Tagsets

- What is a tagset?
- Standards and tagging
- Tags for parts of speech:
  - Nouns, verbs, adverbs, adjectives, articles, etc
  - Subtagging
    - nouns can be singular or plural
    - verbs have tenses
  - Different tagsets have different focuses

Tags are cryptic

```
>>> text = nltk.word_tokenize("And now for something completely different")
>>> nltk.pos_tag(text)
[('And', 'CC'), ('now', 'RB'), ('for', 'IN'), ('something', 'NN'),
('completely', 'RB'), ('different', 'JJ')]
```

- Brown and Treebank established some cryptic tags; everyone tends to use Treebank's
  - CC = coordinating conjunction
  - RB = adverb
  - IN = preposition
  - NN = noun
  - JJ = adjective

NLTK can help

- We don’t even remember what all the tags mean sometimes
- but nltk.help.upenn_tagset(*tag*) does!

Homographs

```
>>> text = nltk.word_tokenize("They
refuse to permit us to obtain the refuse
permit")

>>> nltk.pos_tag(text)
[('They', 'PRP'), ('refuse', 'VBP'),
('to', 'TO'), ('permit', 'VB'), ('us',
'PRP'), ('to', 'TO'), ('obtain', 'VB'),
('the', 'DT'), ('refuse', 'NN'),
('permit', 'NN')]
```

Tags in NLTK

```
>>> tagged_token = nltk.tag.str2tuple('fly/NN')
>>> tagged_token
('fly', 'NN')
>>> tagged_token[0]
'fly'
>>> tagged_token[1]
'NN'
```

- Tags are tuples
- Tags can be converted by NLTK between tagsets

Making tuples from a corpus

```
>>> sent = '''
... The/AT grand/JJ jury/NN commented/VBD on/IN a/AT
number/NN of/IN
... other/AP topics/NNS ,/, AMONG/IN them/PPO the/AT
Atlanta/NP and/CC
... Fulton/NP-tl County/NN-tl purchasing/VBG departments/
NNS which/WDT it/PPS
... said/VBD ``/`` ARE/BER well/QL operated/VBN and/CC
follow/VB generally/RB
... accepted/VBN practices/NNS which/WDT inure/VB to/IN
the/AT best/JJT
... interest/NN of/IN both/ABX governments/NNS ''/'' ./.
... '''
>>> [nltk.tag.str2tuple(t) for t in sent.split()]
[('The', 'AT'), ('grand', 'JJ'), ('jury', 'NN'),
('commented', 'VBD'),
('on', 'IN'), ('a', 'AT'), ('number', 'NN'), ... ('.', '.')
```

Many Tag Sets

- Different corpora have different conventions for tagging.
- There are ISO standards for tagging…
- There were many boring meetings
- NLTK made a simplified, unified tagset
- … which no one uses.

Simplified Tagset of NLTK

Tag	Meaning	Examples
ADJ	adjective	*new, good, high, special, big, local*
ADV	adverb	*really, already, still, early, now*
CNJ	conjunction	*and, or, but, if, while, although*
DET	determiner	*the, a, some, most, every, no*
EX	existential	*there, there's*
FW	foreign word	*dolce, ersatz, esprit, quo, maitre*
MOD	modal verb	*will, can, would, may, must, should*
N	noun	*year, home, costs, time, education*
NP	proper noun	*Alison, Africa, April, Washington*
NUM	number	*twenty-four, fourth, 1991, 14:24*
PRO	pronoun	*he, their, her, its, my, I, us*
P	preposition	*on, of, at, with, by, into, under*
TO	the word *to*	*to*
UH	interjection	*ah, bang, ha, whee, hmpf, oops*
V	verb	*is, has, get, do, make, see, run*
VD	past tense	*said, took, told, made, asked*
VG	present participle	*making, going, playing, working*
VN	past participle	*given, taken, begun, sung*
WH	*wh* determiner	*who, which, when, what, where, how*

