

Higher-level Natural Language Processing with textacy

Burton DeWilde / Chartbeat
PyGotham 2016

https://github.com/bdewilde/pygotham_2016

whoami

- particle physicist turned data scientist
- at intersection of media, data, and social good
- Python coder since 2009 (Py3 since 2014 )
- work at Chartbeat, volunteer with DataKind

Chartbeat

- leading real-time analytics provider for publishers
 - ~50B pageviews/month
 - ~300k requests/second



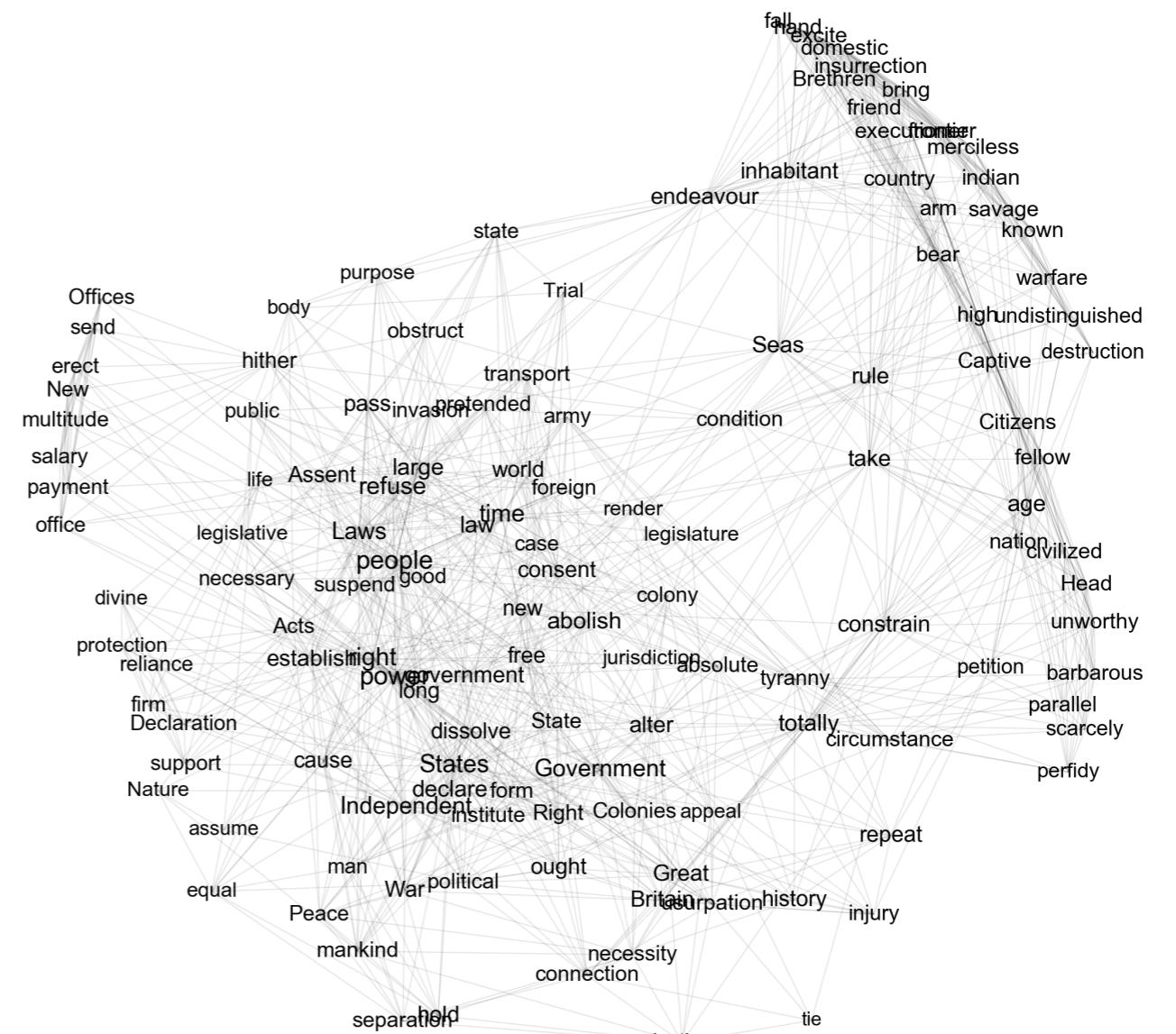
#BigData

Chartbeat

- leading real-time analytics provider for publishers
 - ~50B pageviews/month
 - ~300k requests/second
- founded in 2009, based in NYC
- Chartbeat ❤️ Python

Outline

- hello, textacy
 - features
 - topic modeling demo
 - roadmap



(Declaration of Independence as semantic network)

Hello, textacy

- “higher-level” NLP tasks
 - built on spaCy’s “lower-level” tokenization, POS-tagging, NER, dependency parse, ...

Hello, textacy

- “higher-level” NLP tasks
 - built on spaCy’s “lower-level” tokenization, POS-tagging, NER, dependency parse, ...
- design considerations
 - small- to medium-sized data
 - cover broad range of user needs
 - fun and user-friendly

Hello, textacy

- “higher-level” NLP tasks
 - built on spaCy’s “lower-level” tokenization, POS-tagging, NER, dependency parse, ...
- design considerations
 - small- to medium-sized data
 - cover broad range of user needs
 - fun and user-friendly

“pandas for nlp” — @gryBox, GitHub issue [#8](#)



Hello, textacy

- “higher-level” NLP tasks
 - built on spaCy’s “lower-level” tokenization, POS-tagging, NER, dependency parse, ...
- design considerations
 - small- to medium-sized data
 - cover broad range of user needs
 - fun and user-friendly

“pandas for nlp” — @gryBox, GitHub issue [#8](#)

tex·ta·cy

noun

1. The euphoria you get from a barrage of excellent and meaningful texts.

source: urbandictionary.com

Getting Started

Step 1:

```
$ pip install textacy
```

Step 2:

```
