LING/C SC 581:
Advanced Computational Linguistics

Lecture 11

Today's Topics

- Quick Homework 4 Review
- Homework 5:
  1. Installation of corpus `ptb` (Penn Treebank Brown/WSJ) into `nltk`
  2. Installing Structural Probes
  3. Installing LaTeX

Quick Homework 4 Review

- We get 3 different sequences for ungrammatical:
  - *furiously sleep ideas COLOR colorless*

    COLOR ∈ {*green, orange, black, red*}

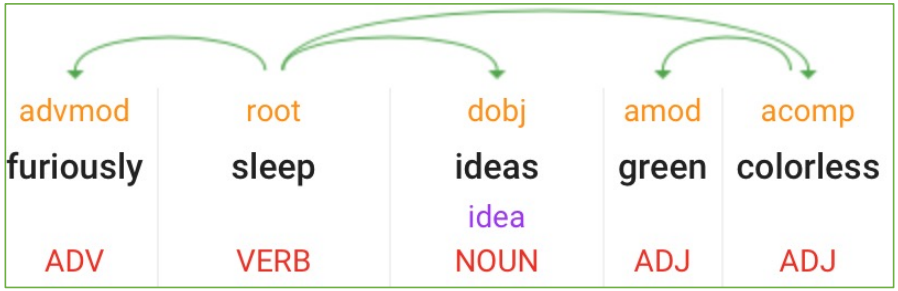
  1. ADV VERB NOUN ADJ ADJ
  2. ADV VERB NOUN NOUN ADJ
  3. ADV VERB NOUN VERB ADJ
- Question: *are these grammatical sequences*?
  - **Quick Homework 4** (*do it before next time*)
  - Come up with grammatical examples of these three sequences (in English), or explain why you think they're not.
  - Email to me (*usual rules*)

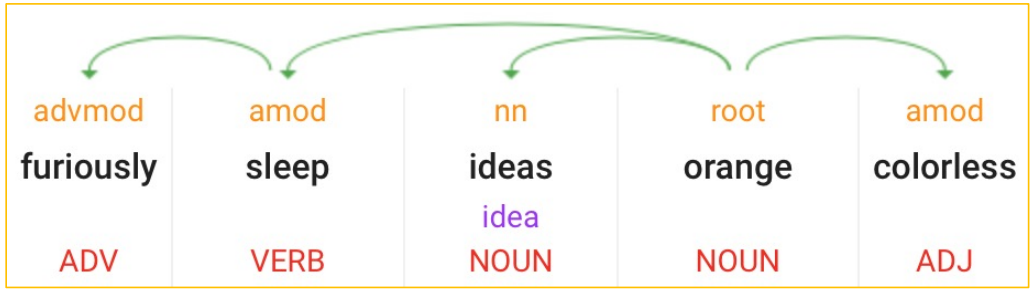
Example

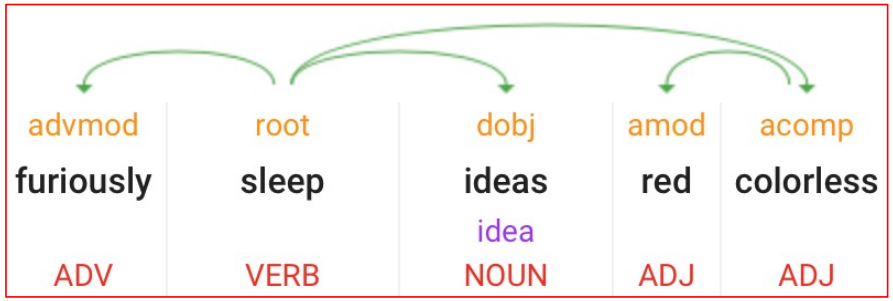
11 *colorless green ideas sleep furiously*

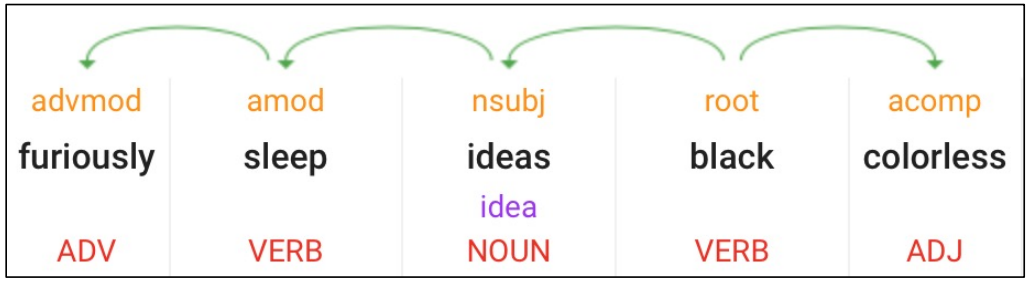
12 *furiously sleep ideas green colorless*

can you parse this?





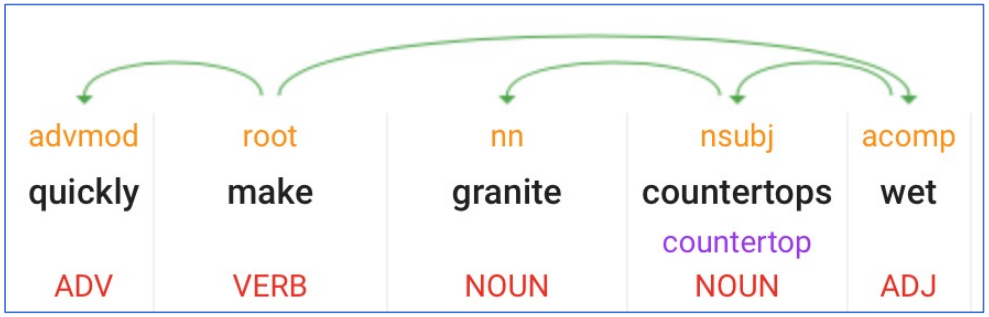




Quick Homework 4 Review

- Some of you came up with nice examples, but most found it very difficult
  1. ADV VERB NOUN ADJ ADJ
  2. ADV VERB NOUN NOUN ADJ

     quickly make granite countertops wet (Konrad N.)
  1. ADV VERB NOUN VERB ADJ



Quick Homework 4 Review

- Let's write a bit of code for searching a tagged word sequence (corpus)

Example:

```
1 seq = [['ADJ', 'ADJ', 'NOUN', 'VERB', 'ADV'],¶
2            ['ADV', 'VERB', 'NOUN', 'ADJ', 'ADJ'],¶
3            ['ADV', 'VERB', 'NOUN', 'NOUN', 'ADJ'],¶
4            ['ADV', 'VERB', 'NOUN', 'VERB', 'ADJ']]¶
5¶
6 def find_tagseq(seq, tagws):¶
7       ans = []¶
8       n = len(seq)¶
9       for idx in (i for i,tup in enumerate(tagws[:-n+1]) if tup[1]==seq[0]):¶
10          d = dict(tagws[idx:idx+n])¶
11          if list(d.values()) == seq:¶
12              ans.append(list(d.keys()))¶
13      return ans¶
```

the "good" sequence

Quick Homework 4 Review

```
>>> import nltk
>>> from nltk.corpus import ptb
>>> tws = list(ptb.tagged_words(tagset="universal"))
>>> from seq import *
>>> len(find_tagseq(seq[0], tws))
108
```