Tagging in Other Languages

```
>>> nltk.corpus.sinica_treebank.tagged_words()
[('\xe4\xb8\x80', 'Neu'), ('\xe5\x8f\x8b\xe6\x83\x85', 'Nad'), ...]
>>> nltk.corpus.indian.tagged_words()
[('\xe0\xa6\xae\xe0\xa6\xb9\xe0\xa6\xbf\xe0\xa6\xb7\xe0\xa7\x87\xe0\xa6\xb0', 'NN'),
('\xe0\xa6\xb8\xe0\xa6\xa8\xe0\xa7\x8d\xe0\xa6\xa4\xe0\xa6\xbe\xe0\xa6\xa8', 'NN'),
...]
>>> nltk.corpus.mac_morpho.tagged_words()
[('Jersei', 'N'), ('atinge', 'V'), ('m\xe9dia', 'N'), ...]
>>> nltk.corpus.conll2002.tagged_words()
[('Sao', 'NC'), ('Paulo', 'VMI'), ('(', 'Fpa'), ...]
>>> nltk.corpus.cess_cat.tagged_words()
[('El', 'da0ms0'), ('Tribunal_Suprem', 'np0000o'), ...]
```

```
Bangla: কুঁড়েঘরগুলির/'NN' আকার/'NN' বাংলার/'NNP' বা/'CC' ভারতের/'NNP' ?/None
ন্য/'JJ' ?/None ঞ্চলের/'NN' প্রচলিত/'JJ' কুঁড়ে/'NN' ঘর/'NN' নয়/'VM' [illegible]/'SYM'
Hindi: पाकिस्तान/'NNP' की/'PREP' पूर्व/'JJ' प्रधानमंत्री/'NN' बेनजीर/'NNPC' भुट्टो/'NNP'
पर/'PREP' लगे/'VFM' भ्रष्टाचार/'NN' के/'PREP' आरोपों/'NN' के/'PREP' खिलाफ/'PREP' भुट्टो/'NNP'
द्वारा/'PREP' दायर/'NVB' की/'VFM' गई/'VAUX' याचिका/'NN' की/'PREP' सुनवाई/'NN'
मंगलवार/'NN' को/'PREP' वकीलों/'NN' की/'PREP' हड़ताल/'NN' के/'PREP' कारण/'PREP'
स्थगित/'JVB' कर/'VFM' दी/'VAUX' गई/'VAUX' ।/'PUNC'
Marathi: ग्रामीण/'JJ' जिल्हाध्यक्ष/'NN' बाळासाहेब/'NNPC' भोसले/'NNP' यांच्या/'PRP' ?/None
ध्यक्षतेखाली/'NN' पक्षाची/'NN' आज/'NN' बै?/None क/'NN' झाली/'VM' ./'SYM'
Telugu: ఖాతాదారుల/'NN' నుంచి/'PREP' వచ్చిన/'VJJ' పత్రాల/'NN' ను/'PREP' సాక్ష్యంగా/'NN'
```

Exercise 1 (simple, 10min)

- Take any book nltk.books or nltk.gutenberg, and part of speech tag it.
  - Start from .raw() text
  - Segment into sentences, then into words.
  - POS-tag the words
- Plot a frequency distribution of the POS-tags
- Plot a frequency distribution of all the nouns in the document

Named Entity Recognition

- See next slides, by Stephan Lesch

  For full slides see:
  http://courses.ischool.berkeley.edu/i290-2/f04/lectures/ner2.ppt

- Example:

  `sentence = "`**`Coca-Cola`**` spokesman `**`Mike Tyson`**` said they will buy `**`Microsof`**`t today."`
  → Coca-Cola: ORG, Mike Tyson: PERSON, Microsoft: ORG

The who, where, when & how much in a sentence

The task: identify atomic elements of information in text

- person names
- company/organization names
- locations
- dates&times
- percentages
- monetary amounts

example from MUC-7

Delimit the named entities in a text and tag them with NE categores:

<ENAMEX TYPE=„LOCATION“>Italy</ENAMEX>‘s business world was rocked by the announcement <TIMEX TYPE=„DATE“>last Thursday</TIMEX> that Mr. <ENAMEX TYPE=„PERSON“>Verdi</ENAMEX> would leave his job as vice-president of <ENAMEX TYPE=„ORGANIZATION“>Music Masters of Milan, Inc</ENAMEX> to become operations director of <ENAMEX TYPE=„ORGANIZATION“>Arthur Andersen</ENAMEX>.

- „Milan“ is part of organization name
- „Arthur Andersen“ is a company
- „Italy“ is sentence-initial => capitalization useless

difficulties

- too numerous to include in dictionaries
- changing constantly
- appear in many variant forms
- subsequent occurrences might be abbreviated

⇒ list search/matching doesn‘t perform well

Whether a phrase is a proper name, and what name class it has, depends on

- Internal structure:
  „<u>Mr.</u> Brandon“
- Context:
  „The new <u>company</u>, SafeTek, will make air bags.“

Applications

- Information Extraction
- Summary generation
- Machine Translation
- document organization/classification
- automatic indexing of books
- increase accuracy of Internet search results (location Clinton/South Carolina vs. President Clinton)

The hand-crafted approach

uses hand-written context-sensitive reduction rules:

1) title capitalized word => title person_name
   compare „Mr. Jones“ vs. „Mr. Ten-Percent“
   => no rule without exceptions
2) person_name, „the“ adj* „CEO of“ organization
   „Fred Smith, the young dynamic CEO of BlubbCo“
   => ability to grasp non-local patterns

plus help from databases of known named entities

Word features

- Easily determinable token properties:

Feature	Example	Intuition
fourDigitNum	1990	four digit year
containsDigitAndAlpha	A123-456	product code
containsCommaAndPeriod	1.00	monetary amount, percentage
otherNum	34567	other number
allCaps	BBN	Organisation
capPeriod	M.	Person name initial
firstWord	first word of sentence	ignore capitalization
initCap	Sally	capitalized word
lowerCase	can	uncapitalized word
other	,	punctuation, all other words

Histories, bin. features & futures

- History h_t: information derivable from the corpus relative to a token t:
  - text window around token w_i, e.g. $w_{i-2},\ldots,w_{i+2}$
  - word features of these tokens
  - POS, other complex features
- Binary features: yes/no-questions on history used by models to determine probabilities of
- Futures: name classes

Next step: Co-reference Resolution

- “**Joe** walks with **his girlfriend**. **She** likes tea.”
- Co-ref resolution important to understand longer texts.
- Step 1: find mentions
- Step 2: cluster connected mentions (which refer to a real-world “thing”)

- “**Mike** walks with **his** **girlfriend**. **She** likes tea.”

Co-reference Resolution – **Step 1: Extract Mentions**

- Mentions refer to something concrete in the world
- Mentions are (for example)
  - **Named Entities**
  - **Pronouns:** He, she, they, his, herself, etc.
  - **Common noun** mentions: the tree, her son, man, the park, the naughty child, …
  - **Embedded mentions:** her son
  - **Other nested mentions:** the CFO of IBM

Co-reference Resolution - **Step 2: Create Mention Chains / Clustering**

- Create mention chains
  - → connect mentions which refer to the some real world thing
- Special case: split antecedent.
  "**Mike** and **Ann** go to **Moscow**. **They** take **a plane**."
- Often not clear how to annotate text, if things point to something in the real world.
- **"Co-referent"** if pointing to the some real-world entity

Coreference Resolution: **Applications**

- Text understanding
  - Eg. understanding a discourse structure, eg in our book series: In conversations often back-and-forth with pronouns, etc.
- Machine translation: eg. some languages have no gender in pronouns
- Information / Relation Extraction, question answering, ...
  - ”**Mike** and **Ann** go to **Moscow**. **They** take **a plane**.”
  - Q: **Who** flew to Moscow?
  - A: They.

Coref Demo: Neural Coref (intergrated in Spacy)

- https://huggingface.co/coref/
  - Let’s try it out
- Demo contains info on how it works and how to train a model (if you are curious)
- Source code is at:
  - https://github.com/huggingface/neuralcoref

Systems 1: Neural Coref

- https://github.com/huggingface/neuralcoref
  - Integrated with spaCy (v2)
  - Easy to use and good accuracy
- Installation:
  - Download spaCy model (for coref, see github)
  - `pip install MODEL_URL`
- Usage:
  - nlp = spacy.load('en_coref_md') # we discuss spacy later
- Misc:
  - Visualisation code: https://github.com/huggingface/neuralcoref-viz

Standford CoreNLP

- Overview: https://nlp.stanford.edu/projects/coref.shtml
- Software: https://stanfordnlp.github.io/CoreNLP/coref.html
- Java-based
- See website for details

- Can also be loaded from within NLTK

Introduction to spaCy

- URL: https://spacy.io/
- Features (they claim):
  - Fast
  - Easy-to-use
  - Mature, gets things done
  - Also for industrial usage
  - Easy to integrate with deep learning
- Python-based

spaCy Installation

- Basics:
  - `bash$ pip install -U spacy`
  - `spacy download en`
- Or with a virtual env (see spacy website)

- First steps:

```
import spacy
nlp = spacy.load('en') # load model

# analyse a document with the model
doc = nlp(u'This is a sentence.')
```

spaCy: First Steps

```
text = open('war_and_peace.txt').read()
doc = nlp(text)

# Find named entities, phrases and concepts
for entity in doc.ents:
    print(entity.text, entity.label_)


# Determine semantic similarities
doc1 = nlp(u'the fries were gross')
doc2 = nlp(u'worst fries ever')
doc1.similarity(doc2)


# Hook in your own deep learning models - or Coref
nlp.add_pipe(load_my_model(), before='parser')
```

ITMO UNIVERSITY

spaCy: visual display

```
from spacy import displacy

doc_ent = nlp(u'When Sebastian Thrun started working on self-driving cars at Google '
              u'in 2007, few people outside of the company took him seriously.')

displacy.serve(doc_ent, style='ent')
```

Examples at: https://explosion.ai/demos/

ITsMOre than a UNIVERSITY

SpaCy: more functions

```
doc = nlp(u"Apple and banana are similar. Pasta and hippo
aren't.")

apple = doc[0]
banana = doc[2]
pasta = doc[6]
hippo = doc[8]

assert apple.similarity(banana) > pasta.similarity(hippo)
assert apple.has_vector, banana.has_vector,
pasta.has_vector, hippo.has_vector
```

ITMO UNIVERSITY

Thank you!

Questions?

Nov 2017