>>> import textacy
```

Getting Started

Step 1:

```
$ pip install textacy
```

Step 2:

```
>>> import textacy
```

Step 3 (Windows users only):



Features!

- stream text, json, and spaCy binary data to/from disk

```
>>> rr = textacy.corpora.RedditReader('RC_2015-01.bz2')
>>> for i, text in enumerate(rr.texts(limit=3)):
...     print(i, text)
```

0 Most of us have some family members like this. Most of my family is like this.

1 But Mill's career was way better. Bentham is like, the Joseph Smith to Mill's Brigham Young.

2 Mine uses a strait razor, and as much as i love the clippers i love the razor so much more. Then he follows it up with a warm towel. I think i might go get a hair cut this week.

```
>>> textacy.fileio.write_file_lines(rr.texts(limit=3), 'comments.txt')
```

Features!

- clean and normalize raw text

```
>>> text = """  
... This first line is nice and clean.  
... But â€” not the 2nd & 3rd!  
... Don't worry, go here: http://textacy.readthedocs.io/.  
... """  
>>> textacy.preprocess_text(  
...     text, fix_unicode=True, no_urls=True, lowercase=True)
```

this first line is nice and clean.
but – not the 2nd & 3rd!
don't worry, go here: *url*

Features!

- unsupervised information extraction

```
>>> text = list(textacy.corpora.WikiReader('enwiki-latest-pages-articles.xml.bz2').texts(limit=1))[0][10:]  
>>> doc = textacy.TextDoc(text)  
  
>>> list(doc.named_entities(good_ne_types={'PERSON'}))[:3]  
[Robespierre, William Godwin, Wilhelm Weitling]  
  
>>> {acro: def_ for acro, def_  
...   in doc.acronyms_and_definitions().items() if def_}  
{'CGT': 'Confederación General del Trabajo',  
'FOTLU': 'Federation of Organized Trades and Labor Unions',  
'LPF': 'League of Peace and Freedom',  
'WTO': 'World Trade Organization'}  
  
>>> doc.key_terms(algorithm='textrank', n=3)  
[('anarchist', 0.03284751614198023),  
 ('anarchism', 0.02039916051740292),  
 ('movement', 0.00841834859681396)]
```

Features!