ptb is Penn Treebank (Brown/WSJ) 1,740,895 words

[['brilliant', 'white', 'flares', 'swayed', 'eerily'], ['prevalent', 'mental', 'disturbance', 'affecting', 'even'], ['little', 'green', 'biplane', 'struggled', 'northward'], ['important', 'factual', 'information', 'is', 'still'], ['female', 'sexual', 'aggressiveness', 'was', 'tremendously'], ['regular', 'medical', 'doctors', 'are', 'especially'], ['most', 'American', 'literature', 'is', 'today'], ['Russian', 'naval', 'power', 'would', 'always'], ['recent', 'Soviet', 'successes', 'is', 'hardly'], ['old', 'black', 'dirt', 'was', 'also'], ['old', 'red', 'wine', 'is', 'often'], ['old', 'red', 'wine', 'should', 'first'], ['former', 'colonial', 'areas', 'has', 'equally'], ['so-called', 'fundamental', 'law', 'had', 'already'], ['single', 'distinct', 'impression', 'is', 'more'], ['Dirty', 'Reactionary', 'bastards', 'comin', 'down'], ['usual', 'congratulatory', 'crowd', 'was', 'conspicuously'], ['sparse', 'brown', 'hair', 'was', 'heavily'], ['strange', 'little', 'man', 'had', 'never'], ['first', 'pink', 'threads', 'stitched', 'together'], ['long', 'black', 'hair', 'streamed', 'out'], ['Many', 'small', 'bones', 'protruded', 'crazily'], ['little', 'sweet', 'potato', 'trilled', 'neatly'], ['nice-looking', 'young', 'officer', 'fell', 'back'], ['odious', 'little', 'toad', 'went', 'along'], ['few', 'medical', 'institutions', 'conducting', 'privately'], ['average', 'male', 'ringer', 'leaves', 'quite'], ['new', 'Polish', 'government', 'can', 'fully'], ['Such', 'multi-crystal', 'materials', 'will', 'probably'], ['so-called', 'male-sterile', 'plants', 'can', 'then'], ['narrower', '30-share', 'index', 'was', 'up'], ['one-year', 'savings-type', 'CD', 'was', 'down'], ['chi-chi', 'airborne', 'activity', 'wins', 'heartwarmingly'], ['other', 'Japanese', 'institutions', 'say', 'privately'], ['many', 'crucial', 'tests', 'are', 'just'], ['unusual', 'seaborne', 'meeting', 'wo', "n't"], ['non-violent', 'civil', 'disobedience', 'are', 'not'], ['methodical', 'Japanese', 'managers', 'do', "n't"], ['parallel', 'federal', 'examination', 'seemed', 'so'], ['other', 'industrial', 'commodities', 'increased', 'nearly'], ['benchmark', '30-year', 'bond', 'was', 'nearly'], ['other', 'short-term', 'funds', 'surged', 'more'], ['coherent', 'Japanese', 'market', 'could', 'also'], ['large', 'mutual-fund', 'companies', 'might', 'soon'], ['Socialist', 'prime', 'minister', 'has', 'simply'], ['federal', 'age-discrimination', 'law', 'does', "n't"], ['proper', 'climatic', 'conditions', 'do', "n't"], ['fiscal', 'first', 'quarter', 'will', 'not'], ['weak', 'corporate', 'profits', 'may', 'further'], ['most', 'prolific', 'producers', 'can', 'ever'], ['Japanese', 'corporate', 'attitudes', 'repel', 'most'], ['new', 'cellular', 'company', 'is', "n't"], ['big-ticket', 'durable', 'goods', 'might', 'actually'], ['gossipy', 'weekly', 'magazines', 'are', 'often'], ['narrow', 'American', 'strategy', 'is', "n't"], ['third-quarter', 'corporate', 'earnings', 'have', "n't"], ['Latin', 'American', 'countries', 'have', 'long'], ['high-tech', 'special', 'effects', 'are', 'continually'], ['traditional', 'American', 'cockiness', 'is', "n't"], ['turgid', 'other', 'films', 'opening', 'now'], ['great', 'many', 'words', 'could', "n't"], ['gross', 'national', 'product', 'is', 'either'], ['big', 'new', 'hotel-casino', 'opened', 'here'], ['many', 'new', 'products', 'are', "n't"], ['detailed', 'financial', 'figures', 'have', "n't"], ['Japanese', 'monetary', 'policy', 'wo', "n't"], ['sympathetic', 'new', 'government', 'wo', "n't"], ['benchmark', '30-year', 'bond', 'gained', 'nearly'], ['small-denomination', 'three-month', 'CDs', 'moved', 'up'], ['Western', 'export-control', 'rules', 'had', 'previously'], ['chief', 'executive', 'officer', 'is', "n't"], ['American', 'institutional', 'investors', 'have', 'never'], ['heavy', 'black', 'smoke', 'billowing', 'high'], ['benchmark', '30-year', 'bond', 'ended', 'about'], ['fiscal', 'fourth-quarter', 'profit', 'plunged', 'more'], ['international', 'financial', 'institutions', 'have', 'virtually'], ['domestic', 'flexible', 'portfolios', 'are', 'up'], ['foundering', 'middle', 'classes', 'are', "n't"], ['Most', 'American', 'investors', 'have', 'just'], ['Fiscal', 'fourth-quarter', 'sales', 'grew', 'about'], ['Year-earlier', 'per-share', 'results', 'are', "n't"], ['big', 'national', 'laboratories', 'are', 'still'], ['hot-dipped', 'galvanized', 'steel', 'increased', 'about'], ['Older', 'SENIOR', 'CITIZENS', 'have', 'long'], ['dynamic', 'free', 'market', 'does', "n't"], ['high-risk', 'corporate', 'bonds', 'have', 'either'], ['stricter', 'protective', 'covenants', 'may', 'increasingly'], ['next', 'few', 'years', 'will', 'greatly'], ['invisible', 'secondary', 'market', 'operating', 'chiefly'], ['last', 'few', 'days', 'will', 'surely'], ['such', 'governmental', 'proceedings', 'are', "n't"], ['many', 'small', 'investors', 'are', 'absolutely'], ['fourth-largest', 'metropolitan', 'region', 'was', 'nearly'], ['several', 'key', 'allies', 'lost', 'out'], ['East', 'German', 'leadership', 'grew', 'still'], ['New', 'clinical', 'trials', 'are', 'already'], ['due', 'last', 'Monday', 'wo', "n't"], ['well-paid', 'paramilitary', 'forces', 'totaling', 'more'], ['Soviet', 'legislative', 'panel', 'rejected', 'as'], ['third-quarter', 'net', 'income', 'dropped', 'nearly'], ['real', 'nervous', 'folks', 'came', 'along'], ['such', 'thin-slab', 'technology', 'is', 'only'], ['national', 'economic', 'interests', 'are', 'much'], ['monthly', 'net', 'purchases', 'are', 'still'], ['next', 'few', 'months', 'is', 'widely'], ['noncriminal', 'congressional', 'hearing', 'does', 'not'], ['negative', 'takeover-related', 'news', 'did', "n't"], ['most', 'such', 'machines', 'are', 'about']]

Quick Homework 4 Review

```
>>> from seq import *
>>> import nltk
>>> from nltk.corpus import ptb
>>> tws =
list(ptb.tagged_words(tagset="universal"))
>>> len(find_tagseq(seq[0], tws))
108
>>> len(find_tagseq(seq[1], tws))
0
>>> len(find_tagseq(seq[2], tws))
0
```

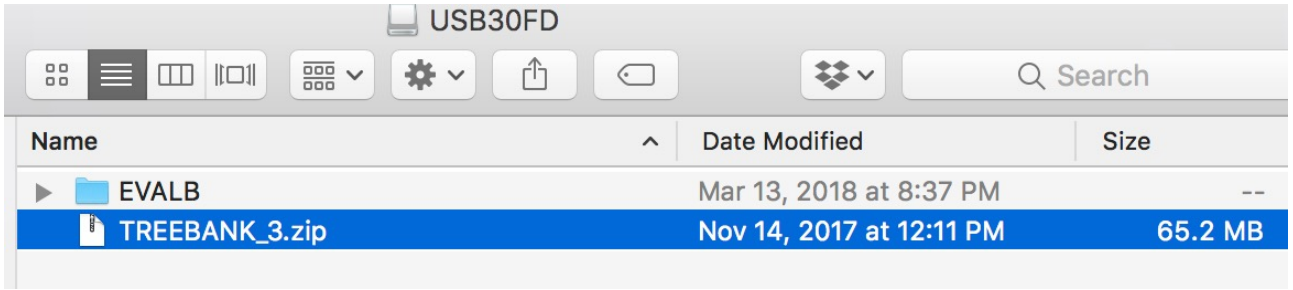
```
>>> len(find_tagseq(seq[3], tws))
6
>>> find_tagseq(seq[3], tws)
[['only', 'made', 'people', 'feel', 'sorry'],
['Closely', 'held', 'Automax', 'has', 'annual'],
['nationally', 'televised', 'address', 'rejecting', 'communist'],
['once', 'signed', 'petitions', 'opposing', 'work-rule'],
['politically', 'charged', 'atmosphere', 'surrounding', 'recent'],
['widely', 'publicized', 'sequester', 'is', 'likely']]
```

Homework 5

- Part 1: install ptb
- Part 2: install Structural Probes

Penn Treebank (version 3)

- Download the file TREEBANK_3.zip from the course website



- This is the full corpus from the Linguistic Data Consortium (LDC)
- Note:
  - http://www.nltk.org/howto/corpus.html#parsed-corpora
  - The NLTK data package includes a 10% sample of the Penn Treebank (in `treebank`), as well as the Sinica Treebank (in `sinica_treebank`).

nltk: Corpus Readers

You should already have *this* 10% sample installed

- Reading the Penn Treebank (Wall Street Journal sample):

```
>>> from nltk.corpus import treebank
1. treebank.fileids()
2. treebank.words(fileid)
3. treebank.tagged_words(fileid)
4. treebank.parsed_sents(fileid)
```

