, if one tag is more likely than another

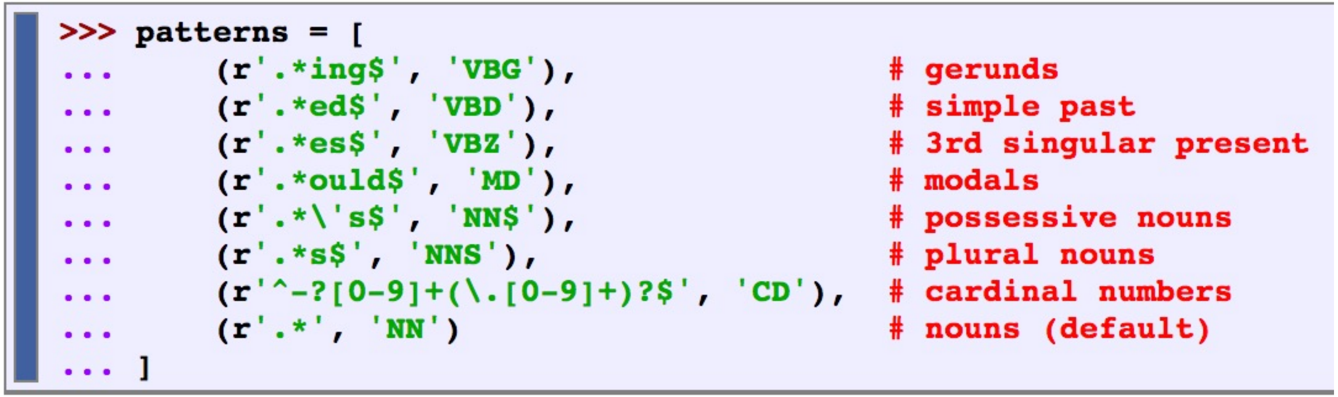
- Tags of surrounding words
  - two determiners rarely follow each other
  - two base form verbs rarely follow each other
  - determiner is almost always followed by adjective or noun

Automatic Tagging

The tag of a word depends on the word and its context within a sentence.

The Default Tagger: assigns the same tag to each token; good for establishing a baseline

The Regular expression tagger: assigns tokens based on matching patterns
E.g., any word ending in **ed** is the past principle of a verb, and any word ending with **‘s** is a possessive noun

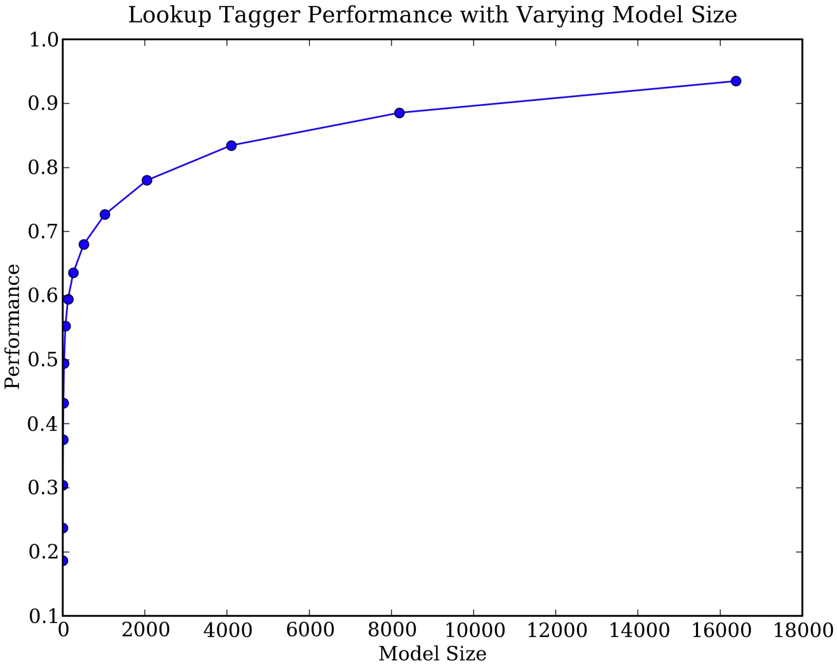


```
>>> patterns = [
...     (r'.*ing$', 'VBG'),               # gerunds
...     (r'.*ed$', 'VBD'),                # simple past
...     (r'.*es$', 'VBZ'),                # 3rd singular present
...     (r'.*ould$', 'MD'),               # modals
...     (r'.*\'s$', 'NN$'),               # possessive nouns
...     (r'.*s$', 'NNS'),                 # plural nouns
...     (r'^-?[0-9]+(\.[0-9]+)?$', 'CD'), # cardinal numbers
...     (r'.*', 'NN')                     # nouns (default)
... ]
```

Automatic Tagging

The tag of a word depends on the word and its context within a sentence.

The Lookup Tagger: Find the x most frequent words and store their most likely tag.
Backoff strategy: words that are not in the lookup list will be assigned ”None”.
For these cases, assign them the default tag.

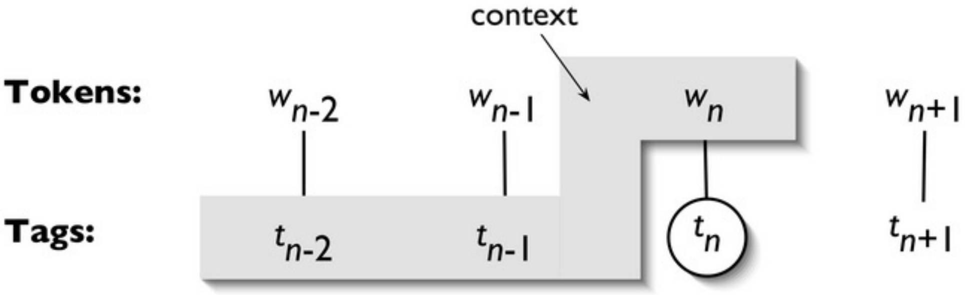


Automatic Tagging : N-Gram tagging

Unigram tagging: uses a statistical algorithm (using training data) to assign the most likely tag for a token

Uses only one item of context. E.g., would tag the word **wind** with the same tag for both ***the wind*** or ***to wind***.

N-gram tagger: generalization of the tagger whose context is the current word together with the POS tags of the n-1 preceding tokens. E.g., n= 3



What did you learn today?

Questions?