- string, set, and document distance metrics

```
>>> doc1 = TextDoc('She spoke to the assembled journalists.')
>>> doc2 = TextDoc('He chatted with the gathered press.)
```

```
>>> textacy.distance.word_movers(doc1, doc2)
0.467695532304
```

```
>>> textacy.distance.word2vec(doc1, doc2)
0.089036465893226113
```

```
>>> textacy.distance.jaccard(
...     [w.lemma_ for w in doc1], [w.lemma_ for w in doc2])
0.8333333333333334
```

Features!

(and more)

Features!

The screenshot shows a documentation page for the `textacy` library. The top navigation bar includes the `textacy` logo, the word "latest", a search bar labeled "Search docs", and links for "Docs" and "Edit on GitHub". The main content area has a title "API Reference" and a section titled "Text Preprocessing". Below this, a note explains that the functions modify raw text *in-place*, replacing contractions, URLs, emails, phone numbers, and currency symbols with standardized forms. It cautions that preprocessing may affect the interpretation of the text – and Spacy's processing of it.

textacy.preprocess.fix_bad_unicode(text, normalization=u'NFC') [source]

Fix unicode text that's "broken" using `ftfy`; this includes mojibake, HTML entities and other code cruft, and non-standard forms for display purposes.

Parameters:

- `text (str)` – raw text
- `normalization (str {'NFC', 'NFKC', 'NFD', 'NFKD'}, optional)` – if 'NFC', combines characters and diacritics written using separate code points, e.g. converting "é" plus an acute accent modifier into "é"; unicode can be converted to NFC form without any change in its meaning! if 'NFKC', additional normalizations are applied that can change the meanings of characters, e.g. ellipsis characters will be replaced with three periods

Returns: str

textacy.preprocess.normalize_whitespace(text) [source]

Given `text` str, replace one or more spacings with a single space, and one or more linebreaks with a single newline. Also strip leading/trailing whitespace.

textacy.preprocess.preprocess_text(text, fix_unicode=False, lowercase=False, transliterate=False, no_urls=False, no_emails=False, no_phone_numbers=False, no_numbers=False, no_currency_symbols=False, no_punct=False, no_contractions=False, no_accents=False) [source]

Normalize various aspects of a raw text doc before parsing it with Spacy. A convenience function for applying all other preprocessing functions in one go.