nltk: Corpus Readers

```
treebank.fileids()
```

```
>>> treebank.fileids()
['wsj_0001.mrg', 'wsj_0002.mrg', 'wsj_0003.mrg', 'wsj_0004.mrg', 'wsj_0005.mrg', 'wsj_0006.mrg', 'wsj_0007.mrg', 'wsj_0008.mrg',
 'wsj_0009.mrg', 'wsj_0010.mrg', 'wsj_0011.mrg', 'wsj_0012.mrg', 'wsj_0013.mrg', 'wsj_0014.mrg', 'wsj_0015.mrg', 'wsj_0016.mrg',
 'wsj_0017.mrg', 'wsj_0018.mrg', 'wsj_0019.mrg', 'wsj_0020.mrg', 'wsj_0021.mrg', 'wsj_0022.mrg', 'wsj_0023.mrg', 'wsj_0024.mrg',
 'wsj_0025.mrg', 'wsj_0026.mrg', 'wsj_0027.mrg', 'wsj_0028.mrg', 'wsj_0029.mrg', 'wsj_0030.mrg', 'wsj_0031.mrg', 'wsj_0032.mrg',
 'wsj_0033.mrg', 'wsj_0034.mrg', 'wsj_0035.mrg', 'wsj_0036.mrg', 'wsj_0037.mrg', 'wsj_0038.mrg', 'wsj_0039.mrg', 'wsj_0040.mrg',
 'wsj_0041.mrg', 'wsj_0042.mrg', 'wsj_0043.mrg', 'wsj_0044.mrg', 'wsj_0045.mrg', 'wsj_0046.mrg', 'wsj_0047.mrg', 'wsj_0048.mrg',
 'wsj_0049.mrg', 'wsj_0050.mrg', 'wsj_0051.mrg', 'wsj_0052.mrg', 'wsj_0053.mrg', 'wsj_0054.mrg', 'wsj_0055.mrg', 'wsj_0056.mrg',
 'wsj_0057.mrg', 'wsj_0058.mrg', 'wsj_0059.mrg', 'wsj_0060.mrg', 'wsj_0061.mrg', 'wsj_0062.mrg', 'wsj_0063.mrg', 'wsj_0064.mrg',
 'wsj_0065.mrg', 'wsj_0066.mrg', 'wsj_0067.mrg', 'wsj_0068.mrg', 'wsj_0069.mrg', 'wsj_0070.mrg', 'wsj_0071.mrg', 'wsj_0072.mrg',
 'wsj_0073.mrg', 'wsj_0074.mrg', 'wsj_0075.mrg', 'wsj_0076.mrg', 'wsj_0077.mrg', 'wsj_0078.mrg', 'wsj_0079.mrg', 'wsj_0080.mrg',
 'wsj_0081.mrg', 'wsj_0082.mrg', 'wsj_0083.mrg', 'wsj_0084.mrg', 'wsj_0085.mrg', 'wsj_0086.mrg', 'wsj_0087.mrg', 'wsj_0088.mrg',
 'wsj_0089.mrg', 'wsj_0090.mrg', 'wsj_0091.mrg', 'wsj_0092.mrg', 'wsj_0093.mrg', 'wsj_0094.mrg', 'wsj_0095.mrg', 'wsj_0096.mrg',
 'wsj_0097.mrg', 'wsj_0098.mrg', 'wsj_0099.mrg', 'wsj_0100.mrg', 'wsj_0101.mrg', 'wsj_0102.mrg', 'wsj_0103.mrg', 'wsj_0104.mrg',
 'wsj_0105.mrg', 'wsj_0106.mrg', 'wsj_0107.mrg', 'wsj_0108.mrg', 'wsj_0109.mrg', 'wsj_0110.mrg', 'wsj_0111.mrg', 'wsj_0112.mrg',
 'wsj_0113.mrg', 'wsj_0114.mrg', 'wsj_0115.mrg', 'wsj_0116.mrg', 'wsj_0117.mrg', 'wsj_0118.mrg', 'wsj_0119.mrg', 'wsj_0120.mrg',
 'wsj_0121.mrg', 'wsj_0122.mrg', 'wsj_0123.mrg', 'wsj_0124.mrg', 'wsj_0125.mrg', 'wsj_0126.mrg', 'wsj_0127.mrg', 'wsj_0128.mrg',
 'wsj_0129.mrg', 'wsj_0130.mrg', 'wsj_0131.mrg', 'wsj_0132.mrg', 'wsj_0133.mrg', 'wsj_0134.mrg', 'wsj_0135.mrg', 'wsj_0136.mrg',
 'wsj_0137.mrg', 'wsj_0138.mrg', 'wsj_0139.mrg', 'wsj_0140.mrg', 'wsj_0141.mrg', 'wsj_0142.mrg', 'wsj_0143.mrg', 'wsj_0144.mrg',
 'wsj_0145.mrg', 'wsj_0146.mrg', 'wsj_0147.mrg', 'wsj_0148.mrg', 'wsj_0149.mrg', 'wsj_0150.mrg', 'wsj_0151.mrg', 'wsj_0152.mrg',
 'wsj_0153.mrg', 'wsj_0154.mrg', 'wsj_0155.mrg', 'wsj_0156.mrg', 'wsj_0157.mrg', 'wsj_0158.mrg', 'wsj_0159.mrg', 'wsj_0160.mrg',
 'wsj_0161.mrg', 'wsj_0162.mrg', 'wsj_0163.mrg', 'wsj_0164.mrg', 'wsj_0165.mrg', 'wsj_0166.mrg', 'wsj_0167.mrg', 'wsj_0168.mrg',
 'wsj_0169.mrg', 'wsj_0170.mrg', 'wsj_0171.mrg', 'wsj_0172.mrg', 'wsj_0173.mrg', 'wsj_0174.mrg', 'wsj_0175.mrg', 'wsj_0176.mrg',
 'wsj_0177.mrg', 'wsj_0178.mrg', 'wsj_0179.mrg', 'wsj_0180.mrg', 'wsj_0181.mrg', 'wsj_0182.mrg', 'wsj_0183.mrg', 'wsj_0184.mrg',
 'wsj_0185.mrg', 'wsj_0186.mrg', 'wsj_0187.mrg', 'wsj_0188.mrg', 'wsj_0189.mrg', 'wsj_0190.mrg', 'wsj_0191.mrg', 'wsj_0192.mrg',
 'wsj_0193.mrg', 'wsj_0194.mrg', 'wsj_0195.mrg', 'wsj_0196.mrg', 'wsj_0197.mrg', 'wsj_0198.mrg', 'wsj_0199.mrg']
>>>
```

```
>>> len(treebank.fileids())
199
```

nltk: Corpus Readers

`treebank.words(`*`fileid`*`))`

```
>>> len(treebank.words('wsj_0003.mrg'))
782
>>> treebank.words('wsj_0003.mrg')
['A', 'form', 'of', 'asbestos', 'once', 'used', '*', ...]
>>> list(treebank.words('wsj_0003.mrg'))[:200]
['A', 'form', 'of', 'asbestos', 'once', 'used', '*', '*', 'to', 'make', 'Kent', 'cigarette', 'filters', 'has', 'caused
', 'a', 'high', 'percentage', 'of', 'cancer', 'deaths', 'among', 'a', 'group', 'of', 'workers', 'exposed', '*', 'to',
'it', 'more', 'than', '30', 'years', 'ago', ',', 'researchers', 'reported', '0', '*T*-1', '.', 'The', 'asbestos', 'fib
er', ',', 'crocidolite', ',', 'is', 'unusually', 'resilient', 'once', 'it', 'enters', 'the', 'lungs', ',', 'with', 'ev
en', 'brief', 'exposures', 'to', 'it', 'causing', 'symptoms', 'that', '*T*-1', 'show', 'up', 'decades', 'later', ',',
'researchers', 'said', '0', '*T*-2', '.', 'Lorillard', 'Inc.', ',', 'the', 'unit', 'of', 'New', 'York-based', 'Loews',
 'Corp.', 'that', '*T*-2', 'makes', 'Kent', 'cigarettes', ',', 'stopped', 'using', 'crocidolite', 'in', 'its', 'Micron
ite', 'cigarette', 'filters', 'in', '1956', '.', 'Although', 'preliminary', 'findings', 'were', 'reported', '*-2', 'mo
re', 'than', 'a', 'year', 'ago', ',', 'the', 'latest', 'results', 'appear', 'in', 'today', "'s", 'New', 'England', 'Jo
urnal', 'of', 'Medicine', ',', 'a', 'forum', 'likely', '*', 'to', 'bring', 'new', 'attention', 'to', 'the', 'problem',
 '.', 'A', 'Lorillard', 'spokewoman', 'said', ',', '``', 'This', 'is', 'an', 'old', 'story', '.', 'We', "'re", 'talkin
g', 'about', 'years', 'ago', 'before', 'anyone', 'heard', 'of', 'asbestos', 'having', 'any', 'questionable', 'properti
es', '.', 'There', 'is', 'no', 'asbestos', 'in', 'our', 'products', 'now', '.', "''", 'Neither', 'Lorillard', 'nor', '
the', 'researchers', 'who', '*T*-3', 'studied', 'the', 'workers', 'were', 'aware', 'of', 'any', 'research', 'on', 'smo
kers', 'of', 'the', 'Kent', 'cigarettes', '.']
>>>
```

nltk: Corpus Readers

`treebank.tagged_words(`*`fileid`*`))`

- list of tuples of the form `(word, POStag)`

```
>>> >>> list(treebank.tagged_words('wsj_0003.mrg'))[:100]
[('A', 'DT'), ('form', 'NN'), ('of', 'IN'), ('asbestos', 'NN'), ('once', 'RB'), ('used', 'VBN'), ('*', '-NONE-'), ('*'
, '-NONE-'), ('to', 'TO'), ('make', 'VB'), ('Kent', 'NNP'), ('cigarette', 'NN'), ('filters', 'NNS'), ('has', 'VBZ'), (
'caused', 'VBN'), ('a', 'DT'), ('high', 'JJ'), ('percentage', 'NN'), ('of', 'IN'), ('cancer', 'NN'), ('deaths', 'NNS')
, ('among', 'IN'), ('a', 'DT'), ('group', 'NN'), ('of', 'IN'), ('workers', 'NNS'), ('exposed', 'VBN'), ('*', '-NONE-')
, ('to', 'TO'), ('it', 'PRP'), ('more', 'RBR'), ('than', 'IN'), ('30', 'CD'), ('years', 'NNS'), ('ago', 'IN'), (',', '
,'), ('researchers', 'NNS'), ('reported', 'VBD'), ('0', '-NONE-'), ('*T*-1', '-NONE-'), ('.', '.'), ('The', 'DT'), ('a
sbestos', 'NN'), ('fiber', 'NN'), (',', ','), ('crocidolite', 'NN'), (',', ','), ('is', 'VBZ'), ('unusually', 'RB'), (
'resilient', 'JJ'), ('once', 'IN'), ('it', 'PRP'), ('enters', 'VBZ'), ('the', 'DT'), ('lungs', 'NNS'), (',', ','), ('w
ith', 'IN'), ('even', 'RB'), ('brief', 'JJ'), ('exposures', 'NNS'), ('to', 'TO'), ('it', 'PRP'), ('causing', 'VBG'), (
'symptoms', 'NNS'), ('that', 'WDT'), ('*T*-1', '-NONE-'), ('show', 'VBP'), ('up', 'RP'), ('decades', 'NNS'), ('later',
 'JJ'), (',', ','), ('researchers', 'NNS'), ('said', 'VBD'), ('0', '-NONE-'), ('*T*-2', '-NONE-'), ('.', '.'), ('Loril
lard', 'NNP'), ('Inc.', 'NNP'), (',', ','), ('the', 'DT'), ('unit', 'NN'), ('of', 'IN'), ('New', 'JJ'), ('York-based',
 'JJ'), ('Loews', 'NNP'), ('Corp.', 'NNP'), ('that', 'WDT'), ('*T*-2', '-NONE-'), ('makes', 'VBZ'), ('Kent', 'NNP'), (
'cigarettes', 'NNS'), (',', ','), ('stopped', 'VBD'), ('using', 'VBG'), ('crocidolite', 'NN'), ('in', 'IN'), ('its', '
PRP$'), ('Micronite', 'NN'), ('cigarette', 'NN'), ('filters', 'NNS')]
>>>
```

nltk: Corpus Readers

`treebank.parsed_sents(`*`fileid`*`)`

- `Tree(…)` structure

```
>>> len(treebank.parsed_sents('wsj_0003.mrg'))
30
>>> treebank.parsed_sents('wsj_0003.mrg')[0]
Tree('S', [Tree('S-TPC-1', [Tree('NP-SBJ', [Tree('NP', [Tree('NP', [Tree('DT', ['A']), Tree('NN', ['form'])]), Tree('P
P', [Tree('IN', ['of']), Tree('NP', [Tree('NN', ['asbestos'])])])]), Tree('RRC', [Tree('ADVP-TMP', [Tree('RB', ['once'
])]), Tree('VP', [Tree('VBN', ['used']), Tree('NP', [Tree('-NONE-', ['*'])]), Tree('S-CLR', [Tree('NP-SBJ', [Tree('-NO
NE-', ['*'])]), Tree('VP', [Tree('TO', ['to']), Tree('VP', [Tree('VB', ['make']), Tree('NP', [Tree('NNP', ['Kent']), T
ree('NN', ['cigarette']), Tree('NNS', ['filters'])])])])])])])]), Tree('VP', [Tree('VBZ', ['has']), Tree('VP', [Tree('
VBN', ['caused']), Tree('NP', [Tree('NP', [Tree('DT', ['a']), Tree('JJ', ['high']), Tree('NN', ['percentage'])]), Tree
('PP', [Tree('IN', ['of']), Tree('NP', [Tree('NN', ['cancer']), Tree('NNS', ['deaths'])])]), Tree('PP-LOC', [Tree('IN'
, ['among']), Tree('NP', [Tree('NP', [Tree('DT', ['a']), Tree('NN', ['group'])]), Tree('PP', [Tree('IN', ['of']), Tree
('NP', [Tree('NP', [Tree('NNS', ['workers'])]), Tree('RRC', [Tree('VP', [Tree('VBN', ['exposed']), Tree('NP', [Tree('-
NONE-', ['*'])]), Tree('PP-CLR', [Tree('TO', ['to']), Tree('NP', [Tree('PRP', ['it'])])]), Tree('ADVP-TMP', [Tree('NP'
, [Tree('QP', [Tree('RBR', ['more']), Tree('IN', ['than']), Tree('CD', ['30'])]), Tree('NNS', ['years'])]), Tree('IN',
 ['ago'])])])])])])])])])])]), Tree(',', [',']), Tree('NP-SBJ', [Tree('NNS', ['researchers'])]), Tree('VP', [Tree('V
BD', ['reported']), Tree('SBAR', [Tree('-NONE-', ['0']), Tree('S', [Tree('-NONE-', ['*T*-1'])])])]), Tree('.', ['.'])]
)
>>>
```

nltk: Corpus Readers

- http://www.nltk.org/howto/corpus.html#parsed-corpora
  - If you have access to a **full installation** of the Penn Treebank, NLTK can be configured to load it as well.
  - Download the ptb package, and in the directory nltk_data/corpora/ptb place the BROWN and WSJ directories of the Treebank installation (symbolic links work as well).
  - `nltk.download('ptb')`
  - Then use the **ptb** module instead of **treebank**:

```
>>> from nltk.corpus import ptb
>>> print(ptb.fileids()) # doctest: +SKIP ['BROWN/CF/CF01.MRG', 'BROWN/CF/CF02.MRG',
'BROWN/CF/CF03.MRG', 'BROWN/CF/CF04.MRG', ...]
>>> print(ptb.words('WSJ/00/WSJ_0003.MRG')) # doctest: +SKIP ['A', 'form', 'of',
'asbestos', 'once', 'used', '*', ...]
>>> print(ptb.tagged_words('WSJ/00/WSJ_0003.MRG')) # doctest: +SKIP [('A', 'DT'), ('form',
'NN'), ('of', 'IN'), ...]
```

Penn Treebank (PTB) with nltk

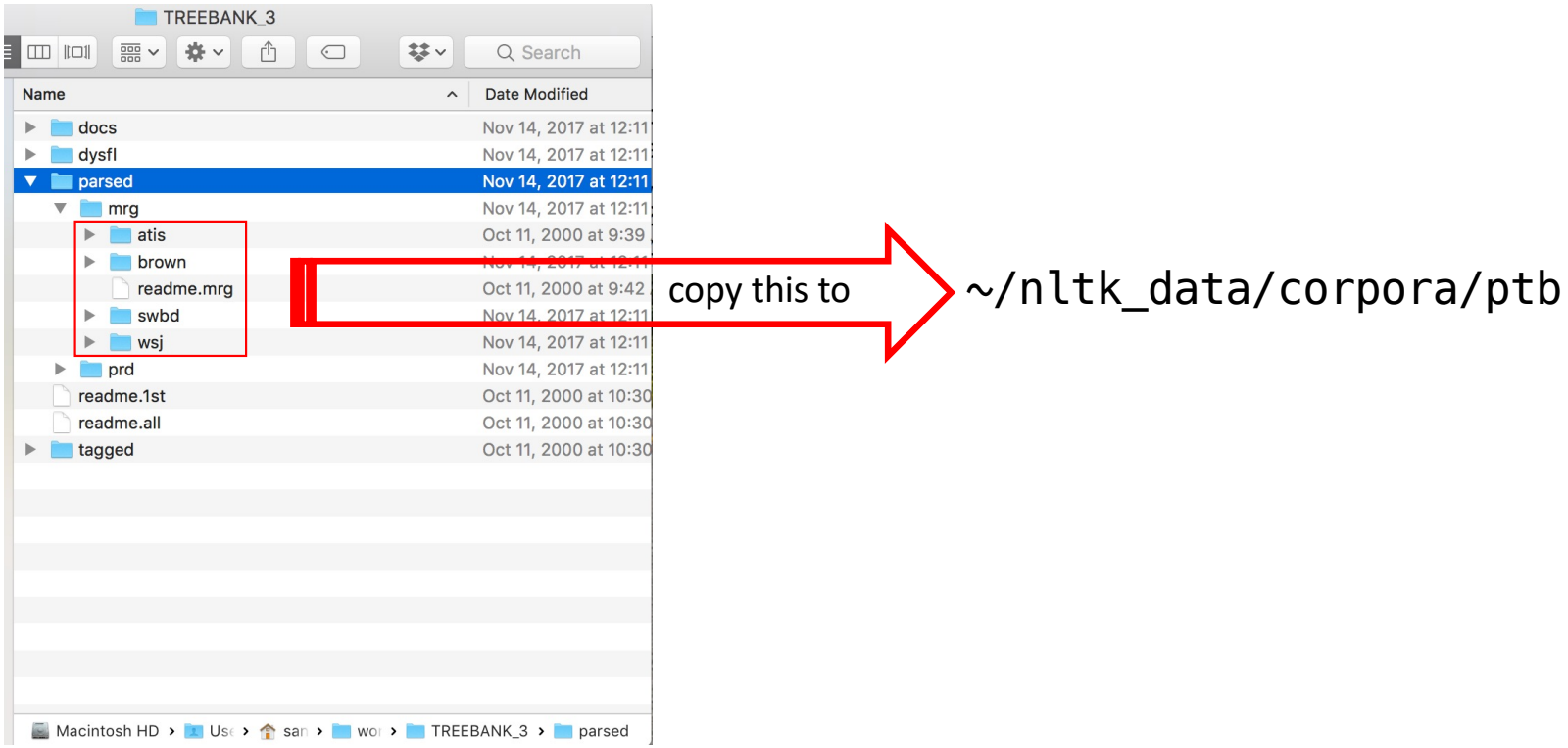
- Unzip TREEBANK_3.zip
- Put your wsj directory (from mrg) here ~/nltk_data/corpora/ptb

```
Sandiways-MacBook:~ sandiway$ python3
Python 3.5.2 (v3.5.2:4def2a2901a5, Jun 26 2016, 10:47:25)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import nltk
>>> nltk.download('ptb')
[nltk_data] Downloading package ptb to /Users/sandiway/nltk_data...
[nltk_data]   Unzipping corpora/ptb.zip.
True
>>>
```

```
Sandiways-MacBook:ptb sandiway$ cd wsj/00
Sandiways-MacBook:00 sandiway$ ls
wsj_0001.mrg    wsj_0021.mrg    wsj_0041.mrg    wsj_0061.mrg    wsj_0081.mrg
wsj_0002.mrg    wsj_0022.mrg    wsj_0042.mrg    wsj_0062.mrg    wsj_0082.mrg
wsj_0003.mrg    wsj_0023.mrg    wsj_0043.mrg    wsj_0063.mrg    wsj_0083.mrg
```