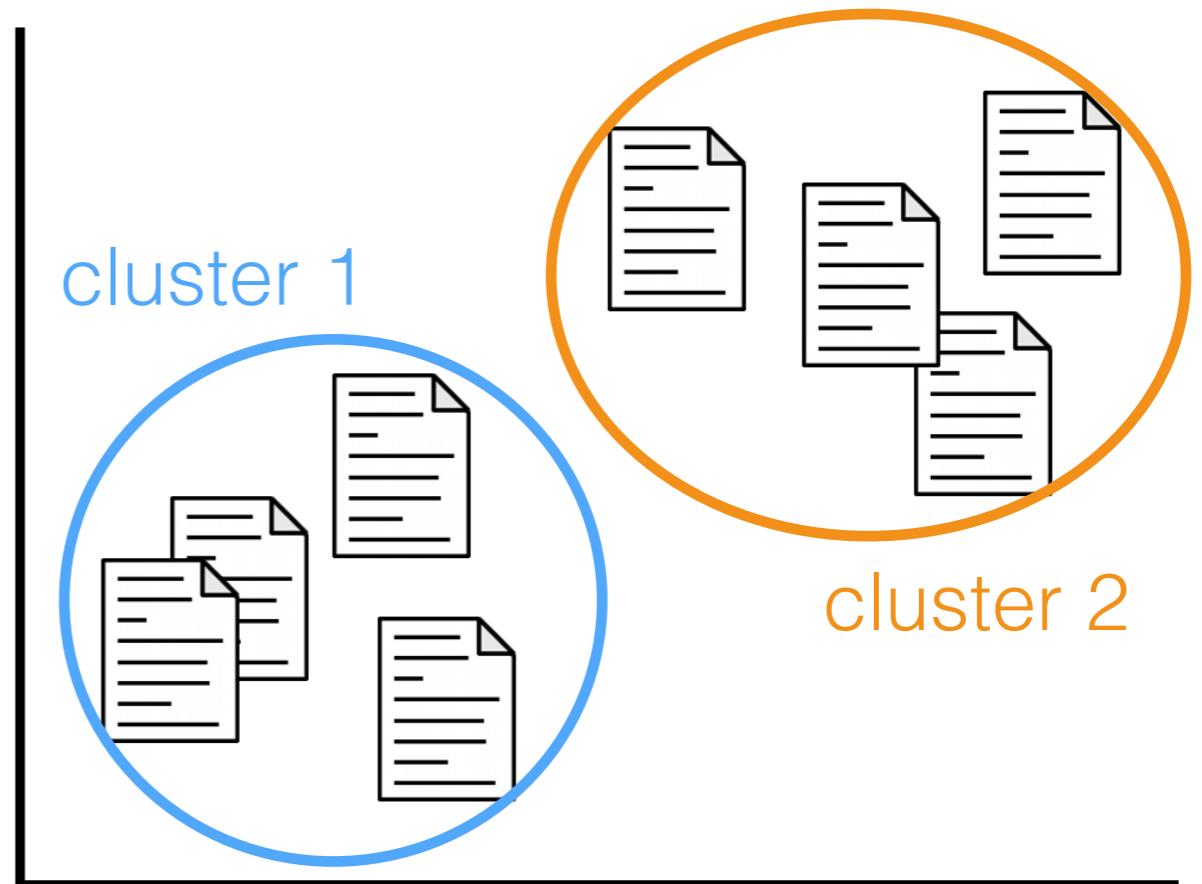
WARNING: These changes may negatively affect subsequent NLP analysis performed on the text, so choose carefully, and preprocess at your own risk!

```
$ jupyter notebook
```