Filename case problem!

Penn Treebank (PTB) with nltk



Penn Treebank (PTB) with nltk

- Rename files to uppercase
  - `for f in \`find wsj\`; do mv -v "$f" "\`echo $f | tr '[a-z]' '[A-Z]'\`"; done`
  - (found on stackoverflow.com)
  - *seems to work but not clean*

```
wsj/14/wsj_1493.mrg -> WSJ/14/WSJ_1493.MRG
mv: rename wsj/22 to WSJ/22/22: Invalid argument
wsj/22/wsj_2236.mrg -> WSJ/22/WSJ_2236.MRG
wsj/22/wsj_2222.mrg -> WSJ/22/WSJ_2222.MRG
wsj/22/wsj_2223.mrg -> WSJ/22/WSJ_2223.MRG
```

directory name needs to be uppercased too!

```
Sandiways-MacBook:WSJ sandiway$ cd 00
Sandiways-MacBook:00 sandiway$ ls
WSJ_0001.MRG    WSJ_0021.MRG    WSJ_0041.MRG    WSJ_0061.MRG    WSJ_0081.MRG
WSJ_0002.MRG    WSJ_0022.MRG    WSJ_0042.MRG    WSJ_0062.MRG    WSJ_0082.MRG
WSJ_0003.MRG    WSJ_0023.MRG    WSJ_0043.MRG    WSJ_0063.MRG    WSJ_0083.MRG
WSJ_0004.MRG    WSJ_0024.MRG    WSJ_0044.MRG    WSJ_0064.MRG    WSJ_0084.MRG
WSJ_0005.MRG    WSJ_0025.MRG    WSJ_0045.MRG    WSJ_0065.MRG    WSJ_0085.MRG
```

Penn Treebank (PTB) with nltk

- **Note**: you may run into problems with file permissions when renaming:

```
atis -> ATIS
override r--r--r--  sandiway/staff for ATIS/ATIS3.MRG? (y/n [n]) ^C
```

- Change permissions (recursively):
  - `chmod –R u+w atis`

Penn Treebank (PTB) with nltk

Renaming script courtesy of *Sandeep Suntwal* (from my 2018 class):

```
import os
import sys

#Change below path as per your computer
path =  'c:\\Users\sandeep\AppData\Roaming\\nltk_data\corpora\ptb\wsj\\'

for subdir, dirs, files in os.walk(path):
    for filename in files:
        newFileName= filename.upper()
        os.rename(os.path.join(subdir, filename), os.path.join(subdir, newFileName))
```

Penn Treebank (PTB) with nltk

```
>>> from nltk.corpus import ptb
>>> print(ptb.fileids())
['BROWN/CF/CF01.MRG', 'BROWN/CF/CF02.MRG', 'BROWN/CF/CF03.MRG', 'BROWN/CF/CF04.M
RG', 'BROWN/CF/CF05.MRG', 'BROWN/CF/CF06.MRG', 'BROWN/CF/CF07.MRG', 'BROWN/CF/CF
08.MRG', 'BROWN/CF/CF09.MRG', 'BROWN/CF/CF10.MRG', 'BROWN/CF/CF11.MRG', 'BROWN/C
F/CF12.MRG', 'BROWN/CF/CF13.MRG', 'BROWN/CF/CF14.MRG', 'BROWN/CF/CF15.MRG', 'BRO
WN/CF/CF16.MRG', 'BROWN/CF/CF17.MRG', 'BROWN/CF/CF18.MRG', 'BROWN/CF/CF19.MRG',
'BROWN/CF/CF20.MRG', 'BROWN/CF/CF21.MRG', 'BROWN/CF/CF22.MRG', 'BROWN/CF/CF23.MR
G', 'BROWN/CF/CF24.MRG', 'BROWN/CF/CF25.MRG', 'BROWN/CF/CF26.MRG', 'BROWN/CF/CF2
7.MRG', 'BROWN/CF/CF28.MRG', 'BROWN/CF/CF29.MRG', 'BROWN/CF/CF30.MRG', 'BROWN/CF
/CF31.MRG', 'BROWN/CF/CF32.MRG', 'BROWN/CG/CG01.MRG', 'BROWN/CG/CG02.MRG', 'BROW
N/CG/CG03.MRG', 'BROWN/CG/CG04.MRG', 'BROWN/CG/CG05.MRG', 'BROWN/CG/CG06.MRG', '
BROWN/CG/CG07.MRG', 'BROWN/CG/CG08.MRG', 'BROWN/CG/CG09.MRG', 'BROWN/CG/CG10.MRG
', 'BROWN/CG/CG11.MRG', 'BROWN/CG/CG12.MRG', 'BROWN/CG/CG13.MRG', 'BROWN/CG/CG14
```

• • •

```
WSJ_2416.MRG', 'WSJ/24/WSJ_2417.MRG', 'WSJ/24/WSJ_2418.MRG', 'WSJ/24/WSJ_2419.MR
G', 'WSJ/24/WSJ_2420.MRG', 'WSJ/24/WSJ_2421.MRG', 'WSJ/24/WSJ_2422.MRG', 'WSJ/24
/WSJ_2423.MRG', 'WSJ/24/WSJ_2424.MRG', 'WSJ/24/WSJ_2425.MRG', 'WSJ/24/WSJ_2426.M
RG', 'WSJ/24/WSJ_2427.MRG', 'WSJ/24/WSJ_2428.MRG', 'WSJ/24/WSJ_2429.MRG', 'WSJ/2
4/WSJ_2430.MRG', 'WSJ/24/WSJ_2431.MRG', 'WSJ/24/WSJ_2432.MRG', 'WSJ/24/WSJ_2433.
MRG', 'WSJ/24/WSJ_2434.MRG', 'WSJ/24/WSJ_2435.MRG', 'WSJ/24/WSJ_2436.MRG', 'WSJ/
24/WSJ_2437.MRG', 'WSJ/24/WSJ_2438.MRG', 'WSJ/24/WSJ_2439.MRG', 'WSJ/24/WSJ_2440
.MRG', 'WSJ/24/WSJ_2441.MRG', 'WSJ/24/WSJ_2442.MRG', 'WSJ/24/WSJ_2443.MRG', 'WSJ
/24/WSJ_2444.MRG', 'WSJ/24/WSJ_2445.MRG', 'WSJ/24/WSJ_2446.MRG', 'WSJ/24/WSJ_244
7.MRG', 'WSJ/24/WSJ_2448.MRG', 'WSJ/24/WSJ_2449.MRG', 'WSJ/24/WSJ_2450.MRG', 'WS
J/24/WSJ_2451.MRG', 'WSJ/24/WSJ_2452.MRG', 'WSJ/24/WSJ_2453.MRG', 'WSJ/24/WSJ_24
54.MRG']
>>>
```

Checking the install:

class BracketParseCorpusReader

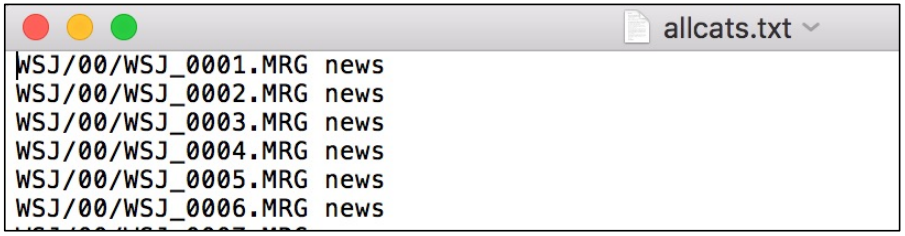
seems to be the Brown corpus + the Wall Street Journal corpus

Penn Treebank (PTB) with nltk

- WSJ only:

```
>>> ptb.categories()
['adventure', 'belles_lettres', 'fiction', 'humor', 'lore', 'mystery', 'news', '
romance', 'science_fiction']
>>> ptb.fileids('news')
['WSJ/00/WSJ_0001.MRG', 'WSJ/00/WSJ_0002.MRG', 'WSJ/00/WSJ_0003.MRG', 'WSJ/00/WS
J_0004.MRG', 'WSJ/00/WSJ_0005.MRG', 'WSJ/00/WSJ_0006.MRG', 'WSJ/00/WSJ_0007.MRG'
, 'WSJ/00/WSJ_0008.MRG', 'WSJ/00/WSJ_0009.MRG', 'WSJ/00/WSJ_0010.MRG', 'WSJ/00/W
SJ_0011.MRG', 'WSJ/00/WSJ_0012.MRG', 'WSJ/00/WSJ_0013.MRG', 'WSJ/00/WSJ_0014.MRG
', 'WSJ/00/WSJ_0015.MRG', 'WSJ/00/WSJ_0016.MRG', 'WSJ/00/WSJ_0017.MRG', 'WSJ/00/
WSJ_0018.MRG', 'WSJ/00/WSJ_0019.MRG', 'WSJ/00/WSJ_0020.MRG', 'WSJ/00/WSJ_0021.MR
G', 'WSJ/00/WSJ_0022.MRG', 'WSJ/00/WSJ_0023.MRG', 'WSJ/00/WSJ_0024.MRG', 'WSJ/00
/WSJ_0025.MRG', 'WSJ/00/WSJ_0026.MRG', 'WSJ/00/WSJ_0027.MRG', 'WSJ/00/WSJ_0028.M
```

- Defined in `~/nltk_data/corpora/ptb/allcats.txt`:



Only WSJ and BROWN corpora

Tagset: Penn/Brown vs. Universal

Several tagged corpora support access to a simplified, universal tagset, e.g. where all nouns tags are collapsed to a single category `NOUN`:

```
>>> print(brown.tagged_sents(tagset='universal'))
[[('The', 'DET'), ('Fulton', 'NOUN'), ('County', 'NOUN'), ('Grand', 'ADJ'), ('Jury', 'NO
 [('The', 'DET'), ('jury', 'NOUN'), ('further', 'ADV'), ('said', 'VERB'), ('in', 'ADP'),
>>> from nltk.corpus import conll2000, switchboard
>>> print(conll2000.tagged_words(tagset='universal'))
[('Confidence', 'NOUN'), ('in', 'ADP'), ...]
```

Brown and PTB Mappings:

- https://github.com/slavpetrov/universal-pos-tags/blob/fca8727e9424255f0732d1bc437f432f45a0c166/en-brown.map
- https://github.com/slavpetrov/universal-pos-tags/blob/c8e49bf1654d337d55553fabd75b4073596feac7/en-ptb.map

```
10  Interface for converting POS tags from various treebanks
11  to the universal tagset of Petrov, Das, & McDonald.
12
13  The tagset consists of the following 12 coarse tags:
14
15  VERB - verbs (all tenses and modes)
16  NOUN - nouns (common and proper)
17  PRON - pronouns
18  ADJ - adjectives
19  ADV - adverbs
20  ADP - adpositions (prepositions and postpositions)
21  CONJ - conjunctions
22  DET - determiners
23  NUM - cardinal numbers
24  PRT - particles or other function words
25  X - other: foreign words, typos, abbreviations
26  . - punctuation
```

Tagset: Penn vs. Universal

https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html

!	.	CC	CONJ	MD	VERB	VB	VERB	PRT	PRT
#	.	CD	NUM	NN	NOUN	VBD	VERB	RB	ADV
$	.	CD|RB	X	NNP	NOUN	VBD|VBN		RBR	ADV
''	.	DT	DET	NNPS	NOUN		VERB	RBS	ADV
(	.	EX	DET	NNS	NOUN	VBG	VERB	RB|RP	ADV
)	.	FW	X	NN|NNS	NOUN	VBG|NN	VERB	RB|VBG	ADV
,	.	IN	ADP	NN|SYM	NOUN	VBN	VERB	RN	X
-LRB-	.	IN|RP	ADP	NN|VBG	NOUN	VBP	VERB	RP	PRT
-RRB-	.	JJ	ADJ	NP	NOUN	VBP|TO	VERB	SYM	X
.	.	JJR	ADJ	PDT	DET	VBZ	VERB	TO	PRT
:	.	JJRJR	ADJ	POS	PRT	VP	VERB	UH	X
?	.	JJS	ADJ	PRP	PRON			WDT	DET
LS	X	JJ|RB	ADJ	PRP$	PRON			WH	X
``	.	JJ|VBG	ADJ	PRP|VBP	PRON			WP	PRON
								WP$	PRON
								WRB	ADV

Homework 5

- Part 1: install ptb
- Part 2: install Structural Probes

structural probes: installation

Follow the instructions here:

- https://github.com/john-hewitt/structural-probes

3. Install the required packages. This mainly means `pytorch`, `scipy`, `numpy`, `seaborn`, etc. Look at pytorch.org for the PyTorch installation that suits you and install it; it won't be installed via `requirements.txt`. Everything in the repository will use a GPU if available, but if none is available, it will detect so and just use the CPU, so use the pytorch install of your choice.

Two ways: use *miniconda* or from scratch

```
conda install --file requirements.txt
pip install pytorch-pretrained-bert
```

requirements.txt

```
Cython
seaborn
scipy
PyYAML
numpy
h5py
tqdm
```

structural probes: installation

1. Clone the repository:

```
% git clone https://github.com/john-hewitt/structural-probes/
Cloning into 'structural-probes'...
remote: Enumerating objects: 258, done.
remote: Total 258 (delta 0), reused 0 (delta 0), pack-reused 258
Receiving objects: 100% (258/258), 591.69 KiB | 296.00 KiB/s,
done.
Resolving deltas: 100% (193/193), done.
% cd structural-probes
```

structural probes: installation

- If doing package install manually:
  - `pip3 install torch`
  - `pip3 install seaborn`
  - `pip3 install cython`
  - `pip3 install pyyaml`
  - `pip3 install h5py`
  - `pip3 install tqdm`
  - `pip3 install pytorch-pretrained-bert`
  - `bash ./download_example.sh` (must have `wget` installed; brew on mac)

structural probes: installation

- Step 4: pre-packaged data

`bash ./download_example.sh` (*must have* wget *installed*; brew on mac)

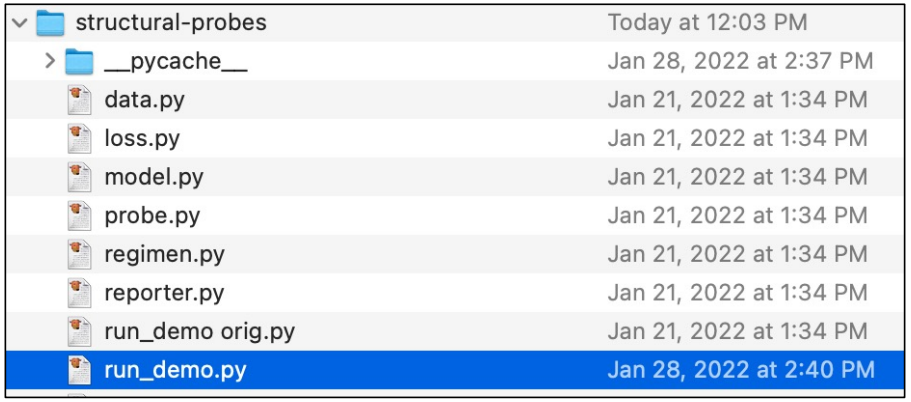
```
https://nlp.stanford.edu/~johnhew/public/en_ewt-ud-sample.tgz
Resolving nlp.stanford.edu (nlp.stanford.edu)... 171.64.67.140
Connecting to nlp.stanford.edu
(nlp.stanford.edu)|171.64.67.140|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 43733896 (42M) [application/x-tar]
Saving to: ‘en_ewt-ud-sample.tgz’
en_ewt-ud-sample.tg 100%[===================>] 41.71M 2.74MB/s in
15s

2022-02-17 10:49:49 (2.84 MB/s) - ‘en_ewt-ud-sample.tgz’ saved
[43733896/43733896]
--2022-02-17 10:49:49--
  https://nlp.stanford.edu/~johnhew/public/sp/bertlarge16-
distance-probe.params
Resolving nlp.stanford.edu (nlp.stanford.edu)... 171.64.67.140
Connecting to nlp.stanford.edu
(nlp.stanford.edu)|171.64.67.140|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 4194722 (4.0M) [text/plain]
Saving to: ‘bertlarge16-distance-probe.params’
bertlarge16-distanc 100%[===================>] 4.00M  3.39MB/s in
1.2s
2022-02-17 10:49:50 (3.39 MB/s) - ‘bertlarge16-distance-
probe.params’ saved [4194722/4194722]
```

```
--2022-02-17 10:49:50--  https://nlp.stanford.edu/~johnhew/public/sp/bertlarge16-
depth-probe.params
Resolving nlp.stanford.edu (nlp.stanford.edu)... 171.64.67.140
Connecting to nlp.stanford.edu (nlp.stanford.edu)|171.64.67.140|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 4194722 (4.0M) [text/plain]
Saving to: ‘bertlarge16-depth-probe.params’
bertlarge16-depth-p 100%[===================>] 4.00M  2.97MB/s in 1.3s
2022-02-17 10:49:52 (2.97 MB/s) - ‘bertlarge16-depth-probe.params’ saved
[4194722/4194722]
x en_ewt-ud-sample/
x en_ewt-ud-sample/en_ewt-ud-dev.elmo-layers.hdf5
x en_ewt-ud-sample/en_ewt-ud-test.elmo-layers.hdf5
x en_ewt-ud-sample/en_ewt-ud-test.conllu
x en_ewt-ud-sample/en_ewt-ud-train.conllu
x en_ewt-ud-sample/en_ewt-ud-train.elmo-layers.hdf5
x en_ewt-ud-sample/en_ewt-ud-dev.txt
x en_ewt-ud-sample/en_ewt-ud-test.txt
x en_ewt-ud-sample/en_ewt-ud-train.txt
x en_ewt-ud-sample/en_ewt-ud-dev.conllu
```