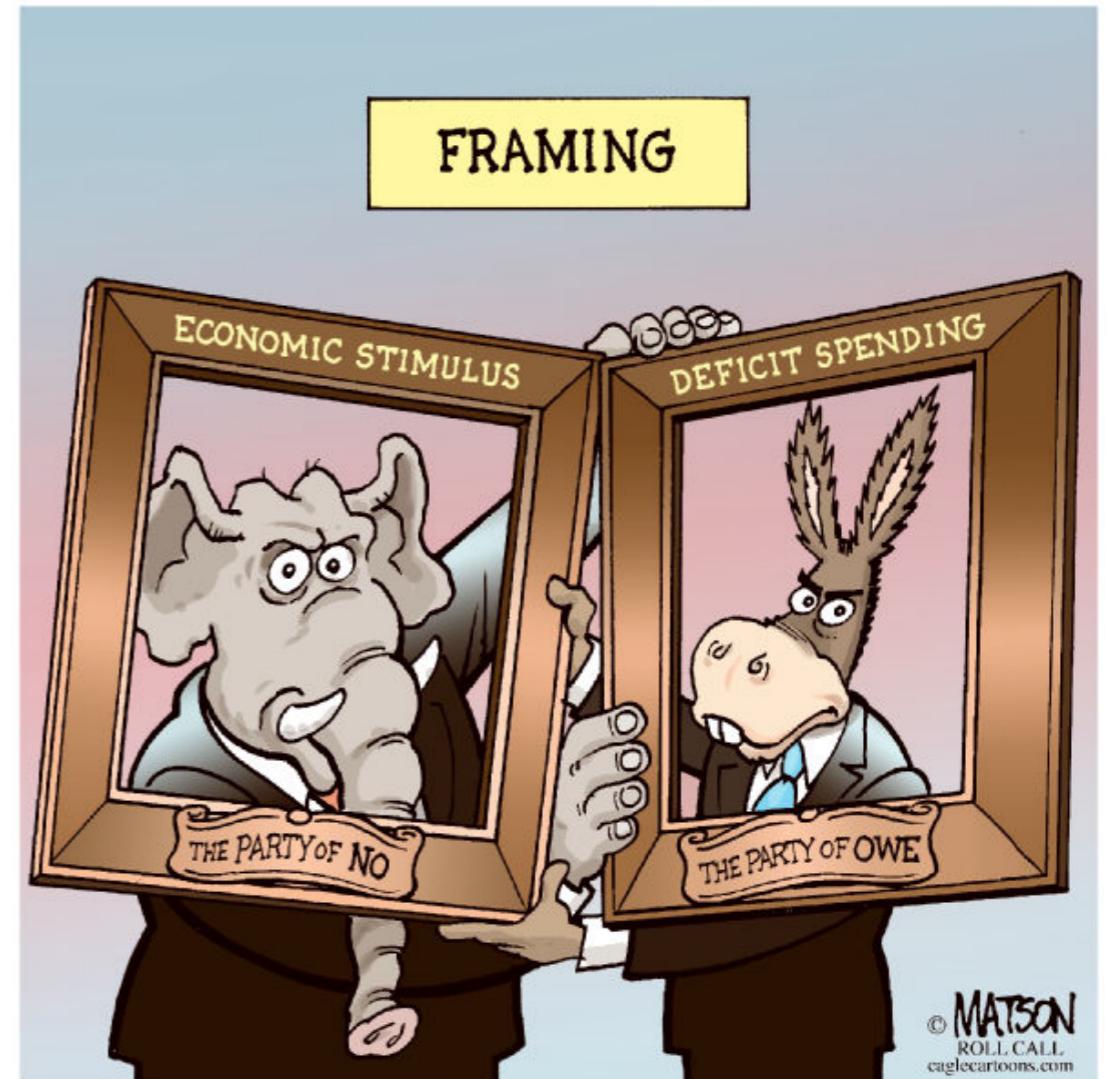
Roadmap

- document clustering



Roadmap

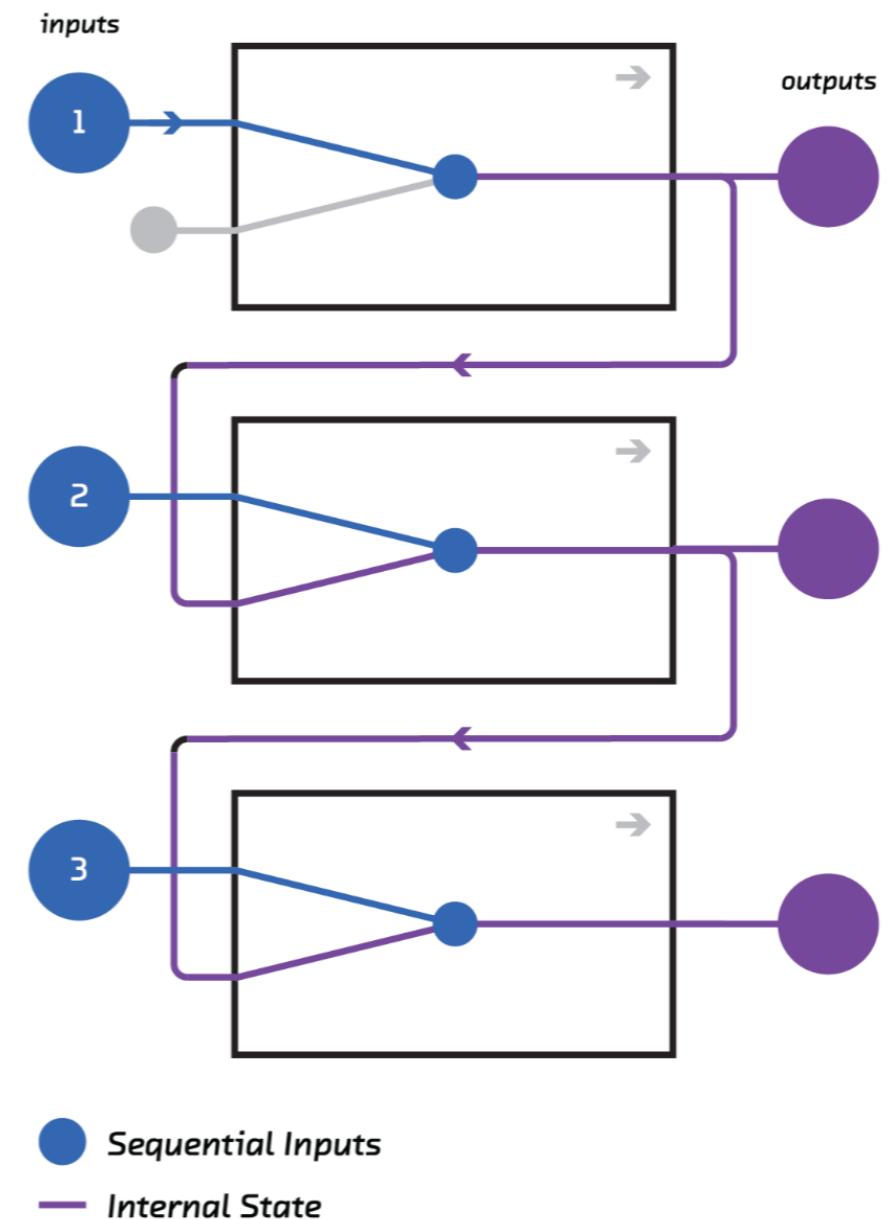
- document clustering
- media framing



Roadmap

- document clustering
- media framing
- #DeepLearning

RNN from yesterday's
“Summarizing Documents” talk:
<http://mike.place/talks/pygotham/>



Roadmap

- document clustering and classification
- media frames analysis
- #DeepLearning
- under active development...



Hello? World?

Users and contributors wanted!

THANK YOU



twitter: @bjdewilde, #textacy

slides: https://github.com/bdewilde/pygotham_2016

docs: <http://textacy.readthedocs.io/>

code: <https://github.com/chartbeat-labs/textacy>

BONUS

Features!

- easy access to and filtering of linguistic elements

```
>>> text = list(textacy.corpora.WikiReader('enwiki-latest-pages-articles.xml.bz2').texts(limit=1))[0][10:]  
>>> doc = textacy.TextDoc(text)  
  
>>> list(doc.sents)[0]  
Anarchism is a political philosophy that advocates self-governed societies with voluntary institutions.  
  
>>> list(doc.words(filter_stops=False))[:5]  
[Anarchism, is, a, political, philosophy]  
  
>>> list(doc.ngrams(2, filter_stops=True))[:3]  
[political philosophy, advocates self, governed societies]  
  
>>> list(doc.noun_chunks())[:3]  
[Anarchism, political philosophy, self-governed societies]
```

Additional Resources

- topic modeling
 - theory: Blei’s “Probabilistic Topic Models” review
<https://www.cs.princeton.edu/~blei/papers/Blei2012.pdf>
- other NLP Python packages
 - spacy: <https://spacy.io/>
 - nltk: <http://www.nltk.org/>
 - gensim: <https://github.com/RaRe-Technologies/gensim>
 - textblob: <http://textblob.readthedocs.io/>