structural probes: installation issues

- **PyYAML** problem
  - YAML is a readable data representation language for storing data (in files), bit like JSON (JavaScript Object Notation).
- Error:
  - `echo "The chef that went to the stores was out of food" | python3 structural-probes/run_demo.py example/demo-bert.yaml`
  - `Traceback (most recent call last):`
  - `  File "/Users/sandiway/work/structural-probes/structural-probes/run_demo.py", line 174, in <module>`
  - `    yaml_args= yaml.load(open(cli_args.experiment_config))`
  - `TypeError: load() missing 1 required positional argument: 'Loader'`

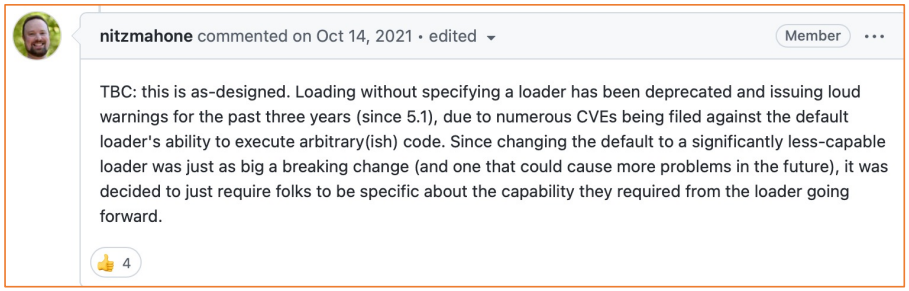
structural probes: installation issues




```
174  yaml_args= yaml.load(open(cli_args.experiment_config), Loader=yaml.Loader)
175  run_experiment.setup_new_experiment_dir(cli_args, yaml_args, cli_args.resu
   lts_dir)
176  device = torch.device("cuda:0" if torch.cuda.is_available() else "cpu")
177  yaml_args['device'] = device
178  report_on_stdin(yaml_args)
```

structural probes: installation issues

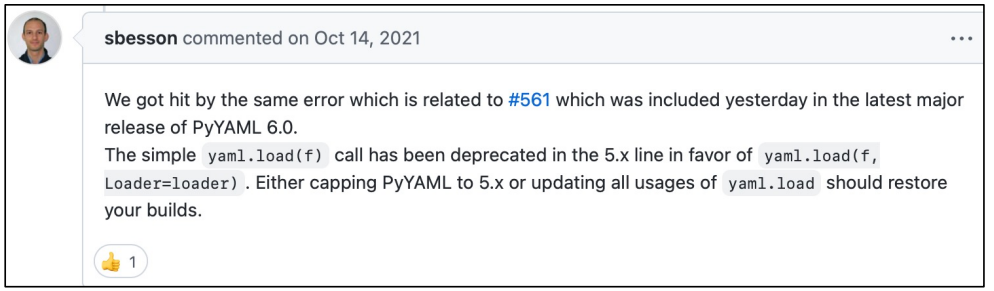
https://github.com/yaml/pyyaml/issues/576



nitzmahone commented on Oct 14, 2021 • edited — Member

TBC: this is as-designed. Loading without specifying a loader has been deprecated and issuing loud warnings for the past three years (since 5.1), due to numerous CVEs being filed against the default loader's ability to execute arbitrary(ish) code. Since changing the default to a significantly less-capable loader was just as big a breaking change (and one that could cause more problems in the future), it was decided to just require folks to be specific about the capability they required from the loader going forward.

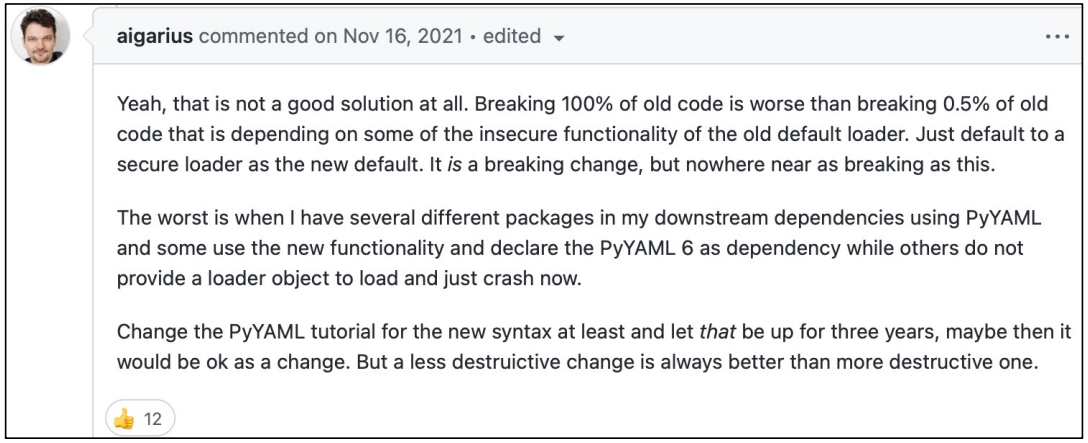
👍 4



sbesson commented on Oct 14, 2021

We got hit by the same error which is related to #561 which was included yesterday in the latest major release of PyYAML 6.0.
The simple `yaml.load(f)` call has been deprecated in the 5.x line in favor of `yaml.load(f, Loader=loader)`. Either capping PyYAML to 5.x or updating all usages of `yaml.load` should restore your builds.

👍 1



aigarius commented on Nov 16, 2021 • edited

Yeah, that is not a good solution at all. Breaking 100% of old code is worse than breaking 0.5% of old code that is depending on some of the insecure functionality of the old default loader. Just default to a secure loader as the new default. It *is* a breaking change, but nowhere near as breaking as this.

The worst is when I have several different packages in my downstream dependencies using PyYAML and some use the new functionality and declare the PyYAML 6 as dependency while others do not provide a loader object to load and just crash now.

Change the PyYAML tutorial for the new syntax at least and let *that* be up for three years, maybe then it would be ok as a change. But a less destruictive change is always better than more destructive one.

👍 12

structural probes: demo

- Make sure you're in the structural-probes directory in your Terminal

```
printf "The chef that went to the stores was out of food" | python3 structural-probes/run_demo.py example/demo-bert.yaml
Constructing new results directory at example/results/BERT-disk-demo-2022-2-17-10-55-25-468055/
The pre-trained model you are loading is a cased model but you have not set `do_lower_case` to False. We are setting `do_lower_case=False` for you but you may want to check this behavior.
100%|████████████████████████████████████| 213450/213450 [00:00<00:00, 228889.25B/s]
100%|████████████████████████████████| 1242874899/1242874899 [14:20<00:00, 1444011.87B/s]
Constructing TwoWordPSDProbe
Constructing OneWordPSDProbe
[demoing]: 1it [00:02,  2.90s/it]
```

1st time only

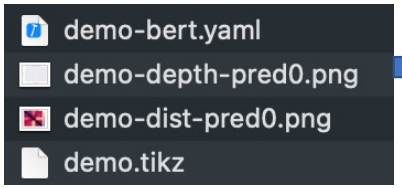
structural probes: demo

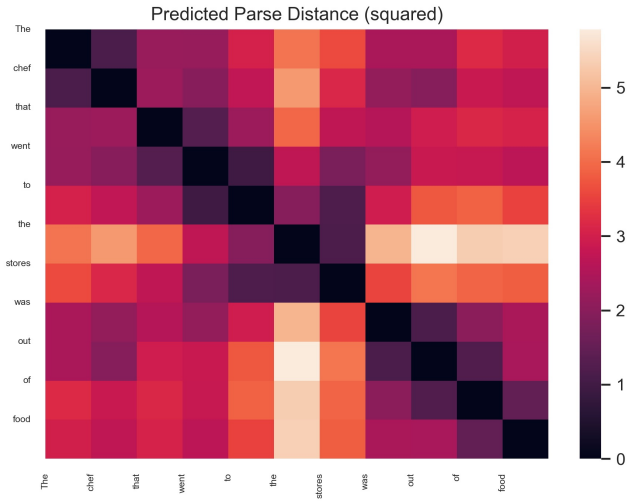
- In the structural-probes directory:

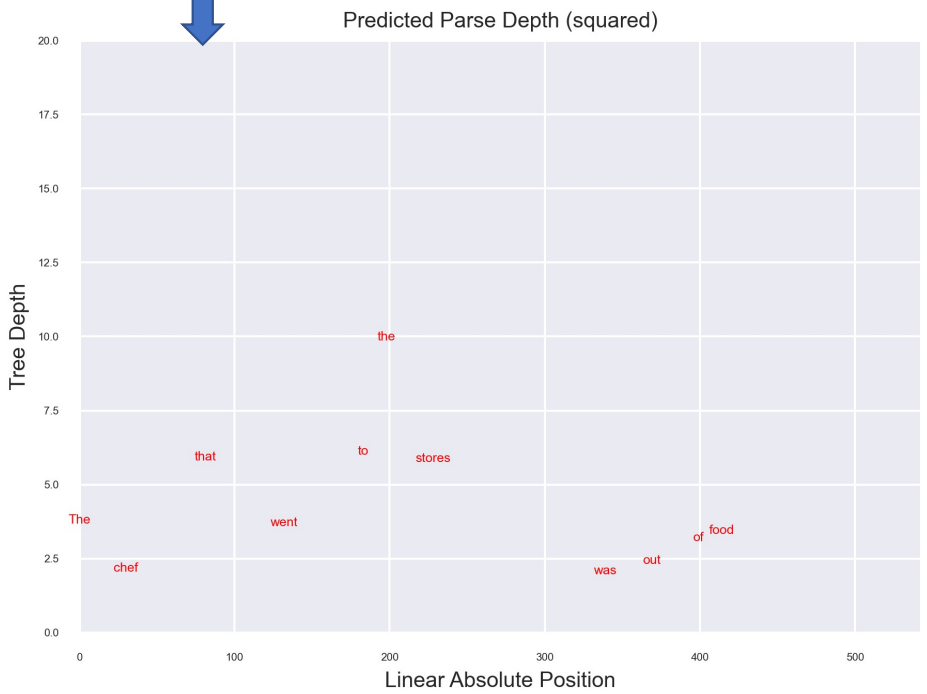
```
(base) structural-probes$ echo "The chef that went to the stores was out of food" | python structural-probes/run_demo.py example/demo-bert.yaml
Constructing new results directory at example/results/BERT-disk-demo-2022-2-16-20-27-9-540275/
The pre-trained model you are loading is a cased model but you have not set `do_lower_case` to False. We are setting `do_lower_case=False` for you but you may want to check this behavior.
Constructing TwoWordPSDProbe
Constructing OneWordPSDProbe
[demoing]: 1it [00:00,  1.31it/s]
```

structural probes: demo

`example/results/BERT-disk-demo-2022-2-16-20-27-9-540275/`







structural probes: demo

`demo.tikz`

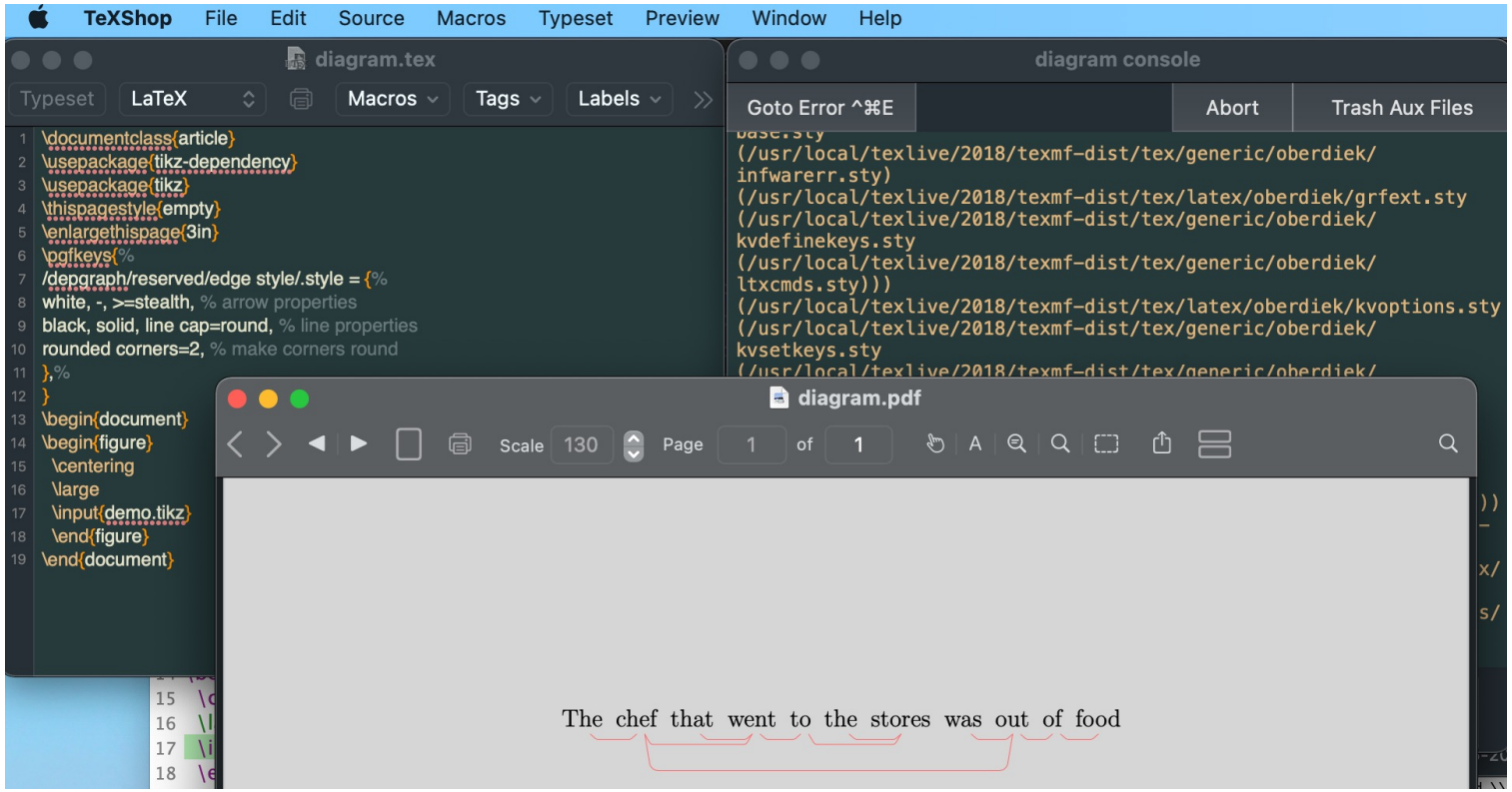
```
\begin{dependency}[hide label, edge unit distance=.5ex]
  \begin{deptext}[column sep=0.05cm]
  The\& chef\& that\& went\& to\& the\& stores\& was\& out\& of\& food \\
\end{deptext}
\depedge[edge style={red!60!}, edge below]{4}{5}{.}
\depedge[edge style={red!60!}, edge below]{1}{2}{.}
\depedge[edge style={red!60!}, edge below]{8}{9}{.}
\depedge[edge style={red!60!}, edge below]{6}{7}{.}
\depedge[edge style={red!60!}, edge below]{5}{7}{.}
\depedge[edge style={red!60!}, edge below]{9}{10}{.}
\depedge[edge style={red!60!}, edge below]{3}{4}{.}
\depedge[edge style={red!60!}, edge below]{10}{11}{.}
\depedge[edge style={red!60!}, edge below]{2}{9}{.}
\depedge[edge style={red!60!}, edge below]{2}{4}{.}
\end{dependency}
```

`diagram.tex` (*download from course website*)

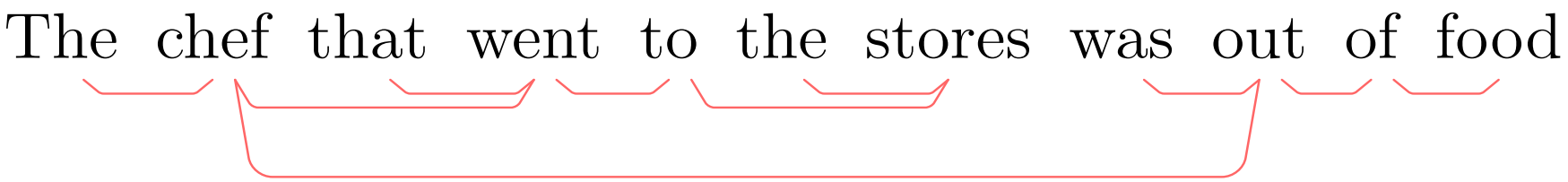
```
\documentclass{article}
\usepackage{tikz-dependency}
\usepackage{tikz}
\thispagestyle{empty}
\enlargethispage{3in}
\pgfkeys{%
/depgraph/reserved/edge style/.style = {%
white, -, >=stealth, % arrow properties

black, solid, line cap=round, % line properties
rounded corners=2, % make corners round
},%
}
\begin{document}
\begin{figure}
  \centering
  \large
  \input{demo.tikz}
  \end{figure}
\end{document}
```

structural probes: demo

- Run LaTeX:



structural probes: demo



https://cloud.google.com/natural-language